

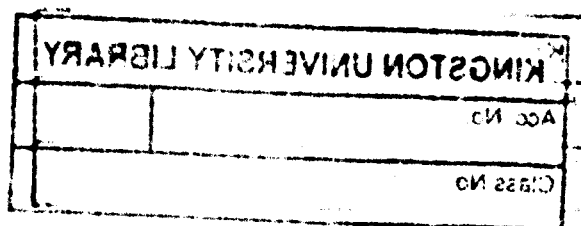
An Integrated Modeling Framework for Concept Formation: Developing Number–Sense, a Partial Resolution of the Learning Paradox

Gerard Vincent Alfred Rendell

A thesis submitted in partial fulfillment of the requirements of
Kingston University for the degree of Doctor of Philosophy

Faculty of Science, Engineering and Computing.

February, 2012



© Copyright 2012
By
Gerard Vincent Alfred Rendell
All Rights Reserved

Abstract

The development of mathematics is foundational. For the most part in early childhood it is seldom insurmountable. Various constructions exhibit conceptual change in the child, which is evidence of overcoming the learning paradox. If one tries to account for learning by means of mental actions carried out by the learner, then it is necessary to attribute to the learner a prior structure, one that is as advanced or as complex as the one to be acquired, unless there is emergence. This thesis reinterprets Piaget's theory using research from neurophysiology, biology, machine learning and demonstrates a novel approach to partially resolve the learning paradox for a simulation that experiences a number line world, exhibiting emergence of structure using a model of *Drosophila*. In doing so, the research evaluates other models of cognitive development against a real-world, worked example of number-sense from childhood mathematics. The purpose is to determine if they assume a prior capacity to solve problems or provide parallel assumptions within the learning process as additional capabilities not seen in children. Technically, the research uses an artificial neural network with reinforcement learning to confirm the emergence of *permanent object invariants*. It then evaluates an evolved *dialectic system* with hierarchical finite state automata within a reactive Argos framework to confirm the reevaluated Piagetian developmental model against the worked example. This research thesis establishes that the emergence of new concepts is a critical need in the development of autonomous evolvable systems that can act, learn and plan in novel ways, in noisy situations.

Acknowledgment

Thank you to my supervisor Chris Tompsett for his intellectual honesty, whose depth and breadth of knowledge, and his continued encouragement, kept me on this journey. I thank Dr. Tim Ellis, for getting me to the completed thesis, and for keeping me on track, I wish to thank Dr. Andy Augousti, and Dr. Souheil Khaddaj.

Thank you to Dr. Aaron Sloman of the University of Birmingham for very valid observations, and criticisms of this work. I wish to acknowledge the support of Dr. Juan Pascual-Leone of the department of psychology at the York University, Toronto, for early access to papers on the Theory of Constructive Operators and discussions on Piaget. Thank you to Dr. Christian Jacob of the department of computer science, at the University of Calgary for providing the “Evolvica evolutionary framework,” which I expanded upon in this research. Thank you to Dr. Kathleen Cramer of the University of Minnesota for answering questions on the development of early mathematical knowledge from the archives of the Rational Number Project.

I would also like to thank my examiners Dr. Alison Pease and Dr. James Orwell, who made my viva an enjoyable, stimulating and challenging event. They have made many valuable contributions to the thesis and have been particularly inspiring as to future directions that this project may now take. I would also like to thank Dr. Joe Hester for the clarity of his prose and composition, that brings this work to life.

To my wife, Carole and my children (James, Kylah, Lisa and Valerie), thank you for giving me the love, time, resources and support to complete this undertaking. A special thank you goes to Kathleen Rast, for giving me peace of mind during my studies.

Lastly, to my late father Vincent Rendell, for providing me with an understanding of childhood mathematics and a lifelong interest in paradoxes.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that the work has not been submitted for any other degree or qualification except as specified.

(Gerard Vincent Alfred Simon Rendell)

Publications

Parts of chapter 2 and 4 have appeared in the *International Journal of Learning* (Rendell, 2008).

TABLE OF CONTENTS

Abstract	iii
Acknowledgment.....	iv
Declaration	v
Publications	vi
List of Tables.....	xx
List of Figures	xxii
Keywords	xxx
1. Introduction	1
1.1 Motivation.....	5
1.1.1 Research Purpose.....	5
1.1.2 Research Questions.....	5
1.1.3 Research Hypotheses and Research Problems.....	6
1.1.4 Research Validity	6
1.2 Aims of the Project	6
1.3 Contributions	7
1.4 Structure of the Thesis	7
1.5 Summary.....	9
2. Literature Review.....	10
2.1 Epistemological Issues.....	11
2.1.1 The Debate on the Learning Paradox	11
2.1.2 Chomsky.....	11
2.1.2.1 Chomsky: Where is the problem?.....	12
2.1.3 Bereiter	14
2.1.3.1 Bereiter: Where is the problem?	15
2.1.4 Fodor	15
2.1.4.1 Fodor: Where is the problem?	17
2.1.5 Piaget.....	18
2.1.5.1 Piaget: Where is the problem?	21

2.1.6	Pascual–Leone	22
2.1.6.1	Pascual–Leone: Where is the problem?	25
2.1.7	Constraints of a Simulation	26
2.1.7.1	An Environment for a Simulation.....	26
2.1.7.2	LP1: Works Within the Constraints Imposed By Fodor’s Arguments	27
2.1.7.3	LP2: Exhibits Emergence of Hierarchical Concepts using Evolutionary Process	28
2.1.7.4	LP3: Operates Autonomously.....	28
2.1.7.5	LP4: Mirrors Real World Behavior of Children Learning Number–Sense using a Number line.....	28
2.1.7.6	LP5: Learns Incrementally with Minimal Innate Knowledge and Reflexes	28
2.1.7.7	LP6: Develops its Learning Process	29
2.1.7.8	LP7: Acts in Novel, Opportunistic and Noisy Situations	29
2.1.8	Conclusion.....	29
2.2	Biological Evolution as a Basis for Emergence	30
2.2.1	Summary.....	30
2.2.2	Conclusion.....	31
2.3	The Roots of Genetic Epistemology	32
2.3.1	Causality	32
2.3.2	Assimilation and Accommodation	33
2.3.3	Relationship of Assimilation to Figurative Scheme and Accommodation to Operative Schemes	33
2.3.4	The Primacy of Action	33
2.3.5	The Knowing Circle is an Abstract Process on Innate Scheme and Primary Reaction.....	35
2.3.6	The Deeper Structure of the Knowing Circle	36
2.3.6.1	Perception and Recognition	37
2.3.6.2	Imitation and Reconstruction.....	38
2.3.6.3	Mental Image and Evocation	38
2.3.6.4	Summary.....	39
2.3.7	Relationship of Piagetian Model to Biological Processing.....	39

2.3.8	The Importance of Developmental Trend.....	40
2.3.9	The Impact of Reflective Abstraction on Development	41
2.3.10	Impact of Evolutionary Trend on Development through Interiorization	42
2.3.11	The Impact of Equilibration and Disequilibration	43
2.3.12	The Impact of Equilibration and Disequilibration Piaget and Mathematics	44
2.3.13	Development of Symbols	45
2.3.14	Worked Example	46
2.3.14.1	WE1 – Base Level	47
2.3.14.2	WE2 – Constrained Level.....	48
2.3.14.3	WE3 – Differentiated Level – Object	49
2.3.14.4	WE4 – Hierarchical Level – Segmentation into Units.....	49
2.3.14.5	WE5 – General Level – Relationships of Unit Values	50
2.3.14.6	Summary.....	51
2.3.15	Conclusion.....	52
2.4	Number–Line Number Sense.....	52
2.4.1	Number–Sense from an Educational Perspective	53
2.4.1.1	Implications for the Research	54
2.4.2	Biology and Neurophysiology on Development of Number–Sense	54
2.4.2.1	Implications for the Research	54
2.4.3	Cultural Impact on Development of Number–Sense using a MNL.....	54
2.4.3.1	Implications for the Research	55
2.4.4	Number–Sense is Innate but Culture Makes a Difference	56
2.4.4.1	Number–Sense is Innate	56
2.4.4.2	Culture Impacts Development of Number–Sense.....	56
2.4.4.3	Evolution Provided a Deep Foundations for Number–Sense.....	56
2.4.4.4	Observations of Logarithmic Use of Number line.....	56
2.4.4.5	Implications for the Research	57
2.4.5	Conclusions	57
2.5	Concept Formation in Automated Mathematics	58
2.5.1	Introduction	58

2.5.2	The HR System.....	58
2.5.2.1	Searching for Definitions – The Process of Theory Formation in HR.....	60
2.5.2.2	Future Work for HR.....	61
2.5.2.3	HR: Where is the Problem?	61
2.5.3	Conclusions	62
2.6	The Underlying Need of Emergence in Metaphors and Conceptual Blending	62
2.6.1	Background.....	63
2.6.2	Metaphor: Where Mathematics Comes From.....	63
2.6.2.1	Criticisms.....	64
2.6.2.2	Metaphor: Where is the problem?.....	65
2.6.3	Conceptual Blending	66
2.6.3.1	Networks and Prototypes	67
2.6.3.2	Example of Conceptual Blending on a MNL.....	69
2.6.3.3	Conceptual Blending: Where is the problem?	70
2.6.4	Conceptual Blending as a Basis for Mathematics.....	72
2.6.4.1	Conceptual Blending: Where is the Problem?	73
2.6.5	Conclusions	73
2.7	Cognitive Development Models and the Need for Emergence.....	74
2.7.1	Foundations	74
2.7.1.1	Background.....	74
2.7.1.2	Why a Solution is Needed	75
2.7.1.3	What is the composition of that solution?.....	75
2.7.1.4	A Suitable Problem.....	75
2.7.1.5	Grading Cognitive Theories.....	76
2.7.1.6	Problems with Newell’s 12 Criteria Test.....	76
2.7.2	Symbolic Models.....	77
2.7.2.1	Symbol Grounding Problem	77
2.7.2.2	McCarthy as an Early Symbolic Model.....	78
2.7.2.3	Searle’s Chinese Room Argument.....	79
2.7.2.4	SOAR.....	80

2.7.2.5	Resolution of Symbol Grounding Using Sub-Symbolic Sensory Data.....	81
2.7.2.6	Resolution using Theory Tethering	82
2.7.2.7	Resolution of Symbol Grounding using SOAR-R	84
2.7.2.8	Resolution using Piagetian Approach.....	84
2.7.2.9	Symbolic Models: Where is the Problem?.....	84
2.7.3	Emergent Models.....	85
2.7.3.1	IBCA.....	85
2.7.3.2	ART	86
2.7.4	Hybrid Models.....	86
2.7.4.1	The Necessity of Dual-Process Theory	87
2.7.4.2	Conceptual Change	88
2.7.4.3	Analysis of Hybrid Models.....	89
2.7.5	Conclusions	89
2.8	Piagetian Implementations.....	90
2.8.1	Pascual-Leone	90
2.8.2	Drescher.....	90
2.8.2.1	Drescher: Where is the problem?.....	93
2.8.3	CLA	93
2.8.3.1	CLA: Where is the problem?	93
2.8.4	Conclusions	94
2.9	Other Frameworks	94
2.9.1	Case Based Reasoning.....	94
2.9.2	Unsupervised Visual Learning	95
2.9.2.1	Adaptive Resonance Theory (ART)	95
2.9.2.2	Self-Taught Learning Framework.....	96
2.9.2.3	Unsupervised Learning – Where is the Problem?.....	96
2.9.3	Emotional / Intuitive Models.....	97
2.9.4	Conclusions	97
2.10	Summary – The Two-Fold Need.....	98
3.	Research Methods	99

3.1	Research Area	99
3.1.1	Research Question	99
3.1.2	Research Purpose.....	99
3.1.3	Research Problem.....	100
3.2	Research Method	100
3.2.1	Research Goals	101
3.3	Research Framework	102
3.3.1	Observations	102
3.3.2	Research Hypotheses	102
3.3.3	Research Tenets.....	104
3.3.4	Research Constraints	104
3.4	Research Design	104
3.5	Research Experiments.....	105
3.5.1	Experiments: Artificial Neural Network Implementation	106
3.5.2	Experiments: Dialectic System.....	107
3.6	Summary.....	108
4.	Architecture and Design.....	109
4.1	Artificial Neural Network Implementation	109
4.1.1	Architecture	109
4.1.2	Design.....	109
4.1.2.1	The Learning Task	110
4.1.2.2	Sensing the Environment	111
4.1.2.3	Example Execution of ANN Implementation.....	112
4.2	Dialectic System	112
4.2.1	Architecture	112
4.2.1.1	Key Implementation Features.....	114
4.2.1.2	Summary of Major Architectural Features and Design Elements	119
4.2.2	Design.....	120
4.2.2.1	Number line World	120
4.2.2.2	Simulated Student	122

4.2.2.3	Simulated Teacher	126
4.2.2.4	Auditor.....	127
4.2.2.5	Learning as Evolution – Using Evolutionary Computing to Learn Schemes.....	127
4.2.3	Algorithms.....	145
4.2.3.1	Algorithm 1 – Prediction Quality	146
4.2.3.2	Algorithm 2 – Automaton Response.....	147
4.2.3.3	Algorithm 3 – Evaluate Automaton Fitness.....	147
4.2.3.4	Algorithm 4 – Operators.....	148
4.2.3.5	Algorithm 5 – Evolve Classes	149
4.2.3.6	Algorithm 6 – Evolve Chromosome for Class.....	151
4.2.3.7	Algorithm 7 – Evolution Strategy.....	153
4.2.3.8	Algorithm 8 – Create Chromosomes	154
4.2.3.9	Algorithm 9 – Best Selection.....	154
4.2.3.10	Algorithm 10 – Elite Selection	155
4.2.3.11	Algorithm 11 – Fitness Proportionate Selection.....	155
4.2.3.12	Algorithm 12 – Rank Based Selection.....	156
4.2.3.13	Algorithm 13 – Random Selection	157
4.2.3.14	Algorithm 14 – Tournament Selection	157
4.2.3.15	Algorithm 15 – Add Class	158
4.2.3.16	Algorithm 16 – Act on Environment	159
4.2.3.17	Algorithm 17 – Sense Environment.....	159
4.2.3.18	Algorithm 18 – Imitation	160
4.2.3.19	Algorithm 19 – Reconstruction.....	160
4.2.4	Example Execution of Dialectic System	161
4.2.4.1	Level 1 – Regularity	161
4.2.4.2	Level 2 – Coordinated Action.....	162
4.2.4.3	Level 3 – Internalized Structure.....	162
4.2.4.4	Level 4 – Symbolic Functions	162
4.3	Summary.....	163
5.	Evaluation	164

5.1	Evaluation of Artificial Neural Network Implementation	165
5.1.1	Level 1 – Identification of Permanent Objects Invariants	165
5.1.1.1	Detecting Regularities in the environment.....	165
5.1.1.2	Coordinated Actions in the Number Line World.....	167
5.1.1.3	Planning Action Tests.....	167
5.1.2	Level 2 – Coordinated Movements.....	170
5.1.3	Summary.....	171
5.2	Evaluation of Dialectic System.....	172
5.2.1	Level 1 – Regularities in the Environment	172
5.2.1.1	Detecting Regularities in the environment.....	172
5.2.1.2	Acting and Sensing the Environment	178
5.2.1.3	Summary.....	182
5.2.2	Level 2 – Coordinated Action.....	182
5.2.2.1	Prediction Scheme: Move Right	183
5.2.2.2	Prediction Scheme: Move Right 1 Micro Unit	185
5.2.2.3	Mutation Rates for Coordinated Action.....	186
5.2.2.4	Support for Worked Example	187
5.2.2.5	Resolution of Bead Problem through Conservation.....	187
5.2.2.6	The Need for Interiorization to Develop Number–Sense	189
5.2.2.7	Summary of Level 1 and Level 2 Results	190
5.2.3	Level 3 – Internalized Structure (Interiorization).....	190
5.2.3.1	Prediction Scheme: OR Prediction Scheme.....	190
5.2.3.2	Prediction Scheme: XOR Prediction Scheme.....	192
5.2.3.3	Final Evolution of Classes	192
5.2.3.4	Support for Worked Example	194
5.2.3.5	Summary.....	194
5.2.4	Level 4a– Argos Schemes for Symbolic Functions	194
5.2.4.1	Argos Scheme: Internalize Values.....	196
5.2.4.2	Argos Scheme: Binary Encoder.....	196
5.2.4.3	Argos Scheme: Full–Adder	202

5.2.4.4	Argos Scheme: Full-Adder Evaluation of Resource Usage	204
5.2.4.5	Support for Worked Example	205
5.2.4.6	Summary.....	206
5.2.5	Level 4b- Piagetian Schemes for Symbolic Functions.....	206
5.2.5.1	Piagetian Scheme: Counting as a Worked Example	206
5.2.5.2	Evaluation of Complexity Values	215
5.2.5.3	Support for Worked Example	216
5.2.6	Satisfying the Constraints (LP1 – LP7)	217
5.2.6.1	LP1: Works Within the Constraints Imposed By Fodor’s Arguments	217
5.2.6.2	LP2: Exhibits Emergence of Hierarchical Concepts using Evolutionary Process	217
5.2.6.3	LP3: Operates Autonomously.....	217
5.2.6.4	LP4: Mirrors Real World Behavior of Children Learning Number-Sense using a Number line.....	218
5.2.6.5	LP5: Learns Incrementally with Minimal Innate Knowledge and Reflexes	218
5.2.6.6	LP6: Develops its Learning Process	218
5.2.6.7	LP7: Acts in Novel, Opportunistic and Noisy Situations	219
5.2.7	Scalability.....	219
5.2.8	Resemblance of Argos Schemes to Macaque Brains.....	220
5.2.9	Summary of Dialectic Evaluation.....	222
5.3	Summary.....	223
6.	Further Work.....	224
6.1	Improving the Implementation of Piagetian Methods.....	225
6.1.1	Singular Evolution for Finite and Hierarchical Automata	225
6.1.2	Developing the Predictive Model	225
6.1.3	Using Intrinsic Rewards	225
6.1.4	Incorporating Emergent Schemes.....	225
6.1.5	Expanded Use of Memories.....	226
6.1.6	Using Play to Develop Metaphors	226
6.1.7	Constraining Resources to Force Planning	226

6.2	Deepening the Model of Cognition.....	227
6.2.1	Extended Evaluation for Rational–Number–Sense	227
6.2.2	Comparisons to Models of the Mental Number line.....	227
6.2.3	Comparison to Other models of Mathematical Cognition	227
6.3	Extending the Model.....	228
6.3.1	Using Vision to see Beyond a Number line.....	228
6.3.2	Optimization and Simplification through Formal Methods.....	228
6.4	To Fully Resolve the Learning Paradox.....	228
6.5	Applications of this Research System.....	229
6.5.1	Classification Systems.....	229
6.5.2	Concept Formation using a Piagetian Model.....	229
6.5.3	Piagetian Studies.....	229
6.5.4	Conceptual Blending	229
6.6	Conclusion	230
7.	Conclusions	231
7.1	Have the Aims Been Achieved?	231
7.2	Contributions	233
7.2.1	Computational Cognitive Psychology	233
7.2.2	Evolutionary Computing	233
7.2.3	Concept Formation Systems	234
7.3	Conclusions.....	234
A.	Evolution and Emergence	235
A.1	Simple Models of Emergence of Structure.....	235
A.2	Fruit Fly Models of Emergence.....	236
A.3	Techniques to Model Neural Networks.....	236
A.3.1	Artificial Neural Networks.....	236
A.3.1.1	ANNs: Where is the Problem?.....	236
A.3.2	Finite State Automata.....	236
A.3.2.1	FSA: Where is the Problem?.....	237
A.3.3	Conclusions.....	237

A.4 Hierarchical Nature of Brain Networks	238
A.4.1 Conclusions.....	239
A.5 The Need to Validate a Biologically Plausible Model.....	239
A.6 Mechanisms for Structuring Networks.....	239
A.6.1 Biological Basis of Reinforcement Learning	239
A.6.2 Actor–Critic Temporal Difference Learning as an Implementation Mechanism	241
A.6.3 Conclusions.....	243
A.7 Mechanisms to Model Evolution.....	243
A.7.1 Evolutionary Strategies: Where is the Problem?.....	245
A.7.2 Conclusions.....	245
A.8 Resolving Issues with FSA.....	245
A.8.1 HFSA	245
A.8.2 Further Problems.....	247
A.8.3 Conclusions.....	247
A.9 The Need for a Model of To Test for Emergence.....	247
A.9.1 Intuitive Emergence	247
A.9.2 Pattern Formation.....	248
A.9.3 Intrinsic Emergence	248
A.9.4 Conclusions.....	249
A.10 Conclusions	249
B. Models of Development	251
B.1 Hybrid Model	251
B.1.1 SOAR.....	251
B.1.1.1 Bottom–Up Learning using SOAR–R.....	258
B.1.2 PolyScheme.....	259
B.1.2.1 PolyScheme: Where is the Problem?	259
B.1.3 Neuro–SOAR.....	259
B.1.3.1 Neuro–SOAR: Where is the Problem?.....	260
B.1.4 ACT–R.....	260
B.1.4.1 ACT–R: Where is the Problem?	260

B.1.5 CogAff.....	261
B.1.5.1 CogAff: Where is the Problem?	261
B.1.6 RCS	261
B.1.6.1 RCS: Where is the Problem?	262
C. Piagetian Models and Mathematics.....	264
C.1 Piagetian Mathematics.....	264
C.1.1 Logical Connectives	265
C.1.2 Spatial Concepts	265
C.1.3 Proximity.....	266
C.1.4 Separation.....	266
C.1.5 Order	266
C.1.6 Surrounding	267
C.1.7 Conservation of Point.....	267
C.1.8 Conservation of Length.....	267
C.1.9 Conservation of Distance	267
C.1.10 Equivalence	268
C.1.11 Subdivision.....	268
C.1.12 Substitution	269
C.1.13 Conservation of Measurement.....	271
C.1.14 Intersection	271
C.1.15 Reversibility	272
C.1.16 Inclusion Relation	272
C.1.17 Transitivity	272
C.1.18 Seriation	273
C.1.19 Conservation of Number	276
C.1.20 Counting.....	276
C.1.21 Commutative Property	276
C.1.22 Multiplication and Division.....	277
C.1.23 Introspection.....	277
C.1.24 Classification.....	277

C.1.25 Ordinality and Cardinality.....	277
C.1.26 Fractions and Proportions.....	278
C.1.27 Generalization	278
C.1.28 Other Conditions	279
C.2 Rational Number–Sense	280
C.2.1 WE6 – Rational Level – Relationships of Unit Values to Continuous Values	280
D. Further Examples of Dialectic Evaluation.....	282
D.1 Argos Scheme – Movement	282
D.1.1 Argos Scheme – Start Number line.....	283
D.2 Argos Scheme – Full Subtractor.....	283
D.3 Argos Scheme – Decoder	287
D.4 Argos Scheme – Counter.....	289
D.5 Argos Scheme – Full–Adder	292
D.6 Piagetian Scheme: Using Planning to Count.....	298
Bibliography	302
Glossary.....	323
Index.....	330
End Notes	343

List of Tables

Table 2–1 This table defines the traceability matrix of relationships between the features exposed through biology and evolution to the concepts in Piagetian theory.....	39
Table 2–2 This table describes the anticipated observations of a simulation as it acts and senses in a number line world. It is initially seeded with a Piagetian model of learning and developing along with a set of innate schemes and primary reactions. As the system executes, it is anticipated that it will exhibit these levels of development. An example is presented that exhibits the progressive emergence of more complex structures, and this is mirrored in an external representation. Since all constraints (LP1 – LP7) apply throughout the life of the simulation, the level in which the constraint is most clearly observed to have been adhered to, is identified.	51
Table 2–3 Evaluation of cognitive development models to support LP1 – LP7	89
Table 2–4 Evaluation of Piagetian models to support LP1 – LP7	94
Table 2–5 Evaluation of Other Piagetian Frameworks to support LP1 – LP7.....	98
Table 3–1 Research Methods	100
Table 4–1 The set of actions that the simulated student can randomly select from to adapt to the environment.	117
Table 4–2 This table defines the traceability matrix of relationships between the features exposed through biology and evolution, to the concepts in Piagetian theory and the architecture and design features used in the simulation.....	119
Table 4–3 Depiction of the channel capacity of FSAs. This channel capacity is used when automata are randomly combined.	134
Table 4–4 A Table of states in the Argos equivalence scheme	135
Table 4–5 Equivalence – synchronizations (transitions).....	136
Table 4–6 Equivalence Table – Example Outputs. These outputs, provided by the environment are used in the evaluation of this class of machines.	137
Table 4–7 Prolog and Epilog processing for Piagetian Schemes.....	139
Table 4–8 This table lists the name of the algorithms employed in the system and their relationship to the Piagetian processes. Since the overall process consists of the processing of a state machine (as described in figure 4–24), only the included algorithms are listed.....	144
Table 4–9 Global structures used in the research.	145
Table 4–10 Normal Operation of the system.....	162
Table 5–1 A summary evaluation of the sensory motor development (actions in the external world) of an agent composed of linear neural network operating with continuous and discrete sensors (as radial basis function neural network) as they explore their number line world and develops an internal network that corresponds to objects in the external world.	171
Table 5–2 Parameters to change the penState to penUp in the environment.....	175
Table 5–3 Parameters for the reinforce action.....	177

Table 5-4 Parameters for Move Right 1 Unit in the environment.....	183
Table 5-5 This table describes the states in a Argos binary encoder scheme. Since these Argos schemes are randomly evolved, this is the depiction of viable schemes.	197
Table 5-6 Synchronizations (Transitions) for Argos binary encoder scheme.....	198
Table 5-7 These are the outputs of Argos binary encoder scheme, which are used in the evaluation function to test the random schemes. When schemes scale up in complexity, the reward is an aspect of the environment, which weights the network in the Argos scheme.	199
Table 5-8 Summary of resource usage in Argos schemes	205
Table 5-9 The steps in the UML sequence for counting.....	210
Table A-1 Relationships of temporal difference learning (TD Learning) to biology	241
Table A-2 Differences between early evolutionary computing and biology (Holland, 1975).....	243
Table D-1 States in an Argos Full Subtractor Scheme	284
Table D-2 Synchronizations (transitions) in an Argos full subtractor scheme	285
Table D-3 Outputs of an Argos full subtractor scheme.	285
Table D-4 States in an Argos decoder scheme. Each state is a Prediction scheme.....	287
Table D-5 Decoder – Synchronizations (Transitions).....	288
Table D-6 Outputs of an Argos decoder scheme	288
Table D-7 States in an Argos full-adder scheme.....	292
Table D-8 Synchronizations (Transitions) in an Argos full Adder scheme	293
Table D-9 Outputs of an Argos full-adder scheme.	293
Table D-10 Steps in the UML sequence for planning.....	300
Table D-11 Sets of Actions to represent 1 bit.....	300

List of Figures

Figure 1–1 Piaget’s bead problem which exhibits stage like variation – young children fail to detect the 1:1 correspondence, whereas older children recognize that they are the same and achieve conservation of number	4
Figure 2–1 Piaget’s bead problem resolved through reflective abstraction: Moving from processing points and lines to processing the action itself.	27
Figure 2–2 An interpretation of the processes of assimilation and accommodation showing how operative schemes are created through accommodation and figurative schemes through assimilation using an external reward provided by the environment. The reward mechanism is based on the actor–critic model (Sutton and Barto, 1988).....	34
Figure 2–3 A depiction of Piaget’s knowing circle. There is a constant dialectic interchange with the environment, first accommodation then assimilation. These processes deepen over time, as the child progresses through developmental stages and matures.....	35
Figure 2–4 A model of Piaget’s innate scheme and primary reactions, which are the abstract processes of assimilation and accommodation separated into act and sense, reinforce and predictive model (Furth, 1969, p162).....	37
Figure 2–5 A model of Piaget’s developmental trend that shows the impact of increasing reflective abstraction and progressive interiorization on the abstract processes of assimilation and accommodation (Furth, 1969, p162 and 154).	41
Figure 2–6 A depiction of the impact of Piaget’s evolutionary trend and developmental trend (Furth, 1969, Piaget 1983) as it affects the learning process to develop more complex schemes, and so resolve LP6. .	43
Figure 2–7 A simple depiction of Piaget’s number example (Furth, 1969, p104).....	45
Figure 2–8 In WE1 – Base Level, there is emergence of the capability to act and sense in a number line world. The simulation learns to move, change its penState and use its innate schemes and primary reactions to adapt to the continuous number line environment.	47
Figure 2–9 In WE2 – Constrained Level, there is emergence of “Stop” action. This action, not inherent in the innate schemes and primary reactions, is perceived as repeated constrained movement in a number line world. This constrained movement is the precursor for unitized movements.....	48
Figure 2–10 In WE3 – Differentiated Level – Object, there is observation of the emergence of differentiated objects as sets of coordinated movements in the number line world. The repeated observation by the simulation is necessary in order to provide the capability to derive sets of values.	49
Figure 2–11 In WE4 – Hierarchical Level, there is emergence of hierarchies. This hierarchy supports the necessary relationships of a value system and it is with this internally constructed value system that “units” emerge. The appearance of, and use of “connected units” in the value system, is the precursor for ordering, seriation and of equivalence.	50
Figure 2–12 In WE5 – General Level, there is observation of more complex behavior by the simulation including the construction of hierarchical relationships with earlier constructed schemes. In this case, the	

“units” developed earlier. It is with the repeated observation of these relations, that the simulation has emergence of concepts such as equivalence, “less than” and “greater than”. 50

Figure 2–13 Conceptual blending of points on a line, requires a material anchor (the line) shown as enclosing box to blend the input spaces of I_1 and I_2 using the gestalt principle of linearity and the conceptual structure notion of sequential order to construct the queue of points such that they can be counted. In Hutchins’ interpretation, the external environment is a material anchor for the points themselves (Hutchins, 2005, p1561). 69

Figure 2–14 A depiction of Searle’s Chinese room argument (Searle, 1980, p417) 79

Figure 3–1 The relationship of the research hypotheses..... 103

Figure 3–2 An outline of the design and runtime components of the research..... 104

Figure 3–3 This diagram summarizes the anticipated emergent structure that will be observed in the executing system. The artificial neural network implementation exhibits regularities in the environment (the permanent object invariant) and coordinated movement. the dialectic system identifies regularizes using the permanent object invariants identified by the artificial neural network implementation and exhibits other, more complex structures such as coordinated movement, internalized FSA structures and interactions as HFSA. Each of the structures in the *dialectic system* is imagined to occur in levels (stages of development) as the system matures. 106

Figure 4–1 An entity relationship diagram of the EBNF grammar of the system, which defines the relationships in the number line world. 115

Figure 4–2 The ERD of actions (through accommodation) that the simulated student can take to process their number line world. The student makes random choices and the environment responds. Then resultant observation and rewards are passed back to the student through the process of assimilation..... 116

Figure 4–3 An image of the user interface provides the simulated student access to its number line world using a worksheet metaphor. 121

Figure 4–4 A depiction of the number line world that the simulated student receives observations from and performs actions on, using the information from the ERD. 121

Figure 4–5 The abstract processes of act and sense, is a generalization of the processes of assimilation and accommodation and is used to produce the Piagetian processes of perception, recognition. This is the mechanism by which system automatically responds to and acts on the environment. 123

Figure 4–6 A depiction of reinforce, which is the mechanism through imitation and reconstruction which generates (learns) new schemes that can predict environmental information (internal/external) and act differently..... 123

Figure 4–7 A depiction of the Piagetian processes of mental image and evocation that enables the use of an internally constructed model of the world from existing schemes. It is used for planning and play. 124

Figure 4–8 A depiction of the simulated student as a state machine. This is seen as a critical piece of the architecture and design, since it suggests that the student operates using the same processes of assimilation and accommodation. 125

Figure 4–9 An example execution that retrieves populations of chromosomes from external storage. This allows the agent to be reactivated and restarted from known positions.	127
Figure 4–10 A simple depiction of a chromosome with transitions	128
Figure 4–11 Structure of a chromosome	128
Figure 4–12 The entity relationship diagram for chromosome. it contains the rules by schemes are mutated. All the details are included in the h[] structure in the chromosome. It supports both FSA and HFSA (Prediction, Argos and Piagetian schemes).....	129
Figure 4–13 The structure of a simple mealy machine, The base for all other FSA.....	130
Figure 4–14 The evolution and evaluation of a class (Evclass 1.) of mealy machines, in this example a simple buffer machine from the simple mealy machine.	131
Figure 4–15 Evaluation function for BufferNot class.....	132
Figure 4–16 Prediction quality function for BufferNot class.	132
Figure 4–17 The definition of a class of machines (EVClass 1.), in this case a BufferMachine.	133
Figure 4–18 A simple depiction of chromosome (FSAs) used in the random construction of more complex as classes, such as Argos full adder schemes.	134
Figure 4–19 The depiction of an Argos Equivalence scheme using initial, final, input, output and ordinary states. This 4 th example shows that the process of randomly selection “States” of machines, works.	135
Figure 4–20 Piagetian Equivalence Scheme. In this image, the twin processes of assimilation and accommodation are clearly seen as separate action. This process follows Piaget’s knowing circle and makes use of chromosomes as Argos and Piagetian machines to process the inputs from, and act on the environment. The processing is separated into Prolog, Main Thread and Epilog processing.....	138
Figure 4–21 A depiction and composition of a binary equivalence machine that compares the binary values of two machines. From the composition, one can imagine that its simple construction of only and, or and buffer machines, would appear early, which is part explains Piaget’s observation of seriation and order, before more complex structures such as counting.	140
Figure 4–22 Example code of the creation of a class of chromosomes used in class–based evolution.	141
Figure 4–23 The definition of a class of machines (EVClass 1.), in this case a Buffer Machine.....	141
Figure 4–24 The process of adaptation as described using an event process chain diagram, shows the switching from act and sense to reinforce when the inputs from the environment cannot be handled by the sense activity. When the schemes have been constructed to handle the input, the system switches back to using its repertoire of schemes. This two state processing of assimilation and accommodation, is the key to understanding the Piagetian approach used in this research.	143
Figure 4–25 This figure provides an executable trace of algorithm 18 (to imitate through building a chromosome class), and algorithm 19 (to reconstruct, through building schemes that can predict the given input sequence, as finite state automata for the chromosome class). In the example code provided, the system passes the inputs from the environment that it cannot processes, and creates a scheme that can process this information. Initially the chromosome class and then the chromosomes themselves are	

constructed. In the example shown, warning messages from the system show the attempted mutation that have taken place. In the final step, chromosome 300519 is generated with a prediction quality of 1.0, having reached perfection, and a fitness of 1.75.	145
Figure 5–1 A comparative state space representation of a radial basis function neural network with discrete and continuous sensors – used in the detection of regularities in the environment. In this case, RBFs detect the point “1” in the environment.	165
Figure 5–2 A depiction of a state value function using discrete sensors (from test 26), that show that the agent using a linear neural network and reinforcement learning can draw a line and identify points in a number line world and so achieve WE1 – WE3 of the worked example.	166
Figure 5–3 A heap map of a state value function using discrete sensors and continuous sensors (from test 26), that show that the agent using a linear neural network and reinforcement learning can draw a line and identify points in a number line world and so achieve WE1 – WE3 of the worked example.	166
Figure 5–4 Comparison of learning performance – continuous vs. discrete sensors for a verve agent navigating an a number line world the specific test was to identify “3” as 1 + 1 + 1 as a number line.	167
Figure 5–5 A comparatives graph of reinforcement learning performance using reward sum for learning only, planning and with curiosity. This shows that it is possible to learn over a reward gradient, with curiosity improving performance. In this example, the aim is to draw 1 + 2.	168
Figure 5–6 A comparative analysis of mean squared error (MSE) using a predictive model to detect objects in the environment with discrete and continuous sensors. It shows that planning with continuous sensors provides benefits over other models. In this test, the aim was to receive a reward after drawing 1 + 1 then 2.	169
Figure 5–7 A comparative model of the number of steps to reach the maximum reward, showing the improvement when curiosity is embedded into the reward gradient. Thus an agent which explores achieves greater reward. These processes provide the basis for sensory motor schemes in a Piagetian model. In this test, the aim was to receive a reward after drawing 2 + 2 then 4.	169
Figure 5–8 A plot of the actions of trial run 4 trial 9 test–26 of a agent composed of linear neural networks, as the operate in a number line world receiving a reward gradient. In this example, the red dotted line represents penup and the black line, pendown.	170
Figure 5–9 The evolution and evaluation of a class (Evclass 1.) Of a prediction buffer scheme. The top two graphs describe the fitness and prediction quality (x axis) against population (y axis). In both the distributions of the measures are non–random, as would be expected. The central image describes the input and output values of the prediction scheme. The bottom images provides a representation of the prediction scheme with the starting states in green (state 4) and the 2 final states (states 1 and 5) in blue. The transition values are also shown.	173
Figure 5–10 Commands to evolve a class of schemes, in this case the buffer class as a Prediction scheme. Initially, the population of chromosomes is empty (pop []). After evolution, chromosomes are added and	

mutated to produce machines that meet the requirements of the evaluation function. In this example, chromosome 300604 has a fitness of 1.55 and a prediction quality of 1. 174

Figure 5–11 An example execution of class-based evolution showing the top ranked chromosomes from generation 1..... 175

Figure 5–12 An example of a prediction scheme that correctly detects regularities in the environment, in this case the penState of penUp as action 2. The graphs in red show the value that needs to be detected. The graphs in blue show the values the scheme predicts. The bottom right graph shows the prediction scheme has correctly predicted the input values. Since this is only a single value, the bottom right image is a perfect 1:1 match on input values. 176

Figure 5–13 A depiction of chromosome 1354679 as Class-14 as a prediction scheme, that updates penState. This scheme has a prediction quality of 1 and a fitness value of 1.66. In the image on the left, the start state is shown in green, and the two end states in blue. In image on the right shows the same chromosome with the values of the transitions shown..... 176

Figure 5–14 Evolution of a prediction scheme that moves the simulation into the reinforce-learn process. The switch is provided as a permanent object invariant. In this example, the scheme correctly generates the required structure with a prediction quality of 1 and a fitness of 1.6..... 177

Figure 5–15 This graph provides an example of the mutation rates for Prediction schemes to detection of regularities in the environment. This process is complete when the prediction quality gets to 1. In the examples provided, the maximum number of mutations required is 42 for the penUp action..... 178

Figure 5–16 This diagram shows the number of times a particular prediction scheme is used over a number of executions. It shows that simulation can use its prediction schemes to act in the environment. There is however, no discernible pattern, these are random actions. 179

Figure 5–17 Sensing the environment using learn prediction schemes. This diagram compares the number of prediction schemes created and used against the maximum number of permanent object invariants in the EBNF grammar..... 180

Figure 5–18 WE1–Base tests shows the maximum continuous movements made in six selected runs by the simulation using the prediction schemes that it has created. These are continuous movements because the notion of “number” is not known; they are just actions that have been selected by the simulation..... 181

Figure 5–19 This figure illustrates the WE2–Constrained Tests. It compares the number of instances of constrained movements by the selected run. It illustrates a random pattern of movements..... 182

Figure 5–20 Example chromosome class code used in the evolution of a set of chromosomes to move right 1 unit on a number line. 183

Figure 5–21 The representation of a Prediction scheme, chromosome 327264 in class 5, that can move 1 unit on a number line. The image on the right shows the Prediction schemes with the transition values. The image on the left is the base scheme. 184

Figure 5–22 An alternate example of a class 5 chromosome. This Prediction scheme can move right 1 unit on the number line, as well as sense it. In the image on the left four states are shown, with state 4 being the

start state and state 2 the end state. The same chromosome is shown on the right hand side with named transitions and transition values. 184

Figure 5–23 Chromosome 1510137 is an example of a Prediction scheme that moves left 1 micro unit. The top row images show the input values (red) vs. the prediction values (blue). These values are overlaid in the bottom left image. In the bottom right image a perfect 1:1 match of input values to prediction is shown. 185

Figure 5–24 Depiction of the Prediction scheme that moves 1 micro unit in the number line world..... 186

Figure 5–25 Example mutation of Prediction scheme that enables coordinated action and sensing in the environment. 186

Figure 5–26 WE3–Differentiated Object Tests – showing number of instances of structures appearing in the simulation over time 187

Figure 5–27 A depiction of Piaget’s bead problem (Figure 1–1) as number line segments with pen–up in red and pen–down in black. The accompanying text indicates that a more primitive simulation would determine that the spread–out line is bigger, because the penUp movements are greater. After conservation of number, this is no longer an issue: space has no value. This figure therefore presents a solution to the bead problem through conservation. 188

Figure 5–28 Attempts to construct the bead problem as a mutation process, shows that a simulation can accurately produce a prediction scheme that exhibits the bead problem as “penUp” and “penDown” variants after 802,000 mutations. 189

Figure 5–29 A depiction of an evolved prediction scheme that possesses the properties of a logical OR. Chromosome 626128 has a prediction quality of 1, and the results show that it accurately performs its logical or function. What this diagram shows is that mutation can easily produce these logical components. 191

Figure 5–30 An example of an XOR Prediction scheme that has been mutated from the simple mealy scheme. Chromosome 1559639 has a prediction quality of 1 and a fitness of 1.68, which suggests that it is an efficient processor. 192

Figure 5–31 Evaluation of classes 193

Figure 5–32 Learning rates for Prediction schemes to develop 2 bit propositional logic components..... 193

Figure 5–33 The depiction of an Argos Scheme (chromosome 2000056, class 48) that includes a copy of chromosome 327264 and 300093 that allows the Argos scheme to draw a number line segment. The prediction quality and fitness are set on these schemes to be 0.5..... 195

Figure 5–34 Chromosome 2000061 is an example of an Argos Scheme mutated to internalize a value using Prediction Scheme 306204. When executed, it can pass a binary value to another Argos or Piagetian scheme. This low–level scheme allows for the construction of more complex hierarchies. 196

Figure 5–35 A classic depiction of a complete 4–bit Argos Binary Encoder scheme. In this diagram, the initial, final, input and output states are shown. During mutation, these states are randomly chosen, which

partially explains why now and then the simulation makes mistakes, but over time, corrections can be, and are made.	197
Figure 5–36 A depiction of a binary encoder as Argos schemes that marshal information across prediction schemes to allow them to process the information. The Argos schemes are in red and green. The Prediction schemes are blue and grey. The lines between the points represent the transitions and synchronizations.	201
Figure 5–37 The composition of chromosomes that make use up an Argos binary encoder scheme as chromosome 1009191	202
Figure 5–38 The depiction of a full adder as an Argos scheme as chromosome 2000019, class 45. It includes a set Prediction schemes and Argos schemes. This is only a single full–adder. It can be connected in series to other full–adders using the Argos mutation process.	203
Figure 5–39 An alternate depiction of a full adder as a set of Argos schemes (red node) and prediction schemes (blue and green nodes). The Argos schemes marshal information across their prediction scheme network using a series of numbered transitions in grey. it is an example of a binary executable HFSA machine that is mutated to interact with the environment.	204
Figure 5–40 This is the legend for examples of Piagetian Schemes implementations.	207
Figure 5–41 A worked example of counting using Piagetian schemes showing assimilation, accommodation and the embedding of Argos schemes which enable interaction with the environment. The interesting feature is the alternate processing of accommodation and assimilation and the use of named chromosome – the inclusion of new copies of chromosomes is seen as critical to the development of a simulation that can adapt to the environment through mutation of its internal structure.	208
Figure 5–42 A UML sequence diagram of the Piagetian counting scheme. It shows how disequilibrium and the stabilizing processes of equilibration occur to enable the system to count larger values. This is important because it shows how the system can regain stability using the internally constructed schemes and confirms Piaget’s approach to adaptation.....	210
Figure 5–43 A depiction of a Piagetian prolog scheme with Argos schemes (red nodes) and contained Prediction schemes (blue and green nodes) that form part of the basic process of assimilation and accommodation.	213
Figure 5–44 A depiction of a Piagetian scheme that moves (reads) along a number line, internalizes each of the values and then counts them producing the resultant counted value. The Argos schemes (red nodes) and the Prediction scheme (green and blue nodes) are connected through synchronization and transitions to enable adaptation with the environment.	214
Figure 5–45 This image depicts the contents of the Piagetian scheme for counting, showing how the different Argos and Prediction schemes are connected together using synchronization and transitions as described in the architecture and design.....	215
Figure 5–46 Measures of complexity of instances of scheme classes by type of scheme. Here the complexity is a simple measure of machine (scheme) size.	216

Figure 5–47 Comparison of processing in Macaque (left Side) and Piagetian schemes (Right Side), (Modha and Singh, 2010, p13485).	221
Figure B–1 A depiction of the SOAR Architecture (Laird, 2008, p225–228).	252
Figure B–2 The SOAR decision cycle – SOAR v9 2008 (Laird, 2008, p227)	253
Figure C–1 A UML sequence diagram that shows the Piagetian mathematical relations of substitution and subdivision being used in a less than machine to determine if two counted values are different. It shows the interaction of the executive with the process of act and sense.	270
Figure C–2 A worked example of a Piagetian “less than” scheme that includes a Piagetian equivalence scheme. It shows how reversible machines are recombined into hierarchies. In this example, “less than” becomes an operator.	274
Figure C–3 A UML sequence diagram showing an alternate view of a Piagetian “less than” scheme, which uses more sophisticated subdivision and substitution.	275
Figure C–4 WE6 – Rational – Level – the sixth action to appear in a number line world, where there are sets of repeated relations. This designates that the processing by the student is making more complex arrangements on their number line world. Though listed, the rational feature will not be evaluated in the research and is included only for completeness.	280
Figure D–1 A depiction of an Argos Scheme that exhibits coordinated movement on number lines. In this depiction, the movements on the number line are random, there is no coordinated control.	283
Figure D–2 Example of a mutated Argos Scheme to start number line using a prediction scheme that is used to tell the environment that it wants to build a new number line. This has the effect of initializing the worksheet.	283
Figure D–3 The classic depiction of an Argos full subtractor scheme using randomly selected initial, final, input, output and ordinary states.	284
Figure D–4 The depiction of a ripple carry subtractor as a collection of full–subtractors as an Argos scheme.	286
Figure D–5 An Argos–Scheme that implements a full subtractor.	287
Figure D–6 A Depiction of an Argos decoder scheme using initial, final, input, output and ordinary states...	287
Figure D–7 Depiction of an Argos decoder scheme with Argos schemes in red and Prediction schemes in green and blue. The transitions and synchronizations are shown in grey and green lines with the appropriate transition values.	289
Figure D–8 A depiction of the composition of Argos scheme as a counter showing the graph of initial, input, output and final states. These states are all randomly chosen as the Argos scheme is mutated.	290
Figure D–9 A depiction of an Argos scheme counter scheme which includes full adders, decoders and internalized actions.	291
Figure D–10 A depiction of a full adder as an Argos scheme showing that common schemes can be randomly constructed using mutations of their initial, final, input and output states.	292

Figure D-11 A depiction of an Argos scheme as a ripple carry adder. Showing the inputs, outputs, initial and final states. This is only one example, others exist. 294

Figure D-12 The depiction of a full adder as a Argos hierarchical finite state machine otherwise called an Argos scheme. 295

Figure D-13 The depiction of the structure of an Argos scheme as a full-adder. In execution, these synchronizations are chosen randomly and evaluated using external rewards by the simulation. They are depicted here in a similar fashion to FSA. 296

Figure D-14 The depiction of a full adder as a set of Argos schemes (red node) that marshal information using their parallel synchronization tables across prediction schemes (blue and green nodes) that process the information using a series of numbered transitions in grey. It is an example of a binary executable machine that is evolved to interact with the environment. 297

Figure D-15 A UML sequence diagram that shows the use of a predictive model to overcome disequilibrium in a worked example of counting. 299

Keywords

Concept Development, Constructivism, Emergence, Evolutionary Bootstrapping, Number-Sense, Learning Paradox, Machine Learning, Modeling Cognitive Processes, Piaget, Knowing Circle, Simulation, HFSA, FSA, Evolutionary Computation, Cognitive Development

Chapter 1

1. Introduction

“To put it most simply, the paradox is that if one tries to account for learning by means of mental actions carried out by the learner, then it is necessary to attribute to the learner a prior structure that is as advanced or complex as the one to be acquired” (Bereiter, 1985, p202).

Pascual-Leone defines Fodor’s problem as the learning paradox (Pascual-Leone, 1976, 1980 cited by Bereiter, 1985, p202). This is a meta-theoretical problem defined by a meta-theoretical question: “How can a structure generate another structure more complex than itself?” Piaget’s position on the learning paradox is that new concepts can be learned through the normal learning process without requiring that additional, but unused capabilities, exist (Fodor, 1980, p142).

The earliest example of the learning paradox is attributed to Plato’s Meno dialogue (Plato, 1985). It is known as the “Paradox of Inquiry” (White, 1976, Matthews, 1999), or simply “Meno’s paradox” (Moline, 1969; Benson, 1990). The thesis of the dialogue is restated by Socrates: “A man cannot inquire either about what he knows or about what he does not know – for he cannot inquire about what he knows (ed. deductively), because he knows it and in that case is in no need of inquiry; nor again can he inquire about what he does not know, since he does not know about what he is to inquire” (Plato, 1985). Thus, inquiry is impossible and by extension, learning is impossible: one can never acquire new knowledge. Yet, people know that they learn, hence the paradox.

In the common usage of everyday discourse, a paradox is a judgment or an opinion, which is contrary to the general opinion or common sense. The learning paradox on this basis would be an obviously anomalous contention that someone seriously propounds, despite its conflict with what is generally regarded, as being true (Rescher, 2001, p6).

Among philosophers and logicians, however, the term has come to acquire a more specific sense, with a paradox arising when plausible¹ premises entail a conclusion whose negation is also plausible. Thus, there is a paradox when a set of individually plausible theses $\{P_1...P_n\}$ validly entails a conclusion C whose negation $\neg C$, is plausible. This means that the set $\{P_1, P_2...P_n\}, \neg C$ is such that all of its members are individually

¹ Plausible, meaning presumably true. What this means is that “we will endorse and employ the proposition (which may actually be false) insofar as we can do so without encountering problems, but are prepared to abandon it should problems arise” (Rescher, 2001, p16).

Chapter 1. Introduction

plausible while nonetheless logically inconsistent overall. Rescher makes the statement that “A paradox arises when a set of individually plausible propositions is collectively inconsistent and the inconsistency at issue here must be real rather than merely seeming” (Rescher, 2001). Paradox is thus the product not of a mistake in reasoning but a defect of a substance: a dissonance of endorsements. A variant way of looking at a paradox is therefore as a conflict–engendering argument, a piece of deductive reasoning to a contradictory result (Rescher, 2001, p6). When graded in this way the learning paradox, is a paradox, since it instills the interest of the inquirer and its deeper meaning is the root of knowledge.

The central question is, “How is complex knowledge constructed by the learner?” Learning obviously takes place, such as when students develop rational number–sense after developing whole number–sense (Bereiter, 1985, p202). Bereiter suggests that the process of learning is far more complex than is currently realized and that educators have overlooked important factors in the promotion of learning. If it is accepted that individuals possess and use an existing higher–level schema, then a large amount of everyday learning occurs by avoiding the learning paradox (Bereiter, 1985, p202). Understanding how the learning paradox can be resolved will improve an educator’s capability to support more effective concept formation in their students (Bereiter, 1985, p202).

Attempts have been made to resolve Meno’s paradox: rejected as verbal trickery based on the equivocation of the two senses of *manthanein* (to learn in any way) (Nehamas, 1985, p3); establishing a broader semantic context to knowledge (Day, 1994); disambiguation of knowledge and belief (Fodor, 1980) or even dismissing it (Moline, 1969, p154). Socrates’ provides a solution by using intuitive induction, based on the “sense–perception” of universals. As Hoffmann points out, intuitive induction faces the same problem as the learning paradox (Hoffmann, 2003). Perceptions are theory laden and based on preconceived representations, which is also related to the problem of possessing knowledge (as described in the Socratic example of geometry) and of having knowledge and using it to solve a given problem (Hoffmann, 2003). The theory of recollection is an original and inventive way of accounting for the understanding of mathematical kinds of truth. It has however been a great source of inquiry. Researchers have viewed different aspects of the learning paradox and applied it to the observation of emergence of new forms (novelty), both behavioral and cognitive on the ontogenetic timescale². Researchers principally focus on an individual’s acquisition of language; however, it equally applies to the development of mathematics (Hoffmann, 2003, p123).

We could argue that the learning paradox is Kant’s critique of pure reason in disguise – whereas empiricism asserts that all knowledge comes through experience, rationalism maintains that reason and innate ideas exist first (Kant, 1999). Kant argued that experience is purely subjective, without first being processed by, pure reason. That using reason without applying it to experience will only lead to theoretical illusions, hence the paradox.

² This notion of emergence defines a constraint under which any simulation that attempts to resolve the *learning paradox* would need to work under. We define this constraint as LP2: Exhibits Emergence of Hierarchical Concepts using Evolutionary Process.

Chapter 1. Introduction

There may be an alternate way of viewing the learning paradox. Consider the dialogue in “Through the Looking Glass” by Lewis Carroll, specifically the discussion between Alice and the Red Queen, “Can you do Division? Divide a loaf by a knife, what’s the answer to that?” (Carroll, 2004, p255). The quote from Lewis Carroll hints at a paradox, which if there is sufficient interest by the reader, will lead them on to stop and ponder the conflict, “A loaf by a knife.” The researcher believes this perplexing situation, in which conflict occurs, is central to self-awareness. If self-awareness of learning and knowing is considered the ultimate human accomplishment then rather than a meta-cognitive approach³, this research paper considers that it is instead reflexivity. Reflexivity occurs “where we experience or are shown a situation where our existing beliefs are inadequate [so] our awareness of our own state of knowing is enhanced,” which is central to growth through education (Duffy and Cunningham, 1996, p181). It is believed that this process of introspection is critical to the emergence of structure that enables us to resolve conflicts in the external world as well as resolve the learning paradox.

The specific answer to the Red Queen’s question was the response, “Bread and butter of course” (Carroll, 2004, p255). Yet this is a difficult linguistic topic because of its metaphorical reasoning, and it is believed that it is far beyond the scope of a single Ph.D. research topic. A simpler problem, can be found in childhood mathematics.

The study of mathematics and specifically fractions is foundational in mathematics, yet it is among the most difficult topics for school students (Cramer, Behr, Post and Lesh, 1997a and 1997b). Students have difficulty recognizing when two fractions are equal, putting fractions in order by size and understanding that the symbol for a fraction represents a single number. Students also rarely have the opportunity to understand fractions before they are asked to perform operations on them such as addition or subtraction. The development of fraction-sense from whole number-sense can be considered a conceptual change and such changes lie at the core of all learning (Bereiter, 1985, p202).

Conceptual development clearly does take place, but some non-Piagetian theories have difficulties in accounting for this (Bereiter, 1985). These theories¹ are forced to assume that the learner already possesses the high-level concepts, or an equivalent higher-level capability; hence the learning paradox. It is believed that fraction-sense⁴ is too complex a problem for a research project and which requires a much simpler problem.

Counting is one of the first number ideas taught to children, but the numbers they recite often have little meaning for them (Copeland, 1974, p82; Chilampikunnel, 2010, p147). Children learn many things about the physical world in which they live before they develop the abstract idea of number, and this learning is incremental⁵, yet has observable stages (Furth, 1969; Copeland, 1974). Number, as a classification (based on

³ A meta-cognitive approach stresses strategies for efficient processing.

⁴ Fraction-sense is equated with rational number-sense.

⁵ The notion of incremental learning is seen as a critical feature of any system that attempts, even partially, to resolve the learning paradox, hence the definition LP5: Learns Incrementally with Minimal Innate Knowledge and Reflexes.

Chapter 1. Introduction

properties shared in common by a given set of objects) is not a physical object such as the color of the object. For instance, does the child who can count by matching number names to objects “being counted” understand the real meaning of number? One way to test this is to spread them out and give them more space. This is illustrated below (Piaget, 1952, p36 and Copeland, 1974, p83):

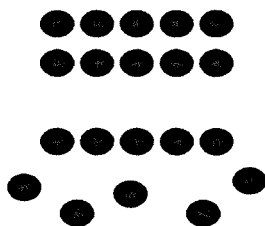


Figure 1–1 Piaget’s bead problem which exhibits stage like variation – young children fail to detect the 1:1 correspondence, whereas older children recognize that they are the same and achieve conservation of number

A youngster observes two sets of beads. He agrees that they are the same (in number). If one set were spread out, would the child still say they are the same? Most children of five or six years of age respond that there are more in the spread out set⁶. The spatial configuration of, or space occupied by the beads is to the child, the number idea. The spread out beads occupy more space, so there must be more beads (Copeland, 1974, p83). The same child at seven or eight hardly pays any attention to the problem and returns the correct answer. This difference is the stage like variation that Piaget consistently refers to in his research (Piaget, 1952, p36). Some researchers believe that the number space problem is related to estimation skills with research suggesting that children ages five to six, frequently estimate numbers using a logarithmic scale, rather than a linear scale (Booth and Siegler, 2006, p189). The cognitive shift from logarithmic to linear scales at ages five to seven which results in better measurement, judgment, numerosities and computational estimation skills, may account for this discrepancy (Booth and Siegler, 2006, p189). What is interesting in the bead problem is that it is a reversible transformation, which the student resolves through maturation. An examination of the development of number–sense, including the bead–problem, is the scope of this research.

Research from cognitive science, machine learning and conceptual development all have different explanations for various forms of concept formation, and these will need to be reviewed to determine how they individually resolve the learning paradox, and specifically if they can resolve the bead problem devised by Piaget through maturation in their systems.

Other researchers have considered concept formation, including conceptual blending (Fauconnier and Turner, 2002), metaphors (Lakoff and Núñez, 2001; Lakoff and Johnson, 1999), but these may not have considered the problems of the learning paradox.

⁶ Piaget’s bead problem is an example of conservation of number. It is used to define a constraint, LP4: Mirrors Real World Behavior of Children Learning Number-sense using a number line. It is also used in conjunction with the observation of children See § 2.5.

Chapter 1. Introduction

Bereiter suggests that when educators overlook the learning paradox, they simplify learning and miss important factors in the promotion of learning (Bereiter, 1985). The implication of Bereiter's analysis is that understanding how the learning paradox can be resolved will improve the educator's ability to support more effective concept formation in their students (Bereiter, 1985).

1.1 Motivation

1.1.1 Research Purpose

The purpose of this research is to study the learning paradox as defined by Bereiter (Bereiter, 1985, p202) and Fodor (Fodor, 1980) by applying principles from Cognitive Modeling, Computer Science and Information Systems to the "real-world" development of number-sense in children. In doing so, a set of objective criteria are determined which are used to grade any potential (partial) resolutions to the paradox from the research literature.

1.1.2 Research Questions

Since Piaget suggests that resolution of the learning paradox occurs naturally in individuals as they mature, this produces a series of questions:

- 1) How have researchers tackled the epistemological issues associated with the learning paradox?
- 2) Since biology and evolution have already resolved the learning paradox, what can be learned by studying these disciplines?
- 3) How does Piaget's model explain the process of maturation of an individual i.e., does his theory sidestep the main issues and can it be modeled in a simulation?
- 4) If one were to construct an example problem based on the observed behavior of children as they develop number-sense, then how would other researchers view this example problem?
- 5) Have other researchers in automated theorem formation (ATF) and automated theorem proving (ATP) in pure mathematics considered the learning paradox?
- 6) Since much of human reasoning is metaphorical, how have researchers in the concept formation discipline tackled the learning paradox?
- 7) How has the field of cognitive development, which uses various forms of machine learning (symbolic, emergent, hybrid), tackled the problem of the learning paradox?
- 8) Have prior implementations of Piagetian models resolved the learning paradox as specified by Piaget's model?
- 9) How have other machine learning frameworks, those not specific to cognitive development models (Duch, Oentaryo and Pasquier, 2008, p123), resolved the learning paradox?

1.1.3 Research Hypotheses and Research Problems

The hypotheses of this research are that (i) a Piagetian model of human cognition exists from which conceptual structures can emerge and resolve the learning paradox, (ii) a biologically plausible model of mathematical concept formation exists, and (iii) it is possible to emulate concept formation in a simulation by using a cognitive learning framework provided by Piaget. Hypothesis (i) assumes that Piaget's theories of learning and development (Piaget, 1964, p8), is far closer to a biologically plausible model than is currently suspected. To support this notion, it will be necessary to identify novel areas of Piaget's work (Furth, 1969 and Copeland, 1974) that were missed by earlier researchers. With this, the research problem can be defined as "Determine the conditions under which it is possible to exhibit emergence in a simulation and so overcome the learning paradox (albeit partially), within a constrained number line world using a biologically plausible model."

1.1.4 Research Validity

A Piagetian model of self-learning and development is used to establish a solution that handles novel situations by developing a digital network of circuits (propositional logic components as finite state automata (FSA) where information is marshaled across this network using hierarchical finite state automata (HFSA). This is important for three reasons. First, current implementations of cognitive development models typically follow the dual-mind hypothesis⁷ (Paivio, 1986; Paivio 2006; Anderson, 2005; Lieberman, Gaunt, Gilbert, Trope, 2002; Pyysiäinen, 2003, p617) and insert already formed symbolic language (propositional or predicate based) on top of sub-symbolic forms. In doing so they skirt around the symbol grounding problem. This thesis takes the genetic epistemological view and affirms that for a machine to approach resolution of the learning paradox on its own, it must form its own neural concepts. Second, by relating the learning paradox to the development of number-sense in childhood mathematics, the educational validity of this research is established.

1.2 Aims of the Project

In this research, the aims are to:

- 1) *evaluate epistemological solutions to the learning paradox*, by interpreting research from the perspective of various researchers (Plato, Chomsky, Fodor, Piaget, Bereiter and Pascual-Leone) and compare and contrast a solution to the learning paradox as a form of emergence in Piaget's theory of genetic epistemology;

⁷ The dual mind hypothesis, often referred to as dual coding theory, asserts that visual information processed by the brain are analogue codes with verbal information being symbolic (Paivio 1986; Paivio, 2006). These already formed symbols, *allow* cognitive modeling researchers, to incorporate symbolic information into their systems in the form of propositional and predicate logics (Anderson, 2005).

Chapter 1. Introduction

- 2) *evaluate sources of emergence*, by reviewing models of neurophysiology and biological evolution to determine how emergence can be simulated in a computer system;
- 3) *provide a computational reading of Piaget's theory*, by re-interpreting his work from the perspective of neurophysiology, biological evolution and the epistemological constraints imposed by the learning paradox and in doing so, produce a *worked example* and a mechanism to grade models of emergence;
- 4) *clarify emergence in a real-world worked example using a number-line*, by reviewing observations of the development of number-sense (as a form of mathematical conceptual learning) in children who are observed developing number-sense using a number line;
- 5) *compare and contrast different notions of mathematical concept formation*, by reviewing human, machine learning, cognitive development and Piagetian models to understand how these support emergence to resolve the learning paradox;
- 6) *provide arguments to support a separation into an artificial neural network implementation and a dialectical system*;
- 7) *evaluate the artificial neural network implementation*, by testing the hypothesis that it is possible for a neural architecture to implement a model of Piaget based on *Drosophila* that exhibits emergence; and
- 8) *evaluate the dialectical system*, in particular, by testing the hypothesis that it is possible to develop number-sense using Piaget's developmental model implemented as HFSA, and so partially overcome the learning paradox.

1.3 Contributions

This is a systematic analysis of Piaget's work that applies recent knowledge of neurophysiology, biology, evolution and machine learning to his theories. Since the works of Piaget span many decades, are, at times confusing, often misquoted and certainly misrepresented, this research confines itself to two theoretical reviews (Furth, 1969 and Copeland, 1974). In doing so, it was realized that there is a biological basis for Piaget's genetic epistemology and that a simple "generative model" of act/sense, learn and plan from *Drosophila* (Miesenböck, 2008, p52; Shang, Claridge-Chang, Sjulson, Pypaert and Miesenböck, 2007, p601) can be mapped to Piaget's genetic epistemology (Furth, 1969).

The second contribution is the development of algorithms for class-based evolution, which enable the evolution of FSA and HFSA, provides evidence of resolution of Piaget's bead problem and of conservation of number. To do this, this thesis extended a computational model of evolution (Jacob, 2001) and binary FSA (Levy, 2002).

Last, the contribution of a definition as a set of constraints (LP1 – LP7) and a worked example (WE1 – WE5), that can be used to grade potential solutions to the learning paradox is made.

1.4 Structure of the Thesis

In chapter 2 is provided a review of the literature. In § 2.1, is introduced the epistemological issues of the learning paradox by describing theories of concept formation by Chomsky, Fodor and Piaget. This provides the

Chapter 1. Introduction

motivation to model Piaget's methods and places some constraints⁸ on a solution. The next step is to describe biological evolution as a basis for emergence of structure that enables children, naturally, to overcome the learning paradox, in § 2.2. Neurophysiology of *Drosophila* (fruit flies) is outlined in which it is suggested that reinforcement learning provides a neuronal construction mechanism, which can be implemented in an artificial neural network, to support emergence of concept primitives. Also identified are hierarchical finite state automata as a simulation mechanism for a *dialectic system*. In § 2.3, a review of the roots of Piaget's genetic epistemology is discussed with a re-interpretation of Piaget's work from Furth (Furth, 1969) and Copeland (Copeland, 1974). This interpretation is then mapped to the neurophysiology of *Drosophila* (fruit flies). Further developed is a set of features required by any simulation, which attempts to acquire number-sense using a Piagetian model along with a *worked example*⁹ against which this solution will be tested. In §2.4, the emergence of number-sense is reviewed in terms of the cultural effects of the number line as well as current theoretical approaches in the development of mathematical ideas. In doing so is confirmed the validity of the worked example. In § 2.5 the development of mathematical concepts in automated theorem formation and proving systems is compared and contrasted to the development of number-sense in children. In § 2.6 is reviewed the underlying basis of concept formation in metaphors and conceptual blending theories; Fauconnier and Turner's view that emergence must be based on evolution. In § 2.7 cognitive development models are reviewed which shows that emergent models provide a suitable basis for resolving the learning paradox, yet they are not capable of resolving the constraints based on the theoretical works of Piaget. Prior Piagetian implementations are reviewed and contrasted to the re-evaluated Piagetian theory, in § 2.8. Other frameworks that could resolve the learning paradox are surveyed, and it is concluded that some, such as visual perception may provide useful future features to extend this research, in § 2.9. The key points of the literature review are summarized and conclusions are reached by discussing the need for a hierarchical neural network implementation of the Piagetian model, to resolve fully the learning paradox. In addition, a solution is provided as an *artificial neural network implementation* and a *dialectic system*, to partially resolve the learning paradox, in § 2.10

In chapters 3 and 4 the research methods utilized are defined along with a description of the architecture and the design of the *artificial neural network implementation* and the *dialectic system*.

In chapter 5, an evaluation of a biologically inspired *artificial neural network implementation* of a Piagetian / *Drosophila* model using reinforcement learning and the *dialectical system* which implements a Piagetian model of development, is provided.

Chapter 6 further describes ways to develop and apply this implementation model and chapter 7 concludes this study.

Appendix A contains more details on biological evolution as a basis for emergence. Appendix B reviews more issues with cognitive development models and their capability to resolve the learning paradox. Appendix

⁸ Those constraints are identified as LP1 - LP7.

⁹ The worked example is described as WE1 - WE5.

Chapter 1. Introduction

C discusses Piagetian approaches to mathematics. Appendix D contains further examples of the evaluation of the *dialectic system*.

1.5 Summary

By observing the development of children, Piaget penned the theory of operative meaning¹⁰ (Furth, 1969 and Copeland, 1974) which provides a model of learning and development, which, he believed, resolved the learning paradox. While many current researchers are aware of his work, none has provided a thorough analysis of his work from the perspective of neurophysiology, biology, evolutionary processes and machine learning. This research holds (i) that a Piagetian model of human cognition exists from which there is emergence of structure and so resolve the learning paradox, (ii) that a biologically plausible mechanism of mathematical concept formation exists and (iii) that it can be implemented in a simulation, where number-sense is an emergent property of the executing system. To evaluate these hypotheses two systems are developed: an *artificial neural network implementation* and a *dialectic system*. It is shown that the *artificial neural network implementation* demonstrates emergence of structure (the permanent object invariant, as described by Piaget), but cannot develop a Piagetian form of number-sense. It is further shown that the *dialectic system* can implement a fuller Piagetian model of development that reveals emergence of number-sense. In doing so, the state of the art of models of emergence and Piagetian models in particular are improved upon.

¹⁰ We refer to Piaget's theories collectively as "genetic epistemology" (Piaget, 1970a) using the two main reference sources of Furth (Furth, 1969) and Copeland (Copeland, 1974).

Chapter 2

2. Literature Review

This chapter begins with an introduction to some of the epistemological issues associated with the learning paradox, from the debate between Chomsky, Fodor and Piaget. A set of constraints are developed that need to be met by any simulation that attempts to resolve the learning paradox in § 2.1.8. Applying the assumption that evolution has already resolved the learning paradox, § 2.2 summarizes some key aspects of biology and evolution, and explains how these can be simulated, without contradicting the identified constraints. A re-evaluation of Piaget's model of cognitive development proceeds using work from Furth, Copeland, Pascual-Leone and Piaget in § 2.3. With this foundation in place, a description of the relationships of Piaget and Mathematics is provided along with a *worked example* that is used in an evaluation of models of cognition. The research approach is then reviewed and confirmed using a number line world to test for emergence of number-sense with research from neuroscience, the cultural impacts of MNL and educational perspectives, in § 2.4. In § 2.5 automated theorem formation and proving systems in pure mathematics are reviewed and their philosophical basis is discussed along with the development of mathematics through simulated creativity to determine how these could support the constraints imposed by the learning paradox and the *worked example*. In § 2.6, is reviewed the underlying basis of emergence in metaphors, mental spaces and conceptual blending theories (Fauconnier and Turner, 2002). Examples of number line processing using conceptual blending are presented which help identify issues with the approach and the need by the authors (Fauconnier and Turner, 2002) for emergence of structure to enable the construction of blends. Cognitive development models in § 2.7 are reviewed to determine if any provide a suitable basis for resolving the learning paradox. This permits the identification of several issues that need to be addressed by present research. In §2.8, existing Piagetian implementations (Pascual-Leone, 1970; Drescher, 2002 and Chaput, 2004), are reviewed and contrasted to the reevaluated Piagetian theory defined in §2.3, and the learning paradox. In §2.9, other frameworks of conceptual development are reviewed to determine if they can resolve the learning paradox, work within the constraints defined, and exhibit emergence as defined by Piaget. The surveyed frameworks include case based reasoning, explanation based learning, unsupervised learning in visual perception and emotional / intuitive models. A summary of the literature review is provided in §2.10.

2.1 Epistemological Issues

This section introduces some of the epistemological issues associated with theories of concept formation (Chomsky, Fodor, Piaget Bereiter and Pascual–Leone), which have been proposed to resolve the learning paradox. In this analysis, a series of constraints (LP1 – LP7) are devised which any solution would need to have resolved. The debate on the learning paradox is outlined in § 2.1.1. A discussion follows the positions of researchers in the debate, Chomsky in § 2.1.2., Bereiter in § 2.1.3, Fodor in § 2.1.4, Piaget in § 2.1.5 and Pascual–Leone in § 2.1.6. The constraints of a simulation are summarized in § 2.1.7 and a set of conclusions is described in § 2.1.8.

2.1.1 The Debate on the Learning Paradox

“No knowledge is possible unless concepts and hypotheses are already in the mind before anything is observed at all” (Piattelli–Palmarini, 1980, p258).

The first major modern debate concerning the learning paradox occurred in October 1975 at the Abbaye de Royaumont (Piattelli–Palmarini, 1980). Two major issues dominated the debate between Noam Chomsky (the originator of generative linguistics), Jean Piaget (the originator of genetic epistemology) and Jerry Fodor (the originator of language of thought):

- 1) The origin of the fixed nucleus; and
- 2) The specificity of the fixed nucleus.

The fixed nucleus is the innate capacity or innate structures required to produce grammatical utterance; whereas the fixed nucleus contains representational content primitives. The specificity of the fixed nucleus are mental skills, especially linguistic ones. The protagonists debated from different positions, with Chomsky and Fodor as nativists opposing Piaget’s genetic epistemology on various grounds.

A summary of the positions of each of the participants is provided in the following sections.

2.1.2 Chomsky

For Chomsky, a genetically predetermined generative grammar is a possible explanation for observations of the quick acquisition of language by children (Chomsky, 1980, p187). The direct origin of the fixed nucleus is a mental organ which is unaffected by cognition, general intelligence or sensory motor activity (Chomsky, 1980, p188). The argument that language is “underdetermined by experience” and the problem of mapping experience onto state transitions¹¹ forced Chomsky to take a nativistic stance¹² to explain the specificity of the fixed nucleus. In later works on binding theory, Chomsky continues to defend this position inferring that certain principles are “built into” the initial state of the human language systems (Chomsky, 2000, p50).

¹¹ This nativistic stance can be explained in terms of the brain creating an endless variety of grammatically correct sentences from a limited vocabulary and being able to map new sentences within this grammar.

¹² The nativistic stance is that language, specifically grammars, must be innate.

Chapter 2. Literature Review

Chomsky's aim was to decompose a particular behavioral¹³ whole into behavioral primitives, which either are no longer determined by experience, or have no internal structure of their own i.e., they can no longer be divided into behavioral sub-systems. Therefore, this logically ensures that these concepts, knowledge or behaviors must be preexisting in evolution, because they cannot depend on ontogenetic learning, nor on making input representations on primitives. Chomsky takes the innate structures to contain representational content and by applying computational theory to cognition, allows them to be decomposed into individual parts as rules and representations. Yet accounting for language from a selection viewpoint offers no more depth on the problem. The updated Chomsky hypothesis suggests that humans have a parameter setting device which sets the logical and syntactic forms of a given language based on sample sentences the child encounters (Chomsky 1986, p146). However, this does not furnish a resolution to the learning paradox.

2.1.2.1 Chomsky: Where is the problem?

Piaget considers Chomsky position to flounder in formalization (Piaget, 1980b). First on the argument that no single logic is strong enough to support the total construction of human logic and that all the logics taken together are too rich to enable logic to form from a single basis i.e., formalization alone is not sufficient. Secondly, Gödel's incompleteness theorems¹⁴ disprove of Chomsky's innate logic approach. Next, since epistemology sets out to explain knowledge as it actually is, any explanation must include the aspects of human psychology. Piaget rejects Chomsky and asserts that, "The fundamental hypothesis of genetic epistemology is that there is a parallelism between progress made in the logical and rational organization of knowledge and the corresponding formative psychological processes" (Piaget, 1970a, p13).

Chomsky leaves the task of explaining the postulated underlying device, the mental organ, to biologists. Cognition is thus computation on representations as symbol structures, with no indication of how symbol formation occurs. Piaget maintains that, the language generated by this "mental organ" is based on logic, and considers the development¹⁵ of this "logic" as innate (Piaget, 1970a, p13).

If it is presumed that some internal mechanism is provided by genetic inheritance, and that, for Chomsky, this genetic inheritance is the potential to learn grammars, (with the grammars being inherent structures supporting manipulation of, more or less, formal transformations from surface structures to hierarchical structures and back again), then the learning paradox can be resolved. What is not available is a mechanism to address novel situations, such as those that occur regularly to humans through evolution. Computational psychology also reveals some major problems with respect to a lack of fault or noise tolerance.

¹³ Behavioral in this context can be interpreted as cognitive.

¹⁴ For Gödel, "within any formal system using only such concepts (recursiveness) and capable of expressing arithmetic, it is impossible to establish its own consistency" (Gödel, 1992, p26).

¹⁵ We refer to this development more appropriately as emergence.

Chapter 2. Literature Review

Chomsky's decomposition position¹⁶ is decried by a theory of evolution¹⁷, which must, in principle, account for both holism and decomposition since both aspects are "observables" in the living (Pinker, 1994). Any valid solution must explain the parallel between the evolutionary and the ontogenetic level (between the whole and the parts). The key aspect is that the evolutionary hierarchy provides a partial ordering, which is reflected in the ontogenetic sequence, as evidenced by cognitive development¹⁸.

Representationalists like Chomsky explain the functioning of a system by assigning to it their own knowledge of the world. They can be seen as epistemic agents outside of the system who define an encoding relation that exists for observers, but is not inherent within the system under consideration itself. That is, the structure is not present in the external world, but only within the observer.

Chomsky's model of language acquisition does not account for the increase in complexity shown by individuals over time. Chomsky's model predicts decreasing structure, which is evidently not the case (Chomsky, 1995). The core question is, "How can the development of complex mental structures be accounted for by mechanisms that are themselves not highly intelligent, nor, richly endowed with knowledge" (Bereiter, 1985, p205)? The central point is how bootstrapping, which is understood to be the progress toward higher levels of complexity and organization, is possible and occurs, without there already being an innate "ladder or rope to climb on" (Bereiter, 1985). For Bereiter, adding complexity is not simply a matter of adding elements or chunking. For Bereiter, the learning paradox is resolved in terms of a computer program, which includes itself as a subroutine and overcomes homunculus¹⁹ (Bereiter, 1985, p 205). Researchers miss the significance of the learning paradox. First, for Chomsky the issue is innateness of knowledge, but more importantly, one should consider how experience modifies these cognitive structures (Bereiter, 1985, p202). Second, crude addition to efficient addition cannot be explained in terms of improved strategy in the learner, utilizing

¹⁶ In the Minimalist Program, a part of generative grammars, Chomsky argues from a naturalistic and reductionist standpoint for the decomposition of sentences (typically) into phases under an optimal design (the universal grammar) that meet the conceptual and phonological needs of the individual (Chomsky, 1995). In linguistics, this holism is based on the belief that it is not possible to understand learning through analysis of small chunks. Often referred to as "whole language," it forms the basis of Chomsky's generative grammars, which is evidenced in skill based instruction and the "whole part whole" approach, through the use of the language acquisition device.

¹⁷ Evolution itself exhibits two fundamental characteristics of holism and decomposition (reductionism) (Smuts, 1927). Whereas holism focuses on how organisms, become increasingly intricate over time, reductionism focuses on how the most adaptive subsystems function and are selected in evolution.

¹⁸ Development is assumed to define LP6: Children develop their learning process. In this, the construction of what is already known and the process by which that knowledge is acquired is altered by the maturing child (Furth, 1969, p105).

¹⁹ Homunculus is the process of infinite regress.

Chapter 2. Literature Review

consistency detection and redundancy elimination, since this is just avoidance (Klahr and Wallace, 1976 cited by Bereiter, 1985).

If the learning paradox is taken as the basis of the distinction between additive learning (elaborative learning) and learning that increases the structural complexity (Kohlberg, 1968 cited by Bereiter, 1985, p217) then rather than merely describing learning at different levels of abstraction (Bereiter, 1985), it becomes clear that the increase in structural complexity is not the difference between kinds of learning. This is the distinction between “kinds of learning that can be accounted for on the basis of knowledge schemas that the learner already possesses and learning that involves a new cognitive structure to which already existing schemes are subordinated” (Bereiter, 1985, p217).

Any solution to the learning paradox must be capable of developing these structurally complex schemes²⁰. Examples being second language development, creativity and rational numbers (Bereiter, 1985, p206). Bereiter defines a set of heuristics, as steps and a direction, for a theoretical solution of how to analyze the bootstrapping process (Bereiter, 1985, p206). Specifically, if some opportunistic kind of cognitive development can be shown to be significantly influenced by instruction, then evidence of learning a more complex cognitive structure has been produced (Brown, Bransford, Ferrara and Campione, 1983 cited by Bereiter, 1985, p208). The implication is that some form of supervised learning including an interaction between a teacher and student is critical to test this form of development.

2.1.3 Bereiter

Developing this approach, Bereiter and Scardamalia suggest the possibility of using defined factorsⁱⁱ within behavioral settings to account for learning new, complex cognitive structures and to overcome the learning paradox through bootstrapping (Bereiter and Scardamalia, 1983 cited in Bereiter, 1985, p204). Bootstrapping is important since it relates to hierarchical machine re-construction, which is, itself, related to theories of emergence (Crutchfield, 1994a and 1994b).

From the assumption that babies have innate knowledge (Spelke, 1982; Carey, 1978 cited by Bereiter, 1985), Bereiter assumes bootstrapping depends on characteristics not shared by computers²¹ and within a rich innate endowment, which places learning, as a nontrivial role in cognitive development. However, this does not explain the whole of development. Bereiter suggests avoiding genius because genius is subsequently so different from the norms of development and to look for areas where bootstrapping is chancy, and where achievements in development are less universal and depend on fortunate events or special circumstances. Most importantly, he assumes that teachability implies bootstrapping: and if some chancy kind of cognitive development can be shown to be significantly influenced by instruction, then evidence of the learning of a more complex cognitive structure has been shown (Brown, Bransford, Ferrara and Campione, 1983 cited in Bereiter, 1985, p208).

²⁰ Scheme refers to the conceptual structures that are developed by an individual.

²¹ As computing was defined in 1980 (Pylyshyn, 1980 cited by Bereiter, 1985).

Chapter 2. Literature Review

In conclusion, Bereiter's remarks point at the necessity of avoiding premature reductionism, since explanations may be more complex than the thing itself. This approach is a key feature in the solution brought forth in this thesis.

2.1.3.1 Bereiter: Where is the problem?

Even in defining the learning paradox, Bereiter and Scardamalia did not provide an implementation model of its resolution. They merely point at areas, like the defined factors and bootstrapping, which allude to a partial solution (Bereiter and Scardamalia, 1983 cited in Bereiter, 1985, p204).

2.1.4 Fodor

Fodor, as with Chomsky, adopted Plato's position and argued that any resolution to the learning paradox required that individuals already had higher level, but as yet, unused, capabilities in order to "learn" them (Fodor, 1980, p149). Fodor argued from a meta-theoretical perspective in terms of the logic that would describe any set of lower level capabilities. Fodor's argument can be seen as a limitation on both human and existing artificial learning systems. Fodor viewed hypothesis formation and induction as central to conceptual learning and suggested:

"... It is never possible to learn a richer logic on the basis of a weaker logic, if what you mean by learning is hypothesis formation and confirmation. ...There literally isn't such a thing as the notion of learning a conceptual system richer than the one that one already has; we simply have no idea of what it would be like to get from a conceptually impoverished to a conceptually richer system by anything like a process of learning" (Fodor, 1980, p149).

Fodor's position is how one can learn a description without already having another description i.e., learning a description that can account for a problem with any degree of detail requires that the same level of detail already exists in the mind, and can be accounted for, at the same level of detail. In the debate and in earlier work on the language of thought (Fodor, 1975), Fodor argued that learning at higher levels of analysis is impossible from lower, more descriptive instances. If thought is mediated through the cognitive structures that are inherent in the brain, then thought is mediated through syntactic (not semantic) models. It does not make sense (logically or otherwise) to support learning descriptions at a higher level of logic. Formulating a higher-level description is not possible unless it is already supported – so the system is not then operating at a higher level.

Continuing these arguments, Fodor, suggests, "What people believe to be a theory of concept learning (concept acquisition or learning) is actually a theory of the fixation of belief by experience of an individual" (Fodor, 1980, p144). Epistemologists including Putnam²² (Putnam, 1975) provided the basis for Fodor's view

²² In multiple realizability (Putnam, 1975), Putnam supports Fodor's argument and provides an effective argument against type-identity theories. The crux is that fixation of belief concerns a balance between interpretation and observation. Whereas identity theories purport a form of reductionism in which the mental item is nothing more than the physical item with which it is associated. The "likelihood argument" implies that

Chapter 2. Literature Review

and to get the theory of the fixation of belief to work and this is understood to mean the impossibility of acquiring more powerful structures requires a radical background in nativism (Fodor, 1980, p144). Fodor explains this as:

- 1) Most people assume that the fixation of belief occurs through inductive inference using some form of non-demonstrative inference using hypothesis formation and hypothesis confirmation, from some set of beliefs, to generalizations, as in the following experiment:
 - a. X is *miv*, if and only if X has [critical attributes: specification attributes, rewards, relationships of attributes and rewards] ...
- 2) The experiment makes the following assumption:
 - a. The critical attributes, which form the hypothesis, are predetermined and fixed in the experimental situation;
 - b. The concept *miv* is also known since it is provided in the hypothesis before it has been learnt.
- 3) All the experiment tells us:
 - a. How various selections of critical attributes affect the individuals convergence through experience (inductive learning) on accepting the right hypothesis;
 - b. That the likelihood that a hypothesis will be accepted by an organism varies with the aspects of the organisms experience in the environment.
 - c. That hypothesis formation and hypothesis confirmation is the only known method that can be used for any belief that can be fixed.
- 4) What the experiment does not tell us is:
 - a. Where the notion of hypothesis comes from; and
 - b. How to acquire the concept, *miv*.
- 5) **Conclusion 1:** Thus, a theory of the fixation of belief is not a theory of concept learning. It uses inductive inference. To accept the arguments of both implies that one is a nativist (Fodor, 1980, p 147).

However, observations of children's behavior can be described in different ways:

- a. **Tenet 1:** Observations of children's behavior show that they develop stronger logics as they mature: As a child progresses through stages of development, they develop stronger forms of logic e.g., they learn transitive concepts then reversibility concepts in later stages.
- b. **Tenet 2:** These stronger logics contain weaker logics: Each stronger logic will asymmetrically contain the weaker logic²³. Examples are:
 - i. **Stage 1:** Propositional logic
 1. It is not possible to give truth conditions in formulas, for example²⁴:

any low-level explanation of higher-level mental phenomena is insufficiently abstract and general. In the *twin earth* thought experiment, Putnam asserted semantic externalism, insisting that "meanings just ain't in the head" (Putnam, 1975, p227).

²³ More expressive logics have the former logic as a proper part (Fodor, 1980, p148).

$$(\forall n)(\exists m)(\text{int}(n) \rightarrow (\text{times}(2, n, m) \wedge \text{even}(m)))$$

2. All you can say in propositional logic is that an instance is true: similarly the assertion $X > 1$ where x is a variable is not a proposition because you cannot tell whether it is true or false, unless you know the value of x . This can be restated, “you cannot quantify all the attributes in a propositional logic”²⁵.

ii. Stage 2: Predicate logic

1. A theory in propositional logic is always a theory in predicate logic, but a theory in predicate logic is not a theory in propositional logic.

By accepting that children develop stronger logics as they mature and stronger logics contain weaker logics (from **Tenet 1** and **Tenet 2**) then transition between developmental stages in Piaget’s genetic epistemology cannot be a learning process since there is no such process as concept learning (from **Conclusion 1**). A general form of argument for this is:

- 1) If one assumes that learning is inductive inference:
 - a. One has to be able to create hypotheses when **Stage 2** concepts are instantiated
 - b. One has to be able to characterize truth conditions on formulas containing the concepts of **Stage 2** while in **Stage 1**.

For Fodor, all that one can say is that concept development in humans occurs as a part of maturation. Secondly, that hypothesis and hypothesis confirmation through experimentation are the only models of learning known to exist.

2.1.4.1 Fodor: Where is the problem?

Fodor uses the separation between propositional and predicate calculus to suggest that it is not possible for an organism that can do propositional calculus to formulate problems at a predicate level, from within the system itself.

According to Fodor, formulating the learning paradox introduces a number of problems, which need to be resolved. First, identifying what concepts could be learned in the potential space of concepts to learn. Second, identifying what kind of description might be adequate for this problem i.e., how could the system learn it? Third, identifying what might be relevant from the current repertoire of concepts that might be worth noting which defines the description space. Last, identifying the evaluation strategy i.e., how does one decide between

²⁴ In the given example of predicate logic “every integer times itself is even,” exhibits skolemization, the introduction of constants or functions.

²⁵ This infers the definition of **LP1**: Works within the constraints imposed by Fodor. Which is, using a propositional logic, one cannot use quantifiers (universal nor existential). Secondly, the starting point for processing, must, at most, only be propositional. This could be taken to imply a structure, with *predicate logic* operating in some hierarchy, and *propositional logic* doing some form of processing.

Chapter 2. Literature Review

alternative strategies. Lastly, it is necessary to identify the mental resources that are needed in order to support concept formation by the individual.

Fodor claimed that hypothesis formation is an inductive process (Bereiter, 1985, p201). This is a technical expression by researchers who create new knowledge by spending considerable time collecting “data” and that examination of these “data” then induces the hypotheses they define. This induction, Fodor asserts, is only possible if the logical structure of the hypothesis was in some form already present in the researcher. In review, it makes no sense to claim, as Fodor and Bereiter did, that because hypothesis formation is an inductive process, there is a learning paradox concerning all theoretical conceptual structures that cannot be gleaned directly from experiential data. Research shows (Von Glasersfeld, 1998; Fodor, 2008, p132) that every inductive inference involves the spontaneous creation of an idea that may turn out to fit the “data” but was not actually inherent in them. The same is true of conceptual accommodations and even of many elementary accommodations on the sensory motor level. In both cases, a conceptual step fits the pattern of abduction. The conceptual step generates new knowledge whenever the abduction proves viable (Von Glasersfeld, 1998, p9).

Fodor’s arguments for the learning paradox unravel when one considers encoding for novel situation: one can only learn what one already knows; experience can only select the appropriate inborn representation. This correspondence, between internal symbol and external phenomena, must have been constructed somewhere, since in Fodor’s arguments there is no crossing between symbol and external phenomena. However, if, as Chomsky and Fodor suggest, learning cannot account for the basic encodings, how could evolution succeed in accounting for them²⁶ ?

The implication for the research is that a solution must model evolutionary development and in doing so account for both Fodor’s arguments as well as Bereiter’s. What is required is a model of how development could occur, one that accounts for the childhood issues such as incorrectly applied knowledge, the differences in levels of knowledge, novel situations, problems and, above all, the development of logics.

2.1.5 Piaget

For Piaget, the origin of the fixed nucleus is sensory motor activity, which is grounded in mechanisms of perceiving, behaving (acting), of feeling, and, the biological mechanism. Researchers referred to the sensory motor schemes as the “cognition–first assumption,” or “front door approach” and, the “back door” to the processes of self–organization by assimilation and accommodation. Piaget assumed that sensory motor schemes are the pre–conditions for language, providing the mechanisms for linguistic structures e.g., point, penState, direction, movement and line for number–sense. Schemes are the conceptual relations, the “prime movers” in the process of acquisition, from which syntax is derived. In the debate, Piaget assumed that language is continuous, with generalization of those sensory motor schemes developing the specificity of the fixed nucleus (Piaget, 1980b).

²⁶ This becomes the definition for LP7: Children act in novel, noisy and opportunistic situations.

Chapter 2. Literature Review

Piaget can be viewed as representing holism with a mechanism of adaptation that uses accommodation and assimilation to account for increasing complexity. This increase in complexity is observed as the differentiation and hierarchical integration, of cognition, within the individual.

Piaget converged with Chomsky on five central issues (Piaget, 1980, p57):

- 1) The inadequacy of empiricism²⁷ – “fundamental relationship that constitutes all knowledge is not, therefore a mere association between objects, for this notion neglects the active role of the subject, but rather the assimilation of objects to the schemes of that subject” (Piaget, 1980a, p377; Furth, 1969, p93);
- 2) Transitions between states using logic as rules derived through deduction, principles and constraints are formally constrained by logical necessity, which is constructed through interiorization and equilibration (Furth, 1969, p68–82);
- 3) The internal activity of the subject generates language (for Chomsky) using its generative grammar and develops objects of thought (for Piaget) using its innate schemes and primary reactions following evolutionary and developmental trends (Furth, 1969);
- 4) Development²⁸ must be studied in real time using experiments with subjects; and
- 5) Language is the product of intelligence.

The main issues that wrestled between Chomsky and Piaget was the innateness of the fixed nucleus, which consists of cognition, self-organization, universal trajectory²⁹ and generalization (Piaget, 1980a, 1980b).

First, cognition is adaptive self-organization, which occurs through the processes of assimilation and accommodation. Piaget addresses Fodor’s concern:

²⁷ Piaget’s position can be considered anti Kantian, since Piaget disagreed with the whole representation theory of knowledge upon which most symbolic processing is built (Furth, 1969, p93).

²⁸ The development of knowledge is spontaneous and tied to the whole process of *embryogenesis*. Embryogenesis concerns the development of the body, but it also concerns the development of the nervous system and the development of mental functions. In the case of the development of knowledge in children, embryogenesis ends only in adulthood. It is a total development process which one must resituate in its general biological and psychological context. In other words, development is a process which concerns the totality of the structures of knowledge. Learning presents the opposite case. In general, learning is provoked by situations – provoked by a psychological experimenter; or by a teacher, with respect to some didactic point; or by an external situation. It is provoked, in general, as opposed to spontaneous. In addition, it is a limited process - limited to a single problem, or to a single structure (Piaget, 1964, p8).

²⁹ The universal trajectory is variously referred by Piaget as the purposeful developmental trend and, evolutionary trend. “The evolutionary trend towards a lesser degree of immediacy and specificity in behavior becomes, in intellectual development, a process of increased reflection, a turning inwards or an interiorization of action that changes coordinated external actions into systems of interior, reversible operations. Equilibration can be viewed as a compensatory response to these biological trends or better as guiding and regulating this trend towards the building up of more advanced structures” (Furth, 1969, p209).

Chapter 2. Literature Review

“the fundamental relationship that constitutes all knowledge is ... the assimilations of objects to the schemes of that subject. This process ...prolongs the various forms of biological assimilations, of which cognitive association is a particular case as a functional process of integration. Conversely, when objects are assimilated to schemes of action, there is a necessary adaptation to the particularities of these objects ... It is this exogenous mechanism that converges with what is valid in the empiricist thesis, but ... adaptation does not exist in a pure or isolated state, since it is always the adaptation of an assimilatory scheme; therefore this assimilation remains the driving force of cognitive action” (Piaget, 1980a, p24).

Second, self-organization and self-stabilization are real features of biological systems, which can be described in logico-mathematical schemes. Piaget comments, “these mechanisms, which are visible from birth, are completely general ... The role of assimilation is recognized in the fact that an observable or a fact is always interpreted from the moment of its observation, for this observation ... requires the utilization of logico-mathematical frameworks such as the setting up of a relationship or a correspondence, proximities or separations, positive or negative quantifications leading to the concept of measure ...” (Piaget, 1980a, p377).

Third, within the process of self-organization, a universal trajectory can be observed in which self-stabilization occurs repeatedly. Stability is lost just before the system makes a transition to a next stable state. Assimilation, accommodation³⁰ and adaptation³¹ provide concepts appropriate for describing the process of change, which are called “development”³². However, this stability may well only exist as the observer and not the observed. This trajectory and the inherent nature of the arguments for inheritance (where living systems increase their level of organization) in evolution, reduced Piaget’s arguments in the debate. For Piaget, innateness alone cannot explain novel solutions. He says, “If there were innateness, reason and language would be the result of selected accidents, but selected subsequently, after the fact, whereas the formation itself would be the result of mutations and would therefore occur at random ... I absolutely refuse, for my part, to think that logico-mathematical structures would owe their origin to chance; there is nothing fortuitous about them. These structures could not be formed by survival selection but by an exact and detailed adaptation to reality” (Piaget, 1980a).

Since the structure of language and the structure of learning are generalizations of already developed sensory-motor schemes, these schemes form the logical premises for cognitive structures. Specifically, “The functioning of intelligence alone is hereditary and creates structures only through an organization of successive actions performed on objects” (Piaget, 1980a). Later reviewers (Spencer, Smith and Thelen, 2001, p1327; Smith and Thelen, 2003; Van Geert, 2000 cited in Smith and Thelen, 2003) discuss this inheritance in terms of dynamic systems and accord Piaget with understanding the need to balance entropy with organization. Individuals are negative entropy systems, in which, through self-organization (equilibration), order spontaneously emerges out of chaos. These self-organizing systems seek new solutions that may be more efficient, effective or complex in direct opposition to the continually increasing disorganizing entropy. This

³⁰ Accommodation is seen as differentiation.

³¹ Adaptation is seen as hierarchical integration.

³² Development is seen as inclusion and generalization.

Chapter 2. Literature Review

self-organization leads to adaptive behavior (accounting for novelty as accidental) and Darwinian selection acts on these behaviors. An increasing energy exchange with the environment is required to generate the increase in structure (Prigogine, 1980, p17).

2.1.5.1 Piaget: Where is the problem?

Monod clearly identifies issues with Piaget's approach (Piattelli-Palmarini, 1980, p140). If the development of language in the child is closely related to sensor-motor experience, one can suppose that a child born paraplegic, for example, would have very great difficulties in developing language. In support of Monod, Fodor reasoned, "But then, why not say that any case of ethological triggering, where an arbitrary action is innately connected with a highly developed scheme of intelligence would count as learning? However, that merely trivializes the doctrine that intelligence arises out of sensory-motor activity and there is really hardly anything left" (Piattelli-Palmarini, 1980, p1401). In this case, sensory-motor schemes are only a "trigger" and are not necessarily isomorphous or analogous to the structure that puts it on the move.

Some neo-Piagetian researchers have elaborated on Piaget's mechanisms of developmental change, incorporating the impact that biology, task requirements, culturally determined experiences and peer influence have upon a child's intellectual performance (Case, 1984; Case, 1996; Griffin, 2004; Schwartz and Fischer, 1994). These researchers rejected the stage-like development so evident in Piaget's own observations of children and concluded that Piaget had also underestimated the capabilities of these children. However, if one were to apply the stage-like development only to schemes (§ 2.3.6) then one can clearly see that some children could possess unique capabilities earlier and/or later than other children could. However, other researchers have clearly seen that the child does go through some developmental cycles (Copeland, 1974, p83; Liebeck, 1984; Karmiloff-Smith, 1996, p19; Furth, 1969).

With Chomsky and Fodor, one could easily adopt the belief that complete cognitive skills or abilities can be performed only because they can be decomposed into individual parts or features of the cognitive structures, which came into being on the evolutionary timescale. However, whenever confronted with the fact that many cognitive processes are deceptively simple and effortless, one may conclude with Piaget that these processes are governed by holistic principles, not necessarily inborn in an evolutionary blueprint, but at least to some extent dependent on the organization by "systems" with their "global dynamics" (Furth, 1969).

The backbone of Piaget's genetic epistemology (epigenesis) consists of explanatory categories such as equilibration, self-organization and self-adjustment. These reflect the fundamental heuristic role played by Piaget's evolutionary model, which some researchers perceived as being anti-Darwinian. One can agree that Piaget's ideas are convoluted, awkward and sometimes just plain wrong, but it may be possible to review his work from a systems perspective and use this to construct a system that approaches human problem solving in novel situations. For this to work, it will have to deal with the major issues as outlined by Fodor and Monod (Piattelli-Palmarini, 1980, p140, Fodor, 1980, p149).

Chapter 2. Literature Review

Piaget's concept of "phenocopy" complicated the debate, since deeper analysis showed it to be Lamarckian – Piaget imposed a feedback from experience to genetic makeup. This approach may simply have been due to Piaget's lack of knowledge on genetics.

Some researchers have noticed problems in the Piaget's methods and achieved a balance with Fodor's nativism (Karmiloff-Smith, 1996). In presenting a "mosaic like" approach to cognition, they assume a representational model of development in children, as objects of cognitive attention, which change over time. In this approach, Karmiloff-Smith accepts the concept of an innate architecture for each of the five domains (the child as linguist, physicist, mathematician, psychologist and notator) and concludes that there are phases (not stages) of development. Rejecting Piaget's stage-like variation, Karmiloff-Smith favors a "recurrent system in which three phases characterize the mastery of each conceptual domain" (Shultz, 1994, p728) and it is the internalization of experience that is critical and becomes the focus of change (Karmiloff-Smith, 1996, p19). These objects are re-described into a "higher level language" that is common across all of the domains and it is this interlingua which opens up the "inter domain representational links" (Karmiloff-Smith, 1996, p19). What is lacking in Karmiloff-Smith's research is the mechanism that explains the development of this "higher level language." Piaget's genetic epistemology explains this mechanism in rational terms as the development of this language from logic (Piaget, 1970a, p13). Secondly, as Shultz points out "Verbalizations may be quite easy to document in *representational redescription* (RR), but nonverbal explicit representations seem to be more challenging" (Shultz, 1994, p729). Researchers have constructed machine learning systems based on RR (Wei and Fu, 2009) using a hierarchical object oriented processes. Yet, as will be seen, these still fail Fodor's arguments.

It still needs to be proven that radical innatism and a mutation-selection hypothesis, can, in principle, provide a joint pattern of explanation for both phylogenesis and ontogenesis. One researcher, who attempted to resolve the learning paradox and implement a Piagetian model of development, explaining the stage-like development and identifying flaws in Piaget's work, was Pascual-Leone (Pascual-Leone, 1980, p280).

2.1.6 Pascual-Leone

Pascual-Leone defines the learning paradox as a meta-theoretical problem, which can be defined as the meta-theoretical question "How can a structure generate another structure more complex than itself?" (Pascual-Leone, 1976, 1980 cited by Bereiter, 1985).

The abstract and technical theory of constructive operators (TCO) seeks to "represent explicitly the underlying mechanism of dialectical equilibration and structural growth" (Pascual-Leone, Goodman, Ammon and Subelman, 1978, p252). TCO integrates three sorts of organismic constructs: schemes, basic factors and basic principles within a process model of the psychological organism, the meta-subject. It is called the meta-subject, to indicate that it is a "highly active hidden organization which is causally responsible for the subject's performance" (ibid, p252). Pascual-Leone posits that TCO is a viable process that explains and extends the Piagetian concepts of general stages and equilibration (Pascual-Leone, 1980, p280) and resolves the learning paradox. When TCO is used in conjunction with the meta-subjective task analysis, research shows that it is a

Chapter 2. Literature Review

determinant of the demand for mental attentional energy (M Demand) required for cognitive tasks, which is consonant with the developmental stage of the subjects in the test. The theoretical conclusion of TCO is that it provides a simulation and explains why complex embedded figures tasks (EFT) are accessible to 9 and 10 year olds given careful and sufficient practice but not to younger children that do not have sufficient mental attentional energy (Pascual-Leone and Goodman, 1979, p347; Pascual-Leone, 1980, p271; Pascual-Leone, 1996, p87). EFT is as an example of the Bi-Polar Dimensions style of cognitive, learning and teaching styles (Pascual-Leone and Goodman, 1979, p308). The other main reason for reviewing Pascual-Leone is that it provides a theoretical model of Piaget's work. Thus, its utilization avoids "the theory is in the code" problem of many complex information systems.

Piaget initially identified the mental attentional energy of an individual in terms of attention span, field of centration and field of equilibrium (Pascual-Leone, 1970, p302). Pascual-Leone refined these into M-Capacity and posits that if its numerical characteristic were proven then it could be used to explain the transition from one stage to another. Pascual-Leone describes a machine like, abstract psychological model which is capable of explaining for instance, the appearance of the conservation of substance before the conservation of weight (Pascual-Leone, 1970, p303, p336). A key issue is that Piaget was not committed to an abstract representation and so could not provide an abstract model, whereas Pascual-Leone has built a complete model of genetic epistemology (Pascual-Leone, 1970, p336).

In developing the TCO, Pascual-Leone enriches Piaget's concepts of general stages and equilibration with a theoretical model of deep information processing with two significant claims regarding cognitive psychology:

Claim 1: That a subject's psychological organism is composed of a generative model and psychogenetic model. The generative model stimulates the processes that generates the subject's performance using recursive operators and is balanced by the psychogenetic model, which stimulates the organism's developmental process of change using innate recursive operators that develop in its lifetime (Pascual-Leone, 1980, p263).

Claim 2: Scientific theories and their constructs reflect the structural invariants of their databases, as filtered by the data analysts (scientists). Which is understood in terms of Claim 1, that the "epistemological function of reflection must be understood as signifying that the theories or constructs in question can simulate the psychogenetic or generative processes which caused the performances reflected in the data" (Pascual-Leone, 1980, p264).

Pascual-Leone considers that the learning paradox does not apply to human cognition because there is no need for computational closure (generality) in human cognition (Pascual-Leone, 1980, p279). Further, that since situation specific structures cannot be learned without experience (the learning paradox), some unlearned (situation-free) organismic factor (F, which is similar to stimulus-response compatibility) must exist to support spontaneous inference in generative constructive tasks such as conservation or inclusion of classes when they are solved for the first time (Pascual-Leone, 1980, p275).

Chapter 2. Literature Review

In explaining the learning paradox as acquisition without learning, Pascual-Leone identifies an evolutionary paradox in which lesser animals (and included within this are computer simulations³³) cannot exhibit or develop abstract processing forms (Johnson and Pascual-Leone, 1989 cited in Pascual-Leone, 1996, p85). In rejecting the neo-Piagetian approach where processional forms are innate, Pascual-Leone proposes that the principles of praxis (goal directed activity), modular organization and equilibration cannot be fully expressed without the use of schemes, reflective abstraction (which is a processes that mediates transition from one level to another) and mental attentional energy (Pascual-Leone, 1996, p86-87). In addressing mental attentional energy, Pascual-Leone notes that only in misleading situations³⁴ and novel situations³⁵ do the Piagetian developmental stages appear. These stages appear “exhibiting a non-linearity, growing in time with a stepwise characteristic growth curve as a function of chronological age” (Pascual-Leone, 1996, p87). In attempting to explain the relationship between the psychogenetic difficulty of Piagetian tasks and the “rank” or dimensionality of their problem solving strategies, Pascual-Leone (Pascual-Leone, 1980, p271) considers there to be a correlation to the M-Demand (mental attentional energy demand) for mental attentional energy. This M-Demand can be determined using meta-subjective task analysis and is realized in experimental results. These stage like variations occur because “learned habits (or innate automatisms³⁶) become obstacles for good performance” as these highly activated schemes tend to dominate (Pascual-Leone, 1996, p88). Obviously, performance does improve over time and one reason for this may be that there are multiple forms of learning:

- 1) LC-learning – which is understood to be associative content learning, and
- 2) LM-learning – which is understood to be logical (structural) learning³⁷.

The effect of LC/LM learning is that it makes accessible at the current M-Capacity of children (and of adults) previously inaccessible solutions; hence, it explains the learning paradox. It is also this LM-Learning which the organism uses with mental attentional energy (M-Capacity) to centrally inhibit (using an interrupt operator or I-Operator) schemes using the executive schemes (E-Schemes), that are task irrelevant.

In TCO, Pascual Leone identifies two key problems that cognitive psychology is still struggling since originally posed by Plato. First, why, and how are proper concepts (universals, logico-mathematical or generic knowledge) irreducible to, yet psychogenetically derivable, from proper experiences. Second, why, and how are, conceptual and experiential forms of knowing / processing continually adapting to the constraints of the current situation. In addressing the first problem, Pascual Leone suggests that mental attention is a determinant within both a Piagetian constructivist and interactionist framework that does not artificially separate the

³³ Computer Simulations in this context refers to work circa 1989, before evolutionary computation and machine learning had been widely read by cognitive science researchers.

³⁴ For Pascual-Leone, a situation is misleading when it elicits schemes that interfere with the task at hand.

³⁵ For Pascual-Leone, problems, are situations which are typically misleading.

³⁶ Innate Automatisms assume a definition of **LP3: Operates Autonomously**.

³⁷ As we shall see in the following chapters, cognitive architectures such as SOAR-R also introduce processes of optimization as a distinct form of learning.

Chapter 2. Literature Review

emergence of both modes of processing from the same origin (Pascual-Leone, 1996, p84). In addressing the second problem, Pascual-Leone identifies the principle of scheme over-determination of performance (SOP) and the gestalt field factor (stimulus response compatibility, S-R compatibility or F-operator) as the drivers for the continuous representation problem. In this model, SOP restricts the dominance of schemes and the F-operator minimizes the number of schemes that apply to the stimulus. Mental attentional energy explains Plato's first problem of "proper concepts" since it drives the development of the organism and increases with age (Pascual-Leone, 1996, p89).

In rejecting Fodor's absolute, meta-theoretical argument on the impossibility of creating more complex structures, Pascual-Leone argued that although hypothesis formation is an inductive process, it is based on limited experimental data and that there is no limit to the theoretical conceptual structures that could be gleaned from experiential data (Pascual-Leone, 1976, 1980 cited by Bereiter, 1985, p201). Research by Von Glasersfeld suggests that inductive inference – which involves the spontaneous creation of an idea, which may turn out to fit the data – is not inherent in the data (Von Glasersfeld, 1998). The same is true of conceptual accommodations and even of many elementary accommodations on the sensory motor level. In both cases, there is a conceptual step that fits the pattern of abduction, a step that generates new knowledge whenever the abduction proves viable (Von Glasersfeld, 1998, p9). However, if abduction is all that is required and Von Glasersfeld is rather short on how abstraction can be supported as a process of learning to support abduction, this is less problematic in the sense that it requires less than Chomsky but there is a danger that it still assumes just what is needed to support his model of learning, it is also no less presumptive in metaphysical terms. Pascual-Leone identifies some success conditions for a computer simulation: namely, it must exhibit truly novel performances, cognitive conflicts, and affective conflicts, which include internal and external motivation (Pascual-Leone, 1980, p288).

2.1.6.1 Pascual-Leone: Where is the problem?

Like its Piagetian predecessor of dimensional analysis, TCO and meta-subjective task analysis must begin by "assuming the strategy (and the structures) used by the subject" (Pascual-Leone, 1980, p271). Further, the method's abstractness (its distance from the step-by-step concreteness of performance in data protocols) makes it "appear, and sometimes become, unreliable" (Pascual-Leone, 1980, p271).

Meta-subjective task analysis still has validity in defining the resource constraints of children. First, for any simulation that attempts to implement a Piagetian model of development, it provides a mechanism to restrict the processing capacity of a solution. Second, the scheme over determination of performance principle (SOP) in itself assumes a traversal across all potential schemes that could fire as the result of a situation. Hence, Pascual-Leone fails **LP1**. Third, over time, the performance of the system would degrade since more schemes are continually being added, and this degradation of performance is not seen in human systems.

Although Pascual-Leone asserts that schemes, equilibration, M-Capacity, C-Learning, LM-Learning, I-Operator, Executive Schemes, SOP, F-Operator and a modular organization together will resolve the proper concepts and the learning paradox, the description of the solution does not provide sufficient information for

Chapter 2. Literature Review

the emergence of structure from a biological perspective. Also, attempts to develop a workable pure cognitive model based solely from printed works on the TCO proved to be too difficult, and so as an approach was rejected in this thesis.

2.1.7 Constraints of a Simulation

The key problem is the lack of a simulation which is capable of developing concepts in the way that most children do naturally, while also overcoming the learning paradox. Creating a mechanism that has emergent properties, ones that enables it to develop concepts (and so overcome the learning paradox), will be a significant advancement for the development of intelligent systems. Fortunately, Piaget provides descriptions of how this system might work. An outline definition is provided with a description of the constraints **LP1 – LP7** believed to be necessary features of any simulation that attempts to resolve the learning paradox in a number line world.

2.1.7.1 An Environment for a Simulation

The obvious representation of the number line is a graph and it would be a trivial problem to adapt a graph grammar for concept formation (Jonyer, Holder, Cook 2002, p71), albeit using a high language. To implement a solution without the representational form of symbols becomes a much more difficult problem. The approach defined in figure 2-1 applies Piaget's genetic epistemology to the learning of number-sense in a number line world. It also serves to introduce a set of characteristics that are used to grade cognitive models, namely **LP1 – LP7**.

The original counting bead problem of figure 1-1, is reused to show how reflective abstraction allows the development of hierarchical concepts for schemes that develop from innate schemes through the sensory motor stage and into the pre-operational stage and beyond.

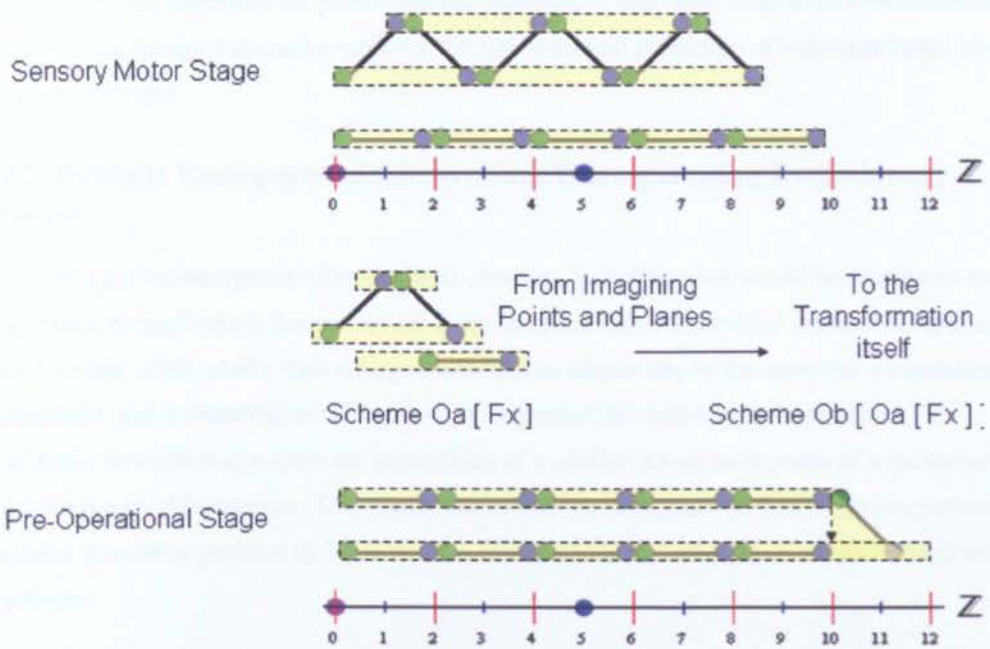


Figure 2–1 Piaget’s bead problem resolved through reflective abstraction: Moving from processing points and lines to processing the action itself.

For Piaget, reflective abstraction arises from a shift of centration (attention) from action on objects, to the action itself. For the child generating mental images of linear transformations of points on planes using their figurative and operative sensory–motor schemes, (depicted as Scheme Oa $f[x]$ in figure 2–1) is transformed through reflective abstraction into the emergence of pre–operational schemes that operate on the figurative and operative schemes themselves (depicted as function $h[f[x]]$ in figure 2–1). Using reflective abstraction “generalized” schemes emerge to support adaptation by the individual. Other researchers have observed this and referred to it as reification of process, perceptual development or as algebra of transformations (Mason, Drury and Bills, 2007, p49). For Piaget, the emergence of pre–operational schemes from sensory motor schemes is due to the evolutionary trend, which alters the learning process of assimilation and accommodation³⁸.

For this process to occur, it is believed that there are seven key constraints that need to be overcome. Each constraint is described in the following sections.

2.1.7.2 LP1: Works Within the Constraints Imposed By Fodor’s Arguments

The solution must be consistent with Fodor’s arguments concerning logic (§ 2.1.4). First, that there is some external focus for concept creation (e.g. satisfying a need) and second that the proof does not depend on quantifying over all possible situations. This implies that the solution cannot have an instantiated predicate

³⁸ It is this emergence of structure from sensory motor schemes which adds clarity to the definition of LP5: Learns Incrementally with Minimal Innate Knowledge and Reflexes.

Chapter 2. Literature Review

logic or include instantiated grammars for processing and planning. It must also come to its own conclusions about the nature of truth, though this can be achieved through statistical prediction of outcomes based on its own view of the environment.

2.1.7.3 LP2: Exhibits Emergence of Hierarchical Concepts using Evolutionary Process

The system exhibits the emergence of hierarchical concepts, including what would be considered as the simulations “symbolic forms,” which themselves are more complex than it’s provided innate structures and reflexes (Pascual–Leone, 1980, p263). This emergence is seen as adaptation, by the agent (as a simulation), within its environment, and is observed as the appearance of generalized behavior in its schemes. The development of these concepts may require the capabilities of evolution across an expanse of populations that increasingly exhibit projectable behavior. The assumption is that the emergence of these concepts partially resolves the symbol grounding problem (§ 2.7.2.1). Also anticipated is that emergence occurs through some evolutionary process.

2.1.7.4 LP3: Operates Autonomously

The system functions autonomously, with minimal innate knowledge and can overcome internal problem situations by being able to process using emergent concepts; it sidesteps the issue of designing perfect algorithms, but requires a contingency recovery procedure for each failure in which the system is organized to detect failures and recover from them as opposed to never failing (Firby, 1989). The autonomous operation may require epigenetic development.

2.1.7.5 LP4: Mirrors Real World Behavior of Children Learning Number–Sense using a Number line.

As a real–world problem, the solution must mimic the behavior that children demonstrate when they learn about numbers (Furth, 1969; Copeland, 1974). This must include the resolution of the bead problem (§ figure 1–1.). It must include a preponderance for estimation and prediction of future states, before effective control is achieved. Next, the solution must exhibit the emergence of its own “concepts” through interaction with the environment. Finally, the system must exhibit properties and processes of maturation including the acquisition of skills (procedural processes), facts (declarative knowledge), the development of conservation and the capability to make mistakes as well as to learn from them. The capability to predict is critical since it underlines the notion of truth and from this equivalence and composition is attainable.

2.1.7.6 LP5: Learns Incrementally with Minimal Innate Knowledge and Reflexes

The system uses incremental (sub–symbolic) learning based on an intrinsic reward based value system. In addition, the learning mechanism exhibits the capability to optimize the available knowledge base. Finally, the

Chapter 2. Literature Review

learning mechanism provides the capability to bootstrap the complete developmental process from minimal innate knowledge and reflexes.

2.1.7.7 LP6: Develops its Learning Process

The observed behavior of the system exhibits stage like variation with emergence of more complex structures that are limited by the available mental attentional energy (M–Capacity), this, through the development of its learning process. This development is presumed to be through a process of bottom–up learning where the agent acquires capabilities during its lifetime.

2.1.7.8 LP7: Acts in Novel, Opportunistic and Noisy Situations

The system (as an agent) must act in novel, opportunistic and noisy situations. In the present research, the simulated student (as an agent, composed of various Piagetian schemes) acts in a number–line world to develop number–sense to solve presented problems where misleading information is provided.

2.1.8 Conclusion

The learning paradox is concerned with the emergence of more complex structures from less complex ones. Piaget resolves the paradox through descriptions of a developmental cognitive process, which contrasts with Fodor and Chomsky’s innateness argument (Bereiter, 1985, p202). Bereiter and Scardamalia argue it is possible to overcome Fodor’s logical arguments through bootstrapping because the less complex and more complex structure are composed of the same defined factors (Bereiter and Scardamalia, 1983 cited in Bereiter, 1985, p204)³⁹. Any simulation to test Piaget’s theories needs to follow very closely his theoretical works. Therefore, two theoretical works will be referenced (Furth, 1969; Copeland, 1974)⁴⁰.

Pascual–Leone argued that there is no limit to the conceptual structures that can be gleaned from experience. However, there is a theoretical limit (M–Demand) that constrains what can be solved based on the meta–subjective analysis of the task difficulty. Though highly relevant, the work of Pascual–Leone will not be fully implemented in the present research. Further, the resolution to the bead problem (Furth, 1969), the development of number–sense (through counting) and the adherence to a set of constraints LP1 – LP7 are seen as necessary conditions under which any simulation that attempts to resolve the learning paradox, using a Piagetian model, would need to operate.

Since biological evolution, as exposed through children, has already overcome the learning paradox, § 2.2 describes research from neurophysiology, biology and evolution to support the argument that to foster emergence, the simulation must apply evolutionary principles.

³⁹ Within this research, bootstrapping is understood to be the evolution from limited complexity to greater complexity within a simulation using a form of machine learning with external rewards that fosters emergence.

⁴⁰ The full depth of analysis of these works, cannot, for reasons of space be addressed in this thesis, but are contained in other research works (Rendell, 2008).

2.2 Biological Evolution as a Basis for Emergence

“How does evolution produce increasingly fit organisms in environments, which are highly uncertain for individual organisms? How does an organism use its experience to modify its behavior in beneficial ways (i.e., how does it learn or adapt under sensory guidance)? How can computers be programmed so that problem-solving capabilities are built up by specifying what is to be done rather than how to do it?” (Holland, 1975, p1).

In § 2.1.7, a set of constraints were defined (LP1 – LP7) that would need to be met by any simulation that attempted to tackle the learning paradox using a Piagetian model. Since evolution has already resolved the learning paradox. This section summarizes some key aspects of biology and evolution, and discusses how these can be simulated, without contradicting the identified constraints. A more detailed discussion of these topics can be found in the appendices (§ A).

2.2.1 Summary

There is emergence of structure in the collective behavior of ant colonies, for instance in searching for food (Johnson, 2001, p45). This is evidence of them solving LP5 and LP7, as an optimization problem. This process can be simulated using swarm techniques where simple rules, embodied in agents using vast populations, are shown to exhibit novelty in opportunistic and noisy situations (Dorigo, Birattari and Stutzle, 2006, p34). Current models are limited, since their simulations do not produce the more complex “ant social model” (Gordon, 2010). Thus, there is a need to model aspects of human cognition. The assumption is made these swarm rules do not support a Piagetian model, since they do not have the capability to develop their learning process and so fail LP6.

Wolfram sees emergence as order from randomness and uses simple cellular automata to simulate biological processes (Wolfram, 2003, p383 and p398) and this research agrees that simulations can mirror biological processes (Wolfram, 2003, p223).

A model of *Drosophila* provides an internal engine that is more complex than a simple set of ant rules (Miesenböck, 2008, p52; Shang, Claridge-Chang, Sjulson, Pypaert and Miesenböck, 2007, p601). Since many aspects of the basic neural pathways are probably conserved, evolutionarily, the fly studies research provides a simple model for cognition can provide a mechanism to resolve LP6. This model consists of acting /sensing, learning and planning (Shang, Claridge-Chang, Sjulson, Pypaert and Miesenböck, 2007, p601). This requires implementation as a neural model, and it is assumed that the *Drosophila* model has similarities to a Piagetian model of learning and development.

Given that evolution has produced a solution to the learning paradox in the cell and neural networks in fruit flies and human beings, a simulation mechanism that mirrors this process is needed. Two approaches are suggested, artificial neural networks (Carpenter, Grossberg, Rosen, 1991; O'Reilly and Munakata, 2000; Sun, 2004; Drescher, 2002; Sutton and Barto, 1998 and Streeter, Oliver and Sannier, 2006) and FSA (Minsky, 1967; McCulloch and Pitts, 1943, p115; Kauffman, 1969; Kumar and Bentley, 2003; Hjelmfelt, Weinberger & Ross, 1992, p383; Koza, 1999, p386). Good examples of ANNs are *radial basis neural networks* and *linear neural*

Chapter 2. Literature Review

networks since these have been shown to mirror neuronal processing (Streeter, Oliver and Sannier, 2006). However, a lack of hierarchies in current implementations means that they cannot resolve LP2. FSA are useful solutions, because they mirror neuronal and cell processing, and are simpler to implement and evolve than ANNs. Recent brain research confirms the approach of using HFSA to model the brain structures with twin pillar hierarchies of perception and executive /action (Albus, 2008; Grainger, 2006a; Granger, 2006b).

The dopamine–norepinephrine reward process is a mechanism for autonomously building internal neural structures based on external interaction. Actor–critic temporal difference learning is a simulation mechanism for reward processing that can be used to build artificial neural networks that learn incrementally (Suri and Schultz, 1998, p350; Schultz, 2000; Schultz and Dickinson, 2000, p474; Fiorillo, Tobler and Schultz, 2003, p1898). Using this mechanism, LP3 can be resolved. It is also possible to use reinforcement learning with FSA and HFSA to build their network, though this thesis will use a more constrained version.

Since the set of constraints (LP1 – LP7) dictate a biologically plausible model be used in a number line world, confirmation is required in this thesis that can show emergence of structure by identifying regularities in the environment e.g., point, line, direction, penState, and so be capable of addressing LP4.

Since evolutionary computing is a software simulation mechanism that parallels evolution (De Jong, 2006; Haynes, Sen, Schoenefeld and Wainwright, 1995; Koza, 1992; Jacob, 2001), it is assumed that this is an appropriate method to use with HFSA to resolve LP2. By separating out the propositional components (FSA) from the hierarchical components (HFSA) the simulation sidesteps LP1.

To develop a simulation of *Drosophila* using binary FSA, a mechanism is needed to resolve some of the core issues, such as storage and concurrency (Maraninchi and Remond, 2001). A reactive systems approach using Argos (Maraninchi and Remond, 2001), provides this mechanism, which when combined with evolutionary computing (Jacob, 2001) and a form of reinforcement (Streeter, Oliver and Sannier, 2006), provides a way of resolving LP6 and produces a mechanism similar to evolvable hardware (Greenwood and Tyrrell, 2006).

Using Crutchfield’s model of intuitive emergence⁴¹ (Crutchfield, 1994a, p2) provides a mechanism for detection of the emergence of structure, and thus a strong foundation for a solution to attempt to resolve the learning paradox with a biologically plausible model.

2.2.2 Conclusion

The combined structure of an *artificial neural network implementation* (using radial basis and linear neural networks with TD learning) and a *dialectic system* implemented using a model of *Drosophila* with

⁴¹ In *intuitive emergence* something new appears, but the “pattern is always referred outside the system to some observer that anticipates the structure via a fixed plate of possible regularities” (Crutchfield, 1994a, p2). Since our solution is a closed system, the emergence is of coordinated behavior that develops to control the internal processes. These internal processes, as HFSA would be observable. This is the approach that will be taken in this research.

Chapter 2. Literature Review

HFSA using evaluation functions based on a model of reinforcement can address **LP1, LP2, LP3, LP5, LP6** and **LP7** using a biologically plausible model.

2.3 The Roots of Genetic Epistemology

“I take from Jean Piaget a model of children as builders of their own intellectual structures. Children seem to be innately gifted learners, acquiring long before they go to school a vast quantity of knowledge by a process I call Piagetian learning or learning without being taught. For example, children learn to speak, learn the intuitive geometry needed to get around in space and learn enough of logic and rhetoric to get around parents – all this without being taught” (Seymour Papert, 1980, p7).

This section provides a re-evaluation of Piaget’s model of cognitive development using work from Furth, Copeland, Pascual-Leone and Piaget. Piaget’s notion of causality is outlined in § 2.3.1. Assimilation and accommodation are introduced in § 2.3.2. The association of assimilation and accommodation to figurative and operative schemes is described in § 2.3.3, the primacy of action in § 2.3.4 and a description of Piaget’s knowing circle is provided in § 2.3.5. An outline of the inner working mechanism of the knowing circle is provided in § 2.3.6. This mechanism is then related to a model of *Drosophila*, in § 2.3.7. The importance of the developmental trend is explained in § 2.3.8 from which is described the impact of reflective abstraction on the developmental trend, in § 2.3.9. The impact of the evolutionary trend in development is illustrated by explaining the importance of interiorization, in § 2.3.10. Equilibration and disequilibrium are explained in § 2.3.11. With a foundation in place, the relationships of Piaget and Mathematics is clarified in § 2.3.12. The development of symbols using the knowing circle is examined in § 2.3.13, followed by a worked example as a series of tasks WE1 – WE5 that is used in evaluating this research in § 2.3.14. A set of research conclusions is described in § 2.3.15.

2.3.1 Causality

A central issue is the notion of causality (causal relations). Piaget’s contention is that “no physical living structure can be adequately explained in physico-causal terms alone and this is even truer of the structure of mature intelligence” (Furth, 1969, p207). For instance, “nor does $2+7$ cause 9”; it is at the level of structural implication (equilibration) where “implication is a relation of logical necessity proper to intelligence and consciousness which cannot be reduced to the lawful relations of causality typical of physical manifestations” (Furth, 1969, p207). Piaget contends that operations are causal-temporal relations between constructive, operative knowing and symbol formation: thus the known object is real, only in the sense that it is present (has an existence) in the act of knowing. This act of knowing is not just the representational use of a symbol (Furth, 1969, p92).

An area where Piaget’s thinking goes most clearly against established views is in the area of figurative knowing (*figurative scheme*) and operative knowledge (*operative scheme*). Figurative knowing, defined as the sensorial content of an event (Furth, 1969, p47), is balanced by operative knowledge. Piaget defines operative knowledge as knowing that is transforming as a structuring kind of knowing action (Furth, 1969).

2.3.2 Assimilation and Accommodation

For Piaget “Intelligence is studied through the general process of adaptation and adaptation is studied as behavior” (Copeland, 1974, p9). The most relevant aspect of intelligence is not what the person does (as external action), but the rules or organization within the individual that control or govern the action, which is much broader than stimulus–response (S–R) approach (Furth, 1969, p74).

Piaget holds that behavior at all levels demonstrates aspects of structuring and he identifies that structuring with knowing (Furth, 1969, p45). In this light, *assimilation* is the tendency for an internal structure to draw external environmental events toward itself, contrasted with *accommodation*, which is the tendency for an internal structure to adapt itself to external environmental events (Furth, 1969, p45). Further, every scheme of assimilation “must be accommodated to the objects to which it is applied. Otherwise the assimilation would be deforming (assimilation pressure) or centered on the affectivity of the self, as is the case in symbolic play in which reality is modified according to the arbitrary desire of the moment” (Furth, 1969, p156; Piaget, 1962a).

2.3.3 Relationship of Assimilation to Figurative Scheme and Accommodation to Operative Schemes

Piaget refers to internal knowing (a person knows something) through active structures by different, yet equivalent psychological terms (Furth, 1969, p76). Piaget used the term figurative as the configurations or outlines “Gestalten” being the chief carriers of perceptive information (Furth, 1969, p101), hence figurative schemes are created by assimilation. Piaget further considered figurative knowledge (as a *figurative scheme*) as the sensorial content of an event (Furth, 1969, p47). This sensory stimulation (perception) is not knowledge unless there is a structured scheme prepared to assimilate it and accommodate to it. In Piagetian terms, “operations (*operative scheme*) are no more than interiorized actions” (Furth, 1969, p101). Operability is the essential, generalizable structuring aspect of intelligence insofar as knowing means constructing, transforming and incorporating (Furth, 1969, p263). This implies that action is an aspect of intelligence, at all periods, including sensory–motor intelligence. Further, internal knowing (as a concept) is both the figurative as well as the operative aspect (existing as reversible operations) which cannot and must not be separated (Furth, 1969, p102 and Furth, 1969, p263).

2.3.4 The Primacy of Action

Figurative schemes process the sensorial content of an event (the stimulus) as the perception, imitation and mental image aspects of cognition (Furth, 1969, p271). Since *operative schemes* consume these *figurative schemes*, so internal knowing is ultimately action based (Furth, 1969, p75; Furth, 1969, p92; Furth, 1969, p102 and Furth, 1969, p163; Furth, 1969, p156).

Before these operations (*operative schemes*) are formulated in language, there is a kind of logic of action coordination in the sensory motor stage. There is a sort of “generalizable action that prefigures classes and relations,” examples include seriation of blocks of decreasing size or number on a number line, but these structures are not capable of being represented, until processing has reached symbolic functioning (Piaget,

Chapter 2. Literature Review

1963). These structures (schemes) are pre-configurations of the later notions of conservation and reversibility. These operations are independent of language during the *concrete operations stage* and are tied as actions related to objects (Piaget, 1963 cited in Furth, 1969, p125). As an example of operational knowing, “The number system as an object of a person’s knowledge is not something that exists apart from his number operations (as *operational schemes*)” (Furth, 1969, p61). The two expressions, “number operations” (*operational schemes*) and “the known number system” are two sides of the same coin. An argument could be made that the number system, be represented as symbols and then written into books. It would thus appear to be separate from the knowing operation⁴² but unless there is somebody who can reproduce them, the numbers and words in the book would remain empty lines and strokes, bereft of any intellectual meaning (Furth, 1969, p61).

A visualization of the relationship of *figurative scheme* and *operative schemes* to *assimilation* and *accommodation* respectively can be defined (Furth, 1969, p78 and p102). If one assumes a form of reinforcement for building hierarchical networks, then a conceptual design for the knowing circle and its scheme process will be the following:

The Design of Assimilation and Accommodation with Figurative and Operative Schemes PMIAS AR-50.VSD

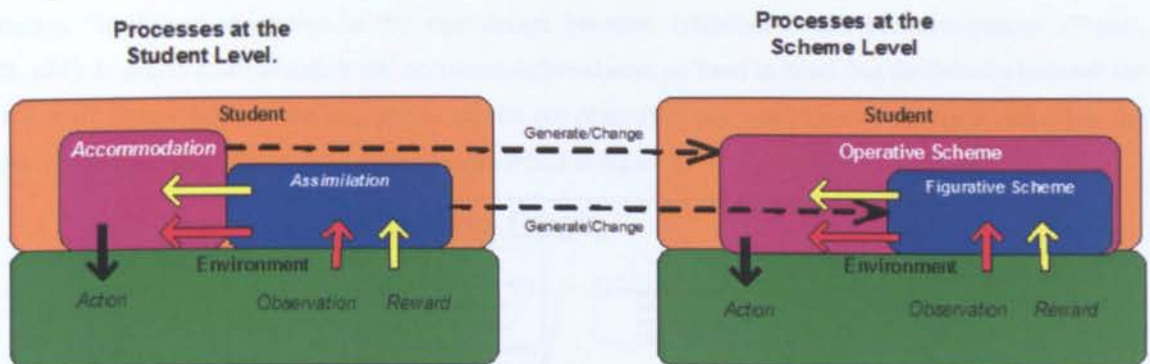


Figure 2–2 An interpretation of the processes of assimilation and accommodation showing how operative schemes are created through accommodation and figurative schemes through assimilation using an external reward provided by the environment. The reward mechanism is based on the actor–critic model (Sutton and Barto, 1988).

In figure 2–2, the observation and the reward associated with the action in the environment by the student are shown on two levels. First, processing by the student is through *assimilation* and *accommodation*. Second, the processing of the active operative scheme consumes the active *figurative scheme*. In this, the figurative scheme receives the reward from the environment, while the operative scheme performs the action on the environment. The processes at the student level are the innate scheme and primary reactions provided by the individual’s genetic heritage.

⁴² The separation of symbol from the knowing act is referred to as reification.

Chapter 2. Literature Review

Within the state machine, the figurative scheme responds to and processes inputs from the environment. The operative scheme takes actions associated with the configuration of the environment. Thus the state switching, which takes place at execution, efficiently utilizes the two levels of processing. This is seen most clearly in the evaluation of the architecture and design which describes these state changes in terms of assimilation and accommodation, using counting as a worked example (§ 5.2.5.1).

2.3.5 The Knowing Circle is an Abstract Process on Innate Scheme and Primary Reaction

Furth aligns the notion of primary reactions to Gestalts (Furth, 1969, p146). Specifically, gestalts are not equated to schemes and confirms that “neither perception nor assimilation leads to a more direct contact between subject and an object...objectivity is constructed progressively from the indissociable interaction of assimilation and accommodation” (Furth, 1969, p147). These innate schemes are used to develop the other schemes through the process of the knowing circle and are seen as the innate gestalts that are not affected by development, remaining stable over time.

Sometimes there is a primacy of *accommodation*, sometimes a primacy of *assimilation*. When there is a balance of the two, this is *equilibrium* (equilibration). Piaget saw equilibration, as the optimal path of adaptation, “Intelligent adaptation is the equilibrium between assimilation and accommodation” (Piaget, 1962b, p84). In practice, *assimilation* and *accommodation* always go hand in hand, but the balance between the two will shift. Piaget refers to the two partial aspects (or phases) of any one behavior pattern (conduct) as the process of *assimilation* and *accommodation*, as described in figure 2–3.

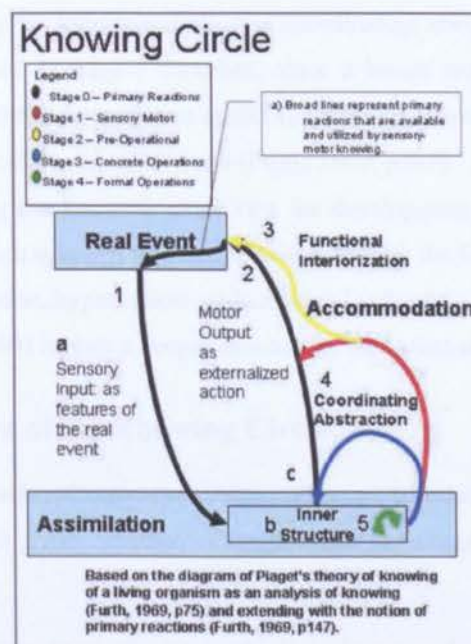


Figure 2–3 A depiction of Piaget’s knowing circle. There is a constant dialectic interchange with the environment, first accommodation then assimilation. These processes deepen over time, as the child progresses through developmental stages and matures.

Chapter 2. Literature Review

In figure 2–3, two principal processes are at work:

- Environment → organism as *assimilation* and
- Organism → environment as *accommodation*.

Piaget's knowing circle consists of a "main control loop," which undergoes progressive development i.e., the learning behavior changes over time. This development is referred to as being embryogenetic, which exhibits the various developmental functions.

The knowing circle assimilates (incorporates) the real event into the inner structure (scheme) and at the same time accommodates their inner structure to the particular features of the real event (Furth, 1969, p75). The process is progressive with increasing *interiorization* and reflection (on the schemes created through assimilation–accommodation from the *innate structures* and *primary reactions*).

Piaget observed four other stages of functional *interiorization*⁴³ over the lifetime of an individual:

- 1) Before the operations are formulated in language, there is a kind of logic of *action coordination* (Furth, 1969, p125). Only through closing the knowing circle (via externalized action of the *semiotic function*) is stage 1 complete. This development of a practical logic of seriation, hierarchical linking, ordering and generalization is referred to by Furth, as *action coordination* and includes the conservation of the *permanent object invariant* (Furth, 1969, p248).
- 2) Only through closing the knowing circle (via functional interiorization and the development of the *symbolic function*) is stage 2 complete since actions are no longer tied to sensory–motor knowing and are related to the scheme of the permanent object, though the coordinated external action may remain an accompanying phenomenon (Furth, 1969, p98).
- 3) Only through closing the knowing circle (via coordinating abstraction and the development of the *structural function*) is stage 3 complete, since it builds symbols using a figurative aspect (sensory motor event) and an operative aspect (significate). These symbolic relationships are the invariants of the *concrete operations stage* (Furth, 1969, p207).
- 4) Only through closing the knowing circle (via the development of the *operational function*) is stage 4 complete which supports formal operations through the further development of the logics of implication, exclusion, hypothetical–deductive and syllogistic reasoning (Furth, 1969, p66).

A close reading of Furth (Furth, 1969) reveals a deeper structure of the innate schemes and primary reactions.

2.3.6 The Deeper Structure of the Knowing Circle

Piaget asks, "[how] three levels of memory – recognition, reconstruction and evocation– fit in with developmental stages" (Furth, 1969, p161). Further, "The figurative aspects of memory (perception, imitation

⁴³ *Interiorization* is considered to be the internal construction of a model of the environment. For instance a value from the environment is *converted to an internal value* to be used in both figurative and operative schemes.

Chapter 2. Literature Review

and images)⁴⁴ (Furth, 1969, p162) and “The schemes of memory are viewed as identical to the general operative schemes of intelligence... all figurative knowledge is part of and controlled by some system of operative schemes” (Furth, 1969, p163). This approach is clarified by Furth in the quote, “with reference to figurative aspects...of the three types of memory; perceptual (*perception*) for *recognition*, imitative (*imitation*) for *reconstruction*, imaginable or linguistic (*mental image*) for *evocation*” (Furth, 1969, p154).

In figure 2–4, these structures are related to the abstract classes of assimilation and accommodation.

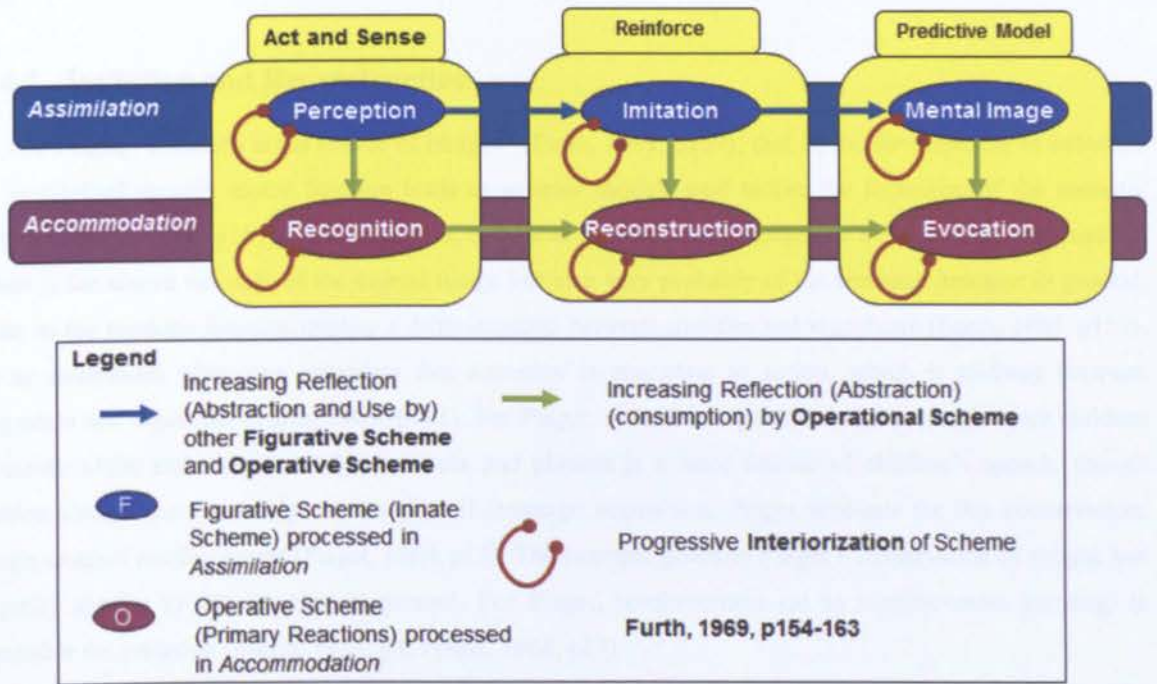


Figure 2–4 A model of Piaget’s innate scheme and primary reactions, which are the abstract processes of assimilation and accommodation separated into act and sense, reinforce and predictive model (Furth, 1969, p162).

These abstract classes are named act / sense, reinforce (to indicate the learning style) and predictive model (to indicate the impact of planning). Each set is described in terms of Piaget’s work in the following sections.

2.3.6.1 Perception and Recognition

Piaget describes *perception* as “a balanced process including centration (with the senses focusing on a given point in the sensory field) and decentration (coordinating movements guided by perceptual schemes)” (Furth, 1969, p211). Perception thus illustrates an elementary state of perceptual equilibration where regulating activities (decentration) tend to compensate for the deforming effects of single centration (Furth, 1969, p211). These “Perceptual structures (*perception*) as instruments of recognition” (Furth, 1969, p159), are interpreted to mean that these *figurative schemes of perception* are embodied within the *operative schemes of recognition*.

⁴⁴ We equate images with mental images.

Chapter 2. Literature Review

In all sensory–motor knowing, “recognition occurs in the act of reacting meaningfully to a given object and requires no representational presence of any sort. For Piaget, infants of the sensory–motor stage are incapable of representational production and can have no mental images as understood by the term” (Furth, 1969, p150).

Recognition is an intrinsic part of every sensory–motor habit and this recognition is different from perception (Furth, 1969, p160), since perception is used by the operative scheme, *recognition* (Furth, 1969, p158).

2.3.6.2 Imitation and Reconstruction

For Piaget, “imitation is the source of images” (Furth, 1969, p159); that is, the development of *imitation* as a specialized sensory motor function leads to exterior models well before the formation of the semiotic function⁴⁵ (Furth, 1969, p151). Thus, *imitation*, once it is capable of functioning in a deferred and internalized manner is the source not only of the mental image but also very probably of the semiotic function in general, insofar as the semiotic function implies a differentiation between signifier and significate (Furth, 1969, p157). This is understood when one considers that *imitation* is *evocation* in action, which is midway between *recognition* and *evocation* (Furth, 1969, p151). For Piaget, *imitation* is positive reinforcement, since children do imitate adults and repetition of new words and phrases is a basic feature of children’s speech, though imitation alone cannot possibly account for all language acquisition. Piaget accounts for this conservation, through external reinforcement (Piaget, 1964, p13). The example given by Piaget’s conservation of weight, but it equally applies to conservation in general. For Piaget, reinforcement (as in reinforcement learning) is responsible for imitation (Piaget, 1954, p4; Piaget, 1964, p13).

Further, Piaget says that “Reconstructive memory is based on imitation” (Furth, 1969, p158). The schemes of reconstructive memory have an increasing capacity for accommodation and hence figurative power (Furth, 1969, p163). Further, a memory reconstruction is different from an imitation (Furth, 1969, p160).

2.3.6.3 Mental Image and Evocation

The derivation of *mental images* is “from *perception* to the image (*mental image*) by way of *imitation*” (Furth, 1969, p160). In this, a *mental image* is the internal representation of an external event. The image is one of the products of the symbolic function⁴⁶, hence of intelligence in its total functioning; it is not a mere trace from passive perception (Furth, 1969, p261). Furth further clarifies Piaget’s theory of images as “images are partially derived from what the child understands or misunderstands” (Furth, 1969, p142–143). The notion of an image to a pre–operational child is as static as the notion of space using topological rules that predominate

⁴⁵ It could be argued in this research that the semiotic function is the instantiation of the imitation / reconstruction process.

⁴⁶ It is argued that the mental image and evocation is the symbolic function, which is the key issue in development of symbols.

Chapter 2. Literature Review

over Euclidean metrics i.e., the developing child does not have conservation. Images of spatial relations become adequate precisely at the point at which children in the *concrete operations stage* have mastered the horizontal and vertical system of spatial co-ordinates i.e., when they have sufficiently increased their operative knowledge such that they can bring this to bear on their figurative knowledge. This is further clarified, “Images (mental Image) as instruments of *evocation*” (Furth, 1969, p159).

Further, “Evocative memory is based on the image (mental image)” (Furth, 1969, p158) and the “Specific function of memory...consists in the evocation of a particular past; this evocation is specifically related to the accommodative activity of knowing focused on the figural aspect of a particular event and temporally located at a certain point in time” (Furth, 1969, p152). This is understood to mean that evocation (as a type of memory) is a memory of a past event as belonging to one’s past – it is an accommodated knowing, which may or may not have figurative aspect.

2.3.6.4 Summary

What is clear from this short summary is that Piaget’s model of assimilation and accommodation can be re-interpreted into a simpler, generative structure (Furth, 1969).

2.3.7 Relationship of Piagetian Model to Biological Processing

Piaget’s model of assimilation and accommodation as provided by Furth (Figure 2.4) is strikingly similar, to the model of *Drosophila* (Miesenböck, 2008, p52; Shang, Claridge-Chang, Sjulson, Pypaert and Miesenböck, 2007, p601). Further, this model is consistent with the neuronal processing by Albus (Albus, 2008; Albus, 2010) and Granger (Granger, 2006a; Granger 2006b). These features are summarized in table 2-1:

Table 2-1 This table defines the traceability matrix of relationships between the features exposed through biology and evolution to the concepts in Piagetian theory.

Features as exposed through biology and evolution	The concept in Piagetian Theory
Thalamic loop (Albus, 2008; Albus, 2010b, p193; Granger, 2006a; Granger 2006b)	Knowing circle (Furth, 1969, p147)
Event Hierarchies (Albus, 2008; Albus, 2010a and 2010b)	Accommodation as an abstract process (Furth, 1969)
Receptive Field Hierarchies (Albus, 2008; Albus, 2010b, p193)	Assimilation (Furth, 1969)
Model of <i>Drosophila</i> (Miesenböck, 2008, p52; Shang, Claridge-Chang, Sjulson, Pypaert and Miesenböck, 2007, p601) and Straital System (Granger, 2006a; Granger, 2006b).	Knowing Circle (Furth, 1969, p147)

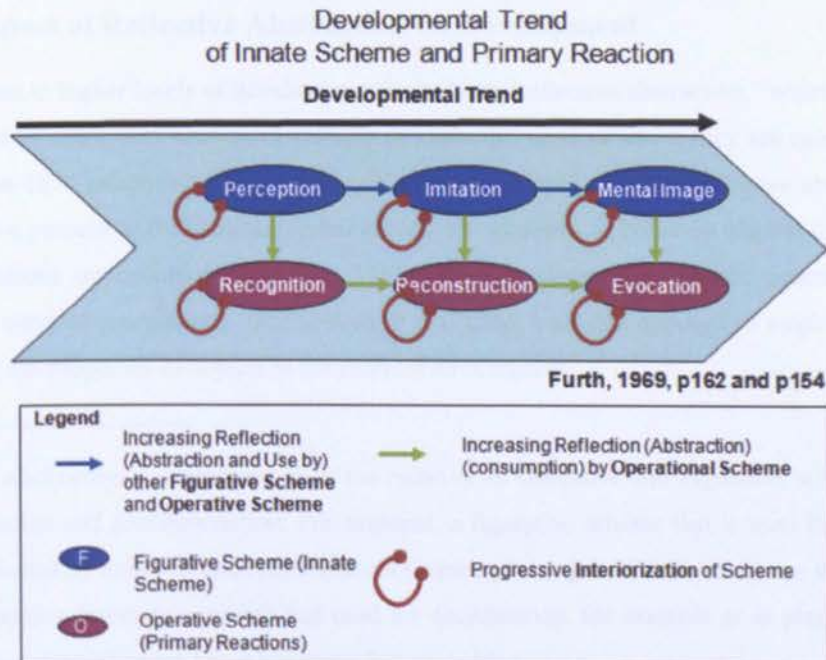
Chapter 2. Literature Review

Features as exposed through biology and evolution	The concept in Piagetian Theory
1) Cortex → matrisome projections (acting)	1) Act (Furth, 1969, p154–163)
2) Sense	2) Sense (Furth, 1969, p154–163)
3) SNc dopamine (DA) projections to both matrisomes and striosomes (learning through reinforcement)	3) Learn (Furth, 1969, p154–163)
4) TAN projections to matrisomes (exploration) which relates to predictive modeling (planning)	4) Plan (Furth, 1969, p261)
CCU (Albus, 2010b)	The structure of a scheme (Furth, 1969; Copeland, 1974) as a HFSA.
Dopamine Reward Processing (Cannon and Bseikri, 2004; Schultz, 1997; Schultz, 2000)	Reinforcement through the processes of imitation, which constructs schemes (Piaget, 1954, p4; Piaget, 1964, p13).

The implication of this is that there is a biological basis for Piaget's work.

2.3.8 The Importance of Developmental Trend

According to Piaget, the progressive interiorization of a scheme goes hand in hand with an increase in mobility – where mobility is the range of a scheme's potential applications within the totality of available schemes (Furth, 1969, p162 and 154). The development trend is a progression of each scheme through the knowing circle. This process is visualized in figure 2–5.



Chapter 2. Literature Review

Figure 2–5 A model of Piaget’s developmental trend that shows the impact of increasing reflective abstraction and progressive interiorization on the abstract processes of assimilation and accommodation (Furth, 1969, p162 and 154).

This developmental trend (Furth, 1969, p162 and p154) performs three key functions:

- 1) First, it increases the coordination and use of existing figurative schemes by other *figurative* and *operative schemes*. *Reflective abstraction*⁴⁷ enhances the inter–scheme communication, by providing the interfaces between accommodated schemes (Piaget, 1983 p125). The increasing depth and complexity of schemes e.g., the capability to increase perception through the reuse of existing schemes, can be partly explained in the increase of M–Capacity (Pascual–Leone, 1970, p336)
- 2) Second, it increases the consumption (and therefore abstraction, through generalization) of *figurative schemes* by *operative schemes*. Examples include from immediate reaction to less immediate reaction to events, i.e., some events are deliberately ignored.
- 3) Third, it increases the progressive *interiorization* of a *figurative scheme* and an *operative scheme*, by optimizing it. This is understood, to be an increase from less predictable to more predictable behavior, through an increase in the time spent in the use of a predictive model.

As an example: a four year old child may be able to work out slowly that $5 + 3 = 8$ by means of a scheme from the pre–operational stage of development that cannot, by itself, be applied to the problem of $3 + 5 = 8$. Piaget explains this, as “the child’s scheme of addition is not mobile enough to disregard the order of the elements and conserve itself as a stable system” (Furth, 1969, p62). In this example, the child has not fully developed a reversible scheme, and it needs further refinement through interiorization.

2.3.9 The Impact of Reflective Abstraction on Development

The movement to higher levels of development depends on “reflective abstraction,” which means coming to know properties of one’s own actions, or coming to know the ways in which they are coordinated (Furth, 1969, p259). In his 1970 essay, titled simply “Piaget’s theory,” Piaget says that reflective abstraction “is the general constructive process of mathematics: it has served, for example, to construct algebra out of arithmetic, as a set of operations on operations” (Piaget, 1983 p125). It abstracts from and generalizes over, the individual’s prior ways of coordinating their actions. It is distinct from and opposed to empirical abstraction, which ranges over the properties of objects in the external environment.

⁴⁷ *Reflective abstraction* is considered to be the reuse of an Operative and Figurative scheme by another process of assimilation and accommodation. For example, a figurative scheme that is used for sensory motor perception is consumed by imitation and reinforcement (learning) changes this scheme. Later this same scheme can be used in planning (predictive model) and used for coordination, for example as in play. Each time, the scheme is becoming more *abstract*, i.e., it has more features added.

Chapter 2. Literature Review

In considering *reflective abstraction*, an example is poignant: “Why is multiplication harder to understand than addition? “ They are both operations on numbers, after all. Piaget’s analysis was that to understand multiplication it is not enough to center your thinking “on the objects that are being put together with other objects and thus on the result of this union. Multiplication also involves isolating the number of times that the objects are being brought together; it means enumerating operations as such, not just the results of those operations (i.e., the number of objects transferred each time)” (Piaget, 2000b, p57). Further, any system would need to support the development of invariants since to understand Piaget’s terms, it is necessary to distinguish between the external transformation and, the external instability of the physical attribute from the internal process of thinking (Furth, 1969, p215). Hence, there is a need to understand reflection.

Piaget refers to reflection and in particular reflective abstraction as the notion of reflection on mental operations using abstraction and generalization. Some researchers refer to this process as interiorization – where students solve problems presented to them with the existing schema using actions in their imaginations – where feedback from the coordination or operational activities to the interior organization which enables it to “reflect” on the general form of the activities. This formal, reflecting abstraction is the principal growth of intelligence as general, logical knowledge (Furth, 1969, p259).

In the developmental trend, there is increasing reflective abstraction (as consumption of schemes by other schemes) and lesser immediacy leading to generalization of action. This process of reflective abstraction generates more interconnected and complex processes, and so resolves LP6. Reflective abstraction is fundamental to the development of number–sense is a conclusion reached in this present research thesis.

2.3.10 Impact of Evolutionary Trend on Development through Interiorization

For Piaget, the evolutionary trend towards a lesser degree of immediacy and specificity in behavior becomes in intellectual development a process of increased reflection. This is a turning inwards or an interiorization of action that changes coordinated external actions into systems of interior, reversible operations. One important comment on Piaget’s work on evolutionary development is that “what is transmitted through the genes cannot go beyond sensory motor intelligence because gene–transmitted structures are tied to specific organs and are therefore inseparably linked to the organs and the sensory content to which they selectively respond. This stable operative knowledge is predicated upon an increased freedom of schemes from subjective actions and specific content. Such knowledge cannot be handed through psychological devices alone. Not by chance are humans born more poorly equipped with innate mechanisms than any other species. This fact enables humans to exercise schemes of adaptive activity with a freedom from the specific and the immediate like no other animal. But at the same time it forces each individual to go through the process of development” (Furth, 1969, p232). The child’s activity serves him both as a source of further progress and as an obstacle to overcome (Furth, 1969, p232). Thus, the basic mechanism of learning is not different from the equilibration process of the whole developing intelligence (Furth, 1969, p234).

This interiorization or “turning inwards” is critical to development, and as a feature, is often mistaken for assimilation and accommodation, as seen in the work of (McClelland, 1995; Shultz, Schmidt, Buckingham and

Mareschal, 1995). The process of interiorization as described by Furth (Furth, 1969, p65, p209, p259; Piaget, 1983, p125; Piaget, 2000b, p57) is provided in figure 2–6.

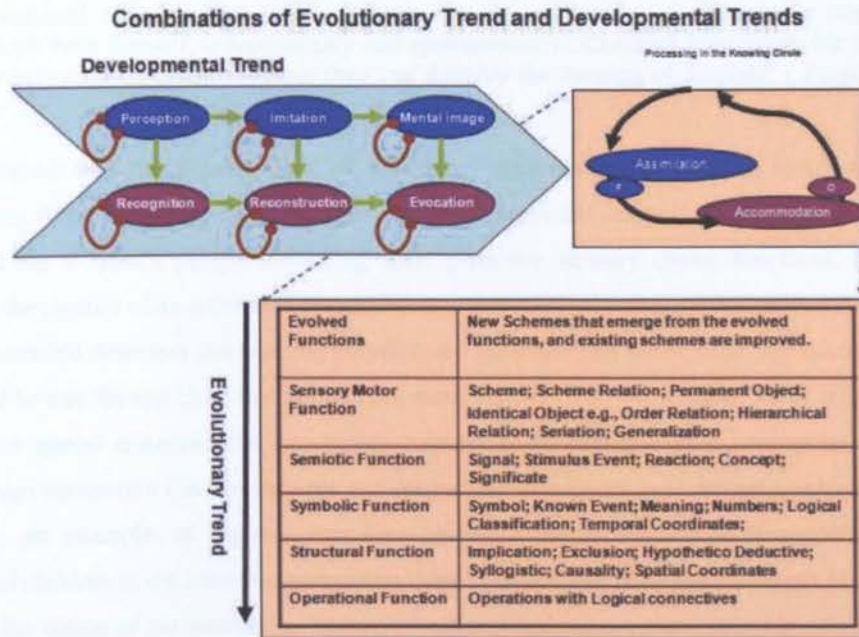


Figure 2–6 A depiction of the impact of Piaget’s evolutionary trend and developmental trend (Furth, 1969, Piaget 1983) as it affects the learning process to develop more complex schemes, and so resolve LP6.

In the evolutionary trend, there is increasing interiorization (as the internalization of sensory data by schemes) and lesser immediacy leading to generalization of action. This process of interiorization generates more interconnected and complex processes, by allowing their networks to use the internalized values and so resolves LP6. Interiorization is fundamental to the development of number–sense, which is a conclusion reached in this research.

2.3.11 The Impact of Equilibration and Disequilibrium

Equilibration can be viewed as a compensatory response to these biological trends (Furth, 1969, p209). Equilibration is the inner regulating factor, which in development leads to an increasing dissociation of the general forms of structured behavior from particular content i.e., the building up of more advanced structures (Furth, 1969, p209).

Instability in this sense occurs when, for instance, the organism is acting / sensing and cannot process information from the environment. This instability forces the individual to change its processing state into learning or planning. In learning and planning, new schemes are generated that can handle the “input data” from the internal or external environment.

2.3.12 The Impact of Equilibration and Disequilibrium Piaget and Mathematics

“It is a great mistake to suppose that a child acquires the notion of number and other mathematical concepts just from teaching. On the contrary to a remarkable degree, he develops them himself, independently and spontaneously...Children must grasp the principle of conservation of quantity before they can develop the concept of number” (Piaget, 1953, p74).

Piaget argued that the development of set theory and mathematics is an abstraction that humans developed using formal operations (Furth, 1969, p66). This present research focuses on the development of mathematics from a child’s perspective using their primitive sensory motor functions, thus, there is no application of the rigidity of an axiomatic system.

Von Glasersfeld describes the learning paradox, as “how can you know what you do not already know,” and this would be true for any child that approaches mathematics (Von Glasersfeld, 1998, p9). To the child, the knowledge that spatial distance does not define number is an example of a conceptual system, one that develops through maturation (see for instance the bead problem § figure 1–1). Piaget’s work on the conception of number is an example of rigorous empirical methods being applied to a cognitive psychological investigation of children in the *concrete operations stage* of development (Ages 7 through 11).

Further, the notion of permanent (or “stable”) mathematical concepts are rooted in perception and of trial and error processing that is finally solidified as logical and reversible mental operations (Piaget, 1952). This research operates within the dichotomy of empiricism and rationalism which considers whether number conceptions arise from experiencing number in the external world or, instead, from reasoning using thought. Piaget suggests a general shift from the former to the latter in the child’s construction of number. That is to say, number first manifests as a qualitative perceptual intuition (on the figurative plane, as a figurative scheme) of small numbers (up to about cardinal 5) but cannot progress further – one does not empirically experience all integers on the real number line, after all. It follows that the construction of number cannot remain within the field of perceptual intuition and can therefore be completed only on the operational plane – as an operational scheme, that embodies action (Piaget, 1952, p154). This research thesis demonstrates this as action on a number line. For Piaget, these schemes are subjective operators of the meta-subject because they determine and operate upon both the content of the subject’s experiences and their performance (Piaget and Morf, 1958 p 86 cited in Pascual-Leone et al, 1978).

In the development of mathematical ability (Copeland, 1974), Copeland stresses the importance of developmental stages for each relation being not specifically tied to the child’s developmental stage. Congruent with this are the development of various relations (Copeland, 1974, see also § C Piagetian Models and Mathematics for further clarification). The capability for the system to generate its own logical system is a critical aspect and underlies any development of language, which includes conservation of number, conservation of measurement/distance and length, equivalence, counting, subdivision and substitution (Copeland, 1974).

2.3.13 Development of Symbols

For Piaget, “The representation theory of knowledge, reduces knowledge to a signal reaction even when it calls its mediators images, words, or symbols and it leaves unexplained the active relation of the knowing person to the representation which would be inherent in any true symbol behavior” (Furth, 1969, p93). Further, “A symbol is thus an observable state that derives from accommodating imitation and as a symbol, represents something other than itself” (Furth, 1969, p102). This process of relations is a critical feature for Piaget. Specifically, there are a set of relations of knowing that Piaget defines which are often referred to in other sources as causal relations, representational relational (representation relation) and intentionality. These differ in content to the representational theory of knowing as presented by other researchers (Furth, 1969, p78). For Piaget, the “relation of knower to representation as well as the relation of material cause and effect lies at the heart of Piaget’s entire theory” (Furth, 1969, p93).

In the representation theory of knowing, “we have a causal–associative relation of thing to sign and vice versa” (Furth, 1969, p95). This is balanced in Piaget’s theory of operative knowledge (operative theory of meaning), which is unique in “dispensing with a meditational representation as far as the essential aspect of critical, objective knowing is concerned.” With this approach, the words “representation” or “internalization” are not used in the developmental stages (Furth, 1969, p75).

To support internal knowing, Piaget employs representation in the active sense and relates it to the *symbolic function* (symbol or linguistic sign) or *semiotic function* of intelligence. For Furth, Piaget’s theory is never a matter of representation (Furth, 1969, p77). Piaget also distinguishes a representational sign (an imaginable symbol) from a linguistic sign. For Piaget, every symbol has two differentiated aspects. First, a figurative aspect⁴⁸ (the plane of representation or representative plan) that refers to some sensory or motoric event in itself. Second, an operative aspect (the plane of operative or operative plane⁴⁹) that refers to meaning i.e., its significate. A concept stays fully within the plane of operative knowing. Each significate is thus a concept (Furth, 1969, p78).

A symbol is experienced as some figurative thing that derives developmentally through the knowing circle. Further, the direct significate of a symbol is a knowing or concept (an operative scheme) and it is only through this concept that it can be said to represent the external thing. The development of symbols is the final stage in a development process that begins with action. A symbol is thus an observable state that derives from accommodating, imitation and, as a symbol, represents something other than itself (Furth, 1969, p102). In illustrating the role of symbols and to point out that operations are real actions, Piaget uses the example in figure 2–7 (Furth, 1969, p104):

$$2 + 3 = 5$$

Figure 2–7 A simple depiction of Piaget’s number example (Furth, 1969, p104)

⁴⁸ The figurative aspect and plane of representation is referred to as the figurative scheme.

⁴⁹ The operative plane is referred to as the *operative scheme*.

Chapter 2. Literature Review

When the operation of addition is performed on symbols, the specific symbols $2,3,5,+ =$ constitute the figurative aspect of thinking (figurative scheme) and are tied to the representation and communication of specific states. The reason it is real, is that transforming operations are manifestations of a thinking process constructed by the thinking subject (Furth, 1969, p104).

Furth further clarifies Piaget's position on this symbolic function as "By being outside the thinking process, a symbolic instrument (symbol) can become a means of communication between persons and thus forms part of the external communication and thus forms part of the external symbolic environment to which a person responds. Knowing as a living process cannot be experienced or observed in itself, but only inferred from its manifestations; a symbol, in contrast, has an integral part that is external to thinking" (Furth, 1969, p110). Thus, a sign is only as good as the knowing structure, which uses it since "the act of comprehending a symbol is similar to the operative process by which we know the referent" (Furth, 1969, p111). A symbol is generated through a combination of the processes of assimilation, accommodation (as imitation and reconstruction) and influenced by equilibration with the evolutionary and developmental trends. In the development of symbols, the system must allow the relationship of actions and perceptions together (Furth, 1969, p78; Furth, 1969, p92) in different but related reversible operations, to form a "concept" (Piaget, 1963 cited in Furth, 1969, p125).

Though important to Piaget, the notion of play (Furth, 1969, p96; Piaget, 1962a), is not covered in this research.

2.3.14 Worked Example

In this section, a worked example using a number line to describe the levels of a partial resolution to the learning paradox (WE1 – WE5), is developed. It is assumed that Piaget's empirical observations are correct and it is considered that there are both intrinsic and extrinsic rewards that allow the development of a scheme, as well as various levels of conservation, along with evolutionary and developmental trends. The positions of Piaget and Fodor at each of the levels of the development of mathematical relations are included (Copeland, 1974).

Piaget described three stages in the development of *concepts* (Copeland, 1974, p84): (1) no understanding. (2) partial understanding. (3) complete understanding. As the child develops a *concept*, the child's ability to develop and use the *concept* passes through each stage of understanding, until finally it has mastery of that concept. It is anticipated that the simulation would develop *concepts* in a similar way, and thus those *relations* normally observed in the development of a child's understanding of number, namely: equality, addition (counting), subtraction, "less than" and "greater than" (Fennell and Landis, 1994; Liebeck, 1984), would be observable in the executing simulation, in similar stages. It is suspected that all these *relations* (as behaviors) would be observable in WE5. However, if we assume that (i) the aim of the simulation is a permanence of equivalence relations (Copeland, 1974, p94) and (ii) that addition (counting) is used by the relations of "less than" and "greater than" then addition is anticipated to appear earlier than WE5. Further, the Piagetian *concepts* of spatial relations (proximity, separation, order and surrounding) and seriation upon which

Chapter 2. Literature Review

these *relations* (addition, subtraction, “greater than”, “less than”) are based, are anticipated to appear even earlier, potentially in **WE3**. This is expected because genetic epistemology predicts the emergence of more complex schemes from simpler forms⁵⁰. Since the simulation exists within a number-line world, the development of these *relations* must occur as some function of the interaction with the environment and this interaction is understood to occur using a Piagetian model. Thus, first, in early development, the system must treat the number line and its components as continuous values, being based on perceptual shape, not value. The differentiation into units comes later and is used in counting (Piaget, 1965, p80 cited in Copeland, 1974, p91). Second, Piaget’s genetic epistemology defines a series of relations that are observed to occur, with stage like variation throughout the life of the child. As the child matures, these relations become increasingly interiorized and generalized. At any point in time, these relations are combined together using the processes of assimilation, accommodation and equilibration to produce the observable behavior.

Since all constraints (**LP1 – LP7**) apply throughout the life of the simulation, the level in which the constraint is most clearly observed to have been adhered to is identified.

2.3.14.1 WE1 – Base Level

In this level, the student has an interface that provides a mechanism to change the pen state (either pen up or pen down) as well as to move forward and back. The starting point is a discrete quantity such as a point on a line where the distance property is initially ignored.



Figure 2–8 In WE1 – Base Level, there is emergence of the capability to act and sense in a number line world. The simulation learns to move, change its penState and use its innate schemes and primary reactions to adapt to the continuous number line environment.

- 1) For Piaget, it is assumed that a student has a set of innate schemes and primary reactions that enable the student to move along a continuous number line to perceive, recognize, imitate, reconstruct, use mental images and evoke those images. An operative scheme is learnt that consumes the figurative aspects and supports action on the number line by the student’s pen. It supports the developing spatial relations of topology (proximity, enclosure, separation and order) then the Euclidian properties (Copeland, 1974, p91, p214, p226 and p232).
- 2) For Fodor, the key is to generate an appropriate descriptive response to a given situation – but this is common to most of the living world, so is not considered to be a learning paradox since a case can be developed that attaches properties to observations.
- 3) In the example presented, “move” is emergent as the simulation achieves a steady state of fluency of its movement and the use of its penState. At this level the simulation is operating in

⁵⁰ In the following description, the level within the worked example (**WE1 – WE5**) where observations of the *relations* are anticipated, are described.

Chapter 2. Literature Review

the Piagetian Sensory Motor stage and so number-sense would not be directly observable, yet the schemes it is building lay the foundation for the future.

- 4) In this level, adherence to **LP3** and **LP7** should be observed as: (i) **LP3**, the system should work unaided on its goals. (ii) **LP7**, the responses from the environment should include elements of noise. Further, the simulation should show that it takes opportunities, as they are presented.

2.3.14.2 WE2 – Constrained Level

In this level, the continuous movement that the student perceives is prevented by the system, which leads to differentiation of a new type of scheme, the permanent object.



Figure 2–9 In WE2 – Constrained Level, there is emergence of “Stop” action. This action, not inherent in the innate schemes and primary reactions, is perceived as repeated constrained movement in a number line world. This constrained movement is the precursor for unitized movements.

- 1) For Piaget, emergence of a “stop” scheme is clearly at a different level. Piaget explains this in terms of learning and it is this feature that clearly must occur. Reversibility (Copeland, 1974, p90), is also required and would need to be tempered with the use of the penState, such that the student develops a scheme that can begin and stop at the same place. Each stopping point would essentially be a different value. It must support the development of the relations of conservation of measurement, length and distance (Copeland, 1974, p248, p267 and p253) and in many ways, the implied point must be observed.
- 2) For Fodor, the key is to account for a higher-level descriptive framework “on top of” the lower level framework that has any meaning above **Base Level**. This is treated as a modification of a descriptive lower level. The learning paradox starts with continuous movement which is then classified into discrete “terminal” points, which in turn are differentiated by distance. For this to occur, the student has initially to learn how to stop and in the same space, move left, stop of its own accord, as well as move right without being constrained.
- 3) In the example presented, “Stop” is an observation of the emergent behavior of the simulation. At this level the simulation is still operating in the Piagetian Sensory Motor stage and so number-sense would not be directly observable. Yet, the appearance of the stop action along with repeated movement is the precursor of number-sense.
- 4) In this level, adherence to **LP6** should be observed as: (i) **LP6**, there should be evidence that the simulation can extend its existing learning process.

2.3.14.3 WE3 – Differentiated Level – Object

The lines and points on the number line are classified in terms of lengths, and then with these lengths, comparisons can be made based on classifications and order. Much of the permanence of objects is based on the prediction that the object will be present.



Figure 2–10 In WE3 – Differentiated Level – Object, there is observation of the emergence of differentiated objects as sets of coordinated movements in the number line world. The repeated observation by the simulation is necessary in order to provide the capability to derive sets of values.

- 1) For Piaget, at this level, the appearance of unit lengths should agree with comparative human studies (Fennell and Landis, 1994; Liebeck, 1984). When the lengths of movement of the operative scheme is reinforced to a *unit*, the scheme will have developed object permanence (Furth, 1969, p125). It is also the first appearance of the developing relations of equivalence, intersection, reversibility, inclusion, transitivity and seriation (Copeland, 1974, p84 and p85–87, p59, p90–92, p120, p108 and p170, p79–80) leading to a development of numbers. It is anticipated that the simulation would be, in Piagetian terms, in the border region between sensory motor and the pre-operational stages.
- 2) For Fodor, it is at the simplest “next” level of differentiated classification (forward and back) from where you start. This is an easy approach because any problem that needs to be solved within this framework can be solved by enumeration of the cases that are currently known.
- 3) In the example presented, “1” is the observed behavior of the simulation as it moves “unit” values on the number line. However, it would not be possible to observe number-sense, because it had not yet internalized the concept of units. It is expected that the bead problem (§ figure 1-1) would be directly observable (or indirectly within the simulation itself), since the simulation would only be capable of differentiating objects - allowing it to manipulate its environment using these objects - there would be no value system upon which they could be mapped, and hence no notion of equivalence. The value of “1” in this system, is arbitrary.
- 4) In this level, adherence to **LP5** should be observed as: (i) **LP5**, there should be evidence of the system reusing its knowledge, and incrementally changing it.

2.3.14.4 WE4 – Hierarchical Level – Segmentation into Units

At this level, there is an appearance of seriation and of ordering in general, such that points are made contiguous and there are progressions of lines and relationships. Much, if not all of these relationships are based on developing the concepts of equivalence and of truth. Truth is understood in this aspect to be prediction of perception of both external and of internal relationships.



Figure 2-11 In WE4 – Hierarchical Level, there is emergence of hierarchies. This hierarchy supports the necessary relationships of a value system and it is with this internally constructed value system that “units” emerge. The appearance of, and use of “connected units” in the value system, is the precursor for ordering, seriation and of equivalence.

- 1) For Piaget, at this level, operative schemes should appear to add onto an existing line, rather than going back to the start. This would indicate a progression, such as $3+1$ or $1+1+2$ and show that lower level schemes are being consumed. The underlying relationships of signals and significate become important to differentiate how symbols can eventually be formed. It is the appearance of schemes for conservation of number, counting (Copeland, 1974, p83) combined with schemes of reversibility, transitivity, equivalence and seriation that populate the hierarchy.
- 2) For Fodor, at the hierarchical level, *general laws*, need to be construed (and learnt) in terms of the lower levels (**Base Level, Constrained Level, Differentiated Level**). For instance, classification of the lines and points into meaningful units and series.
- 3) In the example presented “1,1”, the values of one unit movement on the number line, would be followed by a similar movement. At this level, the simulation should be observed to have acquired the concepts of seriation and ordering through the reuse of the units concept it had identified earlier. To achieve this would require the emergence of hierarchical structures to support the internalized concept of sets. It should also be observed that the bead problem (§ figure 1-1) is no longer a problem, since the simulation would have acquired the notion of units and with this, conservation of number.
- 4) In this level, adherence to LP2 and LP4 should be observed as: (i) LP4, the actions on the number-line should most clearly mirror those of children. (ii) LP2, there should be evidence of hierarchies being developed e.g., sets of sets of repeated patterns.

2.3.14.5 WE5 – General Level – Relationships of Unit Values

At this level, there should be comparisons of values in sets and series and general conformance to values and predictions.

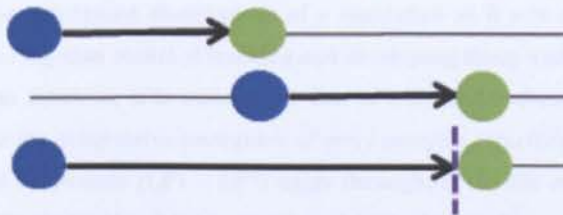


Figure 2-12 In WE5 – General Level, there is observation of more complex behavior by the simulation including the construction of hierarchical relationships with earlier constructed schemes. In this case, the “units” developed

Chapter 2. Literature Review

earlier. It is with the repeated observation of these relations, that the simulation has emergence of concepts such as equivalence, “less than” and “greater than”.





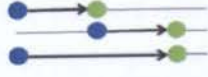
- 1) For Piaget, within these general laws, the expression of equivalence, truth, and logical connectives such as add, subtract, need to be considered. This would be possible if the evolutionary trend of stage like variations, which takes imaging of points and lines, is converted into the transformation itself using a process that acts on the processes of assimilation and accommodation (§ figure 2–1, the bead problem, which is resolved through reflective abstraction). The appearance of part–whole relationships should be expected, along with the combinations of values. The external behavior may evoke the response 1+1, but that is merely within the representational system of the observer. At this level, there should be the appearance of the commutative property, along with multiplication and division and more appearance of introspection, classification with subdivision and substitution (Copeland, 1974, p123, p146, p184, p51–60, p80, p159 and p252).
- 2) For Fodor, at the **General Level** “better general laws” need to be learnt in terms of the lower levels (**Base Level, Constrained Level, Differentiated Level, Hierarchical Level**). Within this simple framework, the relationship of numbers to sets of units is introduced.
- 3) In the example presented “1+1→2”, the general *relations* of addition (counting) and equivalence are realized. At this level, the simulation should also be observed as having and using the *relations* of “greater than” and “less than”. This is anticipated through the addition of more complex hierarchical schemes which build upon the previous schemes.
- 4) In this level, adherence to **LP1** should be observed as: (i) **LP1**, the relationships should not be defined in the system i.e., there should be no rules upon which actions are taken. All actions must necessarily be self-generated.

2.3.14.6 Summary

It is believed that there are a number of constraints (**LP1 – LP7**), that need to be accounted for by a simulation that attempts to develop number–sense using Piaget’s genetic epistemology. Further, it is affirmed that it is necessary to go through “stage like” development to overcome the learning paradox and equate the levels to a facet of these developmental stages. Table 2–2 summarizes the relationships of the worked example to the constraints.

Table 2–2 This table describes the anticipated observations of a simulation as it acts and senses in a number line world. It is initially seeded with a Piagetian model of learning and developing along with a set of innate schemes and primary reactions. As the system executes, it is anticipated that it will exhibit these levels of development. An example is presented that exhibits the progressive emergence of more complex structures, and this is mirrored in an external representation. Since all constraints (LP1 – LP7) apply throughout the life of the simulation, the level in which the constraint is most clearly observed to have been adhered to, is identified.

External Representation	Level of	Anticipated level in which the constraint is most clearly observed to have been adhered to.	Presented Example of
-------------------------	----------	---	----------------------

	development	LP1	LP2	LP3	LP4	LP5	LP6	LP7	Emergence
	WE1. Base	-	-	X	-	-	-	X	“Move”
	WE2. Constrained	-	-	-	-	-	X	-	“Stop”
	WE3. Differentiated – Object	-	-	-	-	X	-	-	1
	WE4. Hierarchical – Units	-	X	-	X	-	-	-	1, 1
	WE5. General – Relationships	X	-	-	-	-	-	-	1 + 1 → 2

Since the external environment does not directly control the execution of the system, the order of appearance of these “Levels of Development” is indeterminate.

2.3.15 Conclusion

This section provided a re-evaluation of Piaget’s genetic epistemology using work from Furth (Furth, 1969), Copeland (Copeland, 1974) and Piaget (Piaget, 1962a; Piaget, 1962b; Piaget, 1963; Piaget, 1964; Piaget, 1965; Piaget, 1983; Piaget, 2000b). As an empirical observation, Piaget provides a plausible model of cognition that required extensive work by other researchers, such as Pascual-Leone, Furth and Copeland to codify into a workable system, with only a few salient points for the design noted here. The roots of Piaget’s genetic epistemology as biological evolution are confirmed and presented in an abstract model of assimilation and accommodation, which fit a neural model of *Drosophila* (Miesenböck, 2008, p52; Shang, Claridge-Chang, Sjulson, Pypaert and Miesenböck, 2007, p601). Further, this model is consistent with the model by Albus (Albus, 2008; Albus, 2010) and Granger (Granger, 2006a; Granger 2006b). A *worked example* of number-sense acquisition (WE1 – WE5), is provided which describes how these would be impacted by the constraints imposed by the learning paradox (LP1 – LP7). Given these applications, a simulation would be capable of developing symbolic knowledge using a Piagetian model and partially resolve the learning paradox.

2.4 Number-Line Number Sense

“You think because you understand one, you must understand two, because one and one make two. But you must also understand and” (Sufi, 12th Century).

This section reviews number-sense from an educational perspective (Liebeck, 1984 and Fennell and Landis, 1994), in § 2.4.1. The emergence of number-sense from a neuroscience perspective (Bugden, Price, McLean, and Ansari, 2012; de Hevia, Girelli and Macchi-Cassia, 2012; Fuhrman and Boroditsky, 2010; Dehaene, 1997) is then reviewed, in § 2.4.2, along with the cultural impacts of MNL (Shaki, Fischer and Göbel, 2012; Nuerk, Helmreich, Zuber, Moeller, Pixner, Kaufmann, 2011; Núñez, 2011; Dehaene, Izard,

Chapter 2. Literature Review

Spelke and Pica, 2008), in § 2.4.3 and 2.4.4. Some conclusions of this research approach using a number line world are then given to end this section.

2.4.1 Number–Sense from an Educational Perspective

Fennell and Landis provide numerous answers to the question “what is number–sense?” (Fennell and Landis, 1994, p187). For them “Number–sense is an awareness and understanding about what numbers are, their relationships, their magnitude, the relative effect of operating on numbers, including the use of mental mathematics and estimation” (Fennell and Landis, 1994, p187). They suggest that students must understand how numbers relate to each other (Fennell and Landis, 1994) and that the magnitude of numbers is a very important aspect of number–sense: magnitude helps the students break down the problem and look at relative size being supported by mental mathematics and estimation. For example, students must understand and use the basic properties of numbers when using the commutative property to know that $20 + 10$ is the same as $10 + 20$. This determination, as well as the ability of the students to estimate, depends on the basic concepts of number–sense.

Fennell and Landis define number–sense as “the foundation from which all other mathematical concepts and ideas arise” (Fennell and Landis, 1994, p187–188). They continue, “Number–sense is good intuition about numbers and their relationships,” and students with number–sense can automatically tackle a variety of problems and can break down the problem and use the numbers as references (ibid, p188). In other words, they can make connections between their knowledge and newly learned mathematical concepts and skills (Fennell and Landis, 1994, p187). In general, they know how to make sense of numbers, how to apply them and are confident that their problem solving processes will enable them to arrive at solutions.

The importance of “number–sense” was identified by earlier researchers, specifically those that developed the National Curriculum as part of the 1988 education system act in Britain (Liebeck, 1984). Liebeck discusses how Piaget’s theory was woven into the ideas of mathematics development (Liebeck, 1984, p71). A major aspect of this curriculum is summed up by Liebeck: “When we teach only for calculating competence, we get demands for understanding. When we teach only for understanding, we get demands for calculating competence. The real need is for both of these. It is through both of them, that we are equipped to solve real problems” (Liebeck, 1984, p11). Liebeck says that the reasons for teaching children the abstract hierarchical nature of mathematics are aesthetics and usefulness (Liebeck, 1984, p13). Further, this comes through the abstract progressive development of i) experience of physical objects, ii) spoken language that describes the experience, iii) pictures that represent the experience and iv) written symbols that generalize the experience, and facilitate problem solving (Liebeck, 1984, p16). In early concept development, Liebeck sees four basic activities taking place: i) matching, ii) sorting, iii) pairing and iv) ordering (Liebeck, 1984, p17) while avoiding the associated noise that deters learning. Counting is seen as a successively more complex concept (process) that requires numbering, ordering, cardinals, linking which ultimately leads to conservation of number (Liebeck, 1984, p33). Numeration, addition and subtraction are seen as similarly complex forms that occur at similar times to shapes and length, capacity, weight and time (Liebeck, 1984, p55).

Chapter 2. Literature Review

2.4.1.1 Implications for the Research

From an educational perspective, Liebeck reviews Piaget, along with the works of Skemp, Bruner and Dienes and describes features of early concept formation as observed in children developing number–sense (Liebeck, 1984, p237 – 239). Fennell and Landis provide a similar view and assert the fundamental nature of number–sense to mathematics, without providing an abstract model, which can be implemented.

These results confirm the validity of a Piagetian approach and suggest some features that could be observed in a simulation that mirrors child development, including place matching, sorting, pairing, ordering, addition and subtraction. These observations are consistent with the views of Copeland, Fennell and Landis (Copeland, 1974 and Fennell and Landis, 1994).

2.4.2 Biology and Neurophysiology on Development of Number–Sense

While behavioral studies continue to reveal the relationships between an individual’s symbolic number processing and their arithmetic performance, much of the neural foundations of arithmetic learning still remains unknown (Bugden, Price, McLean, and Ansari, 2012, p448). Some studies suggest that number–space mapping is innate, whereby small numbers induce a compression and large numbers an expansion of spatial extent (de Hevia, Girelli and Macchi–Cassia, 2012). Other studies presented evidence suggesting the existence of a mapping between symbolic and non–symbolic numbers and spatial magnitude, with “infants at 8 months of age transferring the discrimination of an ordered series of numbers to an ordered series of line lengths, and learning to productively use a rule that establishes a positive relationship between number and length, while failing to do so with an inverse relationship” (de Hevia, Girelli and Macchi–Cassia, 2012). Other researches also support the notion that children have an innate spatial representation for time and number (Fuhrman and Boroditsky, 2010, p1432).

2.4.2.1 Implications for the Research

The implication is that the approach of this thesis of constructing an abstract representation, in this case HFSA, does not violate existing knowledge from cognitive neuroscience on the development of number–sense. Further, the implication of innate number to space mappings (Fuhrman and Boroditsky, 2010; de Hevia, Girelli and Macchi–Cassia, 2012), confirms the research approach of using a MNL with an interaction mechanism that provides spatial values.

2.4.3 Cultural Impact on Development of Number–Sense using a MNL

Research suggests that cultural differences account for how individuals process numbers on the number line, with western children showing SNARC effect (small numbers associated with left responses and right responses for large numbers) as well as the automatic / plastic mapping between numbers and space (Shaki, Fischer and Göbel, 2012, p275). Without necessarily challenging earlier research on the cultural late development bias of numeric magnitudes in number line processing by preschool children, Opfer, Thompson

Chapter 2. Literature Review

and Furlong hint that development of an early left–right bias improves performance in both estimation and reasoning skills (Opfer, Thompson and Furlong, 2010, p761). However, research shows that performance by children in Siegler’s number line task is indeed influenced by culture–specific language properties (Nuerk, Helmreich, Zuber, Moeller, Pixner, Kaufmann, 2011, p598). Evidence of the cultural impact on the MNL is provided by work on Australian Indigene, who, using merely a spatial strategy, were significantly more accurate than western children who relied on an enumeration strategy supported by counting words to perform addition tasks (Reeve, Lloyd, Reynolds and Butterworth, 2011, p630).

Núñez argues convincingly that humans do not have a hard–wired, innate and consistent number to space relationship that can be mapped to a mental number line (MNL) (Núñez, 2011, p652). Rather, the MNL emerges for each individual through top–down dynamics that are culturally mediated, reusing existing cognitive mechanisms which have been acquired through education, and which make use of the brain phenotypes that support number–to–space mappings (Núñez, 2011, p652). They maintain that the left–right association of numbers is merely cultural. This is contrary to the views of Dehaene and SNARC coding (§ 2.4.3). Núñez also argues for the implausibility of an innate number line by showing diachronically that the ancient civilizations did not use it, with the first recorded publication being in 1685 by John Wallis (Núñez, 2011, p654). In supporting this argument, Núñez refers to the lack of sophisticated numerosity in indigenous groups, concluding that “number line intuition is not universal and that number concepts can exist independently from MNL representations” (Núñez, 2011, p658). Moreover, Núñez says that the number line task “imposes an overtly spatial source domain for the mapping, not a target domain,” which some cultures will have difficulty in manipulating, but that through education, it is possible to manipulate those environments (Núñez, 2011, p661).

In describing cognition, Núñez refers to conceptual mapping – conceptual metaphor (§ 2.6) which “is an inference–preserving cross–domain mapping that allows the projection of the inferential organization from a source domain onto a target domain” (Núñez, 2011, p663). This approach supports an entire network of systematic inferences of concepts in terms of spatial experience (Núñez & Sweetser, 2006 cited in Núñez, 2011, p663). Crucially, these conceptual mappings are universally available, but they are not genetically determined, allowing for cultural variation, historical mediation and development (Núñez, 2011, p663). One such conceptual metaphor is the MNL and the related number space representations. Núñez points to the need for a better understanding of how these conceptual metaphors relate to the neural processing in the brain (Núñez, 2011, p663).

2.4.3.1 Implications for the Research

There are three impacts for this research thesis: 1) The SNARC effect (Shaki, Fischer and Göbel, 2012) will not be accounted for in this research, though potentially, it can be an object of follow–on work for future research. 2) The research approach of using a spatial strategy for the development of number–sense using a MNL, is supported by research in the field (Reeve, Lloyd, Reynolds and Butterworth, 2011, p630). 3) In addressing the need for a better understanding of conceptual metaphors in the development of number–sense,

Chapter 2. Literature Review

Núñez points to conceptual metaphor as a solution (Núñez, 2011, p663). The role of conceptual metaphor is reviewed in the § 2.6.

2.4.4 Number–Sense is Innate but Culture Makes a Difference

2.4.4.1 Number–Sense is Innate

Dehaene, as the subtitle of his 1997 work indicates, understands number–sense and all that goes with it, as created by the mind and brain (Dehaene, 1997). For Dehaene, the brain itself communicates in numerical codes. Nerve cells (neurons) convey messages by electrical pulses that pass down the nerve fiber (axon) until they reach the junction with the next neuron or an effector such as a muscle. There they release little packets of chemical transmitter that trigger the next cell to respond. The response depends on the frequency of electrical pulses and the quantity of transmitter released. Counting is thus an embedded feature of brainwork – much more fundamental than language, which partly explains Piaget’s fascination (Piaget, 1970a, p13).

2.4.4.2 Culture Impacts Development of Number–Sense

For Dehaene, constructivist theories view mathematics as a set of cultural inventions that are progressively refined in the history of mathematics and are slowly acquired during childhood and adolescence (Dehaene, Izard, Spelke and Pica, 2008, p1217). All humans share similar intuitions which map numbers onto space, but that culture specific experiences alter the form of this mapping (Dehaene, Izard, Spelke and Pica, 2008, p1217). Yet, the acquisition requires a complex interaction between symbols and “non–symbolic visual and auditory numerosities” (Dehaene, Izard, Spelke and Pica, 2008, p1218).

2.4.4.3 Evolution Provided a Deep Foundations for Number–Sense

Conclusions drawn from Dehaene’s research suggest that the mental construction of mathematics may have deeper foundations (Dehaene, Izard, Spelke and Pica, 2008, p1217). Mathematical objects may find their ultimate origin in basic intuitions of space, time, and number that have been internalized through millions of years of evolution in a structured environment and emerge early in ontogeny, independently of education (Dehaene, Izard, Spelke and Pica, 2008, p1217).

2.4.4.4 Observations of Logarithmic Use of Number line

Observations of children positioning small and large values on a number line seem to obey Weber’s law⁵¹ and it is compatible with a logarithmic internal representation with fixed noise (Dehaene, Izard, Spelke and Pica, 2008, p1217). Dehaene suggests that since a logarithmic scale provides several orders of magnitude with fixed relative precision, it may have been selected by evolution for its compactness (Dehaene, Izard, Spelke

⁵¹ Weber’s law a ubiquitous psychophysical law whereby increasingly larger quantities are represented with proportionally greater imprecision.

Chapter 2. Literature Review

and Pica, 2008, p1217). Continued research on indigene supports this notion, and shows that it not only occurs, but that indigenous populations, those that do not have access to an educational culture, show continued use and development of logarithmic numerical estimation skills on the number line. They also do not develop the linear skills that are found in the educated children of western cultures (Dehaene, Izard, Spelke and Pica, 2008, p1217). But, Dehaene retorts, “If humans’ initial intuition of number is logarithmic, we wouldn’t have had to wait until the 17th century to see the invention of logarithms through Napier’s painstaking work” (Dehaene, Izard, Spelke and Pica, 2008, p1213). It could be that the sensory motor functions may not be sufficiently pliable.

2.4.4.5 Implications for the Research

There are several implications of Dehaene’s work. First, the deep structures of cognition that impart mathematical ability (of space, time and number) justifies the architectural approach (Dehaene, Izard, Spelke and Pica, 2008, p1217), which enables the random self-assembly of a simulation that exhibits mathematical ability (§ 4.2.2.5 on using evolutionary computing as a framework for learning schemes). Second, the observation of children’s logarithmic use of a number line as described in numerical cognition (Dehaene, Izard, Spelke and Pica, 2008, p1217) and the perceived shift from logarithmic to linear could be an area for future research.

2.4.5 Conclusions

In this section, the emergence of number-sense has been discussed, reviewing the cultural effects of the number line as well as current theoretical approaches in the development of mathematical ideas. This research confirms a number of positions, and identifies areas for future research. First, the choice of a Piagetian model of number-sense development is supported by Liebeck (Liebeck, 1984, p237) with sets of common structures appearing in childhood development (Fennell and Landis, 1994). Second, while behavioral studies continue to reveal the relationships between an individual’s acquisition of symbolic number processing (Dehaene, Izard, Spelke and Pica, 2008) and innate structures (de Hevia, Girelli and Macchi-Cassia, 2012; Dehaene, 1997), and other researchers suggest that there are no clear models of cognitive mathematical development (Bugden, Price, McLean, and Ansari, 2012). This confirms the research approach of using an *artificial neural network implementation* to test for emergence of *permanent object invariants* as well as to implement a *dialectic system* as a Piagetian model with tests for emergence of number-sense.

Even though research suggests a cultural bias of the MNL (Shaki, Fischer and Göbel, 2012; Opfer, Thompson and Furlong, 2010; Nuerk, Helmreich, Zuber, Moeller, Pixner, Kaufmann, 2011), this will not be accounted for in this research. Further, the observation of a logarithmic to linear shift in the use of the MNL (Dehaene, Izard, Spelke and Pica, 2008) will not be considered in this research, though it could be considered for future-work.

2.5 Concept Formation in Automated Mathematics

Since this research examines how the development of number sense in children resolves the learning paradox, in this section is reviewed automated theorem formation and proving systems in pure mathematics to determine if they have resolved the learning paradox. If they have, then potentially, they could be a candidate solution from which to develop the research in the present thesis. Philosophical foundations are presented in § 2.5.1, and HR (Bundy, 1985; Colton, 2000; Colton, 2002) is reviewed as an exemplary system, in § 2.5.2. Conclusions reached are presented in § 2.5.3.

2.5.1 Introduction

Theorem formation and proving systems, otherwise referred to as automated theorem formation and proving (ATF/ATP) continues to be an active area of development (Pease, Colton. and Charnley, 2012), which reasonably began with Siekmann and Wrightson's study on the automation of reasoning (Siekmann and Wrightson, 1983).

The application of AI techniques to mathematical discovery has largely involved computer algebra systems: theorem provers with ad-hoc systems for generating concepts and conjectures (Colton and Muggleton, 2006, p25). Such ad-hoc systems have included the AM system (Lenat, 1982; Lenat, Prakash and Shepard, 1986), which worked in set theory and number theory; the GT system (Epstein, 1988 cited in Colton and Muggleton, 2006, p25), which worked in graph theory; the IL system (Sims & Bresina, 1989 cited in Colton and Muggleton, 2006, p25), which worked with number types and the Graffiti program (Fajtlowicz, 1988 cited in Colton and Muggleton, 2006, p25), which has produced scores of conjectures in graph theory.

While many theorem proving techniques are based on Herbrand's theorem (Chang and Lee, 1973), which uses inference rules in predicate calculus and a proof system using sequent and tableau methods (Bibel, 1981), attempts have been made using natural deduction, induction (Colton, 2002), recursion analysis (Stevens, 1988) or combining the methods of Lakatos to build collaborative agent based systems (Pease, 2007).

Besides the obvious application of mathematical reasoning, many problems have been transferred into theorem proving problems. These include software verification, program-synthesis, hardware verification, question-answering systems, game design, state transformation systems and computational creativity (Colton, 2000, p97).

2.5.2 The HR System

In his 1999 Ph.D. thesis, Colton extended upon Alan Bundy's aim of producing a mathematical assistant, and delivered the HR system (Bundy, 1985; Colton, 2000; Colton, 2002, p14). Colton considers three main applications for HR: (i) to find something about a given concept i.e., a definition or a property; (ii) to find an entirely new concept with a particular property; and (iii) to find a set of concepts, which cover all definitions of a particular form (Colton, 2000, p114). Automated theory formation in pure mathematics requires the invention of new concepts, the computation of examples, the making of conjectures and ultimately the proving of these

Chapter 2. Literature Review

theorems (Colton, 2000). To do this HR uses various pieces of mathematical software, including Otter to prove conjectures and the MACE program to find counter examples (Colton, 2000, p28 and p98). It also uses model generators and databases, to build a theory from the bare minimum of information – the axioms of a domain.

The HR system provides general purpose automated theory formation (ATF) as well as automated theorem proving (ATP) in mathematical domains using a frame representation (Colton and Muggleton, 2006, p31). Unlike other ATF solutions described by Colton such as AM, GT, IL, BACON and the Bagai et al program, Colton’s motivation is focused on finding interesting patterns (Colton, 2002, p144 and p287), in a similar fashion to machine learning programs such as Progol⁵² (Colton and Muggleton, 2006, p25). In so doing, it draws parallels with Piaget’s assertion of the need for intrinsic motivation for the development of concepts (Furth, 1969, p246); however, the definition of what constitutes a concept differs.

HR uses a frame system (for constants, concepts and hypotheses) on top of an object-oriented language (Java) as its knowledge representation (Colton and Muggleton, 2006, p25). Theories are formed using production rules, definitions and “fine grained parameterizations,” which describe how new definitions will be created by exploiting the production rules (Colton and Muggleton, 2006, p32). Three types of production rules are used in this process: 1) nullary (entity-disjunct), 2) unary (exists, match, equals, split, size, linear-constraint embed-algebra, embed-graph and record) and 3) binary (compose, disjunct, negate, forall, arithmetic and num-relation) (Colton and Muggleton, 2006, p35). Theories defined using these rules are stored in first order predicate logic (Colton and Muggleton, 2006, p35).

In HR, theories typically contain examples of the objects of interest and concepts⁵³ (Colton, 2000, p98). These theories also contain proofs, dis-proofs and counterexamples, as well as open conjectures for which the truth is unknown. An example conjecture is “All prime numbers are non-squares.” Using standard terminology, statements are conjectures until they become theorems when proven via a series of logical inferences. Specifically, a theory in HR is a set of classification rules, which are expressed as a predicate definition, and a set of association rules, which are expressed as range-restricted clauses (Colton & Muggleton, 2006, p29).

In HR, concepts can be considered as invariants or parts of invariants if they represent properties, which remain unchanged in the domain (Colton, 2000, p16 and p153). A concept in HR consists of a tripartite of a set of examples (a *data table* or set of *data tables*), a *definition* (with constants, variables or states) and a *categorization*. The categorization is the generation of the theory over the examples HR has available, and includes a set of conjectures involving the concept (Colton, 2000, p6). For example, in the concept of *prime number*, the objects of interest that are specified in the *data table* are the integers from 1 to 10, with a truth flag

⁵² Progol is an implementation of Inductive Logic Programming used in computer science that combines “Inverse Entailment” with “general-to-specific search” through a refinement graph (Colton and Muggleton, 2006, p25).

⁵³ Concepts, which discuss the nature of those examples and statements, and highlight the relationships between concepts (Colton, 2000, p98).

Chapter 2. Literature Review

“of being a prime” being indicated. The *definition* consists of the predicate rules and the *categorization* consists of the primes and non–primes based on the level of interestingness. Colton argues that a desirable property for a set of concepts in a theory is to be able to describe different ways of grouping the objects of interest that i) avoids redundancy, ii) achieves a high number of categorizations and iii) develops all the user driven concepts (Colton, 2002, p168).

Users in HR initially create a domain model of the concept they are interested in evaluating as a user supplied set of data frames. The data frames consist of i) optional set of axioms, ii) examples (as a classification), iii) termination conditions and iv) a background theory which consists of constants and predicates using its frame system (Colton and Muggleton, 2006, p28). This domain model is then used in theory formation.

2.5.2.1 Searching for Definitions – The Process of Theory Formation in HR

ATF in HR is driven by a set of theory formation steps, which attempt to define a new concept, using production rules, and the agenda mechanism (Colton, 2002, p142; Colton and Muggleton, 2006, p46). In the agenda mechanism, HR employs simple breadth first, best–first and look ahead strategies to improve its overall performance in generating interesting concepts (Colton and Muggleton, 2006, p47).

HR generates hypotheses in first order predicate logic using an ILP routine which combines inductive and deductive reasoning to form clausal theories consisting of classification rules and association rules on the user supplied data (Colton, 2000, p12; Colton and Muggleton, 2006, p36). Specifically, HR uses the supplied frames and generates definitions using a set of production rules, interprets the definitions as classification rules, then uses the sets of the definitions to induce hypotheses in first order logic from which it extracts association rules (Colton and Muggleton, 2006, p25). The system makes use of 17 measures of interestingness including: intrinsic, developmental, relative, theorem based and learning to determine what to do based on what it has found (Colton and Muggleton, 2006, p47). Examples of interestingness include applicability – which estimates the proportion of objects of interest for which the concept / conjecture is applied; novelty – which evaluates how novel is the categorization of a concept with respect to others; comprehensibility – which measures how succinct the definition a concept is and variety – which measures the categorizations produced by a concept where less categorization means less change (Colton, 2002, p145).

Colton suggests that “the initial choice of concepts will have a profound effect on the nature of the theory produced,” since all new concepts will be based on them, and all new conjectures, theorems and proofs will involve them to some extent (Colton, 2002, p67). This in part, is due to the system bootstrapping from fundamental concepts to rich theories (Colton, 2002, p67). These existing concepts are supplied as part of the user data.

Noisy and incomplete data is a common aspect of science, yet in mathematics, the opposite is true (Colton and Muggleton, 2006, p44). By supporting noisy data, HR, facilitates its use in more real–life situations. HR fixes faulty hypotheses (and conjectures) using abductive reasoning with methods inspired by

Chapter 2. Literature Review

Lakatos (Lakatos, 1976 cited in Colton and Muggleton, 2006, p44; Pease, 2007, p13). This enables HR to be applied to the more human specific cognitive development approaches.

2.5.2.2 Future Work for HR

HR's ability to induce conjectures from the data alone means that it can construct empirically plausible but non-trivial ones to prove hypotheses about the data. Colton's work supports the differentiation of automated theorem proving systems through the development of a standard library of first order theorems, which are used to benchmark other systems (Colton and Muggleton, 2006, p25). HR also supports modification of non-theorems using MACE (Colton and Muggleton, 2006, p53), and the reformulation of constraint satisfaction problems to find quasigroups (Colton and Muggleton, 2006, p54).

HR has been applied to several machine learning tasks including: the invention of integer sequences, number sequence extrapolation, creative problem solving and puzzle generation (Colton, 2000, p97). The key to HR's success lies in the maturity of the production rules (Colton and Muggleton, 2006, p27). While HR is not mimicking the way mathematicians work (Colton, 2002, p143), or for that matter, how children are observed to develop number-sense, the production rules have proven to be highly general by constructing many important existing concepts and many interesting new concepts for which they were not originally conceived (Colton, 2002, p143).

Over time, HR has been extended with new capabilities including Lakatos style reasoning (Pease, 2007) and formal concept analysis (Colton and Wagner, 2007) to name two. These additional features suggest that HR is evolving towards more noisy and incomplete environments, and at some point may become a contender as a cognitive development model that could support a Piagetian model of development.

2.5.2.3 HR: Where is the Problem?

Interestingly, Lakatos style reasoning (Lakatos, 1975) has been applied to Piaget (Roswell, 1983) as well as to HR (Pease, 2007). The evidence presented by Roswell suggests that in the early stages, the process of equilibration may result in false ideas being retained in a more complex form. What this implies, is that the child may go through a process similar to ATP and ATF, but that the underpinning of a predicate calculus is unlikely to be available directly to them as they are maturing. Further, the interestingness property in HR (Colton and Muggleton, 2006, p47) can be related to Piaget's notion of novelty in adaptation (Piaget, 1980a, p24; Copeland, 1974, p9; Piaget, 1962b, p84). In HR, novelty looks for empirical relationships with existing concepts and forms conjectures when such relationships are found. It is also interesting to note that the desirable property of the concepts in a theory represent the experience of the concept in the domain space (Colton, 2002, p168). Piaget would assert that this is the equivalent to acting in the environment, which is seen as critical to the development of the child. However, unlike a child, HR is not autonomous; it acts in its environment on specification by a user (Colton, 2002, p141).

Chapter 2. Literature Review

This research solution acts in situations that are noisy and incomplete, and in acting the system is not necessarily interested in being 100% accurate nor being necessary mathematically correct. With this in mind HR is not a suitable model to implement a Piagetian model, since Piaget crafted his approach on the development of children, and children are not observed to produce proof of their actions. It could be argued that children are not logical at all; however, once a child develops concrete operations (Piaget, 1963 cited in Furth, 1969, p125; Furth, 1969, p207), the child may need to have the capabilities of an ATP system to enable them to operate in the formal operations stage of development (Furth, 1969, p66).

HR (Colton, 2002), like most automated theorem proving systems comes already prepared with a predicate calculus, and mechanisms by which the user can direct the actions of the system, in doing so, it fails **LP1**.

The random development of logical capability is the goal of this research thesis. There is no formal definition of concepts as in HR. Concepts are emergent from the processes of assimilation and accommodation, and there is no direct parallel to the predicate definitions of HR. What is interesting is that a concept in Piagetian terms is both externalized and internalized action. In Colton's work, there is a similar notion of action with a concept (Colton, 2002, p67). Specifically, it is the "categorization over examples HR has available as well as a set of conjectures involving the concept" that is important (Colton, 2000, p102). Though this is similar, it is not fully compliant with Piaget's notions of "concept." It does however suggest that the rules that the HR system has constructed, could be described as the exploration of the concept in a pure mathematics environment.

2.5.3 Conclusions

This section reviewed automated theorem formation and proving systems in pure mathematics (Colton, 2000; Colton, 2002; Colton and Muggleton, 2006; Colton and Wagner, 2007; Pease, 2007). It reviewed their philosophical basis, the development of mathematics through simulated creativity and determined how these could support the constraints imposed by the learning paradox and the *worked example*. Discovered in this research was that even though ATF and ATP methods exhibit novelty in developing interesting new concepts for which they were not originally conceived (Colton, 2002, p143), they fail Fodor's and Piaget's arguments by having too much "built in" (**LP1**), as compared to the observed capabilities of the developing child. As such, they are not a candidate solution upon which to build a solution.

2.6 The Underlying Need of Emergence in Metaphors and Conceptual Blending

§ 2.6.2 reviews the underlying basis of emergence of concepts in metaphors (Lakoff, 1992; Lakoff and Johnson, 1980; Lakoff and Núñez, 2001). This is followed by a review of mental spaces (Fauconnier, 1985) and conceptual blending theories (Fauconnier and Turner, 2002), which are then correlated with the constraints imposed by the learning paradox, in § 2.6.3. An example of number line processing using conceptual blending (Hutchins, 2005) is then presented along with an example of arithmetic from Guhe (Guhe et al, 2011, p251). In

Chapter 2. Literature Review

both examples, issues are identified with the approaches and then Fauconnier and Turner' views on the need for "emergence" to build cognitive structures in conceptual blending theory are highlighted. Finally, a review of conceptual blending as a basis of mathematics is given in § 2.6.4, followed by conclusions in § 2.6.5.

2.6.1 Background

In its first two decades, much of cognitive science focused on the mental functions of memory, learning, symbolic thought, and language acquisition (Fauconnier, 1985). These are the functions in which the human mind most closely resembles a computer (Fauconnier, 1985). Now, cognitive science is increasingly focused on the more mysterious, creative aspects of the mind, i.e., consider Ryle's "wink" (Geertz, 1973, p7). The "wink," more so than a twitch, imparts information that can be understood in terms of its cultural impact as a greeting or humor or parody. Cognitive psychologists have studied this aspect of the human condition and attributed to it various terms. Mithen refers to it as cognitive fluidity (Mithen, 1996); others as relational networks of symbols (Deacon, 1996), mental spaces (Fauconnier, 1985), blending (Geertz, 1973), conceptual blending (Fauconnier and Turner, 2002), and even to metaphors (Lakoff and Núñez, 2001).

2.6.2 Metaphor: Where Mathematics Comes From

Lakoff's original thesis on conceptual metaphor was inspired by Reddy's work on the conduit metaphor (Lakoff, 1992, p204). Lakoff provides a definition, "metaphor is pervasive in everyday life, not just in language but in thought and action. Our ordinary conceptual system, in terms of which we both think and act, is fundamentally metaphorical in nature" (Lakoff and Johnson, 1980, p3). For Lakoff, metaphor is not a figure of speech (in language) but is a mode of thought and reasoning (Lakoff, 1992, p205).

The idea of conceptual metaphor is pervasive through Lakoff's work and is grounded in the human body, brain, in cognitive capacities, and in common human activities and concerns (Lakoff and Núñez, 2001, p358). Specifically, "Conceptual metaphor is a cognitive mechanism for allowing us to reason about one kind of thing as if it were another...it is a grounded, inference preserving cross-domain mapping—a neural mechanism that allows us to use the inferential structure of one conceptual domain (say, geometry) to reason about another (say, arithmetic)" (Lakoff and Núñez, 2001, p6). In this, Lakoff and Núñez, suggest that many of the sophisticated mathematical procedures humans employ can be traced back to primitive schemas, an example of which is the "container schema" which they believe underlies set theory and Boolean logic (Lakoff and Núñez, 2001, p33 and p123). Like other primitive schema, the "Container Schema" is an abstraction from physical sensations (Lakoff and Núñez, 2001, p101). In this there is a clear parallel to Piaget's work (§ 2.3 on Piaget, § 2.3.6 on innate schemes and primary reactions and § 2.3.10 on sensory motor schemes).

For Lakoff, the development of thought has been the process of developing better metaphors (Lakoff and Núñez, 2001, p102). In the embodied mind, the content of mathematics is not given in advance; it evolves in the individual through the linking and blending of metaphor. This "Cognitive mathematics" also has a cultural dimension, which, from the perspective of embodied mathematics, is entirely natural (Lakoff and Núñez, 2001, p358).

Chapter 2. Literature Review

Applying the assumption that most thoughts are unconscious and unavailable to the conscious mind, Lakoff and Núñez point to general purpose conceptualizations that are used in every day processing, being reused in the development of mathematical concepts (Lakoff and Núñez, 2001, p29). In this reuse, they point to the combination of multiple image schemas (above, contact and support) to represent the English word “on,” as in for instance, the “jar is on the table” (Lakoff and Núñez, 2001, p30). In this sense, “on” is an idea, and the idea does not “float abstractly in the world,” it can only be embodied in the human. The same is true for mathematical abilities. Given that the brain has an innate arithmetic capability, with babies exhibiting characteristics associated with discriminating between collections of two and three items (Lakoff and Núñez, 2001, p15; Dehaene, Izard, Spelke and Pica, 2008). Lakoff and Núñez argue that this occurs through a conceptual metaphor called subitizing (Lakoff and Núñez, 2001, p15). Subitizing develops as the child matures, and leads the child through basic arithmetic and later to more complex mathematics in general (Lakoff and Núñez, 2001, p26). It is the learning and development of these mathematical concepts that is the interest of this research thesis.

2.6.2.1 Criticisms

Madden reviewed the work of Lakoff and Núñez from the perspective of Poincaré, elaborating their work on the foundational basis of intuition and logical processes (Madden, 2001, p1182), by equating Poincaré’s position on unconscious processes to conceptual metaphor, which in turn can be equated to intuition⁵⁴ (Lakoff and Núñez, 2001, p xv). Madden posits that they seem to have missed the essential need for conscious definitions of proof using logic, since without this two fold approach, meaning, for the individual, is lost (Madden, 2001, p1182). Madden condenses his analysis around three core issues: the role of conceptual metaphor in mathematical cognition, the nature of mathematical truth and the techniques of mathematical idea analysis.

Conceptual Metaphors and Mathematics

The hypothesis of conceptual metaphor is based on two tenets. First, that thought is mainly unconscious (Lakoff and Núñez, 2001, p xv) and second, that conceptual metaphors are influenced by the human condition, its bodily senses and cultural identities (Lakoff and Núñez, 2001, p358).

Even though the general theory of metaphors (Lakoff and Johnson, 1999) is referenced, there is little research presented on the actual use and manipulation of the conceptual metaphors (Madden, 2001, p1184). Second, having access to metaphors is quite different to learning to use them (Madden, 2001, p1184).

The Truth of Mathematical Meaning

⁵⁴ Specifically, mathematical intuition.

Chapter 2. Literature Review

Almost more important, than developing one's own conceptual metaphors, is the capability to communicate a shared understanding "of truths" between participants (Madden, 2001, p1186). If one were to follow Lakoff and Núñez, then "meaning, existence and truth" are metaphorical entities that exist "conceptually" only "in the minds of beings with appropriate metaphorical ideas" (Lakoff and Núñez, 2001, p368 and p375). This is understood to mean that each individual holds similar metaphors, only because they have the same internal apparatus that enabled those metaphors to be constructed. In this respect, Lakoff and Núñez, are anti-Platonistic⁵⁵, and argue that because there is no mechanism whereby Platonism can be tested, it cannot really be considered a scientific hypothesis and so is rejected (Lakoff and Núñez, 2001, p4). Anti-Platonism is a difficult proposition to take, because the innate mechanism that allowed the conceptual metaphors to arise in the first place must have been constructed through evolution using those self-same "truths," which logically could not have existed in the external world. Balaguer (Balaguer, 1998) and § 2.1.3 provide a description of universals and the impact to the learning paradox are more fully explained.

Second, truth must be considered unambiguously in context: consider the case of the square root of -1 , there is no answer on the real number line, but two of them in the plane. Alternatively, consider the notion of " $1+2=3$ " on a number line. This seems simple enough but it contains multiple ideas not the least of which is equality, which for Lakoff and Núñez consists in the blending of multiple conceptual metaphors including the grounding metaphors of *object collection*, *object construction*, *measuring stick* and *motion along a line* (Lakoff and Núñez, 2001, p75); as well as the *space set blend* (Lakoff and Núñez, 2001, p75), *spaces are sets of points* (Lakoff and Núñez, 2001, p263) and *rational number line blend* (Lakoff and Núñez, 2001, p300).

Finally, it is also difficult to reconcile Lakoff and Núñez's version of truth with how mathematicians think (Byers, 2007, p27): their metaphorical approach has to deal, in large part, with ambiguity, novelty and noise which is also not covered in their approach (Madden, 2001, p1183).

Mathematical Idea Analysis

In defining mathematical idea analysis, Lakoff and Núñez provide a technique for separating apart the metaphorical parts of the subconscious that has generated the mathematical ideas (Lakoff and Núñez, 2001, p375). However, mathematical metaphors are "frequently misleading, sometimes just plain wrong" (Madden, 2001, p1186), and this would be true of the metaphors in mathematical idea analysis as well.

2.6.2.2 Metaphor: Where is the problem?

An interesting reference is made by Madden (Madden, 2001, p1187), "Mathematics is its own mirror on the very thinking that creates it" (Cuoco, and Curcio, 2001, px). When viewed in the context that Mathematics is the only infinite human activity (attributed to Paul Erdos), it becomes possible to take a closer step to

⁵⁵ Platonism and the transcendental origin theory, is the philosophical doctrine that abstract concepts exist independent of their names.

Chapter 2. Literature Review

conceptual understanding⁵⁶. That is, that mathematics, available innately to the individual, is used by the individual to construct the conceptual metaphors, which are then used to perceive and act in the world. This is also true for learning and planning. The capability to construct new metaphors (and thus handle infinity) is the key to perception, an example of which is the evidence of children having difficulty understanding mathematics, misusing, or being unable to generate new metaphors as conceptual blends (Lakoff and Núñez, 2001, p102). When given examples of a new metaphor, the child can understand the set problems (Madden, 2001, p1184), but this implies that the child can receive new blends and use them appropriately. This sharing and coordinated development is not well covered in Lakoff and Núñez. In addition, the catalog of over-lapping metaphors (Lakoff and Núñez, 2001, p368) suggests that it is not that the metaphorical approach is necessarily wrong, it is that a more appropriate mechanism would be a meta-cognitive metaphor that enables the construction of metaphor.

Conceptual metaphors may explain how an individual develops concepts and reasons, but the conceptual metaphor framework (Lakoff and Johnson, 1980; Lakoff and Johnson, 1999; and Lakoff and Núñez, 2001) does not provide enough details for the construction of a machine-learning framework, and is so rejected as a candidate solution for the present research. Specifically Lakoff and Núñez define a set of categories of metaphors, rather than a mechanism by which individuals act and sense, learn and plan in the world and develop their own categories (§ 2.3 refer to Piaget). The emergence of new structure is seen as a basic need to resolve the learning paradox (§ 2.1.7 refer to LP), but the conceptual metaphors framework does not support the notion of emergence of new structure very well. Potential observations of an implementation of the Piagetian framework may show emergence of metaphor through a shared interaction of more complex forms of reasoning, but this capability to reason in metaphor is deemed to be at too high a level for the current research. It is possible that an analysis of conceptual metaphors could form part of later research, as well as a fuller comparison of Piagetian and conceptual metaphor frameworks. As a final point, metaphorical mappings (Lakoff and Johnson, 1980; Lakoff and Núñez, 2001) alone are not sufficient to account for how mathematical concepts develop in an individual (Guhe et al, 2011, p251).

2.6.3 Conceptual Blending

By suggesting that any theory of human cognition (cognitive powers) must not only account for the richness and variety of human innovation, but also show how that innovation is guided (Fauconnier and Turner, 2002, p310), Fauconnier and Turner form an argument for concept formation, which is analogous to evolutionary development. In this, human beings use conceptual integration to create rich and diverse conceptual worlds, and “the construction of meaning is like the evolution of species” (Fauconnier and Turner, 2002, p171 and p309). The guiding constraints of this conceptual integration are a combination of *constitutive principles* and *governing principles* with the overarching goal to “achieve human scale” (Fauconnier and Turner, 2002, p346).

⁵⁶ This meta-cognitive metaphor (Madden, 2001, p1184), can be interpreted as Piaget’s “knowing circle.”

Chapter 2. Literature Review

The *constitutive principles* make use of matching and counterpart connections, generic spaces, blending, selective projection and emergent meaning, which ultimately leads to the development of emergent structure in the blend by forming *networks* of concepts (Fauconnier and Turner, 2002, p310).

The *governing principles* occur not as all-or-nothing constraints on networks, rather they “characterize strategies for optimizing emergent structure” (Fauconnier and Turner, 2002, p311). These *governing principles* scale a single relation, compress one or more relations, create new relations or they borrow compressions from one concept to another (Fauconnier and Turner, 2002, p312). Other *governing principles* cover areas such as topology, pattern completion, integration, unpacking and relevance (Fauconnier and Turner, 2002, p346). Creativity and novelty are consequences of this conceptual integration, which are themselves dependent on a background of firmly anchored and mastered mental structures or mental spacesⁱⁱⁱ (Fauconnier and Turner, 2002, p382).

The conceptual blending process of networks and prototypes has been evolving since it first appeared in the mammalian line, but it is the human development of double scope blending that provides more benefit (Fauconnier & Turner 2002, p171).

2.6.3.1 Networks and Prototypes

There are four main types of prototypes that anchor intuitive everyday notions about meaning to a unified understanding of the unconscious processes. These are simplex, mirror, single-scope and double-scope (Fauconnier & Turner, 2002, p119). These networks integrate and compress, disintegrate and decompress concepts^{iv} (Fauconnier & Turner 2002, p119). It was the evolutionary emergence of double scope conceptual integration that enabled the rapid development of human culture, and enables the child to understand and develop rational number-sense (Fauconnier and Turner, 2002, p389; Fauconnier and Turner, 2002, p113).

Double Scope Conceptual Integration and Language

Double-scope blending, consists of selectively integrating (using a mapping capacity) the inputs of two or more conceptual arrays whose frame structures and vital relations⁵⁷ conflict to create a novel conceptual array whose frame dynamically develops emergent structure not found in the inputs (Fauconnier & Turner 2002, p391).

Double-scope blending is a necessary feature of human higher-order conceptual singularities grammatical constructions and language (Fauconnier & Turner 1996, p113), for its equipotentiality⁵⁸ (Fauconnier & Turner 2002, p182). Intermediate stages of the mapping capacities (between conceptual blending and double scope blending) are useful and adaptive, but not for language, which demands equipotentiality (Fauconnier & Turner 2002,p182). This explains the absence of intermediate stages of language as an observable product (Fauconnier & Turner 2002, p179).

Of critical importance in the development of meaning, the association of a compression pattern with a linguistic form, which is the maximizing and intensifying of vital relations (Fauconnier and Turner, 2002, p353). These vital relations connect a represented element with a representing element in different input spaces using a representation (Turner, 2011, p48). In the development of meaning lies an assumption, “A language already has all the grammatical forms it needs to express almost any conceptual blend” (Fauconnier and Turner, 2002, p365).

Stability of Events and Time

Reasoning processes require stable representations of constraints achieved whether through cultural stability or material stability (Hutchins, 2005, p1555). One such stability is the vital relation of time, which is bound up in mental spaces (Fauconnier, 1985). Events as singularities are bound up in event spaces, which can include subjective experience of those events. For example motion is an event, which through physical space – from point a to point b – holds corresponding objective and subjective experience (Fauconnier & Turner 2002, p376; Hutchins, 2005, p1568). As structure emerges from the blended space, the universal event becomes a universal spatial length, and therefore a measure, analogous to inches and feet, and so on. This explains why an event has a length – it is a minute long, an hour long, etc., (Fauconnier & Turner 2002, p376). Similarly, in the blend, with objective time, the shared universal events, such as hours, minutes, etc.) exist – egos are constrained to move at the same rate (Fauconnier & Turner 2002, p376). If agency is projected onto the causal constraint, all egos are moved through the shared universal events at the same rate by an agent, in this case the

⁵⁷ Vital conceptual relations include cause-effect, identity, time, space, modality, participant structure, disanalogy and role (Fauconnier & Turner 2002, p391).

⁵⁸ Equipotentiality is the ability of language to be used effectively in any situation, not just those that fit a finite list of frames (Fauconnier & Turner 2002, p182).

Chapter 2. Literature Review

agent is often referred to as “Time” (Fauconnier & Turner 2002, p96). In this new blend, the emergent entity “Time” derives its motion from the network in which times move, but derives its anchor from the network in which individual moves (Hutchins, 2005, p1563).

Emergence through Composition, Completion and Elaboration

In blending theory, the *constitutive principles* and *governing principles* achieve human scale through the emergence of structure in the blend (the blended scheme) in three interrelated ways: composition, completion and elaboration^v (Fauconnier & Turner 2002, p143). In this approach, language is seen as a system of prompts for integration (Fauconnier & Turner 2002, p143), such that when one views words on a page, these are triggers for the imagination, which are used to “call up some of what we know and to work on it creatively to arrive at a meaning” (Fauconnier & Turner 2002, p143). This is very close to the notion of figurative schemes, with the associated operative scheme acting on it to generate meaning (§ 2.3.6 on Piaget and 4.2.2.2).

2.6.3.2 Example of Conceptual Blending on a MNL

A model of counting on a number line is presented using conceptual blending theory (Hutchings, 2005, p1576). This extended example exposes some weaknesses of the theory including the idea that conceptual blending is merely an internalized cognitive process (Hutchings, 2005, p1576; Fauconnier & Turner 2002, p195–216). The objective of the example is to count the number of points in a line and occurs as a set of steps:

- 1) If one were to consider the points on the line, the cultural practice initially creates a spatial memory for the order of the points.

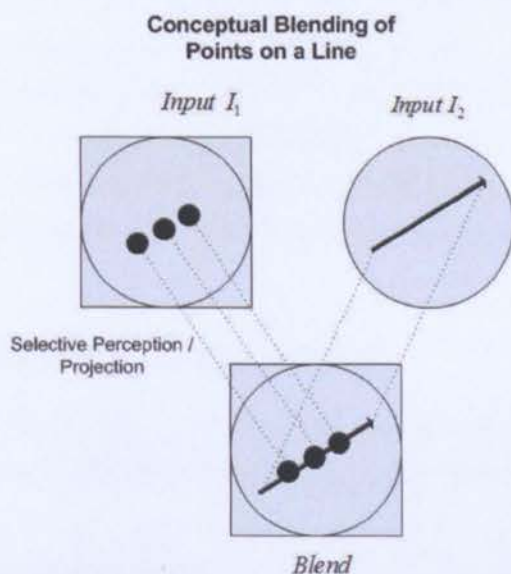


Figure 2–13 Conceptual blending of points on a line, requires a material anchor (the line) shown as enclosing box to blend the input spaces of I_1 and I_2 using the gestalt principle of linearity and the conceptual structure notion of sequential order to construct the queue of points such that they can be counted. In Hutchins’ interpretation, the external environment is a material anchor for the points themselves (Hutchins, 2005, p1561).

Chapter 2. Literature Review

- 2) Next, the gestalt principle of linearity makes the line configuration perceptually salient; however, human perceptual systems have a natural bias to find line like structures.
- 3) Using *composition*, there is a conceptual blending of physical structure of the line with the imagined trajector which enables the experience (phenomenologically) of the emergence of the ordered queue from the line. This ordering of elements was not present in either I_1 or I_2 . The vital relations in the composition are the trajector and the virtual locations.
- 4) Using *completion*, other elements are recruited so that experiencing the queue makes sense in the cultural context for example the symbols and their values: one, two and three.
- 5) Using *elaboration*, once there is emergence of a queue, other reasoning options become possible such as which one is first, which one is in the middle, and which one is bigger, which one is smaller.
- 6) In the standard strategy for counting objects in the queue, attention is applied to each object, which is accompanied by a transition on a sequence of number names (Hutchings, 2005, p1561). Orientation of the objects in the queue on the trajector, are maintained with sufficient immutability, allowing the counting operation to be applied to each one. This counting takes place in a mental space i.e., not in the representation of the external environment (Hutchings, 2005, p1562).

In Hutchings approach, when using material anchors, there is no need to posit a separate mental representation of the material structure of the input space I_1 , since doing so would obscure the use of the environment to define its frame⁵⁹ (Hutchings, 2005, p1561). The blending with the material anchor of the “line” facilitates stability, which enables the more complex operation of counting to take place (Hutchings, 2005, p1562). The physical structure of the input space I_1 is not a material anchor because of the way it is used, and not because it has some intrinsic quality (Hutchings, 2005, p1562).

The benefit of Hutchings’ material anchor approach is the reduction in the amount of cognitive processing required, by a factor of 2 in this number line example (Hutchings, 2005, p1562). Similar benefits would accrue with fictive motion of objects on the number line, in which, for instance, sequences were being predicted (Hutchings, 2005, p1565), as well as the impact of cultural on the conceptual blending (Hutchings, 2005, p1569).

2.6.3.3 Conceptual Blending: Where is the problem?

There are a number of noted differences with Hutchings approach and that of Fauconnier & Turner (Hutchings, 2005, p1572), as well as conceptual blending overall. Each of these, is covered in turn.

Issues with Where Concepts Come From

The key issue in conceptual blending is how the new concepts are formed (Guhe et al, 2011, p250).

⁵⁹ In the alternate real-space approach, it would be necessary to use two stages of selectivity (Hutchins, 2005, p1561).

Chapter 2. Literature Review

Issues with Material Anchors

Fauconnier & Turner extend Hutchings notion of material anchors to include spoken and written language (Fauconnier & Turner 2002, p210–211), implying that “every symbol and sign is a material anchor for a conceptual blend,” which Hutchings rejects on the basis that cross-space mappings are not representations (Hutchings, 2005, p1572). For Hutchings, abstract symbols will appear as the weakest material anchor. For instance, in the example of the number line, the name of the object is the weakest material anchor. In spoken or written language, the temporal or spatial organizations of sentences are the material anchors for some portion of the grammatical conceptual structure (Hutchings, 2005, p1573). Similarly, a sign (§ 2.3.5 on Piagetian signs) is a very weak form of material anchor. This is clearly in line with Piaget’s notion of meaning (§ 2.3.6.1 on Piaget i.e., the theory of objective meaning is not a representational form of meaning). For Piaget, the strongest referent is the counting mechanism that counts the value of the object, since it enables the internalized operation.

Hutchings concludes that “conceptual models embody/express/hold constraints about conceptual elements. In order to play a role in the reasoning process (which is elaboration), a conceptual model must be cognitively stable” (Hutchings, 2005, p1572). This stability is garnered by simplicity, culture (social) and its material or imaginary anchors (Hutchings, 2005, p1575).

The benefits of a conceptual model, be it symbolic, spatial or image, is that it builds the constraints of the task into the structure of the model. Hutchings suggests that the first conceptual models to arise were not symbolic (Hutchings, 2005, p1575), but were material or culture (social). Thus, language emerged after the material anchors had been internalized.

Issues with Double Scope Conceptual Integration and Language

It is the assumption “A language already has all the grammatical forms it needs to express almost any conceptual blend” (Fauconnier and Turner, 2002, p365) that causes problems. Consider how the child learns a language and develops their grammatical forms. The statement by Fauconnier and Turner implies that the language that the child uses is already fully learnt, with enough structure for learning, but from where did the language originate?

Issues with Having Too Much Built In

The approach taken by (Fauconnier & Turner 2002) is too prescriptive, their theory lacks the formalism to describe how the system develops autonomously, bootstrapping from simpler forms. If they had provided this mechanism, then their theory would be a candidate solution to resolve the learning paradox.

Issues with a Lack of an Implementation Model

Chapter 2. Literature Review

There is little mention of learning and development. The examples given are at a very high level. This may be adequate for discussing metaphors, but it makes it difficult to determine an implementation mechanism.

2.6.4 Conceptual Blending as a Basis for Mathematics

In crucially asserting, “blending is not algorithmic” (Fauconnier and Turner, 2008, p59, Fauconnier and Turner contend that new structure spontaneously arises through rethinking, without providing a mechanism for this emergence. Yet, some mechanism must generate this new structure. In addressing this issue, Guhe, from the perspective of multiple worlds, proposes that a combination of *structure mapping theory* and *anti-unification* combined with a theory of intuitions (from *conceptual blending*) provides a valid algorithm for mathematical discovery and the construction of mathematical concepts (Guhe, Pease, Smaill, Martinez, Schmidt, Gust, Kühnberger and Krumnack, 2011, p251).

Guhe’s work is based on the assumption that metaphorical mappings are a special case of conceptual blending (Guhe, 2011, p251), and that metaphor (Lakoff and Johnson, 1980; Lakoff and Núñez, 2001) alone is not sufficient to account for how mathematical concepts develop in an individual (Guhe et al, 2011, p251). Guhe’s work suggests that concepts, and specifically mathematical concepts, evolve over time in children (Guhe et al, 2011, p250).

By taking a different approach than the traditional view of how people make cognitive mathematical discoveries using deduction, mathematical induction, abductive reasoning (Liebeck, 1984), Guhe makes the assumption that analogy, metaphor and conceptual blending are the core processes used (Guhe et al, 2011, p250). In doing so they define an algorithmic process of discovery using conceptual blending that consists of exploration, recognition/goal formulation and discovery using a first order predicate logic (Guhe et al, 2011, p250).

The aim of Guhe’s work is two fold: 1) To ground the mathematical notions in basic cognitive capacities and 2) To define the relationship of these cognitive capacities to abstract arithmetic conceptualizations (Guhe et al, 2011, p249).

An analogical mechanism for mapping between conceptual domains is implemented using *structural mapping theory* (Gentner, 1983; Gentner and Markham, 1997, cited in Guhe et al, 2011, p250), where the theoretical models for concepts is based on a combination of mental spaces (Fauconnier, 1985), conceptual blending (Fauconnier and Turner, 2002; 2008), as well as the underlying metaphors (Lakoff and Johnson, 1980; Lakoff and Núñez, 2001).

Guhe et al. apply *Anti-unification* theory (Plotkin, 1970) in the form of heuristic driven theory projection (HDTP) to enable a mapping between source and target domains via a computed generalization (Guhe et al, 2011, p250). HDTP works by iteratively computing the least general generalization that subsumes each formulae (that maps a domain), using a first-order logic language (Guhe et al, 2011, p250). In doing so HDTP sidesteps the limitations of classical *anti-unification*’s failure to detect structural commonalities. Lakoff and Núñez identified arithmetic as being a conflation of the object collection and object construction metaphors

Chapter 2. Literature Review

(Lakoff and Núñez, 2001, p96), commented: use metaphors of motion along a path (MOP), the measuring stick (MS) for their research on fractions (Lakoff and Núñez, 2001, p69 and 71).

By assuming the innateness of the basic metaphors of arithmetic, Guhe sidesteps the critical issues concerned with how new concepts are formed (Guhe et al, 2011, p250), and wraps these metaphors into conceptual frames consisting of a first order predicate calculus with explicit use of the existential quantifier (Guhe et al, 2011, p256). This allows for the formation and evaluation of various theories manually constructed by conceptually blending a prescribed set of concepts and heuristics.

Guhe provides a worked example of a conceptualization of rational numbers using points on a mental number line, to explain how the measuring stick, *object construction* and *motion along a path* metaphors are blended into a conceptualization (Guhe et al, 2011, p255). In general terms this explains why conceptual blending is a valid mechanism for mathematical discovery that may find a parallel in neurophysiology (Guhe et al, 2011, p264).

In the future, researchers may attempt to test the partial implementation with a larger number of cases studies (Guhe et al., 2011, p263), though there is no hint that it will be attempted to implement this within a machine learning, or conversely within an existing framework such as HR (Colton, 2002).

2.6.4.1 Conceptual Blending: Where is the Problem?

In making use of a first order predicate calculus (Guhe et al., 2011, p255), Guhe sidesteps Fodor's arguments for LP1. Further, it has not been observed that children have an awareness of the first order predicate calculus, so their mechanism cannot be used as an implementation for a Piagetian model. Lastly, Piaget considered metaphor as analogous to symbol which was constructed by the symbolic/semiotic function, bootstrapping from lower levels of complexity (Furth, 1970, p138).

2.6.5 Conclusions

This section provided an evaluation, with examples, of both metaphors (Lakoff and Johnson, 1980; Lakoff, 1992; Lakoff and Núñez, 2001; Lakoff and Johnson, 1999) and conceptual blending theory as they apply to concept formation in mathematics (Fauconnier and Turner, 2002; Fauconnier and Turner, 1996; Fauconnier, 1985; Fauconnier, 1997; Hutchins, 2005; Langacker, 1987; Mithen, 1996; Turner, 2011).

Research presented by Fauconnier and Turner, show that humans use metaphors when communicating with others and that blending theory can classify metaphors and describe the processes used to create⁶⁰, modify and destroy them (Fauconnier and Turner, 2002). A description of the relationship to linguistic forms is provided, but the theory lacks a coherent formalization of how those same linguistic forms are generated (Fauconnier and Turner, 2002, p365). Analysis reveals that the authors of conceptual blending theory understood the need for a biological basis for emergence (Fauconnier and Turner, 2002, p146), and the

⁶⁰ This creation is assumed to be from earlier, more primitive forms, but there is no mention if how these earlier forms are made.

Chapter 2. Literature Review

appearance of an emergent logic (Fauconnier and Turner, 2002, p44). The key issue is how are these new concepts are formed (Guhe et al., 2011, p250). In the examples of the development of mathematical sense presented by Guhe (Guhe et al., 2011, p249), their research uses a first order predicate logic, which is more capacity than is observed in childhood. These metaphors may be at a conceptual level different from the concepts envisioned in this research thesis. Piaget did account for metaphors in later stages of development (Furth, 1969. p59), and these could be considered in future work. The double scope blending (Fauconnier and Turner, 2002, p365) approach differs with Piaget's observations on the development of language using assimilation and accommodation (§ 2.3.6). For Piaget, the set of schemes is not fully prescribed, as in the formulation by Fauconnier and Turner. It may be that double scope conceptual blending is the way humans develop language. The research in this thesis may provide the basis of a mechanism for testing the development of conceptual metaphors; however, that is not the purpose of this research.

2.7 Cognitive Development Models and the Need for Emergence

“There are side effects that deserve attention too. Any program that will successfully model even a small part of intelligence will be inherently massive and complex” (Drescher, 2002, pii).

In this section, cognitive development models are reviewed (Duch, Oentaryo and Pasquier, 2008, p123) to determine if any of these models provide a suitable basis for resolving the learning paradox and its constraints (LP1 – LP7). Also addressed is the worked example (WE1 – WE5) in order to identify several issues that need to be addressed by this thesis. A brief introduction follows in § 2.7.1 after which symbolic models, the issues associated with symbol grounding, Searle's Chinese room argument and various ways of resolution are examined in § 2.7.2. Emergent and hybrid models using examples are illustrated in § 2.7.3, along with dual-process theories and conceptual change, in § 2.7.4. Conclusions follow in § 2.7.5. Further analysis of conceptual models is provided in Appendix B.

2.7.1 Foundations

“What distinguishes generally intelligent entities is their ability to solve not just a single problem using a specific method, but the ability to pursue a wide variety of tasks, including novel tasks, using large bodies of diverse knowledge, acquired through experience, in complex dynamic environments” (Laird, 2008, p225).

2.7.1.1 Background

A cognitive model embodies a scientific hypothesis about those aspects of human cognition, which are relatively independent of task and constant over time (Langley, Laird and Rogers, 2008, p142)⁶¹. The cognitive

⁶¹ Laird's research provides support for the definitions of LP3, LP5 and LP7.

Chapter 2. Literature Review

architecture assigns a design model, which poses constraints on the implementation, providing a definition of an abstract machine to support programming of a cognitive model. The function of the abstract machine is to provide the most useful set of processes and representations for developing such models and the machine usually comes commented: with a programming language to build those components (Langley, Laird and Rogers, 2008). These components provide the underlying structure to tackle a wide range of integrated research and commercial problems that require increasingly intelligent cognitive behavior.

2.7.1.2 Why a Solution is Needed

Beginning with production systems in the early 1970's and, despite the development that has occurred to date in many disciplines, including simulated training, computer tutoring systems and interactive computer games; there is increasingly a need for more general intelligent systems (Langley, Laird and Rogers, 2008, p142).

2.7.1.3 What is the composition of that solution?

Langley, Laird and Rogers posit a number of abstract features that enable systems, composed of them, to exhibit intelligent behavior (Langley, Laird and Rogers, 2008, p145). A system requires the ability to recognize in dynamic situations, those patterns or events as similar situations. Recognition requires categorization, and recognition is closely related to the assignment of objects, situations and events to existing concepts. Other features include decision-making and choice, perception and situation assessment, prediction and monitoring, problem solving and planning, reasoning and belief maintenance, execution and action, interaction and communication, remembering, reflection and learning (Langley, Laird and Rogers, 2008). The architecture should also contain a set of properties, namely: representation of knowledge; organization of knowledge; utilization of knowledge; acquisition and refinement of knowledge (ibid, p142). The architecture should also contain evaluation criteria of cognitive architectures, namely: generality, versatility and task ability; rationality and optimality; efficiency and scalability; reactivity and persistence; improvability; autonomy and extended operation. The fact that these features should be found in any workable solution in one form or another is not disputed (Georgeon, Ritter and Haynes, 2009).

2.7.1.4 A Suitable Problem.

If one were to account for much of natural growth, with a cognitive model of learning and development, then most of what is needed to resolve the learning paradox could be tested in a simplistic simulation (WE1 – WE5), albeit with an as yet, unresolved complex infrastructure. The key problem to address is the capacity of the cognitive model, as a specification of those principles imposed by the problem (LP1 – LP7), to resolve the learning paradox.

“A cognitive architecture specifies the underlying infrastructure for an intelligent system”
(Langley, Laird and Rogers, 2008).

Chapter 2. Literature Review

2.7.1.5 Grading Cognitive Theories

Anderson and Lebiere propose using Newell's 12 Criteria Test to measure the high level capacity of cognitive theories to exhibit the underlying behavior that is representative of human cognitive processes (Anderson and Lebiere, 2003, p579). These criteria are themselves constraints for the whole of cognitive modeling, which have largely defined the development of cognitive models from 1980 onwards. To implement a full human intellectual capacity Anderson and Lebiere suggest that the architecture must satisfy a minimal set of cognitive architectural functional achievement criteria. These are the following: behave as an (almost) arbitrary function of the environment as a universal computational machine, operate in real time, exhibit rational i.e., effective adaptive behavior, use vast amounts of knowledge about the environment, exhibit dynamic behavior⁶², integrate diverse knowledge both sub-symbolic and symbolic, use (natural) language, exhibit self-awareness, a sense of self and learn from its environment. To achieve this human like capacity, the architecture must also satisfy a minimal set of cognitive architecture implementation criteria including: acquiring capabilities through development; arising through evolution and being realizable within the brain. The criteria emphasize that any cognitive architecture must have a correspondence to human cognition and be capable of practical application in a domain such as education (Anderson and Lebiere, 2003, p579).

2.7.1.6 Problems with Newell's 12 Criteria Test

Even though Newell's 12 Criteria Test is not exhaustive, there are noted difficulties. Agassi points to the problem of not understanding how natural language occurs (Anderson and Lebiere, 2003, p594). This is seen as a fundamental limitation of existing systems. Agassi concludes that perhaps a simpler problem should be tackled, one whose solution could improve the overall approach. It is within this frame that the problem set for this research is cast, namely the development of number-sense.

Commons and White suggest that Newell's 12 Criteria Test should include stage like variations, where actions at a higher order of hierarchical complexity are defined in terms of lower order actions, organize and transform lower stage actions and which solve more complex problems through the non arbitrary organization of actions (Commons and White, 2003, p606). Commons and White's arguments run parallel to Bereiter and his examples of second language development, being able to use knowledge critically and rational numbers (Bereiter, 1985). There is also agreement with Piaget and the need to develop hierarchical structures (Furth, 1969 p38; Copeland, 1974, p62).

Newell's 12 Criteria Test (Newell, 1990), have been analyzed and applied to ACT-R, SOAR and classical connectionist architectures but such fine-grained categorization makes comparison of different systems rather difficult (Duch, Oentaryo and Pasquier, 2008, p123). Duch, Oentaryo and Pasquier reclassify the cognitive architectures into symbolic, emergent and hybrid models. It is within this framework and through

⁶² Dynamic behavior includes the capability to behave robustly in the face of error, the unexpected condition and the unknown.

Chapter 2. Literature Review

the constraints of the learning paradox (LP1 – LP7) where the cognitive architectures and models are examined.

2.7.2 Symbolic Models

“Few existing cognitive architectures have the capability to create new symbolic structures – they are a prisoner of the original encodings provided by a human programmer” (Laird, 2008, p232).

One of the most difficult challenges, and this is true of SOAR, is to provide necessary initial structures without compromising flexibility so that the agent can adapt (learn) the specifics of the task and environment (Laird, 2008, p225). By using predicate logics to support both the long term and short-term memories, as well as the structuring of operators, SOAR, like ACT-R and other symbolic architectures, assume the “hypothesis that there are useful abstractions and regularities above the level of neurally based theories” (Laird, 2008, p225). Therein lies the problem of the learning paradox. To resolve the paradox, the system must be capable of generating a structure more complex than that which it already contains. Using a top-down, analytic approach, symbolic architectures focus on information processing using high-level symbols with declarative knowledge (Laird, 2008). These structures provide a representational model of reality and it is questionable if representation alone is sufficient.

Examples of symbolic models include PRODIGY (Carbonell, Knoblock and Minton, 1990), SOAR (Laird, Newell and Rosenbloom, 1987; Newell, 1990; Wray and Jones, 2005), EPIC (Meyer and Kieras, 1997; Kieras and Meyer, 1997), ICARUS (Langley and Choi, 2006; Langley and Messina, 2004), FORR (Epstein, 1992), AIS (Hayes-Roth, Pfeleger, Lalanda, Morignot and Balabanovic, 1995), APEX (Freed, 1998, p922), CIRCA (Musliner, Goldman and Pelican, 2001, p2124), Emile (Gratch, 2000), ERE (Drummond, Bresina and Kedar, 1991), GLAIR (Shapiro and Ismail, 2003), 3T (Bonasso, Firby, Gat, Kortenkamp, Miller and Slack, 1997) and RAA (Pell, Bernard, Chien, Gat, Muscettola, Nayak, Wagner and Williams, 1997).

However, there is a key issue that has bearing on the use of symbolic models and that is the symbol grounding problem.

2.7.2.1 Symbol Grounding Problem

The symbol grounding problem is the difference between human cognition and cognitive modeling using computer systems, whereby a computer system, that merely contains arbitrary symbols, is not be capable of approaching understanding (Harnad, 1990, p337). Even after 15 years of research, all of the current strategies are semantically committed and hence none of them provides a valid solution to the symbol grounding problem (Taddeoa and Floridi, 2005). Any solution that attempts concept formation would need to resolve this problem and, by consequence, satisfy LP2.

Human Vs. Computer Understanding

Chapter 2. Literature Review

Harnad observed that humans have a behavioral capacity (Harnad, 1990). Through this behavioral capacity, discrimination⁶³, manipulation, identification⁶⁴, description of objects and descriptions of events and states are associated to their names. Along with the capability to respond, the human mind conceives syntactic symbols and attaches a semantic meaning to those symbols. Through this, a human mind can be said to understand.

By comparison, computer systems are syntactic, in that they consist solely of strings of symbols (Harnad, 1990). The symbol is arbitrary and is not related to its meaning or content. Because no amount of syntax will ever produce semantics, there is no mechanism that a computer system running a symbol only program will ever be able to understand. Without sub-symbolic to symbolic processing and the capability to learn and develop, the purely symbolic cognitive architectures fail LP2.

2.7.2.2 McCarthy as an Early Symbolic Model

As an early example of a symbolic system, McCarthy's "advice taker," was to form its experience as effectively as humans, using a trial and error approach to learning with innate structures consisting of recursive expressions in a predicate logic (McCarthy, 1968). Whereas a human is instructed mainly in declarative sentences which describe the situation in which action is required together with a few imperatives that say what is wanted, "advice taker" was instructed mainly in the form of sequences of imperative sentences (McCarthy, 1968, p406). Bar-Hillel identifies issues with its transitive relations, the lack of temporal relations and the use of deductive logic (McCarthy, 1968, p11). This is in agreement with Piaget's observations that children have innate schemes and primary reactions but that logics appear through development (Copeland, 1974, p183; Furth, 1969, p216).

Noting that humans are the product of millions of years of evolution, McCarthy attests that the world's structure is not directly describable in terms of the input-output relations of a person (McCarthy, 1997, p5). This is an extension of earlier work on "advice taker" and suggests that the adequate structures for creating a well-designed child are more complex than Chomsky's universal grammar. The implication is that there is also a set of definitions that describe how the world works, a set of mental characteristics and a set of abilities that would be useful to the developing simulation along with a set of features of a language of thought (McCarthy, 1997, p18).

When McCarthy's position is compared to the research constraints (LP1 – LP7), several issues arise. First, McCarthy was attempting to create a robot with child-like capabilities. The research that is the content of this thesis only seeks to resolve the learning paradox and create a more complex structure in a number line world. Thus, some of the arguments for definitions of how the world works seem not to be necessary, such as distributed mechanisms and senses. In terms of innate abilities, McCarthy's system includes a predicate logic,

⁶³ Discrimination is through sub-symbolic iconic representations.

⁶⁴ Identification of the invariant features as categories, as classes of objects or from the discrimination of iconic sub-symbolic representations.

Chapter 2. Literature Review

so fails Fodor's arguments for **LP1**. It does not seem to need to exhibit hierarchical concepts, so fails **LP2**. It also does not seem to have a trajectory of matching the observations of a developing child, so may fail **LP4**, through if it is based on "advice taker," it may learn incrementally and so support **LP5**. In McCarthy's 1997 paper, it could not be seen that it needed to develop its learning process and so fails **LP6** (McCarthy, 1997). Finally, and this is a constraint for all reactionary systems, it was intended to work in novel, noisy and opportunistic situations and so may meet (**LP7**). However, McCarthy's research skirts the symbol grounding problem, with addition of a predicate logic.

2.7.2.3 Searle's Chinese Room Argument

The symbol grounding problem is associated with the Chinese Room Argument (CRA) (Searle, 1980; 1984; 1990) in the sense that both are concerned with forms of cognition. The CRA addresses the symbol system hypothesis, which can be formed by a question, "If a machine can convincingly simulate an intelligent conversation, does it necessarily understand?" A depiction makes it easier to understand the arguments and realize that Searle wraps the original Turing Test (Turing, 1950) between two layers of machine translation. Here the Turing compliant program is an instance of a "Script Applier Mechanism" as a story-understanding program (SAM) (Schank and Abelson, 1977). It is this, which Searle takes for his example (figure 2-14).

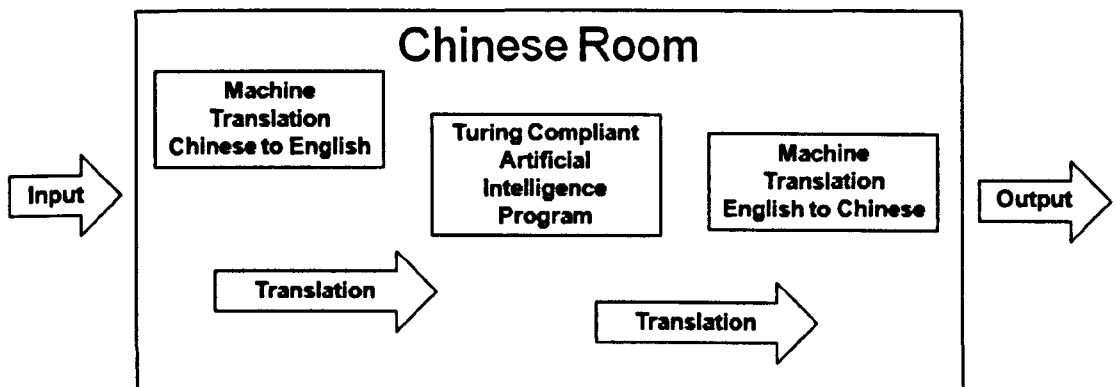


Figure 2-14 A depiction of Searle's Chinese room argument (Searle, 1980, p417)

With the 1980 and 1984 work frequently reviewed and often misquoted, Searle summarized the main argument (Searle, 1999), as:

"Imagine a native English speaker who knows no Chinese locked in a room full of boxes of Chinese symbols (a data base) together with a book of instructions for manipulating the symbols (the program). Imagine that people outside the room send in other Chinese symbols, which, unknown to the person in the room, are questions in Chinese (the input). And imagine that by following the instructions in the program the man in the room is able to pass out Chinese symbols which are correct answers to the questions (the output). The program enables the person in the room to pass the Turing Test for understanding Chinese but he does not understand a word of Chinese" (Searle, 1999).

Chapter 2. Literature Review

For Searle, the person in the Chinese room is doing exactly what a computer would be doing if it used the same rules to engage in a grammatically correct conversation in Chinese. Thus, if manipulating Chinese symbols according to formal rules is not sufficient for the person to understand Chinese, then it is not sufficient for a computer to understand Chinese either, since both are engaging in “mindless” symbol manipulation (Searle, 1990, p27). The symbol system hypothesis states that the key aspects of intelligent behavior are properties of formal symbol systems. For Searle, computation, as it is standardly defined (in terms of the manipulation of formal symbols), is not, by itself, constitutive of, nor sufficient for, thinking. The syntax of the implemented program is not the same as the semantics of actual language understanding.

A reading of some critiques of CRA including those that were part of the initial paper (Searle, 1984; 1990; 1999; Fodor, 1991; Churchland and Churchland, 1990; Rey, 1986) provides three common misconceptions. First the interpretation that “computers cannot think” is restated into computation as standardly defined in terms of the manipulation of formal symbols is not by itself constitutive of, nor sufficient for, thinking. Second, the misunderstanding that “machines cannot think” is corrected by the following argument. The brain is a machine. If a machine is defined as a physical system capable of performing certain functions, then there is no question that the brain is a machine. Since brains can think, it follows immediately that some machines can think. Last, the concept that it is “impossible to build a thinking machine” is restated. If one assumes that thinking is caused by neurobiological processes in the brain and since the brain is a machine, there is no obstacle in principle to building a machine capable of thinking. Furthermore, there is no theoretical argument against the possibility of building a thinking machine out of substances unlike human neurons.

What CRA shows is that it is not possible to build a thinking machine using a formal symbol system. Harnad and Searle conclude that the CRA deals a knockout blow to strong AI⁶⁵ (Harnad, 1993). This can easily be seen with an example of an early symbolic cognitive architecture, the SOAR project (Laird, Newell and Rosenbloom, 1987). The implication for this research is to answer the question “Is it possible to use a different medium, in which to evolve a symbol system?”

2.7.2.4 SOAR

The goal of SOAR was to define an architecture for a system that was capable of general intelligence (Laird, Newell and Rosenbloom, 1987). SOAR is fundamentally a search architecture (Laird, Newell and Rosenbloom, 1987). Its knowledge is organized around tasks, which it represents in terms of problem–spaces, states, goals and operators. SOAR provides a problem–solving scheme as a means to transform initial problem states into goal states. Laird, Rosenbloom and Newell consider universal sub–goaling, which is the property of being able to do problem solving to make any decision, to be one of the most important contributions of

⁶⁵ Strong artificial intelligence is the view that suitably programmed computers (or the programs themselves) can understand natural language and actually have other mental capabilities similar to the humans whose abilities they mimic.

Chapter 2. Literature Review

SOAR. This problem solving capacity is referred to as the “problem space hypothesis”⁶⁶ (Newell, 1980b). In addition, SOAR is based on the physical symbol system hypothesis⁶⁷ (Newell, 1980a, p135). As previously stated, there are problems with the symbol system hypothesis and as will be shown, there are also issues with the problem space hypothesis as well.

The problem space hypothesis assumes that all symbolic cognitive activity can be modeled as heuristic search in a symbolic problem space. In particular, Newell claims that reasoning, problem solving and decision making can all be captured as searches in appropriately defined problem spaces. The problem space is usually an implicit space defined by the combination of all possible states with the transformations similarly atomized and encoded. In this space, an initial state and a goal state are specified and the abstract task is to find, using trial and error a path from the initial state to the goal state via the allowed transformations. The search need not necessarily be blind trial and error, but can use heuristic information to enhance the process; it might even become algorithmic. However, this not only presupposes encoding in the presumed implementation of the problem space, it inherently restricts all such variation and selection searches to the combinatoric possibilities given by generating the set of atomic encodings. In particular, there is no possibility in this view of generating new “emergent” representations as trials toward possible solutions of the problem, as possible satisfiers of the goal criteria. The only representational states allowed are the syntactic combinations of already available atomic “representations.” In this respect, as Bickhard points out, SOAR is over committed to Fodor’s necessary innateness argument (Bickhard, 1991a; 1991b) and therefore fails LP1.

In contrast, CRA does not impact weak AI⁶⁸ and it is within this weak AI approach that this research thesis is undertaken. In fact, the experimental position of the research could be cast into a CRA, with “Chinese” being replaced with integer and number–sense. The key issue is overcoming the physical system hypothesis and its pseudonym the symbol–grounding problem.

2.7.2.5 Resolution of Symbol Grounding Using Sub–Symbolic Sensory Data

To resolve the symbol grounding problem and overcome the physical symbol system hypothesis, Mayo advises the use of sub–symbolic sensory data (Mayo, 2003, p57). The sensory data (of experience) upon which the symbols are grounded is continuous and fluid. Artificial neural networks can be used to provide non–

⁶⁶ The problem space hypothesis, assumes that all symbolic cognitive activity can be modeled as heuristic search in a symbolic problem space (Laird, Newell and Rosenbloom, 1987, p58).

⁶⁷ The physical symbol system hypothesis: “A physical symbol system has the necessary and sufficient means for general intelligent action” (Newell and Simon, 1976, p116). The physical symbol system hypothesis is also called a formal system.

⁶⁸ Weak artificial intelligence claims only that the computer is a useful tool in studying human cognition, as it is a useful tool in studying many scientific domains. Those computer programs which simulate cognition will help us to understand cognition in the same way that computer programs which simulate biological processes or economic processes help us understand those processes.

Chapter 2. Literature Review

symbolic representations of the environment. A category is thus a selection of continuous sensory data, but without some sort of bias, the selection of category is computationally intractable.

A computer system could acquire its symbols as it learns new tasks, the bias is then provided by the task (Mayo, 2003, p 57). The symbol is essentially discrete and static and is an arbitrary assignment to a category of sensory data that can be detected in the environment through a process of decontextualization. The rules for reasoning, which are themselves the combination of strings of symbols, are themselves arbitrary symbol strings. A computer system typically has pre-programmed categories in its sensory motor apparatus and constrains which categories can be developed. By enabling a computer system to develop its own categories, would enable the system to side step the symbol grounding problem.

Symbols can be used to designate abstract concepts, but the issue is where do the abstract concepts come from. Connectionism could provide the important sub-symbolic categorizations; another approach is theory tethering (Sloman, 1985).

2.7.2.6 Resolution using Theory Tethering

On addressing symbol grounding and specifically concept formation, Sloman proposed, “a machine may use symbols to refer to its own internal states and to abstract objects” (Sloman, 1985, p1000). Sloman indicated how it might refer to a world to which it has only limited access, relying on the use of “axiom-systems to constrain possible models and perception-action loops to constrain possible completions” (Sloman, 1985, p1000). In concluding that these constraints leave the “meanings” partly indeterminate and indefinitely extendable and even that some use of causal links reduce some of the indeterminacy, Sloman opens up the resolution of the symbol grounding portion of the learning paradox through a Piagetian approach to causality. For Piaget, the symbol grounding problem is the development of meaning through action. For Sloman, understanding is more than a collection of symbols:

“Without a functional architecture supporting distinctions between beliefs, desires, plans, suppositions, etc., a machine cannot assign meanings in the way that we do. Merely storing information and deriving consequences, or executing instructions, leaves out a major component of human understanding, i.e., that what we understand matters to us” (Sloman, 1985, p999).

Further, Sloman identifies some key areas that would need to be addressed, including the nature of truth, mathematical concepts like “not, all, or, number, minus” that do not directly refer to experience, the incompleteness of semantics, concept learning, abstract concepts as references to inaccessible objects and generalization amongst others (Sloman, 1985).

In rejecting the existing mechanisms to resolve the symbol grounding problem and its earlier definition of concept empiricism, Sloman concludes that AI researchers must not restrict their ontologies to patterns and

Chapter 2. Literature Review

relationships in artificial neural networks (Sloman, 1985). Further and this follows Piaget⁶⁹, they must not ignore the structure of the environment (Sloman, 1985). In addition, concept formation, which is the essential notion of symbol grounding, requires that a successful system must be able to generate new concepts. This approach coincides with the view of emergent behavior (Crutchfield, 1994a, 1994b) as well as evolutionary development (§ 2.2 and A, for a fuller discussion on emergence) in the learning process.

For Sloman, symbol tethering “shows that events and processes in a machine can constitute a model for a significant subset of “axioms implicitly defining mentalistic concepts” (Sloman, 1985, p996). As an example, Sloman constructs a Euclidean model, as a set of axioms of points, lines, intersects, and explains the different ways that this model can be used to interpret the world. The model for this set is not, as sometimes suggested, another symbolic structure denoting the world. These models are portions of the real world (Sloman, 1987, p372). It is symbol tethering which constructs understanding through the rich semantics expressed in a formalism that allows conclusions to be drawn in a formal way. Thus, when a machine interrogates its memory, states of the memory can cause appropriate new symbols to be constructed and stored (Sloman, 1987, p370).

Theory Tethering: Where is the Problem?

There are issues with the theory tethering⁷⁰ approach, especially if one were to try to resolve **LP1 – LP7** (§ 2.1.7), then a number of issues arise. First, it is necessary to work within the constraints imposed by Fodor’s arguments. It is evident that there is too much built in to Sloman’s formal system, so theory tethering fails **LP1**. Second, any system that purports to exemplify human cognition would need, certainly in an educational sense, to be able to recreate the milestones through which a child grows through (§ figure 1–1 on Bead Problem). Sloman must also be careful to ensure that these “concepts” are not forced into having a “formal” logical construction, if they have an operational meaning that is relevant to causality or basic sets. This is not to argue that such concepts are trivial, for this would not justify working so long with “sets” in primary schools if this was so. There is no evidence that theory tethering could successfully resolve **LP4** and it is the relevance to educational research that typifies the complexity of the research problem. The problem is not that Sloman’s formal model with “non-derivative” semantics could exhibit emergence of hierarchical structures, so as to resolve **LP2**. The issue is having adequate examples of where the systems that implement theory tethering have been shown to exhibit the emergence of new structure and so resolve **LP2**.

⁶⁹ A functional exchange between a biological organization and the environment presupposes an internal structure and leads to a structuring of the environment. For Piaget, action is not limited to external action. It is generally synonymous with behavior (Furth, 1969, p259).

⁷⁰ Theory tethering is also referred to as symbol tethering.

2.7.2.7 Resolution of Symbol Grounding using SOAR–R

One potential solution to the symbol grounding problem and **LP2** is to provide hybrid cognitive modeling systems that learn their symbols using bottom up learning (Georgeon, Ritter and Haynes, 2009). This is covered in more detail later (§ 2.7.4).

2.7.2.8 Resolution using Piagetian Approach

Piaget disagreed with the whole representation theory of knowledge upon which most symbolic processing is built (Furth, 1969, p93). Piaget asserted that the whole of knowledge is based on action and that this action, through the processes of assimilation and accommodation differentiates itself into semiotic and latterly the symbolic functions through a process of meta–genesis (§ 2.3.5 on Piaget Knowing Circle).

2.7.2.9 Symbolic Models: Where is the Problem?

By utilizing predicate–based logic for reasoning and optionally, planning, symbolic models such as EPIC, CIRCA, EMILE, RAA and 3T all fail Fodor’s argument for **LP1**. First, these most often require specific domain knowledge to generate an understanding of the required problem space. Second, the developer of the solution provides this domain model. Third, they do not exhibit the learning and developmental characteristics of a young child (**LP4**), nor exhibit the stage like variation. Lastly, many of the system use means end analysis, which earlier research has demonstrated to be necessarily included apriori. In this sense, they also fail **LP1**.

However, symbolic models exhibit interesting features.

First, RAA (Pell, Bernard, Chien, Gat, Muscettola, Nayak, Wagner and Williams, 1997) uses a constrained resource limitation feature similar in nature to Pascual–Leone’s M–Demand (Pascual–Leone, 1980, p271).

Second, the autonomous operation of 3T systems is due in large part to the embodiment of the principle of cognizant failure, in each layer (Gat, 1998). Cognizant failure is based on the idea of designing a system to detect failures and recover from them as opposed to never failing (Gat, 1998, p198). Originating on Firby’s research on adaptive control (Firby, 1989), like nature, it sidesteps the issue of designing perfect algorithms, but requires a contingency recovery procedure for each failure. It also presumes that the multiple possible outcomes of actions are easily categorized as success or failure and those failures are inevitable. The benefits of implementing this approach are shown by the 3–year autonomous operation of 3T systems (Bonasso, Firby, Gat, Kortenkamp, Miller and Slack, 1997).

Systems such as PRODIGY (Carbonell, Knoblock and Minton, 1990) have made use of CBR and EBL algorithms, to learn new search control rules. As will be described in § 2.9.1, these approaches cannot discover new facts, only recombine existing facts into new combinations and as such fail **LP5**.

2.7.3 Emergent Models

“The challenge for the future is to develop general artificial systems that can pursue a wide variety of tasks (including novel ones) over long time scales; with extensive experiential learning in dynamic, complex environments... cognitive architecture provides the building blocks to meet this challenge” (Laird, 2008, p234).

Emergent models use low-level activation signals flowing through an artificial network consisting of numerous processing units, relying on the emergent self-organizing and associative properties of connectionism (O’Reilly and Munakata, 2000). Within the space of emergent architectures, there exists a broad range of biological cognitive features that have been transposed into machine learning architectures. Two examples are reviewed IBCA (O’Reilly and Munakata, 2000) and ART (Carpenter, Grossberg and Rosen, 1991; Carpenter and Grossberg, 2002).

2.7.3.1 IBCA

In the area of computational cognitive neuroscience, the goal of Integrated Biologically-based Cognitive Architecture (IBCA) (O’Reilly and Munakata, 2000) is to understand how the brain embodies the mind by using biologically based computational models comprised of special artificial neural networks. Extending parallel distributed processing (McClelland and Rumelhart, 1988) with biologically sound models; IBCA embodies a model of cognitive psychology. Motivated by physical reductionism, reconstructionism, levels of analysis and scaling issues, computational cognitive neuroscience is bracketed by two important questions: 1) How does emergence occur within an organism and 2) How does one encourage emergence within a simulation. Any simulation addressing these questions includes features such as parallelism, gradedness, interactivity, completion and learning using neural networks. By combining these networks, the constructed systems exhibit capabilities such as visual encoding, spatial attention, episodic memory, working memory, semantic representation, task directed behavior and deliberation, as well as features of explicit cognition that are observed in early child development.

IBCA: Where is the Problem?

First, Poldrack makes the case that much of what is interesting is early development – beyond one and a half years of age – is poorly understood, and at age five, cortical patterning is almost completed (Poldrack, 2010). This is readily understood, by the reasoning that correlations of neural imaging do not provide sufficient information to discern what is going on developmentally. In this sense, IBCA fails LP2 and LP6 because current examples do not exhibit hierarchical concept formation nor development of their own learning process.

Second, common inferences made in IBCA are descriptive and often logically incoherent (Poldrack, 2010). The IBCA architecture may well prove to be a more correct model than a Piagetian model. For now though, it seems to be too immature to account for the observations of childhood development that Piaget has

Chapter 2. Literature Review

observed (Furth, 1969; Copeland, 1974) and so fails LP4 . More importantly, it does not tackle the symbol grounding problem, which is evidently approachable within the confines of this thesis.

2.7.3.2 ART

Adaptive Resonance Theory (ART) (Carpenter and Grossberg, 2002) is a theory on aspects of how the brain processes information. It describes a number of neural network models, which use supervised, unsupervised learning and self-organizing methods to address problems such as pattern recognition and prediction (Carpenter, Grossberg, Rosen, 1991). Using the concept of resonance, ART reconciles sub-symbolic distributed features to symbolic categories, as coherent context sensitive states. By using fuzzy methods, these symbols and rules become resonant, i.e., the system becomes aware of these new states through a process of synchronous equilibrium. ART supports dynamic behavior including mimicking of motor skill learning and performance (Grossberg and Paine, 2000; Carpenter, Grossberg, Markuzon, Reynolds and Rosen, 1992).

ART: Where is the Problem?

One criticism of ART is that it has a statistically unacceptable as a method of learning categories – noisy data can produce a vast proliferation of categories with severe over-fitting (Williamson 1995). Over-fitting of data is itself, not necessarily invalid in educational terms, since language learning typically demonstrates over-generalization and incorrect learning⁷¹. The danger occurs when it becomes impossible to correct invalid learning – as identified by Gold (Gold, 1967, p465) – as occurs with learning a grammar. Secondly, performance has a dependence on the parameters chosen for training. The choice of these parameters requires good knowledge of these architectures and often requires some experimentation to get the best results using these architectures, a computationally intensive proposition. Since foreknowledge is required, ART fails LP3. Researchers have tackled this problem by combining a genetic algorithm to evolve the weights in the neural network (Kaylani Georgiopoulos, Mollaghasemi, Anagnostopoulos, Sentelle and Zhong, 2010). Since generalization occurs in the sub-symbolic layers, it seems that the capability to generalize and generate concepts using a form of concept formation (LP2) in the symbolic layers has not been considered. Therefore, its use in solving the learning paradox is put into doubt. Further, it has not been shown that ART could develop its learning process and so fails LP6.

2.7.4 Hybrid Models

Hybrid architectures combine the properties and processes of symbolic and emergent architectures together. Sun suggests that a lack of support for symbol grounding in SOAR and ACT-R, led to the development of hybrid architectures (Sun, 2004).

⁷¹ Over-fitting, as in learning the wrong thing.

Chapter 2. Literature Review

This review of hybrid models initially covers the necessity of dual-process theories (§ 2.7.4.1) and the need for a structure for conceptual change (§ 2.7.4.2). It then reviews a selection of hybrid models (§ 2.7.4.3) and compares them to the needs of the learning paradox (LP1 – LP7).

2.7.4.1 The Necessity of Dual-Process Theory

Pyysiäinen finds it difficult to reconcile criteria one (generality) and criteria twelve (evolution) in Newell's twelve Criteria Test (Newell, 1990), on the basis that an evolved implementation cannot be computationally universal (Pyysiäinen, 2003, p617). By observing that some tasks are more difficult than others, Pyysiäinen posits that this supports dual-process theories (Pyysiäinen, 2003, p617). The disconnected structure in cognitive modeling, which has connectionist models at one-side and production systems on the other, can be positioned on a continuum with hybrid systems somewhere between both approaches. An example of which is CLARION (Sun, Merrill, Peterson, 2001, p205). Dual process theories (Epstein and Pacini, 1999) have been variously labeled and represent two different ways of processing information. Although definitions vary, Pyysiäinen notes that there is considerable overlap in processing, examples include: intuition and implicit learning versus deliberative, analytic strategy (Lieberman, 2000); a reflexive and a reflective system (Lieberman, Gaunt, Gilbert, Trope, 2002, p210); associative versus rule-based systems (Sloman, 1996; 1999); an experiential or intuitive versus a rational mode of thinking (Epstein and Pacini 1999) and implicit versus explicit cognition (Holyoak and Spellman 1993).

The two supporting biological systems of dual process theories are the spontaneous system (using the lateral temporal cortex, amygdala and basal ganglia) and the rational system (using the anterior cingulate, prefrontal cortex and hippocampus). The rational system either generates solutions to problems encountered by the spontaneous system, or it biases its processing in a variety of ways. To quote Pyysiäinen, "A pre-existing doubt concerning the veracity of one's own inferences seems to be necessary for the activation of the rational system" (Pyysiäinen, 2003, p117). The rational system thus identifies problems arising in the spontaneous system, takes control away from it and remembers situations in which such control was previously required. These operations are plausibly considered to consist of generating and maintaining symbols in working memory, combining these symbols with rule based logical schemes and biasing the spontaneous system and motor systems to behave accordingly (Lieberman, Gaunt, Gilbert, Trope, 2002; Pyysiäinen, 2003, p617).

These hybrid models resolve Harnad's symbol grounding problem by allowing a connectionist category to be related to an arbitrary symbol. A problem with this approach is with the assignment of an appropriate symbol. The symbols themselves cannot be fully formed, otherwise one is back to the symbol grounding problem.

Stressing the importance of the bridge between connectionist and the rule based approaches of dual process theories, Wang, Johnson and Zhang discuss having a theory of mind that provides explanations of the development of symbols from the non-symbolic connectionism that exists in the mind (Wang, Johnson and Zhang, 2003, p626). This provides the backdrop for the learning paradox and Piaget in particular (§2.3.5 on the Permanent Object Invariant).

Chapter 2. Literature Review

2.7.4.2 Conceptual Change

In the general problem of concept formation Clancey comes close to identifying the learning paradox and refers to it as “thinking new thoughts” and even referring to Plato’s “unchanging forms in the mind” (Clancey, 2001, p395–400). Clancey appears to provide a loose definition:

“Concepts fall more on a spectrum, both specific and abstract, not either–or. Abstraction, as it is naturally found in human thinking, should not be contrasted with concrete ideas (Arnheim, 1969)... the spectrum is obvious when we consider that not all abstract relations are mathematical. Other kinds of abstract relations are possible, involving causality, classification, temporality, etc. Abstraction, per se, is the relation of one concept to another, not a specific set of relations which are mathematical (e.g., spatial configuration relations)” (Clancey, 2001, p 394).

It is based on the assumption that general conceptual processes exist and there is no need to separate mathematical and other abstractions. In this respect Clancey is much closer to the original Piagetian approach as a scheme of coordinated actions and not just schemes of objects, or where behavior is a recorded structure. The key theoretical notion is that “something other than specific domain facts and rules, organizes changes in human behavior” (Clancey, 2001, p418). This organizing process is abstraction and is applied to the “progressing from previous organizations of behavior” (Clancey, 2001, p418). Learning is thus a transformative process and “occurs as and through action itself.” Clancey concludes that cognitive modeling as a discipline should view knowledge and learning in terms of conceptual coordination using a neural based process in which sensory motor systems and abstraction is blended, sequenced and composed. The key issue is how this occurs within the individual (Clancey, 2001).

Formally, one can write the symbols as a model in a grammar (such as ENBF), but the question remains, “How did an organism such as a human create such sophistication, grounded as it is within the substrate of a neuronal network?” This alludes to the symbol grounding problem (Harnad, 1990). Similarly, if cognitive architectures embody the framework: “people solve problems by means end analysis,” then this poses a question about the nature of the knowledge of means end analysis in cognitive skills, because if the knowledge of means end analysis is required for the process of skill acquisition, then it must be apriori knowledge. If it is not then would one learn anything, since theories such as ACT–R (Anderson, 2007) require means end analysis for learning. If it is apriori, then the philosophical arguments about conceptual change hypothesis and knowledge apply. If one assumes that people can learn in ways other than means end analysis, then one has to question the boundaries between procedural and declarative knowledge and the functions which transfer declarative knowledge to procedural knowledge.

When one considers the learning paradox and the need to ground symbols, Clancy’s observation of Anderson and Lebiere becomes important (Anderson and Lebiere, 2003, p595). Not only does the conceptual change hypothesis (Clancey 1999a cited in Anderson and Lebiere, 2003, p595) form a bridge between non-symbolic and symbolic knowledge, there is also a direct correspondence of second order categorizations to Piaget’s scheme model. Further, the work of Pascual–Leone (Pascual–Leone and Goodman, 1979, p311; Pascual–Leone, 1980, p280; Pascual–Leone, Goodman, Ammon and Subelman, 1978, p252) has been shown

Chapter 2. Literature Review

to explain this developmental trend. As will be seen in the design, the conceptual change hypothesis is closely related to the Piagetian scheme process (§ 2.3.6 and figure 2–1 on building symbols through development of the learning process).

2.7.4.3 Analysis of Hybrid Models

Like symbolic models, the hybrid models all fail some of the constraints defined in this study. PolyScheme (Cassimatis, Trafton, Bugajska and Schultz, 2004, p19), fails **LP1** and does not mirror observations of child development (**LP4**). CLARION (Sun, 2004, p342 cited in Georgeon, Ritter and Haynes 2009) is not constrained by Fodor’s arguments (**LP1**), but it is difficult to determine if CLARION can fully support autonomy (**LP3**) or observed to exhibit the stage like variation observed in children (**LP4**). Neuro SOAR (Neuro–SOAR (Cho, Rosenbloom and Dolan, 1991 cited in Laird, 2008, p225) is a significant improvement on other cognitive models. However, Neuro SOAR misses the mechanism of the knowing circle and the process of evolutionary trend and the developmental trends (Furth, 1969) and so fail **LP6** and **LP7**. Research on ACT–R has led to the comprehensive computational theories of a wide variety of human phenomena including brain activity (Anderson, 2007, cited in Laird, 2008, p233). However, they have ignored many of the cognitive capabilities in SOAR including episodic memory, emotion, and visual imagery. Like other hybrid models, they fail Fodor’s arguments (**LP1**) and further, are not seen to develop the stage like variation observed in developing intelligence (**LP4**). CogAff (Sloman, 2001, p42) fails because it does not specify an implementation that could resolve the learning paradox; it also fails Fodor’s arguments (**LP1**). Further, it has not been shown to develop its learning process (**LP6**). RCS (Albus, 1975; Albus, Pape, Robinson, Chiueh, McAulay, Pao and Takefuji 1992; Albus, 1991; RCS, 2009) possesses many characteristics that are necessary in any autonomous system, but lacks the capability to generalize or to change the way it learns over time. RCS includes its own abstract symbolic language and with frame based knowledge structures and inbuilt logics, fails Fodor’s arguments (**LP1**). It also has not shown the capability to develop the stage like variation observed in developing intelligence (**LP4**).

2.7.5 Conclusions

The problem with cognitive architectures and cognitive modeling is that they are not theories themselves, but merely implementations using high–level languages, thus they lack sufficient explanatory power (Tadepalli, 2003, p622). Tadepalli suggests that the models extant from these architectures can and should be evaluated empirically. The evaluation criteria used within this research (**LP1 – LP7**) was designed for this purpose with table 2-3 summarizing the results of the analysis.

Table 2–3 Evaluation of cognitive development models to support LP1 – LP7

#	Category	Model	Summary Evaluation
1	Symbolic	McCarthy	Fails LP1 , LP2 and LP5 at least
2		Theory Tethering	Fails LP1 , LP2 and LP4

#	Category	Model	Summary Evaluation
		EPIC, CIRCA, EMILE, RAA and 3T	Fails LP1 at least
3		Symbolic SOAR	Fails LP1
4	Emergent	IBCA	Fails LP2, LP4 and LP6 at least.
5		ART	Fails LP2 and LP6 at least.
6	Hybrid	PolyScheme	Fails LP1, LP2 and LP4 at least.
7		CLARION	Fails LP1, LP2 and LP3 at least.
8		SOAR-R	Fails LP6 and LP7 at least.
9		ACT-R	Fails LP1 and LP4 at least.
10		CogAff	Fails LP1 and LP6 at least.
11		RCS	Fails LP1 and LP4 at least.

From table 2-3 it is clear that no one system provides a complete solution to the evaluation criteria established to resolve, albeit partially, the learning paradox. Further, a series of principles have been identified that will be carried forward into the architecture and design namely: symbol grounding, conceptual change, dual process theories and the need for emergence.

2.8 Piagetian Implementations

This section reviews existing Piagetian implementations (Pascual-Leone, 1970; Drescher, 2002 and Chaput, 2004), and contrasts them to the reevaluated Piagetian theory (§ 2.3).

2.8.1 Pascual-Leone

Pascual-Leone developed the abstract and technical theory of constructive operators (TCO) (§ 2.1.6) as a mechanism to test a Piagetian model (Pascual-Leone and Goodman, 1979; Pascual-Leone, 1970; Pascual-Leone, 1980; Pascual-Leone, 1996; Pascual-Leone and Johnson, 1999; Pascual-Leone, Goodman, Ammon and Subelman, 1978). Though his work has significance e.g., the roles of an executive, resource constraints and the approach to learning (§ 2.1.6 on Pascual-Leone), attempts by this research to develop a workable pure cognitive model based solely from printed works on the TCO proved to be too difficult. The TCO learning mechanism could not be implemented in an artificial neural network, so Pascual-Leone’s approach had to be rejected.

2.8.2 Drescher

Drescher’s implementation of the schema mechanism starts with a set of primitive items and primitive actions (Drescher, 2002). It then explores the environment to create a set of sensory motor schemas. These schemas form the basis of new synthetic items. They are also used in the creation of goal-directed composite actions. Using these techniques, an agent can build a hierarchy of items to describe its environment and a hierarchy of sensory motor schemas can be combined into a plan to achieve some goal. Drescher created a

Chapter 2. Literature Review

micro-world of blocs and made a one-armed, one-eyed simulated agent to reconstruct the main developmental milestones described by Piaget. These included the construction of causality, the fusion of the different sensorial modalities and the construction of the notion of *permanent object invariants* (Drescher, 2002). Drescher's model of emergence of perceptual systems agrees with Bickhard's conjecture that interactive representation is the most fundamental form of representation (knowledge) since it satisfies the crucial meta-criterion of being able to tell when the system is in error (Bickhard, 1996).

Drescher presents a simulation of Piagetian sensory motor skill that invents a series of approximations to the persistent object concept and discovers correspondences among its senses. This schema mechanism is engineered to pursue two fundamental objectives: to gain knowledge by constructing or revising symbolic assertions about the world and to use those symbolic constructs to gain further knowledge (Drescher, 2002, p51). This conflicts with Piaget, who asserted that language develops from action and action is non-symbolic (Furth, 1969, p77-84).

Drescher's research replicates early Piagetian milestones in the infant's acquisition of the concept of the physical object (Drescher, 2002, p3) and to do this it must have minimal innate knowledge of its environment. Drescher uses the assimilation-accommodation aspects of Piagetian development (Drescher, 2002, p23), but assumes that Piaget did not specify explicit rules governing the activity and modification of schemes. A close reading of Furth (Furth, 1969 and Copeland, 1974) provides clear evidence of the complex interactions of assimilation-accommodation and the nature of the schema process through the development of the knowing circle (§ 2.3.5). It is also uncertain, if the overall development of the knowing circle was clear to Drescher, as he makes the assumption "The schema mechanism is engineered to pursue two fundamental, symbiotic objectives: to gain knowledge by constructing or revising symbolic assertions about the world and to use those symbolic constructs to gain further knowledge" (Drescher, 2002, p51). This perspective runs against the notion that development in sensory-motor and pre-operational areas, is non-symbolic. Furth contends that development occurs through a process of action with semiotic functions and symbolic functions emerging through the development of the learning process (Furth, 1969).

A different approach is taken in this research thesis that a neural network will develop actions that can be controlled by emergent schemes, where the schemes themselves go through a developmental process, that enable the attribution of semiotic, then symbolic functions. Essentially, emergence in terms of Crutchfield (Crutchfield, 1994a, p1) occurs through the development of a model that enables prediction of the components of the environments.

An interesting aspect of Drescher's work is "that the bootstrapping of intelligence involves assembly of concepts from special-case fragments" (Drescher, 2002, p40). This is in agreement with Bereiter's notion of bootstrapping (Bereiter, 1985, p206). The notion that there is "no apriori reason to expect a constructivist mechanism to exhibit stage like regularities at all" (Drescher, 2002, p41) can be understood to go against Pascual-Leone with the stage like variation occurring when new growth in schemes occurs. Drescher also notes that empirical learning "having structures learn from within, rather than from without - is crucial for constructivist bootstrapping," which is most clearly seen in the development of concepts (Drescher, 2002,

Chapter 2. Literature Review

p190). Overall Drescher's work can be seen as taking Piaget's work as an approximate working hypothesis, even if it is diluted by modern research. However, Drescher articulates six key problems that will need to be tackled by any research that attempts to develop general intelligence using a symbolic approach:

- 1) *symbol grounding problem* in ascribing meaning;
- 2) problem of naïve induction, the Grue/Bleen paradox;
- 3) problem of non-absurd generalizations, like cognizant failure, one assumes they must occur and are, eventually, overcome;
- 4) problem of using only projectable concepts;
- 5) problem of preferring entrenched concepts, when more than 1 generalization applies; and
- 6) resolving the problem of induction conflicts and deduction overrides and the issues with induction and counterfactuals.

For any system that uses primitives, as a symbol, a key issue is ascribing meaning (the *symbol grounding problem*). Resolving this *symbol grounding problem* can be achieved within a sensory-motor process by assigning them as perceptions and actions (Drescher, 2002, p89). Within Drescher's research, the grounded symbols are the innate schemes that are pre-defined within an artificial neural network, which allows them to be adaptive.

The Goodman Grue/Bleen Paradox of confirmation (Rescher, 2001, p 227–230) alludes to the basic problem of identifying which generalizations are reasonable (Drescher, 2002, p167). To overcome the Goodman Grue/Bleen Paradox (Rescher, 2001) it is important to realize that a deductive override mechanism must exist to escape the fallacy of naïve induction (Drescher, 2002, p173); however, to overcome homunculus, a fixed point must be established through the establishment of explicit beliefs about generalization that resolve conflicts at different levels of abstraction.

In the *problem of non-absurd generalization*, in which applying only non-absurd generalization, a system assumes that it will not generate absurd generalization by ensuring that the generalization itself is verified from experiential data (Drescher, 2002, p168). The determination of which predicates lend themselves to be more useful generalizations and which ones should be avoided (Drescher, 2002, p169).

The problem of *entrenchment* occurs when more than one generalization can apply to a particular situation (Drescher, 2002, p169); this is a determination of the relative reasonableness of the competing concepts.

Given any agent that can act, it will consistently generate a series of generalizations that will require a choice amongst the generalizations – such a measure is “implausibly restrictive – it contradicts the fact that we make generalizations about concepts that could not have been anticipated by evolution – and wholly ineffective” (Drescher, 2002, p174), since, as the Goodman Grue/Bleen Paradox shows, absurd generalizations can be recast in terms of mundane concepts.

Chapter 2. Literature Review

Drescher makes plain the notion that a schema can make *counterfactual assertions*⁷², and a synthetic item, which reifies the validity conditions of its host schema's counterfactual assertions has to be resolved, but his research shows that as at present, this problem is unresolved (Drescher, 2002, p175).

Drescher classifies schemes as symbolic representations (Drescher, 2002, p184), albeit hybrid and discloses that using single layer neural networks could not compute disjunctive normal forms (DNF) formulae. This problem could potentially have been resolved using multi-layer neural networks. Rather than “using intermediate processing units to compute conjunctions, each schema has its own extended context, in effect, its own entire connectionist network” (Drescher, 2002, p186). The advantage of this approach is simplicity to the overall design and alignment with biological reward processing on which it is based. Second, it utilizes these sensory motor objects within hierarchical networks, to facilitate structuring.

2.8.2.1 Drescher: Where is the problem?

By the utilization of a predicate-based logic, Drescher fails Fodor's arguments (LP1). It also does not allow the development of its learning process and so fails LP6.

2.8.3 CLA

Inspired by Piaget, Chaput (Chaput, 2004), proposed a new learning architecture called CLA: Constructivist Learning Architecture which itself is an extension of Drescher's schema mechanism (Drescher 1991).

In reworking Drescher's schema mechanism, Chaput implements schemas as self-organizing maps (SOM) (Kohonen, 1997) because of their neural plausibility. SOMs map the patterns of behavior, as input vectors into a feature coordinate system (feature map). After training, the SOM will organize itself as a network of nodes, where each node represents a prototype vector of the input. By utilizing multiple hierarchical layers of SOMs, with each layer taking input from the layer below, the architecture allows the system to learn increasingly abstract schemas. Chaput shows how the CLA implements the same functionalities as the schema mechanism, but without facing the scale-up limitation. Primarily a rework of Drescher's foraging goals and recovery task with reinforcement learning using delayed feedback and high values for completion of foraging tasks (Compare to § A.6.2 Actor-Critic Temporal Difference Learning as an implementation mechanism). With state value functions and policy functions spread through the prerequisite schemas enabled the robot to organize its behavior according to the assigned task.

2.8.3.1 CLA: Where is the problem?

It is evident from Chaput's research that it did not support the capability to develop its learning mechanism (LP6) nor its own symbols (LP2) nor the observation that it fully exhibits the stage like variation

⁷² A counterfactual assertion is an assertion about what would be the case if some (perhaps false) premise were true.

Chapter 2. Literature Review

observed in developing intelligence (LP4). However, like other hybrid models, CLA identifies the need for emergence.

2.8.4 Conclusions

It is evident that the existing Piagetian implementations (Pascual–Leone, 1970; Drescher, 2002 and Chaput, 2004) were attempting to solve a different problem than the learning paradox. Also, they were not attempting to produce a model that was fully biologically plausible, though they do provide some interesting insights into the development of emergent structures. Table 2–4 provides a summary analysis:

Table 2–4 Evaluation of Piagetian models to support LP1 – LP7

#	Category	Model	Summary Evaluation
1	Symbolic	Pascual–Leone	Unable to implement.
2	Emergent	Chaput	Fails LP2, LP4 and LP6 at least.
3	Hybrid	Drescher	Fails LP1 and LP6 at least.

2.9 Other Frameworks

This section reviews other frameworks of conceptual development to determine if they can resolve the learning paradox, work within the constraints defined by LP1 – LP7 and exhibit emergence as defined by Piaget. The surveyed frameworks include case based reasoning (Schank, 1982; Kolodner, 1983); explanation based learning (Sørmo, Cassens and Aamodt, 2005); unsupervised learning in visual perception (Le, Ranzato, Monga, Devin, Chen, Corrado, Dean and Ng, 2012) and emotional / intuitive models (Maturana, 1988b; Maturana and Varela, 1980; Maturana, 1988a; Jaaregui, Jauregui and Jauregui, 1995).

2.9.1 Case Based Reasoning

Case Based Reasoning (CBR) traces its roots to the work of Roger Schank on dynamic memory (Schank, 1982). Early implementations include CYRUS (Kolodner, 1983) and IPP (Lebowitz, 1983) which solved new problems based on the solutions of similar past problems. The main criticism levied against CBR is that by accepting anecdotal evidence as implicit generalizations without confirming its statistical relevance, does not guarantee that the generalization is correct. This is balanced by the observation that all inductive reasoning, where data is too scarce for statistical relevance, is inherently based on anecdotal evidence.

Explanation–based learning (EBL) as a form of machine learning exploits a very strong, or even perfect, domain theory to make generalizations or form concepts from training examples, is considered in this research to be a specialization of CBR (Sørmo, Cassens and Aamodt, 2005). By inference, it therefore suffers from the same issues.

CBR presumes that a single example can be both learnt as correct and reused “forever” and that the example can be generalized to some extent without further consideration of how/why the generalization does

Chapter 2. Literature Review

occur. This potentially suggests that a learner will only need to be shown one single, difficult example and they will need no further examples. However, instant learning is atypical of ordinary learning behavior and the need for simple examples before complex ones would seem to be a fundamental aspect of understanding why a problem is solved in some particular way. In this respect, CBR fails LP5.

2.9.2 Unsupervised Visual Learning

In machine learning, unsupervised learning refers to the problem of trying to find hidden structure in unlabeled data. Since the examples given to the learner are unlabeled, there is no error or reward signal to evaluate a potential solution. This distinguishes unsupervised learning from supervised learning and reinforcement learning (Sutton and Barto, 1998). Unsupervised learning includes topics such as self-organizing algorithms (§ 2.8.3 CLA and SOM by Chaput), conceptual clustering using discrimination nets (Fisher and Langley, 1985) and connectionism research by Kohonen, Grossberg, Rumelhart and Zipser, amongst others. Two forms of unsupervised learning are considered: Adaptive Resonance Theory and the work at Google (Le, Ranzato, Monga, Devin, Chen, Corrado, Dean and Ng, 2012).

2.9.2.1 Adaptive Resonance Theory (ART)

A fundamental problem of perception and cognition concerns the characterization of how humans discover, learn, and recognize invariant properties in the environments to which they are exposed (Carpenter and Grossberg, 1987, p54). Carpenter and Grossberg considered this stability–plasticity dilemma and devised algorithms (ART) to find patterns, categories and correlations in essentially binary data using massively parallel neural networks without a feedback signal from a teacher i.e., the data is unlabeled.

In their approach, after learning self–stabilizes with recognition codes being detected in the input patterns, the search process is automatically disengaged (Carpenter and Grossberg, 1987, p54). Thereafter input patterns directly access their recognition codes without any search. Thus, recognition time does not grow as a function of code complexity. A novel input pattern can thus directly access a category if it shares invariant properties with the set of familiar exemplars of that category (Carpenter and Grossberg, 1987, p59). The invariant properties of a learned critical feature pattern (prototype) emerge from the statistical regularity in the data (Carpenter and Grossberg, 1987, p55) which itself is centered on the appearance of novelty, using differential equations to characterize its class of network.

ART: Where is the Problem?

Several issues have been identified with the ART approach (§ 2.7.3.2), namely that its structures are determined before hand (and so fails LP3) and that it has not considered symbolic knowledge nor developed its learning process (and so fails LP6). However, one of the goals of this research is to determine if and how other systems could implement a biologically inspired Piagetian model of learning and development. Within Piaget’s model are a number of processes including assimilation, accommodation, interiorization and reflective abstraction which have particular characteristics (§ 2.3). From the review of Carpenter and Grossberg

Chapter 2. Literature Review

(Carpenter and Grossberg, 1987), it is evident that although the processes modeled reflect some aspect of cognitive development, is cannot be directly translated to the work of Piaget (§ 2.3).

2.9.2.2 Self-Taught Learning Framework

An alternate method of “unsupervised learning,” specifically supervised learning on unlabeled data, has been proposed (Le, Ranzato, Monga, Devin, Chen, Corrado, Dean and Ng, 2012). Their work is based on the theoretical existence of specific neurons in the inferior temporal cortex that respond selectively to highly specific complex objects⁷³ (Desimone, Albright, Gross and Bruce, 1984, p2051) and is an example of the self-taught machine learning framework approach (Raina, Battle, Lee, Packer and Ng, 2007, p759).

In the Google approach, a neural network was presented with 10 million digital images each with 200x200 pixels, from randomly selected YouTube videos. Using model parallelism and running on a cluster with 1,000 machines (16,000 cores) for three days, the network learned invariances in the training data (Le, Ranzato, Monga, Devin, Chen, Corrado, Dean and Ng, 2012, p7).

Once trained, it performed far better than any previous efforts and obtained an accuracy of 15.8% accuracy in recognizing faces from 20,000 object categories from ImageNet, a relative improvement of 70% on earlier efforts (Le, Ranzato, Monga, Devin, Chen, Corrado, Dean and Ng, 2012, p2).

In the self-taught machine learning framework approach (Raina, Battle, Lee, Packer and Ng, 2007, p761) the network initially learns basic elements e.g., rows of pixels which are then developed into more abstract representations such as edges using a modified sparse coding algorithm. The self-taught machine learning framework compares favorably with other methods such a principal component analysis (PCA) and other techniques such as Gaussian discriminant analysis (GDA) and support vector machines (SVM) (Raina, Battle, Lee, Packer and Ng, 2007, p763), as is evidenced by recent results (Le, Ranzato, Monga, Devin, Chen, Corrado, Dean and Ng, 2012).

2.9.2.3 Unsupervised Learning – Where is the Problem?

Though sparse coding algorithms have been shown to have a biological parallel in the visual cortex (Lee, Battle, Raina and Ng, 2007), being used to identify regularities in the environment, it is not a Piagetian solution to concept formation. Within Piaget’s model are a number of interacting processes including assimilation, accommodation equilibration and disequilibration which have particular characteristics (§ 2.3.2 and 2.3.11). From the review of unsupervised visual learning (Lee, Battle, Raina and Ng, 2007), it is evident that although the processes reflect aspects of cognition, these cannot be directly translated to the work of Piaget (§ 2.3). Further, it has been shown in the analysis of the Piaget’s theory that there is a coordination of the activities of interiorization and reflective abstraction (§ 2.3.9 and 2.3.10) which reuses the assimilated structures. This reuse is not encompassed within the solution provided by Lee, Battle, Raina and Ng. Thus Google was not applying

⁷³ Neuroscientists have discussed the possibility of what they call the “grandmother neuron,” specialized cells that fire when they are exposed repeatedly or “trained” to recognize a particular face of an individual.

Chapter 2. Literature Review

a Piagetian concept mechanism (§ 2.3), specifically one that changes through maturation and is based on assimilation and accommodation. By this definition unsupervised learning fails **LP5**. Further, though the capability of recognizing a face is seen as an important step in early childhood development (Field, Woodson, Greenberg and Cohen, 1989, p179; Piaget, 1962b, 1952) – making use as it does of the fusiform face area in the brain – it is widely recognized that some 2–2.5% of a people in Germany may suffer from prosopagnosia (Blank, Anwender and von Kriegstein, 2012, p605), the same may be true for other countries. With this in mind, facial recognition is not seen as critical to the development of concepts. What can be taken from the Google approach (Le, Ranzato, Monga, Devin, Chen, Corrado, Dean and Ng, 2012), is the scale of their work. In terms of Piaget and the learning paradox, they are developing the sensory motor schemes (§ 2.1.5) upon which the more complex schemes are laid down. In this respect, they are at too low a level for this research.

2.9.3 Emotional / Intuitive Models

Other scientific models exist to describe human reasoning and in doing so resolve the learning paradox. These models propose much more complex epistemologies requiring the use of terms such as emotion and intuition (Maturana, 1988b; Maturana and Varela, 1980; Maturana, 1988a; Jauregui, Jauregui and Jauregui, 1995). These approaches are not addressed in this research, since they have no formal implementation methodology.

2.9.4 Conclusions

This section reviewed unsupervised learning in visual perception (Le, Ranzato, Monga, Devin, Chen, Corrado, Dean and Ng, 2012), case based reasoning (Schank, 1982; Kolodner, 1983); explanation based learning (Sørmo, Cassens and Aamodt, 2005), emotion and intuition (Maturana, 1988b; Maturana and Varela, 1980; Maturana, 1988a; Jauregui, Jauregui and Jauregui, 1995) frameworks to determine if they could resolve the *worked example* and support a Piagetian model of development. It concluded that these frameworks bypass the need for learning and development, but have valuable features, which could be incorporated into future research to improve on a resolution to the learning paradox.

Specifically, Piaget considered that identifying invariances in the environment (through visible perception for example) occurred through the normal development of the child through the creation and modification of figurative and operative schemes which identify those regularities (§ 2.3.3). The position taken in this research is that the *dialectic system* assumes that such regularities in the environment will have already been generated by the *artificial neural network implementation* (§ 4.1 and 5.1.).

Though it may be useful to add visual processing to this research, it should be considered for future research. For example, the eigen-background approach (Rymel Renno, Greenhill, Orwell and Jones, 2004) could be used to detect moving objects on a number line. Similarly, line trajectories (Ren, Orwell, Jones and Xu, 2004) could be used to predict ending movements on a number line. This research could use these approaches to “see” statistical regularity and “geometry,” which would extend processing into three dimensions.

Chapter 2. Literature Review

Table 2–5 provides a summary analysis.

Table 2–5 Evaluation of Other Piagetian Frameworks to support LP1 – LP7

#	Category	Model	Summary Evaluation
1	Symbolic	CBR	Fails LP5 at least.
2	Emergent	ART	Fails LP2 and LP6 at least.
3		Google	Fails LP5 at least.
4	Unable to determine	Emotional – Intuitive	Unable to determine.

2.10 Summary – The Two–Fold Need

In § 2.1, is described a Piagetian epistemological solution to the learning paradox (Furth, 1969; Copeland, 1974). In § 2.2, is described the biological basis of a solution to the learning paradox and presented a model of *Drosophila* (Miesenböck, 2008, p52; Shang, Claridge–Chang, Sjulson, Pypaert and Miesenböck, 2007, p601), that it is believed, exhibits emergence. It is assumed that any full solution to the learning paradox would need to be fully realized in an *artificial neural network implementation*, one that mirrors the learning and development of children. Since one could not find a hierarchical neural architecture that could be suitably modified, and time was not available to develop such a solution, another approach was used. By taking Minsky’s position, that a neuron can be implemented using FSA (Minsky, 1967; McCulloch and Pitts, 1943), it was conjectured that a neural Piagetian model could be fully implemented using HFSA. Thus, a two–fold solution as an *artificial neural network implementation* and a *dialectic system* was proposed. In § 2.3, a reevaluated model of Piaget was presented. It was asserted that the Piagetian model mirrors *Drosophila*. A worked example on the development of number–sense (WE1 – WE5) was presented and related to the needs of the learning paradox (LP1 – LP7). In § 2.4, the *worked example* was compared to research on the MNL and the cultural effects of its use. It was concluded that the worked example is sufficient, real–world example. In § 2.5, the constraints of the learning paradox were compared to ATF and ATP systems, and it was concluded that they were solving a different problem than envisioned in this research. In § 2.6, a review of the underlying basis of evolutionary emergence in conceptual blending theory was described, and it was posited that this provided an effective mechanism of concept learning in this research. In § 2.7, is reviewed the literature on cognitive development models and concluded that they were not attempting to resolve the learning paradox, nor were they using a Piagetian model. It is anticipated that this research will further these discussions. A review of existing Piagetian implementations was provided in § 2.8. It was concluded that these implementations were attempting to solve different problems and were unaware of the biological basis of Piaget’s model. In § 2.9, other frameworks were reviewed and their potential future impacts on the research was discussed. The research problem and scope can thus be defined.

Chapter 3

3. Research Methods

“The first rule of discovery is to have brains and good luck. The second rule of discovery is to sit tight and wait till you get a bright idea” (Polya, 1988, p172).

This chapter identifies the research question, purpose, defines an effective problem statement, and methods to use in this research thesis. To achieve this, a series of research goals along with a research framework of observations, hypotheses, tenets and constraints are identified. A research design is proposed along with a set of experiments designed to determine the conditions under which structure will emerge in both the *artificial neural network implementation* and the *dialectic system*.

3.1 Research Area

3.1.1 Research Question

A review of the research literature distinguished Pascual-Leone’s definition of Fodor’s problem as the learning paradox (Pascual-Leone, 1976, 1980 cited by Bereiter, 1985, p202), which is a meta-theoretical problem. The problem is defined by a meta-theoretical question: “How can a structure generate another structure more complex than itself?”

3.1.2 Research Purpose

The purpose of this research is to review the research literature on the implications of the learning paradox (Bereiter, 1985, p202; Fodor, 1980) as it applies to concept formation in early childhood mathematics and to discern an implementation of a cognitive modeling framework as defined by Piaget (Furth, 1969; Copeland, 1974 and Pascual-Leone, 1980) that attempts to exhibit emergence as a natural process of maturation in the simulation. The epistemological issues as identified in the debate between Chomsky, Jean Piaget and Jerry Fodor (Piattelli-Palmarini, 1980), will be reviewed (§ 2.1). As part of this review a set of constraints, that any solution which attempts to resolve the learning paradox, will be identified (§ 2.1.7). Since a biologically plausible mechanism of emergence needs to be built, a review of emergence in evolution and biology will be conducted (§ 2.2.). A literature review will be conducted to confirm biologically plausible mechanisms in Piaget’s model, as well as identify experiments (*worked example*) that can be conducted to test for emergence (§ 2.3). An analysis of number-sense will be undertaken to confirm the development of early

Chapter 3. Research Methods

childhood mathematics and define the educational setting (§ 2.4). A literature review of automated theory formation and theorem proving will be conducted to determine if mechanisms used in these approaches resolve the learning paradox (§ 2.5.). The role of metaphor and conceptual blending will be analyzed to determine if there are biologically plausible mechanisms that can be used to implement a Piagetian model (§ 2.5). A review of cognitive development models will be conducted to determine if they exhibit emergence as defined by Piaget (§ 2.7). Similarly, a review of existing Piagetian models will be conducted to determine if they describe a biologically plausible model (§ 2.8). Finally, a review of other machine learning frameworks will be conducted to determine if the constraints imposed by the learning paradox are sidestepped, or if they could implement a Piagetian model of emergence (§ 2.9.).

3.1.3 Research Problem

The research problem is to determine the conditions under which it is possible to exhibit emergence in a biologically plausible simulation and so overcome the learning paradox (albeit partially) within a constrained number line world.

3.2 Research Method

Typical for a research area that covers multiple disciplines and specifically to tackle the needs of the research problem, a combination of approaches is required since each of the scientific, mathematical and engineering methods has a number of limitations. These limitations are described in table 3–1.

Table 3–1 Research Methods

Step	Mathematical Method	Scientific Method	Engineering Method
1	Understanding	Observations: from experience (characterization, observations, definitions and measurements of the subject (phenomena of inquiry))	Define a need and do background research,
2	Analysis	Hypothesis: a proposed explanation, hypotheses (theoretical, hypothetical explanations of observations and measurements of the subject)	Establish design criteria e.g., use a requirements engineering process such as Rational Unified Process (RUP)
3	Synthesis	Deduction (predictions) (reasoning including logical deduction from the hypothesis or theory)	Prepare preliminary designs and build prototype
4	Review–Extend	Experiments to test characterizations, hypotheses and predictions	Test and redesign as necessary
5	Conclusion	Conclusions analyze and present results)	Present results

Chapter 3. Research Methods

To be usable, the research method must address the following needs:

1. **Need-1:** It must enable the research to be described in a scientific way so that a hypothesis is formed without bias, predictions can be made. The method must also define the experiments that will be undertaken as well as support the recording and running of experiments. The method must also remove the bias from the analysis of the results and support publication of the results in a clear and consistent format that shows the relationships of cause and effect.
2. **Need-2:** It must support the analysis of the problem and its solution in a simple, effective and standard way that allows for induction, deduction and proof of the mathematical elements of the experiments.
3. **Need-3:** It must support the engineering of the simulation so the results can be shown to be consistent with other similar projects.

To tackle the needs of the research problem a combination of approaches is required since each of the scientific, mathematical and engineering methods has a number of limitations:

- The scientific method provides a solution to meet **Need-1**, but not **Need-3** and only some of **Need-2**.
- The mathematical method provides a solution to meet **Need-2**, but not **Need-3** and only some of **Need-1**.
- The engineering method (Rational Unified Process) provides a solution to meet **Need-3**, but not necessarily **Need-2** or **Need-1**. The engineering method of the Rational Unified Process (Jacobson, Booch and Rumbaugh, 1999) will be used.

Since one cannot “prove” a hypothesis statistically, many researchers use a null hypothesis. The logic of null hypothesis testing is analogous to proof by contradiction, and since the null hypothesis, as an identity, can and should only be rejected, statistical hypothesis testing is not a relevant approach to use for this research. Thus, the mathematical method and engineering method must be combined and used within a framework of the scientific method so that the problem can be defined, a simulation built and the experiments identified, executed and results evaluated.

3.2.1 Research Goals

The primary research goal is to determine if it is possible to build a *dialectic system* that implements a biologically plausible Piagetian model using an evolutionary computing framework (Jacob, 2001; Levy, 2002) and a reactive systems framework (Maraninchi and Remond, 2001) within a simulated number line world that exhibits emergence of number-sense.

The primary goal is predicated on confirming that an *artificial neural network implementation* using the Verve toolkit (Streeter, Oliver and Sannier, 2006) (i) can identify the *permanent object invariants* as defined by Piaget (§ 2.3.5); (ii) a mechanism can be defined to allow these objects to be transferred to the *dialectic system*; (iii) the solution must adhere to the identified constraints of the learning paradox (**LP1 – LP7**), and (iv)

implement a workable solution to the *worked example* (WE1 – WE5). The intrinsic property of this architecture is the strict adherence to a Piagetian model that is biologically plausible.

3.3 Research Framework

3.3.1 Observations

The literature review defined a series of observations that lead us to the belief that the correct problem is being solved. First, Piaget's observations of the stage like development of number–sense including the bead problem (§ figure 1–1) and the *permanent object invariant* (§ 2.3.5). Second, a set of constraints were defined that must be met by any solution that can overcome the learning paradox (LP1 – LP7) (§ 2.1.7). These constraints have been evaluated against solutions in the research area, and there is consistency in the results of this analysis (§ 2.4 – 2.9.).

3.3.2 Research Hypotheses

The current structure of the research is based on a number of accepted principles and a set of developed tenets. By classifying the research into two dimensions: cognitive modeling and computing / information systems, an appropriate research strategy unfolds through the definition of three major hypotheses, which are described in figure 3–1:

- 1) **Hypothesis 1:** A Piagetian model of human cognition exists that explains emergence and therefore resolves the learning paradox.
- 2) **Hypothesis 2.** A biologically plausible model of mathematical cognition exists.
- 3) **Hypothesis 3.** Hypothesis 1 and 2 can be modeled and implemented in a number line simulation with number–sense as an emergent property of the executing system.

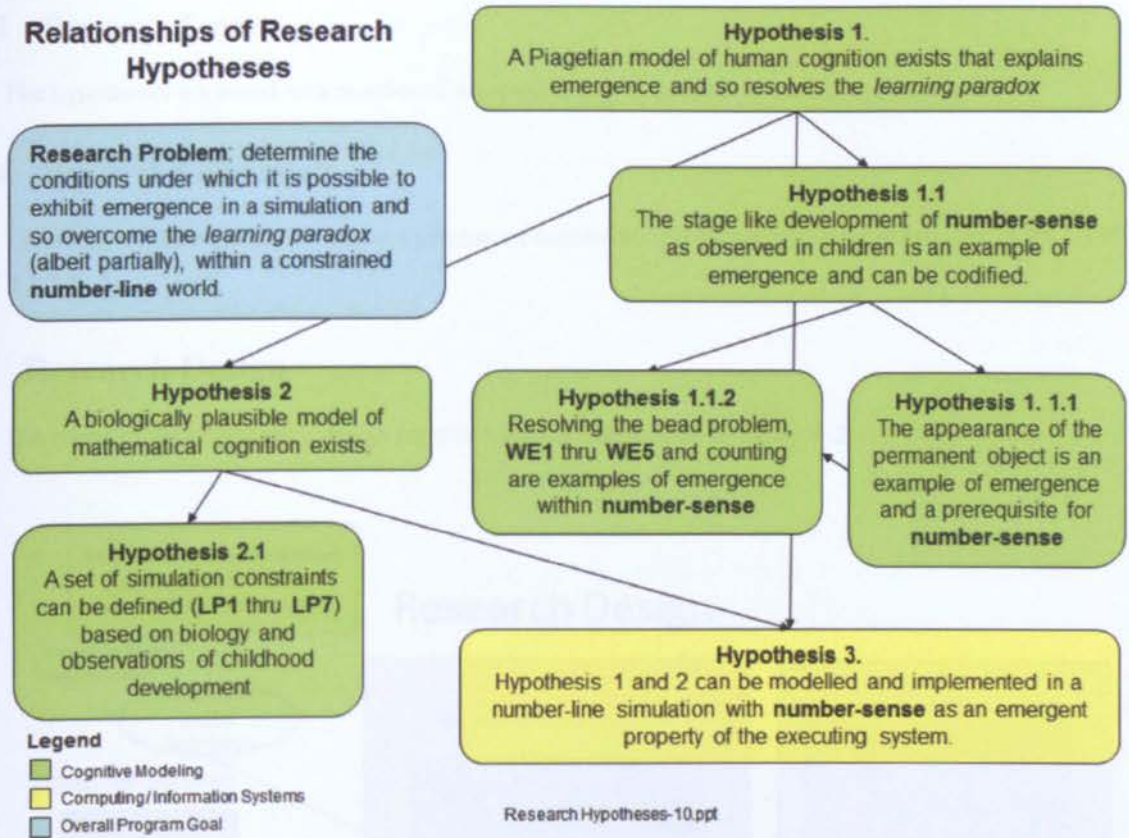


Figure 3–1 The relationship of the research hypotheses.

Each hypothesis is explained in detail below:

Hypothesis 1 can be classified further. A Piagetian model of human cognition exists that explains emergence and resolves the learning paradox (Furth, 1969; Copeland, 1974). Within **Hypothesis 1**, there are a number of other hypotheses:

1. **Hypothesis 1.1:** The stage like development of number–sense as observed in children is an example of emergence and can be codified (§ 2.1, 2.3, 4, and 5.)
2. **Hypothesis 1.1.1:** The appearance of the permanent object is an example of emergence and a prerequisite for number–sense (§ 2.3.6).
3. **Hypothesis 1.1.2:** Resolving the bead problem (§ figure 1–1), **WE1 – WE5** (§ 2.3.14) and counting are examples of emergence within number–sense.

Hypothesis 2 can be further classified. A biologically plausible model of mathematical cognition exists. Within **Hypothesis 2**, another hypothesis is defined:

1. **Hypothesis 2.1:** A set of simulation constraints can be defined (**LP1 – LP7**) based on biology (§ 2.1.7 and 2.2) and observations of childhood development (§ 2.3).

Hypothesis 3 can be classified further. **Hypothesis 1** and **Hypothesis 2** can be modeled and implemented in a number line simulation with number–sense as an emergent property of the executing system (§ 4.)

3.3.3 Research Tenets

The hypotheses are based on a number of accepted tenets as defined in the literature review.

3.3.4 Research Constraints

The scope of the work is limited by a number of constraints defined in the literature review (LP1 – LP7) see § 2.1.7.

3.4 Research Design

The scope of the engineering design for this research is described in figure 3–2.

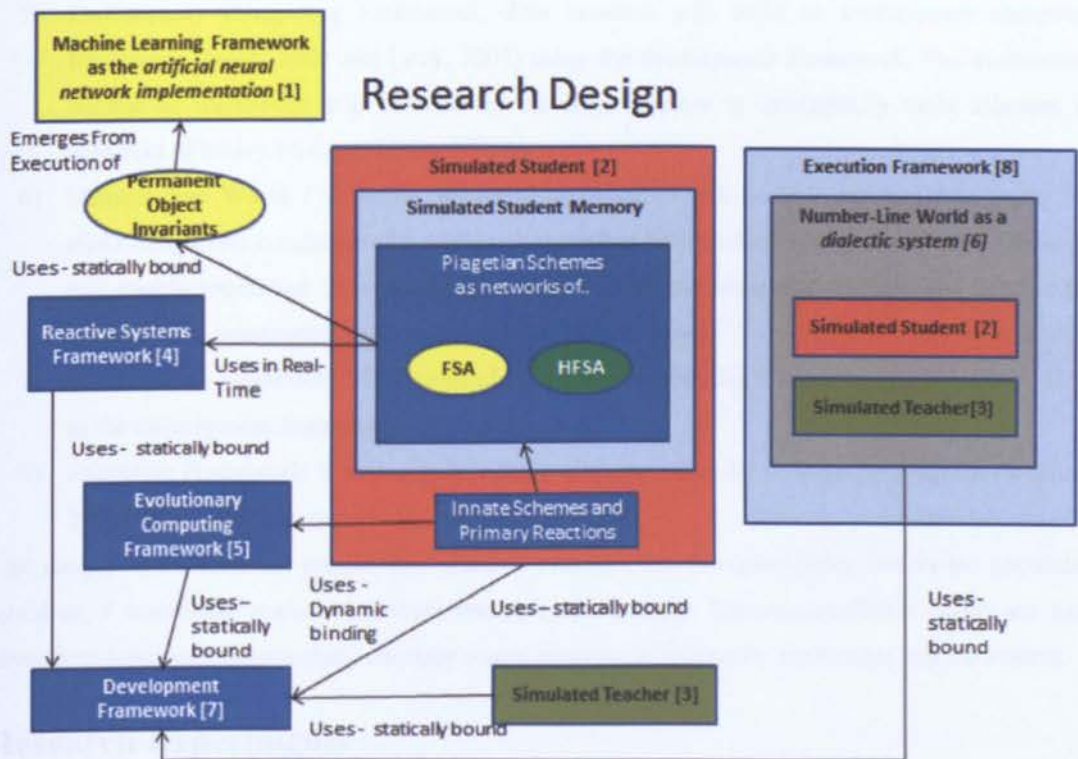


Figure 3–2 An outline of the design and runtime components of the research.

- 1) Machine Learning Framework: This research will reconstruct the Verve toolkit (Streeter, Oliver and Sannier, 2006) to test for the emergence of *permanent object invariant*'s in an *artificial neural network implementation* within a number line world.
- 2) Simulated Student: This research will build a Piagetian / Drosophila model (Furth, 1969; Copeland, 1974; Miesenböck, 2008, p52; Shang, Claridge-Chang, Sjulson, Pypaert and Miesenböck, 2007, p601) as a set of innate schemes and primary reactions using the development framework and instantiate it in a simulated student.

Chapter 3. Research Methods

- a. The innate schemes and primary reactions enable the student to act/sense, learn and plan. The act and sense process uses schemes that have been retrieved from memory, to act and sense in the number line world. The learn process builds new schemes. The plan process reuses and reorganizes schemes to internally model the external environment.
 - b. The simulated student will be composed of networks of schemes as networks binary FSA and HFSA.
- 3) Simulated Teacher: This research will build a simulated teacher using the development framework. The teacher contains various automated testing scripts using a worksheet metaphor.
 - 4) Reactive Framework: This research will build a run-time Argos reactive framework (Maraninchi and Remond, 2001) using the development framework. This will be used to enable run-time connection between the binary FSA and the binary HFSA.
 - 5) Evolutionary Computing Framework: This research will build an evolutionary computing framework (Jacob, 2001 and Levy, 2002) using the development framework. The evolutionary computing framework will be used by the learn process to dynamically build schemes (as networks of binary FSA and binary HFSA).
 - 6) Number line World / *Dialectic system*: This research will build a number line world (the *dialectic system*) consisting of a workbook metaphor with student and teacher number lines that will enable interaction between the simulated student and simulated teacher. The number line world will be constructed using the development framework.
 - 7) Development Framework: This research will use Mathematica 8 environment (Wolfram, 2003) as the development framework.
 - 8) Execution Framework: This research will use a Mathematica 8 run-time environment (Wolfram, 2003) within which to execute the number line world.

In the design of the *dialectic system*, this research assumes that the controllable factors are population size, # children, # mutations, evaluation criteria and selection criteria. The uncontrollable factors are time, randomness, population of chromosomes, resource usage, ordering of actions by the teacher and the student.

3.5 Research Experiments

A series of experiments will be performed against the *artificial neural network implementation* and the *dialectic system* to determine (i) if results show that they meet the predictions made from Piaget's model, (ii) how closely they handled the constraints imposed (LP1 – LP7), (iii) if they solved the worked example (WE1 – WE5), (iv) if they solved the bead problem and (v) exhibit emergence of number-sense.

Figure 3–3 classifies the six levels of experiments that will be conducted across the constructed systems.

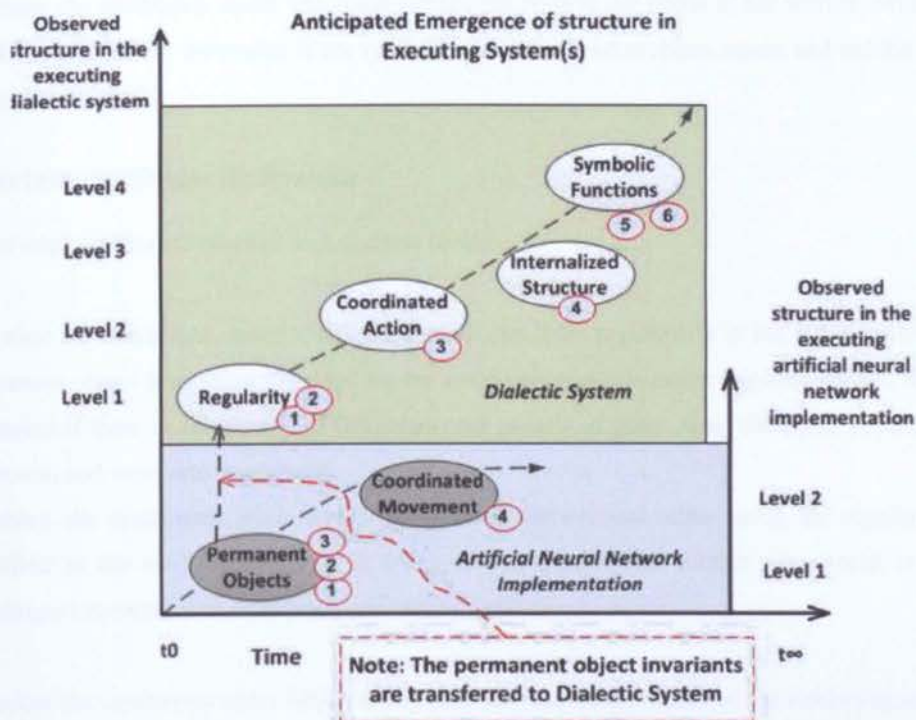


Figure 3-3 This diagram summarizes the anticipated emergent structure that will be observed in the executing system. The artificial neural network implementation exhibits regularities in the environment (the permanent object invariant) and coordinated movement. the dialectic system identifies regularizes using the permanent object invariants identified by the artificial neural network implementation and exhibits other, more complex structures such as coordinated movement, internalized FSA structures and interactions as HFSA. Each of the structures in the *dialectic system* is imagined to occur in levels (stages of development) as the system matures.

3.5.1 Experiments: Artificial Neural Network Implementation

A series of experiments are planned on two levels:

Level 1

- 1) Determine the conditions under which the system can identify regularities in the environment using different configuration values for discrete or continuous sensors and various reinforcement learning rates. Specifically, determine if there is emergence of the *permanent object invariants* of point, line, direction, penState (penUp, penDown, Stop) and movement.
- 2) Determine the conditions under which the system can act and sense using the regularities it has identified in the environment, and in doing so navigate in the number line world as a form of coordinated movement from relative and static positions.
- 3) Determine the conditions under which the system can plan a set of actions rather than relying on merely acting/sensing the environment, and so improve its performance.

Level 2

Chapter 3. Research Methods

- 4) Determine the conditions under which the system can resolve the needs of the worked example (WE1 – WE5). Specifically determine if the system resolve the bead problem, count and exhibit emergence of number–sense.

3.5.2 Experiments: Dialectic System

A series of experiments are planned in 4 discrete levels:

Level 1

- 1) Determine the conditions under which the system can learn regularities in the environment using the *permanent object invariants* provided by the *artificial neural network implementation*. Specifically, determine if there is emergence of the permanent objects of point, line, direction, penState (penUp, penDown, and stop) and movement.
- 2) Determine the conditions under which the system can act and sense using the regularities it has identified in the environment, and in doing so navigate in the number line world in a form of coordinated movement from relative and static positions.

Level 2

- 3) Determine the conditions under which the system can resolve the needs of the worked example (WE1 – WE5). Specifically determine if the system can resolve the bead problem, count and exhibit emergence of number–sense.

Level 3

- 4) Determine the conditions under which the system can learn to build internal structures that represent propositional logic components (AND, OR, Buffer, Not etc.) and dialectic actions (success, failure, student actions, teacher requests etc.). The emergence of internalized structure is a prerequisite for the Piagetian model, since it enables the simulated student to act on its internal structures and so develop number–sense.

Level 4

- 5) Determine the conditions under which the system could construct Piagetian schemes as networks of binary HFSA and binary FSA that mirror predictions from Piaget’s theory. Specifically determine the conditions under which the following could occur:
 - a. Equivalence schemes.
 - b. Less than schemes.
 - c. Processes of Equilibration and Disequilibrium.
 - d. Encoder Schemes.
 - e. Decoder Schemes.
 - f. Counting schemes which can resolve the *worked example* (WE1 – WE5).
- 6) Determine how closely the structure of the solution meets the needs of the Piagetian model.

3.6 Summary

A research design has been proposed for both the *artificial neural network implementation* and the *dialectic system*. A series of experiments have also been defined to determine the conditions under which emergence of structure arises in the systems, and if they can resolve the worked example, bead problem or exhibit emergence of number–sense as defined by Piaget.

Chapter 4

4. Architecture and Design

This chapter describes the architecture and the design of the *artificial neural network implementation* and the *dialectic system* solution to resolve (partially) the learning paradox, with each system being described in turn.

4.1 Artificial Neural Network Implementation

4.1.1 Architecture

The architecture for the Verve toolkit has already been defined (Streeter, Oliver and Sannier, 2006). It uses a model of reinforcement learning using an actor-critic approach (Sutton and Barto, 1998, p151) that mirrors the reward processing and prediction response in animals (Suri and Schultz, 1998, p350, see elsewhere Schultz, 2000; Schultz and Dickinson, 2000, p474; Fiorillo, Tobler and Schultz, 2003, p1898). It uses radial basis function and linear neural networks to implement a simulation that acts/senses, learns and plans its movements in a grid world. The original grid world has been modified to support penState (up, down) directions (left, stop, right) and movements with integer, real, discrete and continuous values from its sensors.

4.1.2 Design

Verve has been constructed with two distinct components that work together to enable the simulation to adapt to its environment by switching between acting/sensing, learning and planning (Streeter, Oliver and Sannier, 2006).

The *reinforcement learning component* works as follows. Based on the given observation, the *policy* chooses an action, and the *state value function* estimates the value of the current state. A prediction error is computed and used to train the *state value function* and the *policy*. Further, the *reinforcement learning component* processes incoming observations into an internal state representation (using radial basis function neural networks) that provides more informative features. Each radial basis function in the state representation is a separate neuron with its activation proportional to its Euclidean distance from the input data point. The system supports two types of sensor states, continuous and discrete. Discrete sensors have an index value. Continuous sensors have a value between -1 and 1 and represent, for example, a distance value returned by measurement. These continuous sensors have a “resolution sensor” to determine their acuity. Agents use a

Chapter 4. Architecture and Design

dynamically growing radial basis function as state representations. This combines sensory inputs into higher-level features and allows generalizations to unseen inputs. Agents are updated in real-time with each time step using a delta learning rule (which is a supervised learning rule, because the actual environment provides real examples that can act as training examples. Thus, it must function differently than the gradient descent rule used for TD Learning) that specifies how much time has elapsed since the previous update. The usual way of setting a neural network learning rate parameter is by using a time constant value that affects how far each weight is adjusted per update. However, this is translated to seconds when displayed. For example, a learning rate time constant of 0.1s causes the system to attempt to reduce errors to 37% of their initial value after 0.1s regardless of the size of the Agent update time delta. The system uses linear neural networks to represent the *state value function* and the *policy*. These neural networks are trained using a TD error signal. In this *reinforcement learning component*, learning can be enabled or disabled based on internal or external decisions. Agents have a policy, which learns to select from a finite number of actions, and are selected using a roulette selection scheme. The agents use a softmax action selection scheme, which maintains separate selection probabilities for each action.

The *predictive model component* includes observations, observation selectors, reward selectors and an uncertainty selector function. The *predictive model* predicts the next observation and reward based on the current observation and action. The *predictive model component* trains itself by computing prediction errors between the actual and predicted values. It also maintains the uncertainty estimate for its own predictions. The *predictive model component* can train its *reinforcement learning component* using its *predictive models* predictions. This allows it to iteratively step through long planning trajectories in the agents “imagined” environment. These planning sequences end when the prediction uncertainty is too high.

The *predictive model component* can also be put into curious planning mode in which case, a small curiosity reward is provided for encountering novel situations (as determined by prediction uncertainty). This drives the system to explore unfamiliar states, and improve its adaptability to the environment (Singh, Barto and Chentanez, 2004). In this respect, curiosity uses the same uncertainty estimations as planning. The system is provided with curiosity rewards at each planning step proportional to the estimated uncertainty for the most recent prediction.

4.1.2.1 The Learning Task

The Verve agent lives in constrained number line world, a square 2D 10*10 grid of number lines. The goal of the agent is to move from a starting location to a goal location within the problem space, and to appropriately make use of its penState (penUp, penDown) to draw lines along its path. The resolution of set problems is not part of the sensory motor evaluation, refer to the subsequent §4.2, for this.

The agent has two sensors that detect the agents’ x and y position. It has five actions move left, right, penUp, penDown, or do nothing. The agent must learn to move to the goal as quickly as possible. As soon as the agent reaches its goal, the trial is ended. In attempting to reach the goal, the agent can run with different learning architecture modes and various configuration options. These include:

Chapter 4. Architecture and Design

- 1) Learning Architecture
 - a. Reinforcement Learning (RL),
 - b. Planning i.e., Model Learning (Model) or Planning with Curiosity (Curiosity)
 - i. Model Learning Rate = 0.1
- 2) Temporal Difference Learning Rate = 0.1 (Low policy learning Rate)
- 3) Policy learning multiplier = 10
- 4) Number of runs averaged = 5
- 5) Number of trials per run = 81
- 6) Maximum number of steps per trial = 1000
- 7) Use Discrete Sensors
 - a. Position input discretization = 20
- 8) Use Continuous Sensors
 - a. Use Continuous Rewards = True/False
 - b. Continuous sensor resolution = 15
 - c. Radial Basis Function resolution = 70% of grid world resolution

Learning performance is measured in a number of ways including the number of steps taken before reaching the goal, the cumulative reward, the mean squared error (MSE) for observations and reward predictions using the predictive model using planning or curiosity and average reward per time step. The actions of the agent, along a simulated number line, are analyzed along with the development of the agent's sensors as radial basis functions and its state value functions.

4.1.2.2 Sensing the Environment

Each agent uses a dynamically growing radial basis function as a state representation for its two types of sensors either *discrete inputs* or *continuous inputs* pertinent to the type of sensor created. This combines sensory inputs into higher-level features and allows generalizations to unseen inputs.

Discrete sensors take an index representing one of several distinct values. When an agent is created, for each discrete sensor state, the number of discrete values that the sensor can represent must be specified.

When an Agent is created, for each continuous sensor used there is a global "resolution" that determines how many radial basis functions span each continuous input dimension. These radial basis functions have no center position, and they are maximally active i.e., activation = 1, when they match the discrete input data, otherwise activation = 0. This value also determines the width (distance of one standard deviation from the center) of each radial basis function, which is set to $2/\text{resolution}$. Where 2 represents the range of -1 to 1. Each continuous sensor has an additional Boolean circular parameter that specifies whether the sensor is detecting a circular input range that can jump instantly from -1 to 1 e.g., the end of a line.

A continuous sensor is converted into a linear value to reduce the complexity. A continuous sensor has a value between -1 and 1 and represents for example, a distance value returned by measurement. A continuous sensor has a "resolution sensor" to determine their acuity and use radial basis function, as recommended by

Chapter 4. Architecture and Design

Sutton and Barto (Sutton & Barto, 1998, p85; p193; p202). The continuous sensors' acuity is $2/Grayscale-1$ therefore with a 10/10 grid, the continuous sensor changes at a rate of 0.222, which is approximately $2/9^{th}$.

4.1.2.3 Example Execution of ANN Implementation

In a typical reinforcement learning setup, the environment provides observations and rewards and the agent (the simulation system) responds with actions (Sutton and Barto, 1998, p52). The system is driven based on the reward provided by the environment. In this research sequence, a grid-world is laid out as a number line world and the simulation attempts to identify invariants in the world such as point, lines and penStates. It receives rewards based on how quickly and efficiently it achieves its desired objectives which may include drawing a line from the origin, drawing a line with a penState of penUp – to represent movement, drawing lines with the penState of penDown to represent a numeric quantity on the number line.

4.2 Dialectic System

4.2.1 Architecture

The architectural strategy is to use the knowledge gleaned from literature review to create a design that meets the constraints (LP1 – LP7) and the research goal (§ 3.2.1).

The research hypothesis 1 (§ 3.3.2) requires that certain specific processing in the Piagetian model must be accounted for. The use of a biologically plausible model of mathematical cognition (research hypothesis 2 § 3.3.2) requires that key features of Drosophila (§ 3.1 and A) and reinforcement learning (§ 2.2) be implemented in the design. To support this need, the research follows the approach of evolvable hardware (Greenwood and Tyrrell, 2006, p12) and builds reusable networks as digital circuits that can count, amongst other things (Wirth, 1995). The simulation evolves networks of binary FSA (as propositional logic components, and schemes that can detect regularities in the external and internal environment) to process information. It evolves networks of hierarchal FSA to marshal information across discrete networks of FSA, with each integrated hierarchy being referred to as a scheme. These networks are stored and retrieved from memory. Various classes of FSA are constructed with each having a discrete externally provided evaluation function (§ figure 5.2.2.1 for example). HFSA are constructed in a similar manner. The simulation uses a control mechanism to switch between acting/sensing, learning and planning. The simulation uses a scheme selection mechanism to determine which scheme to apply in a given situation, and an exception process which identifies when it cannot process information (§ 4.2.2.5 for the process of adaption, as it is exposed through learning as evolution). When an exception occurs, the system switches between ignoring the input or changing its structure to handle the input (which is equated to equilibration). Within each Piagetian scheme there is a mandated switch between the processes of action (which is equated to accommodation § 2.3.2) and structuring (which is equated to assimilation § 2.3.2). The simulation, as a simulated student adapts to its number line world using sensory input and sensory output based on an EBNF grammar (EBNF Interaction Grammar § 4.2.1.1). Rewards are provided by the environment in a similar fashion to (reinforcement learning § 2.2). The

Chapter 4. Architecture and Design

execution and evolution of the simulation is externally observed for its behavior, and the internal structure of the simulation is observed (to determine the types of schemes that have evolved). When schemes are learned, copies of those schemes can be reused in the simulation, and become part of its larger organization:

- 1) When the simulation evolves (learns) a new sensor e.g., to change the penState to penDown to draw a point on a number line, then that scheme can be reused.
- 2) When the simulation evolves (learns) a new sensor e.g., to reuse the penDown sensor and evolve a move, it can draw a line on a number line, then that scheme can be reused (This effectively resolves WE1 § 2.3.14).
- 3) When the simulation evolves (learns) a new sensor e.g., to reuse the penDown sensor and evolve a move, and reuse PenUp it can draw a line on a number line, then that scheme can be reused (This effectively resolves WE2 § 2.3.14).
- 4) When the simulation evolves (learns) to differentiate objects on a number line, by reusing existing schemes (This effectively resolves WE3 § 2.3.14), and is an example of conservation of length).

The literature review identified several architectural issues that need to be resolved. Each issue is treated in turn.

For the symbol grounding problem (§ 2.7.2.1), this research takes a genetic epistemological view, asserting that for a machine to approach resolution of the learning paradox, the simulation must form its own hierarchical concepts. The basis of this structuring comes from using sub-symbolic sensory data (Mayo, 2003, p57), blended into task based experience by the simulation. The sub-symbolic sensory data is provided to the *dialectic system* by the *artificial neural network implementation* as the *permanent object invariants* embedded within the EBNF grammar (§ 4.2.1.1). This approach adheres to the Piagetian notion of substitution (Copeland, 1974, p252). The appearance of the *permanent object invariants* is identified by Piaget (Furth, 1969, p98) as a key aspect in the development of cognition It is also identified as WE2 in the worked example (§ 2.3.14).

To sidestep naïve induction (§ 2.8.2) and the Goodman Grue/Bleen Paradox (Rescher, 2001) – where deductive override mechanisms exist to escape the fallacy of absurd generalizations (Drescher, 2002, p173) – this research builds evolvable hardware circuits. To overcome the associated homunculus, a fixed point is established as the *permanent object invariants*, which resolves the conflicts at different levels of abstraction. This research reuses schemes by using a reflective abstraction mechanism § 2.3.9). The issue of non-absurd generalizations (§ 2.8.2) is similarly sidestepped.

The issue of using only projectable concepts (§ 2.8.2) is resolved by the reuse of skills (scheme) that the simulation has evolved based on action on the environment. Similarly, the issue of preferring entrenched concepts – when more than 1 generalization applies (§ 2.8.2) – are sidestepped by reuse. Similarly, the external reward mechanism and may build sets of conflicting networks, but this is consistent with performance of children, so is not an issue.

Chapter 4. Architecture and Design

The issue of induction conflicts and deduction overrides (§ 2.8.2) occurs in Drescher's work because of the symbolic nature of processing (Drescher, 2002, p89). This research builds binary schemes which execute as evolvable hardware (Greenwood and Tyrrell, 2006), and so sidesteps these issues.

Cognizant failure (§ 2.7.2) is utilized, whereby imperfect algorithms are created with a mechanism to detect failure and rectify the imbalance (§ 2.3.5 on Piagetian Equilibration / Disequilibrium). The identification occurs when inputs (from the external or internal environment) cannot be processed, is rectified by the reconstruction of a scheme that can process the input. The example provided in the evaluation concerns counting (§ 5.2.5.1).

Reinforcement is implemented as an evaluation function for each class of schemes (§ 4.2.3.5 for example).

4.2.1.1 Key Implementation Features

There are several key features of the implementation that need to be accounted for. Each feature is described separately.

In this implementation the Piagetian notion of a "concept" for example number, is the execution of information through the set of schemes, as digital circuits, that can internalize the externalized "value" and processes it through assimilation and accommodation (Furth, 1969, p76–78).

The simulation makes use of an interaction grammar. This is an extension on the use of the *permanent object invariant*, to allow the simulation to interact both externally and internally using the processes of assimilation and accommodation. By using EBNF, the simulation can be constructed in a modular fashion, which reduces the overall development time. As can be seen in the ERD in figure 4–1, the simulation makes use of list structures to hold the environmental information.

Chapter 4. Architecture and Design

All the scripts and interactions make use of the EBNF grammar that defines what can and cannot be interacted on the worksheet and on the number line.

The image below, figure 4-1, describes the entity relationship diagram (ERD) of the grammar:

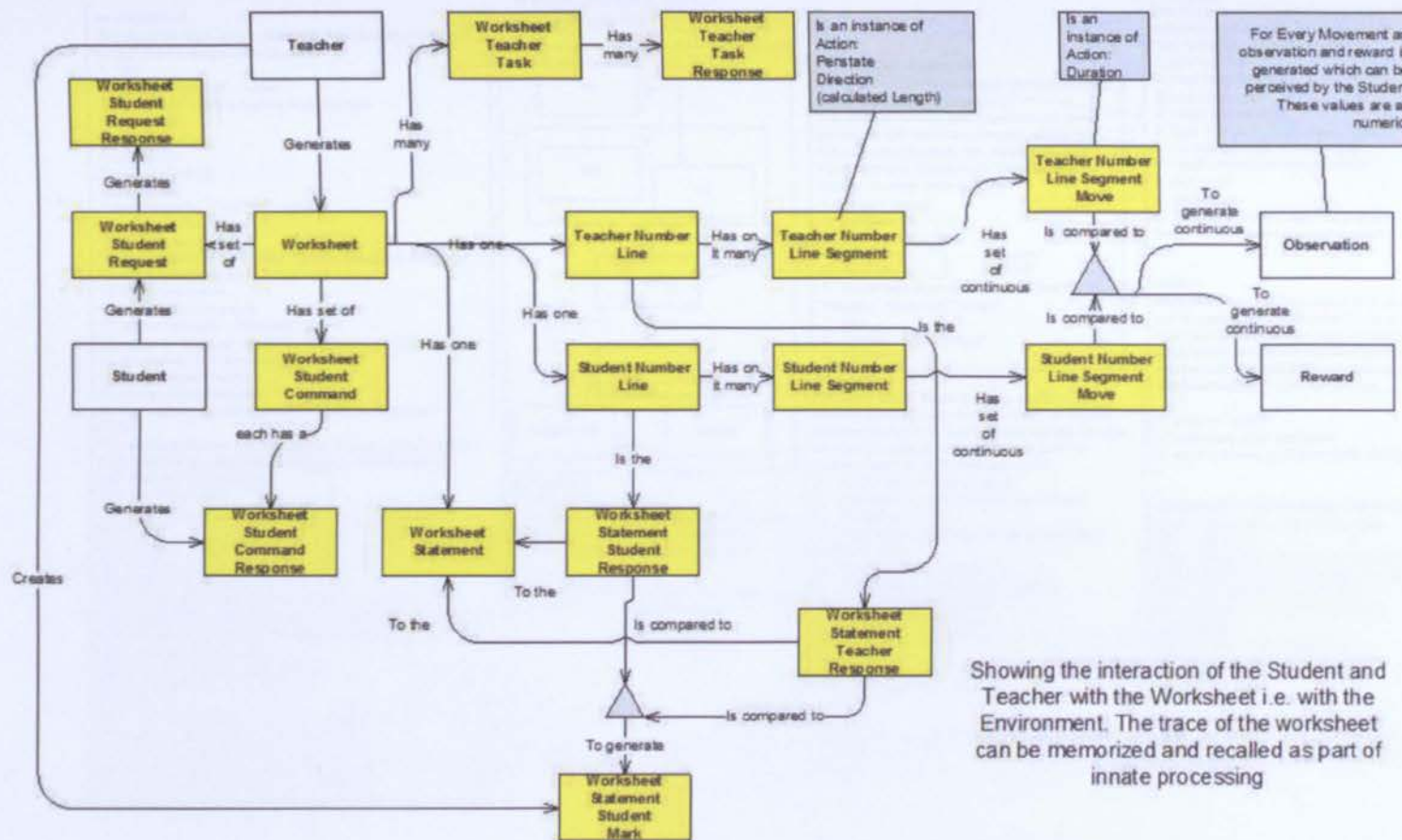


Figure 4-1 An entity relationship diagram of the EBNF grammar of the system, which defines the relationships in the number line world.

The relationships in the EBNF are defined with the following image, figure 4-2:

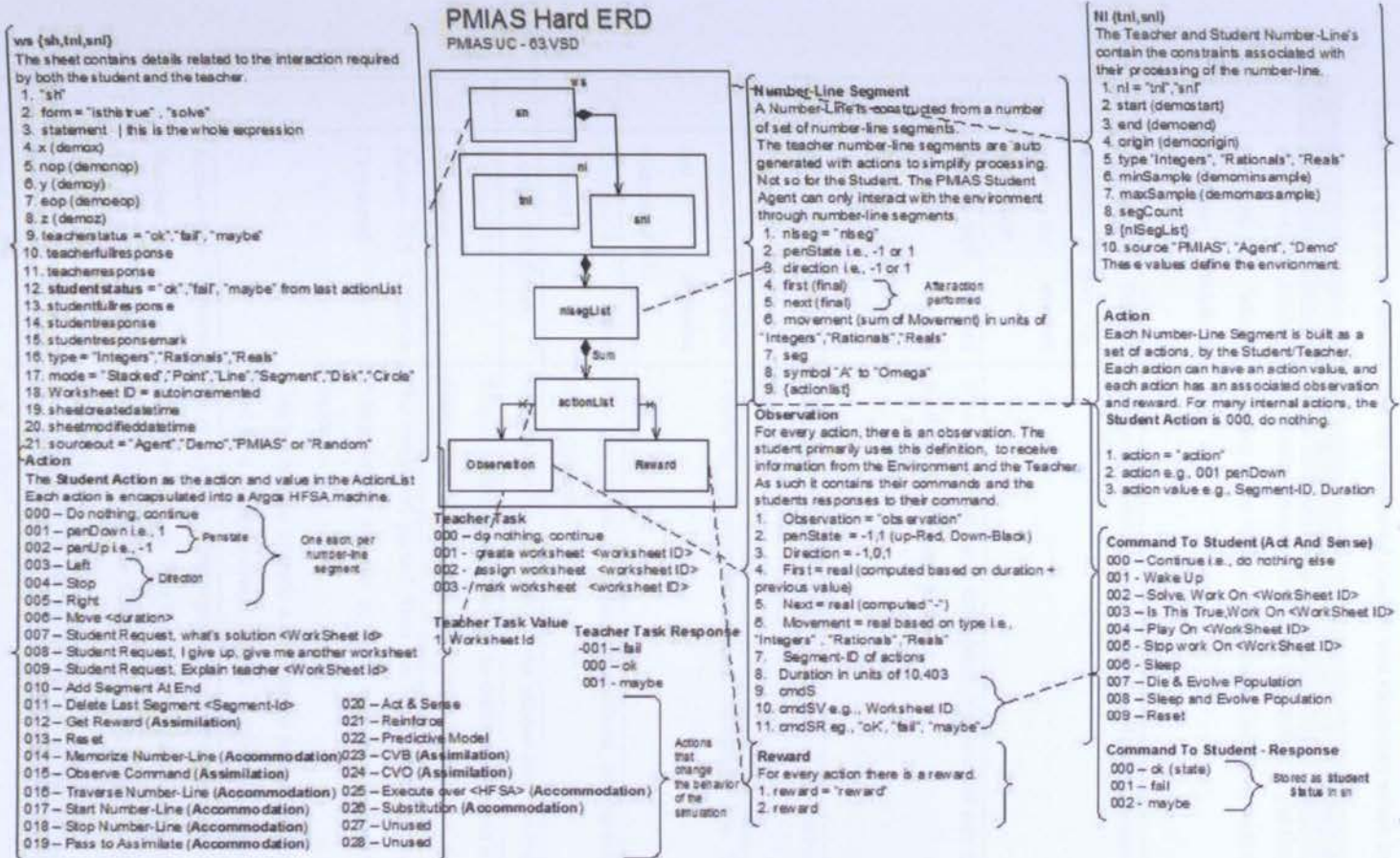


Figure 4-2 The ERD of actions (through accommodation) that the simulated student can take to process their number line world. The student makes random choices and the environment responds. Then resultant observation and rewards are passed back to the student through the process of assimilation.

Chapter 4. Architecture and Design

Another way of viewing the ERD is to see it as a set of actions that the simulation can perform through the process of accommodation. These controlled actions are intercepted by the executing system and perform a series of housekeeping duties, as detailed in table 4.1:

Table 4-1 The set of actions that the simulated student can randomly select from to adapt to the environment.

#	Action	Description
1	Action-007 Student Request, what's solution <worksheet ID>	Randomly chosen, this allows the student to get the answer. Potentially, this could provide a moniker to the student, which could be stored with the HFSA.
2	Action-008 Student Request, I give up, give me another worksheet	Randomly chosen, this allows the student to change the worksheet.
3	Action-009 Student Request, Explain Teacher <Worksheet ID>	When a worksheet is completed, the student can randomly choose this action to get more feedback on the same problem.
4	Action-010 Add Segment at End	Adds a segment at the end at the end of the current number line. This is typically used in play.
5	Action-011 Delete last segment <Segment ID>	Delete the last segment in the current number line. This is typically used in play.
6	Action-012 Get Reward	Gets the reward from the environment and updates the activation of the currently active HFSA (scheme).
7	Action-013 Reset	Reset is used to clear the current actions and return the system to the last stable state.
8	Action-014 memorize	This is called by the Student to deliberately store a memory (which is a combination of the observation and the HFSA that processes it). This is an aspect of accommodation.
9	Action-015 Observe Command	The command passed from the Teacher to the student is decoded by the Executive and a determination of what to do is executed. This entails finding a HFSA to respond to the request. This could also provide a description to the Student, of what the Piagetian scheme is to be used for e.g., equivalence. This is an aspect of assimilation.
10	Action-016 Traverse Number line	The next number line segment is observed and passed to the student. There being an inherent order in the inputs. This is an aspect of accommodation.
11	Action-017 Start Number line	This is called by the Student when they want to start a new number line. This is typically used in play.
12	Action-018 Start Number line	This is called by the Student when they want to stop a new number line. This is typically used in play.
13	Action-019 Pass to Assimilate	Passes the value to assimilation. This is an aspect of accommodation.

Chapter 4. Architecture and Design

#	Action	Description
14	Action-020 Act and Sense	The environment triggers the Student to change into the Act and Sense mode.
15	Action-021 Reinforce	The environment triggers the Student to change into the learning mode of reinforcement.
16	Action-022 Predictive Model	The environment triggers the Student to change into planning mode.
17	Action-023 CVB	Converts the observed value into a binary form. This conversion is akin to pixilation of an image. This is an aspect of Assimilation. An argument could be made that CVB and CVO are synonyms for the action of substitution. Given enough time, this conjecture could be proven.
18	Action-024 CVO	Converts the internal binary form observed external value. This is a critical piece and may require the substitution of machines that are "related" to the observed properties of the environment. An argument could be made that CVB and CVO are synonyms for the action of substitution. Given enough time, this conjecture could be proven.
19	Action-025 Execute over <HFSA>	This command executes the previous HFSA values over the selected HFSA. This serialization is the mechanism that allows for instance the equivalence machines to work and is an aspect of accommodation. It is also the process by which machine can consume one another. The issue is how the machine can consume the processes of the other machine, unless it has some form of introspection...the capability to build a new HFSA. For instance, this would allow the system to bootstrap from FSA to HFSA.
20	Action-026 Substitution	Substitution is used in the Piagetian "less than" machine to replace the result of the Piagetian Equivalence machine of 0, 1 with an internal e.g., 0=Move 1 Left, 1 is Move 1 Right etc. Substitution is used in the Piagetian "counting machine" to replace the external movement of 1, with an internal value of binary 1. Thus, substitution, which takes an external value and replaces it with an internal one and an internal value with an external one, is the basis of interiorization.

There is no need to generate a system action to execute, because the machines are inherently built with the Argos reactive framework with externalized actions in mind e.g., A prediction machine includes commands such as Action-006 which cause a movement on the number line, thus when this machine is executed, it causes actions on the external environment.

4.2.1.2 Summary of Major Architectural Features and Design Elements

A summary of the major architectural features and design elements are related to the literature review in table 4.2 below:

Table 4–2 This table defines the traceability matrix of relationships between the features exposed through biology and evolution, to the concepts in Piagetian theory and the architecture and design features used in the simulation.

Features as exposed through biology and evolution	The concept in Piagetian Theory	Architecture and Design Feature
Thalamic loop (Albus, 2008; Albus, 2010b, p193; Granger, 2006a; Granger 2006b).	Knowing circle (Furth, 1969, p147).	The student as state machine (§ figure 4–8) which executes the inputs over the hierarchical schemes.
Event Hierarchies (Albus, 2008; Albus, 2010a and 2010b).	Accommodation as an abstract process (Furth, 1969).	The thread through the executing hierarchical schemes that process actions (both externally or internally)
Receptive Field Hierarchies (Albus, 2008; Albus, 2010b, p193).	Assimilation (Furth, 1969).	The thread through the executing hierarchical schemes that process observations and rewards.
Model of Drosophila (Miesenböck, 2008, p52; Shang, Claridge–Chang, Sjulson, Pypaert and Miesenböck, 2007, p601) and Straital System (Granger, 2006a; Granger, 2006b).	Knowing Circle (Furth, 1969, p147).	The student as state machine (§ figure 4–8) which switches states.
1) Cortex → matrisome projections (acting).	1) Act(Furth, 1969, p154–163).	1) The accommodation side of a scheme (§ 2.3.2) which acts internally or externally.
2) Sense.	2) Sense (Furth, 1969, p154–163).	2) The assimilation side of a scheme (§ 2.3.2) that receives input from the external or internal environment and builds structures.
3) SNc dopamine (DA) projections to both matrisomes and striosomes (learning through reinforcement).	3) Learn (Furth, 1969, p154–163).	3) The evolution of scheme as networks of FSA and HFSA (§ 4.2.1.1).
4) TAN projections to matrisomes (exploration) which relates to predictive modeling (planning) along with actor–critic TD	4) Plan (Furth, 1969).	4) The reuse and reconstruction of schemes that internally model the external environment to predict future states.

Chapter 4. Architecture and Design

Features as exposed through biology and evolution	The concept in Piagetian Theory	Architecture and Design Feature
learning.		
CCU (Albus, 2010b).	The structure of a scheme (Furth, 1969; Copeland, 1974).	The FSA structure which processes information, combined with the HFSA that processes information across the FSA network.
Dopamine Reward Processing (Cannon and Bseikri, 2004; Schultz, 1997; Schultz, 2000).	Reinforcement through the processes of imitation, which constructs schemes (Piaget, 1954, p4; Piaget, 1964, p13).	The externally constructed evaluation functions for the FSA and HFSA.

The implication of the information in table 4–3, is that there is a plausible biological implementation for Piaget’s work that agrees with other researchers.

4.2.2 Design

The research consists of a set of frameworks (§ 3.4). Each key process of the design is detailed below.

4.2.2.1 Number line World

The following image, figure 4–3 shows the externalization of the simulation, i.e., the teacher; student and auditor all interact within this environment.

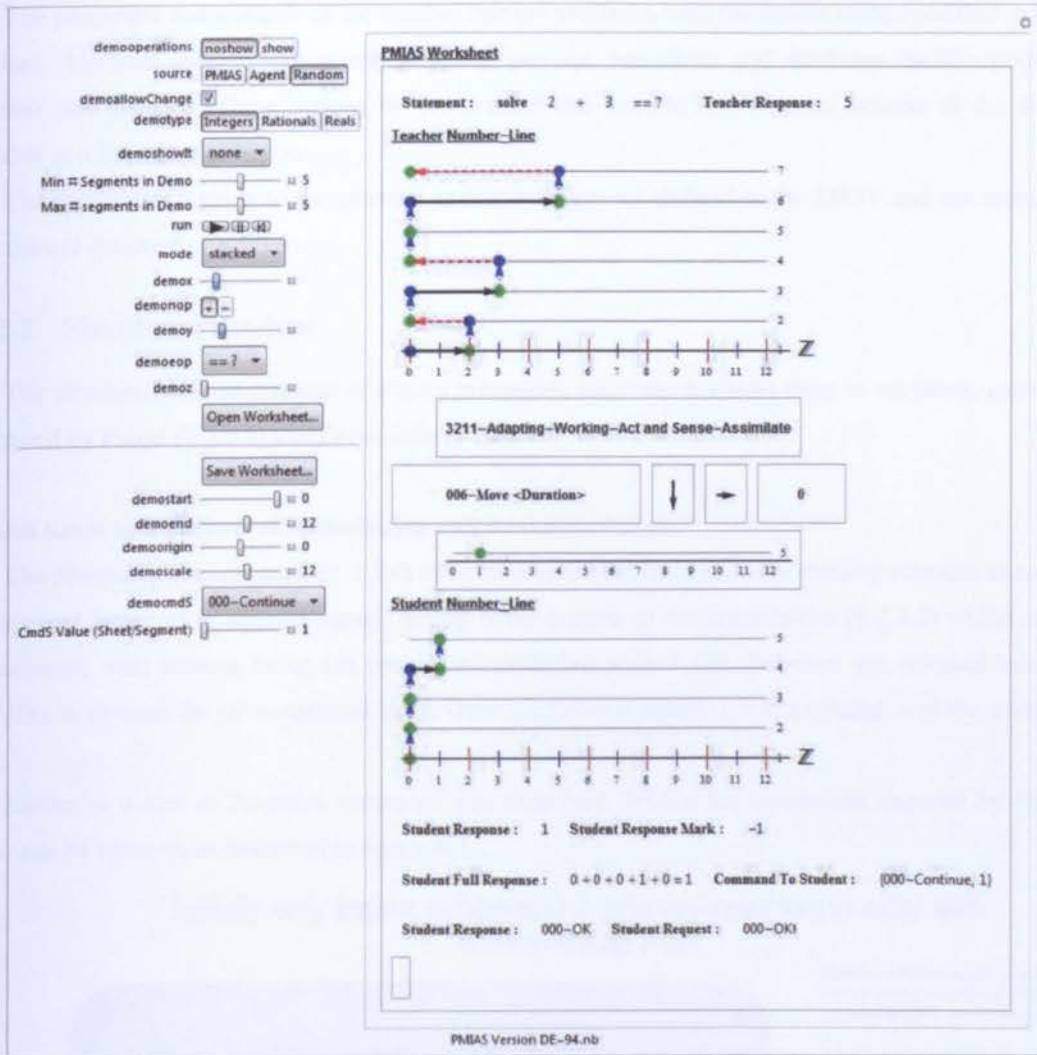


Figure 4-3 An image of the user interface provides the simulated student access to its number line world using a worksheet metaphor.

Displayed within this image, are the worksheet, the automated teacher's number line along with the students own number line. The auditor, as a user, can configure values on the user interface, display results, as well as stop and restart processing. The auditor, can also set checkpoint and export the current content of the students short term, long-term memory. A number line is visualized in the following diagram, figure 4-4:

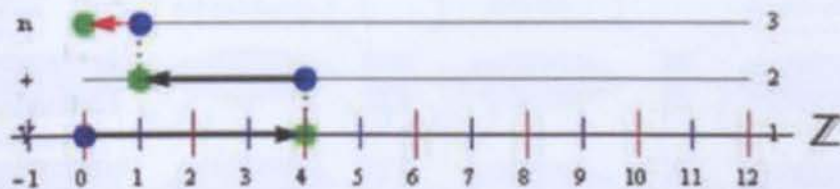


Figure 4-4 A depiction of the number line world that the simulated student receives observations from and performs actions on, using the information from the ERD.

Chapter 4. Architecture and Design

The properties and methods of the number line are available, with the details being specified in the EBNF grammar. Methods include movement, stopping, pen-up, pen-down and attributes include point, length, direction and distance. These actions by the teacher and internal and external actions of the student are available as a list of action values.

The inputs and outputs to the external environment are all defined in the EBNF and are accessed as an environment structure (§ 4.2.1.1).

4.2.2.2 Simulated Student

The simulated student consists of a main processing loop which allows them to act/sense, learn and plan as dictated by Piaget (§ 2.3.5) and Drosophila (§ 2.2).

Act and Sense as a Process of Assimilation and Accommodation

The process of act and sense (§ 2.3.6) consists of selecting from memory existing schemes to adapt to the environment (external or internal input). Acting is the process of accommodation (§ 2.3.2) which acts on the environment, with sensing being the process of assimilation (§ 2.3.2). Schemes are selected based on the capability to process the environmental input. Once a scheme is selected, it is executed over the environmental input.

Earlier, a model of Piagetian processes was presented. Within the constraints imposed by HFSA, this model can be realized, as described in figure 4–5.

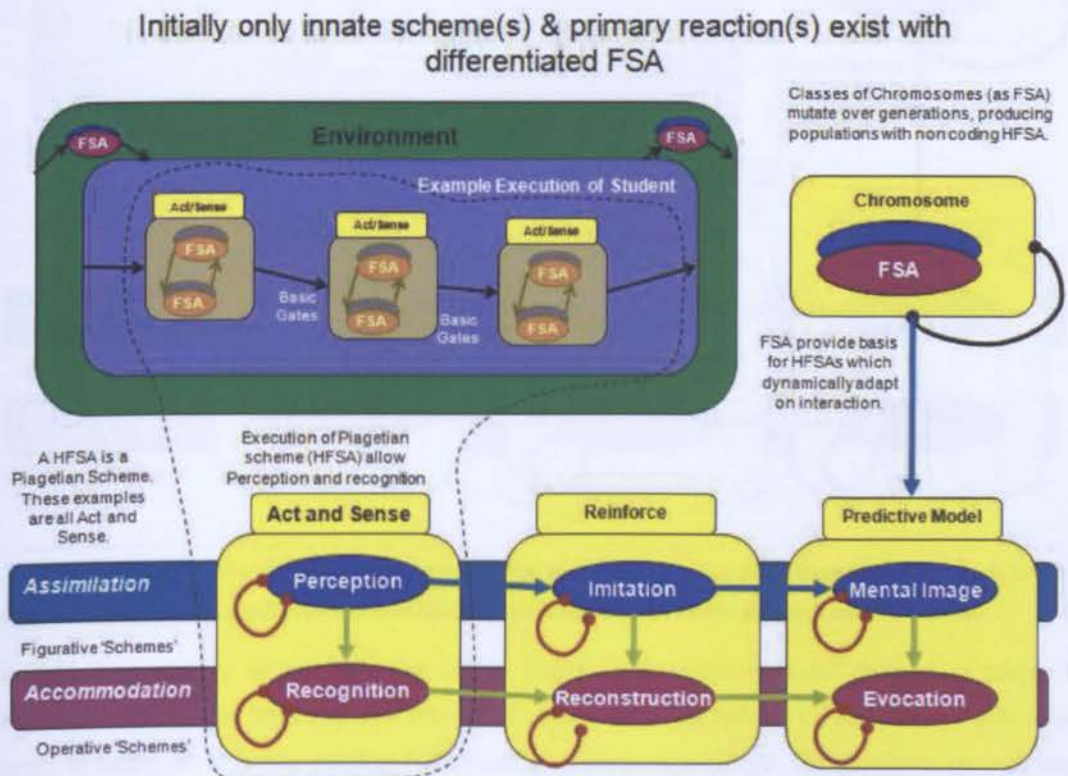


Figure 4-5 The abstract processes of act and sense, is a generalization of the processes of assimilation and accommodation and is used to produce the Piagetian processes of perception, recognition. This is the mechanism by which system automatically responds to and acts on the environment.

In acting and sensing, the system perceives and recognizes the environment using a set of HFSA that marshal information between processing units (FSA). The system is constrained to use only those HFSA (as schemes) that are currently available to sense (process inputs) and act on the environment. All of these machines (HFSA) have been evolved from lower level prediction FSAs. Acting and sensing however, cannot build new machines, it cannot learn. All it can do is reuse what it already has. This is consonant with Pascual-Leone's ideas of M-Capacity, i.e., learning is expensive, so it acts and senses first.

Reinforce as a Process of Assimilation and Accommodation

The process of learning generates new schemes using the evolutionary framework as detailed in figure 4-6.

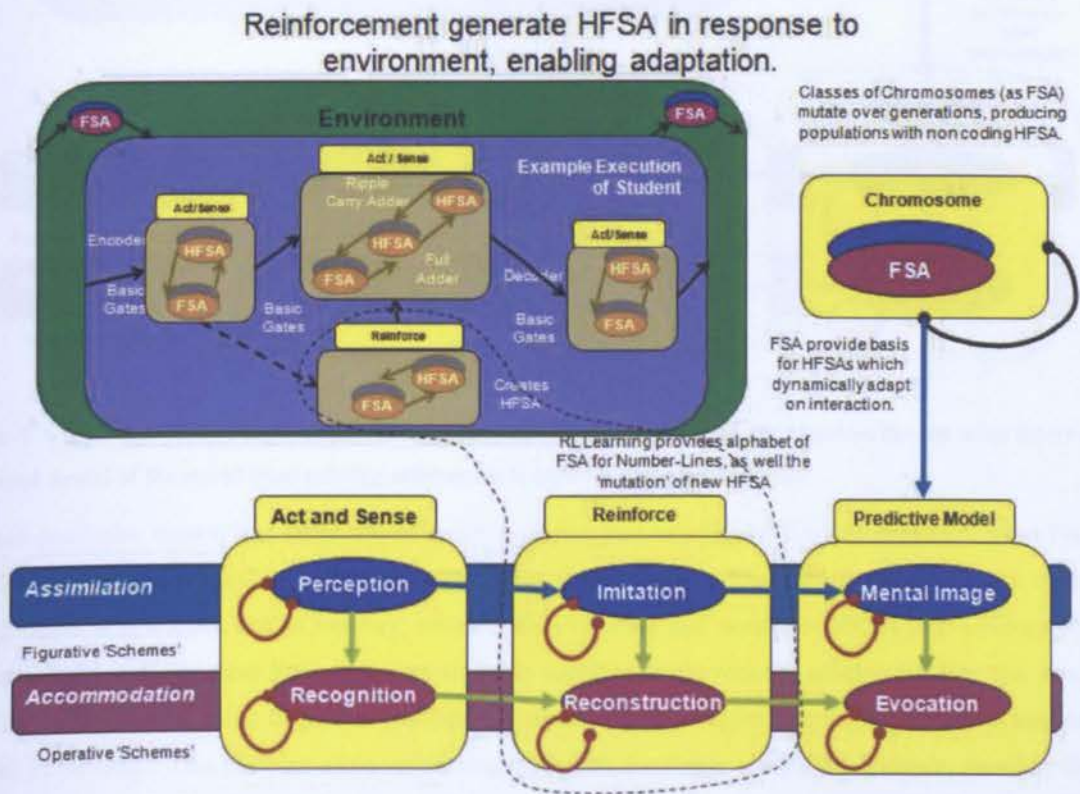


Figure 4-6 A depiction of reinforce, which is the mechanism through imitation and reconstruction which generates (learns) new schemes that can predict environmental information (internal/external) and act differently.

Reinforce, is a process of building new HFSA based on the environmental demand to adapt. This Piagetian process, allows for imitation and reconstruction of the observations of the external and internal environment. In doing so, it reuses existing HFSA and constructs new HFSA. The reinforce process cannot make predictions.

Planning as a Process of Assimilation and Accommodation

The process of planning reorganizes existing schemes to make an internal predictive model of the environment as described in figure 4-7. Planning is not implemented in this research.

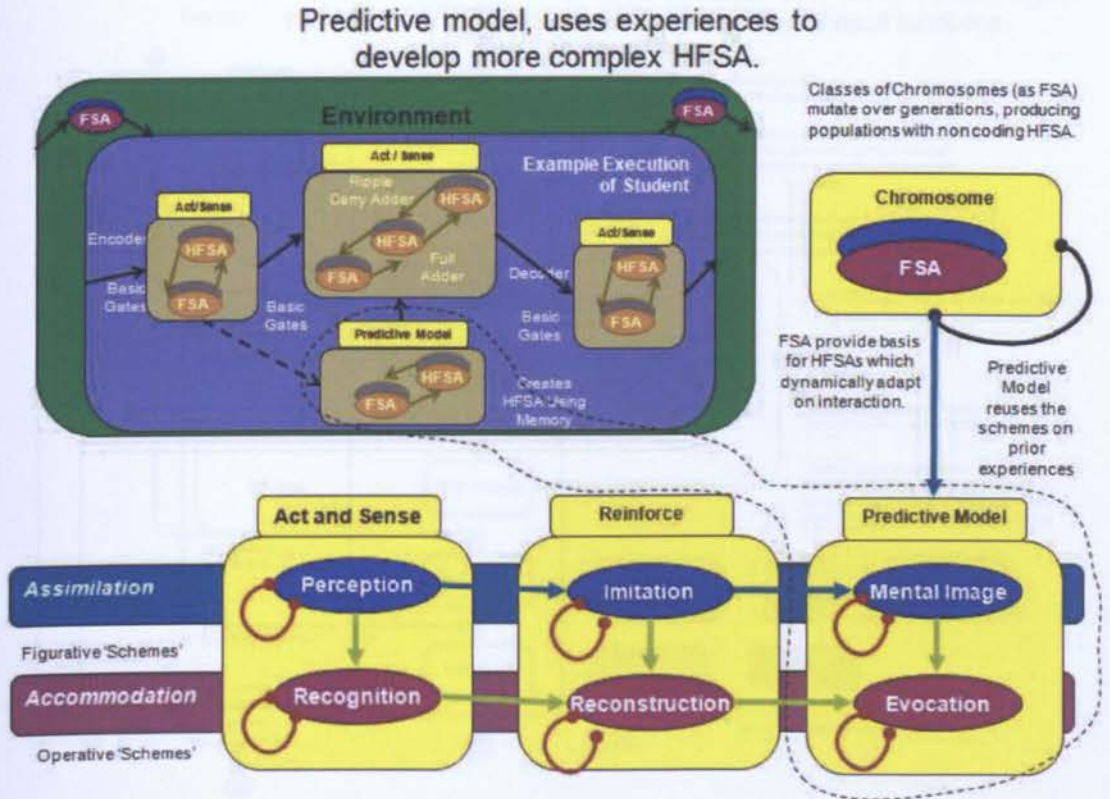


Figure 4-7 A depiction of the Piagetian processes of mental image and evocation that enables the use of an internally constructed model of the world from existing schemes. It is used for planning and play.

This predictive model, can make use of newly acquired and existing HFSA and construct, what Piaget referred to as “mental images” of the environment. From these a HFSA can evoke (as in plan), new actions. The key aspect is that it can use its memory, which is the execution and storage of HFSA with environmental values attached i.e., the number line actions are stored in memory, along with the scheme (HFSA) that process them. This is different to other cognitive development models. It also begins to explain why some memories “feel like childhood.” This presents some interesting properties for future work. For example, recalling from memory a scheme of 1, will in fact, pull from memory, the execution of that HFSA, with its latent values attached. An analogy would be the reconstruction of a number line and the evocation of a change in configuration.

Student as State Machine

The Student itself is implemented as a state machine. This ensures that its processing is consistent and provides a balanced, deterministic implementation of a Piagetian model of processing (§ 2.3.). The following

diagram (figure 4–8) provides details of the valid states and transitions that the Student can pass through. This switching is reflected in the student scheme process, which calls accommodation first, then assimilation, continuously.

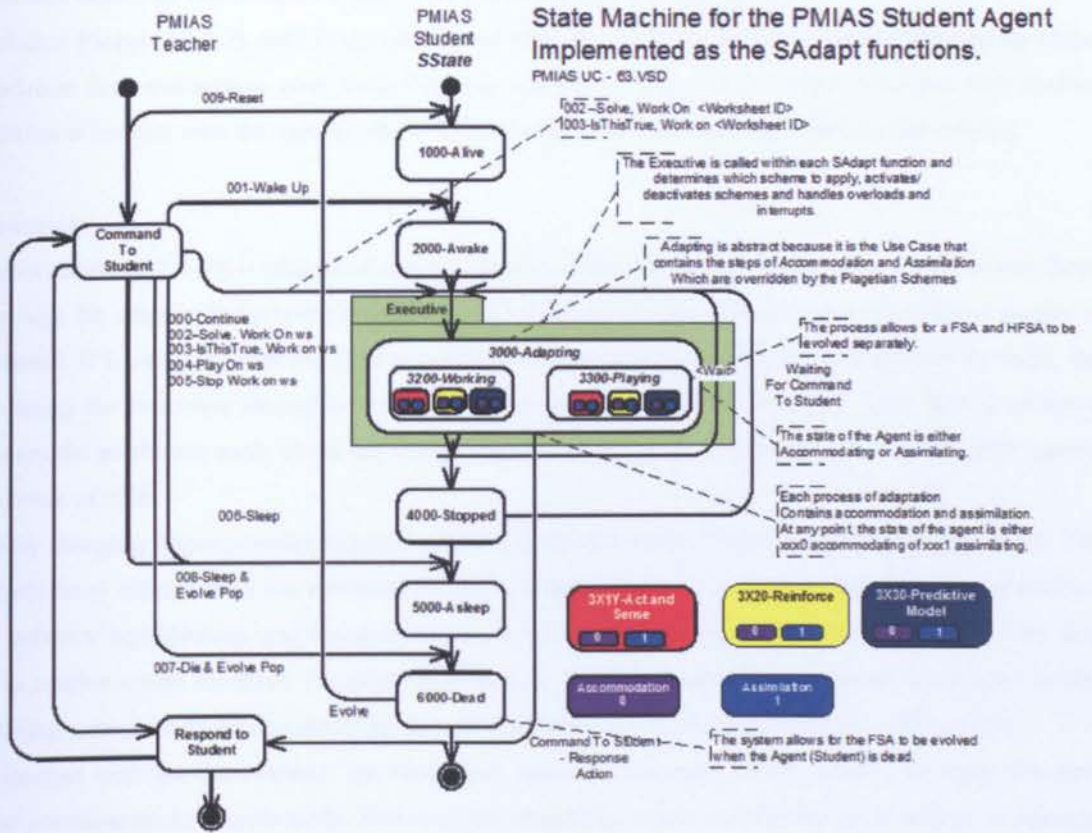


Figure 4–8 A depiction of the simulated student as a state machine. This is seen as a critical piece of the architecture and design, since it suggests that the student operates using the same processes of assimilation and accommodation.

From this, it can easily be seen that the student’s state can change, either by external or internal action. Of significance is that there is a dichotomy of processing, first accommodation then assimilation, consistent across the distinct approaches of acting/sensing, reinforce and predictive model. This dynamic switch is embodied in an executive, modeled after the theory of constructive operators (Furth, 1969, p75; p147).

Schemes

There are 3 classes of machines (EVTypes), that the class-based evolution builds Prediction schemes, Argos schemes and Piagetian schemes.

Prediction machines are purely FSA. There are different classes of prediction machines each with a unique, externally provided evaluation function that constructs it based on a set of inputs and outputs. For instance, prediction machines can be constructed as Boolean components in a network, such as OR, NOT, AND machines or as machines that act on, or receive input from the environment. For instance, the system can construct a scheme to represent 3 as 1+1+1 or 2+1.

Chapter 4. Architecture and Design

Argos schemes are HFSA that process information across networks of FSA. This enables digital circuits to be constructed that follow the synchrony hypothesis (Maraninchi, 1992; Maraninchi and Remond, 2001). Argos schemes contain the static linkages between FSA and other HFSA Argos schemes. By themselves, Argos schemes cannot be executed, they need to be contained with a Piagetian scheme.

To follow Piaget (§ 2.3.5), each Piagetian scheme must consist of assimilation and accommodation steps, accommodation first, assimilation next. Each Piagetian scheme includes a set of Argos schemes. This enables the simulation to interact with the number line world and experience the world as a thread of processing.

Other Processes

Interiorization (§ 2.3.10) is process of conversion of external values into binary values, and reusing these values through the scheme. Reflective abstraction (§ 2.3.9) is considered to be the reuse of existing schemes in a new scheme. It is imagined that the system enables truth (validity, equivalence) as a scheme by using the planning (using the predictive model) to accurately make predictions about the world. Truth then is no more than a successful prediction made about the world, which exists as a set of schemes. This means there can be different levels of truth.

Merely stringing Argos schemes together will not produce a viable Piagetian system that can count. The system itself must interact with the environment and through a process of assimilation and accommodation oscillate between equilibration and disequilibration, acquiring structures that are more complex. The key question to resolve within the given Piagetian framework is “How do more relevant executive schemes finally become strong enough to overcome the irrelevant scheme” (Pascual-Leone and Goodman, 1979, p309).

To interact with the environment, the simulation needs to determine which schemes to apply (become active and consequentially deactivated). This requires Piagetian, Argos and Prediction machines to connect together to respond both to inputs and observations of the environment, as well as to internal structural changes. Since the model proposed by Furth (Furth, 1969) is incomplete, it cannot be executed because the mechanism to marshal information between schemes was undefined. The research by Pascual-Leone^{vi} (Pascual-Leone and Goodman, 1979) is used to provide an executive an interrupt processes and those additional features required to build a workable system.

4.2.2.3 Simulated Teacher

The teacher is a construction of various teaching scripts, configured by the auditor as worksheets using the EBNF grammar. The teacher completes these worksheets, they solve the “teacher side” of the problem e.g., Solve $1 + 1 = ?$ using the features of the environment. In turn, these worksheets are given to the student to interact with. The student works on the worksheets and when completed, these are marked by the teacher. The results are passed back to the student for further downstream analysis. This provides the basic reward mechanism for the student. The rewards are used to strengthen schemes.

4.2.2.4 Auditor

The simulation provides the role of auditor who can perform real-time configuration of the currently executing environment.

4.2.2.5 Learning as Evolution – Using Evolutionary Computing to Learn Schemes

The evolutionary computing framework produces new schemes (as in learn concepts). Each scheme exists in a class of schemes, and there may be many classes of schemes. Each instance of a scheme has a specific set of values in its chromosome structure. Each class of chromosomes may go through many evolutionary cycles. Each cycle of evolution uses mutation to change the structure of the finite state and HFSA. Mutation can randomly change every feature of the automata as states, inputs, output and transitions.

To support restart in processing, generations of chromosomes and their associated classes are stored offline, typically in files. These files can be read and the simulation restarted from its last invocation point, as shown in figure 4-9:

■ Step 1 - Get the population

```
(* Step 1 - Initialize - get the EVGen, EVChromosomeClassID,  
EVChromosomeClass and Parents from the previous generation. In this case, the initialized values *)  
filename = "EVGen"; EVGen = Get[filename]; (* Get the latest EVGen Count *)  
EVGen = 1;  
filename = "EVChromosomeClassID" <-> EVGenText[EVGen]; EVChromosomeClassID = Get[filename]; (* Get the EVChromosomeClassID *)  
filename = "EVChromosomeID" <-> EVGenText[EVGen]; EVChromosomeID = Get[filename]; (* save initialized EVChromosomeID *)  
filename = "EVChromosomeClass" <-> EVGenText[EVGen]; EVChromosomeClass = Get[filename]; (* Get the EVChromosomeClass *)  
filename = "EVParents" <-> EVGenText[EVGen]; EVParents = Get[filename]; (* Get the parent population of chromosomes *)  
EVGen  
getEVClassList[] (* Get the list of the EVClasses *)
```

Figure 4-9 An example execution that retrieves populations of chromosomes from external storage. This allows the agent to be reactivated and restarted from known positions.

In this example, generation 1 is being retrieved from storage. It is conceivable that in long running sequences of interactions, this process of storage and retrieves occurs automatically.

Chromosome Structure

Each instance of a scheme belongs to a scheme class, and has a unique chromosome structure. The FSA are encapsulated within a chromosome that consists of the qualitative values of fitness / prediction quality, the FSA, the strategy parameters that describe its ID, its class and parental history (§ 4.2.3.1 for a definition of prediction quality using Algorithm 1 and § 4.2.3.3 for a definition of fitness using Algorithm 3). A depiction of a simple chromosome is shown in figure 4-10:

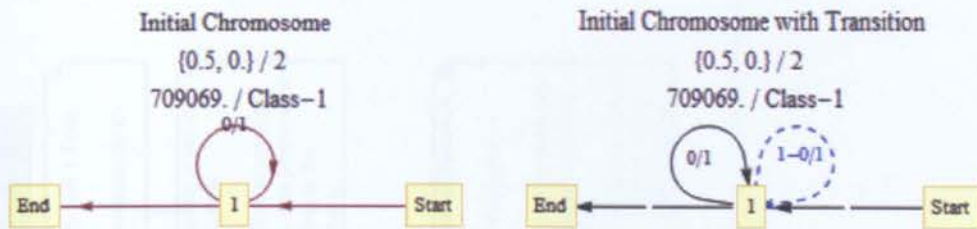


Figure 4-10 A simple depiction of a chromosome with transitions

This initial static chromosome is depicted on the left of figure 4-10 and with the transitions as they can be processed as inputs by the FSA on the right of figure 4-10.

The chromosome structure is depicted in the figure 4-11 below:

```

chromo[q[{0.5, 0.}], p[- Automaton -], s[{709069., 1., Class-1, 0., 0., {}}]]

Head = chromo
q-----
fitness = 0.5
prediction Quality = 0.
p-----
Head = Automaton
Type = Mealy
States = {1, 2}
FromState  Input  ToState  Output
1          0      1         1
Initial State = 1
Final States = {1}
Embedding = {}
Alphabet = {0, 1}
s-----
Chromo ID = 709069.
Chromo Class = 1.
Chromo Class Name = Class-1
Chromo Mutation Attempts = 0.
-----
    
```

Figure 4-11 Structure of a chromosome

There are a set of simple rules for building FSA and HFSA. These are depicted in figure 4-12:

An ERD for a h[] structure in Chromosome

It enables random Evolution of Hierarchical Finite State Automata (HFSA).

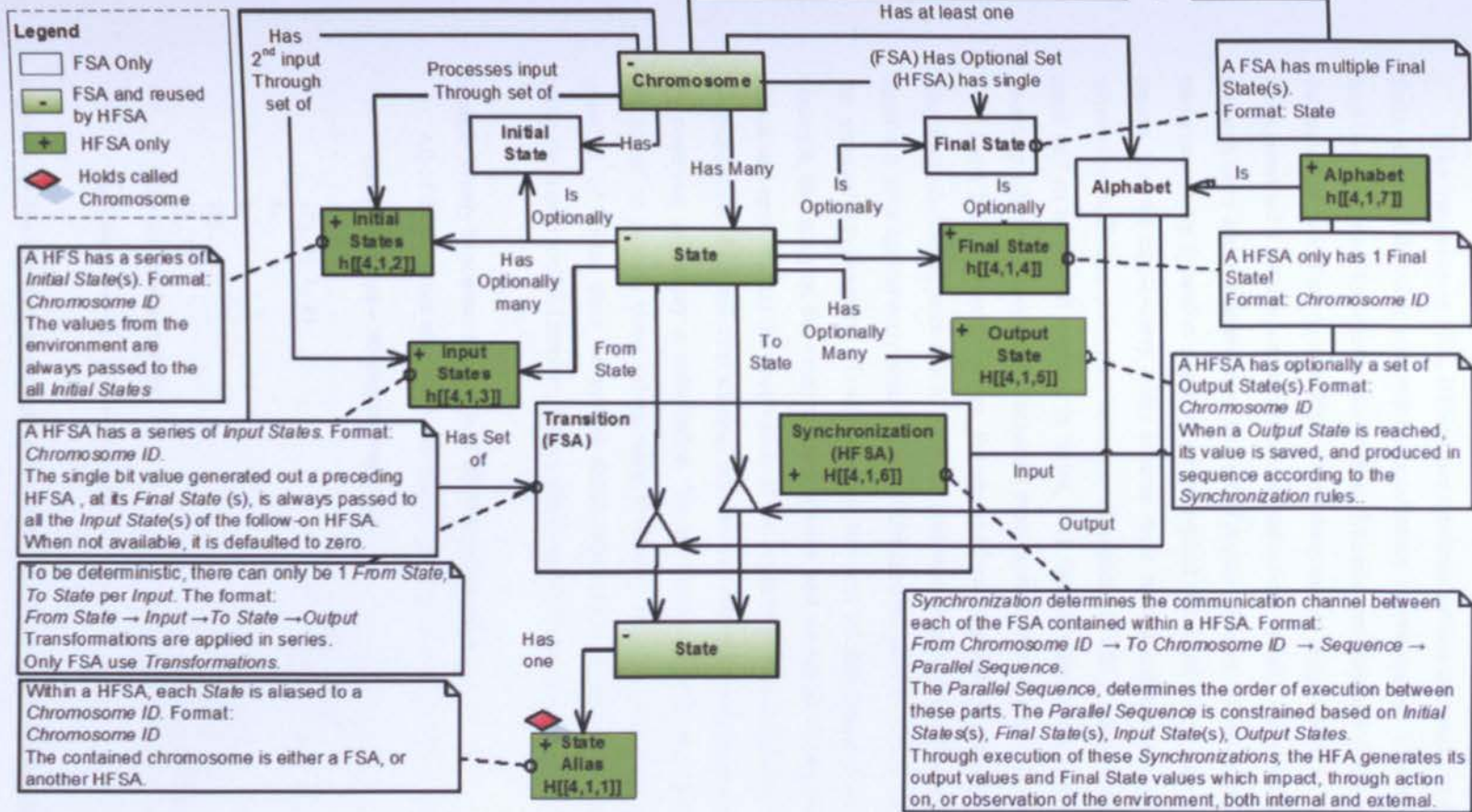


Figure 4-12 The entity relationship diagram for chromosome, it contains the rules by schemes are mutated. All the details are included in the h[] structure in the chromosome. It supports both FSA and HFSA (Prediction, Argos and Piagetian schemes).

Chapter 4. Architecture and Design

All FSA schemes are evolved, with every mutation producing a valid set of transition rules. A reward is provided by the environment based on its utility and warranty, which strengthen their use.

Like transitions in FSA, HFSA's use synchronizations to statically bind chromosomes together. This gives them temporal duration and enables consistency of processing. This static binding does not, however, lend itself to the need for interaction with a dynamic environment (both internal and external). Consider, for instance how counting occurs. First, a structure cannot be pre-built to handle every set of inputs; the system must construct a mechanism to handle the interaction. Second, and more importantly, the mechanism by which, a ripple carry adder, implemented as a set of Argos schemes, works is that, over time, it takes a set of inputs, as observations by figurative schemes, of the typical values of 1 or 2 and accumulates each one separately. This means that the ripple-carry adder scheme must remain active over a period of time and more importantly, when disequilibrium occurs for instance, attempting to add 3 values with a single 2 bit machine, the system, must add an additional machine in series, such that it remains in equilibrium. Since the ultimate goal for the counting scheme is to count, the schemes must be rewarded to generate these structures.

For a FSA to process values, the observed external values must be converted to a binary representation. The most obvious approach is to have the system convert these values automatically. However, this simplistic approach goes against early observations of children (Copeland, 1974), which suggests that they are focused on the external environment. However, since the aim of this research is to test the internal construction of concepts and not just the external manipulation and detection of objects, this research assumes a simplistic stance and requires that the observation of the environment be detectable and constructible by a FSA. Thus the interiorization of a prediction scheme, one that can identify a unit measure, with a binary "1," which is suitable for counting, is simply a substitution. To be consistent with this approach, the system builds a "binary machines" to process these values using Prediction schemes and Argos schemes as digital circuits. The key question, of "when does the system detect regularities in the environment," has previously been addressed using the Verve toolkit (Streeter, Oliver and Sannier, 2006, p3) (see § 2.3.5 on *permanent object invariant*).

Simple Mealy Machine: the Basis of all Other Machines

All of the schemes start from this Mealy machine (Mealy, 1955, p1045), figure 4-13:

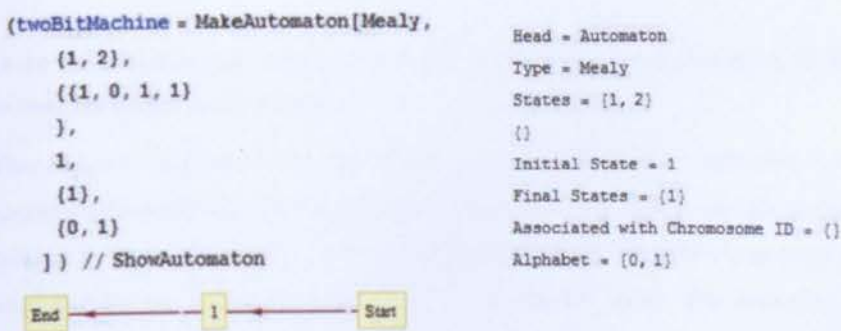


Figure 4-13 The structure of a simple mealy machine, The base for all other FSA

This FSA is composed of two states, with state 1 being the starting state and ending state.

Prediction Scheme: Buffer

Each prediction scheme starts from the simple mealy machine and is evolved to make predictions about the external world based on the characteristics of the class. For instance, if an environmental input is provided, and no machine can process it, then a machine is evolved to process it.

In figure 4–14 the simple Mealy machine is randomly mutated, such that given a 0 as input they produce a zero as output and given an input value of 1, produce a 1 as output. Over 2 generation steps, they generate chromosome 300093, with a prediction quality of 1.0 and a fitness of 1.55 (§ 4.2.3.1 for a definition of prediction quality using Algorithm 1 and § 4.2.3.3 for a definition of fitness using Algorithm 3).

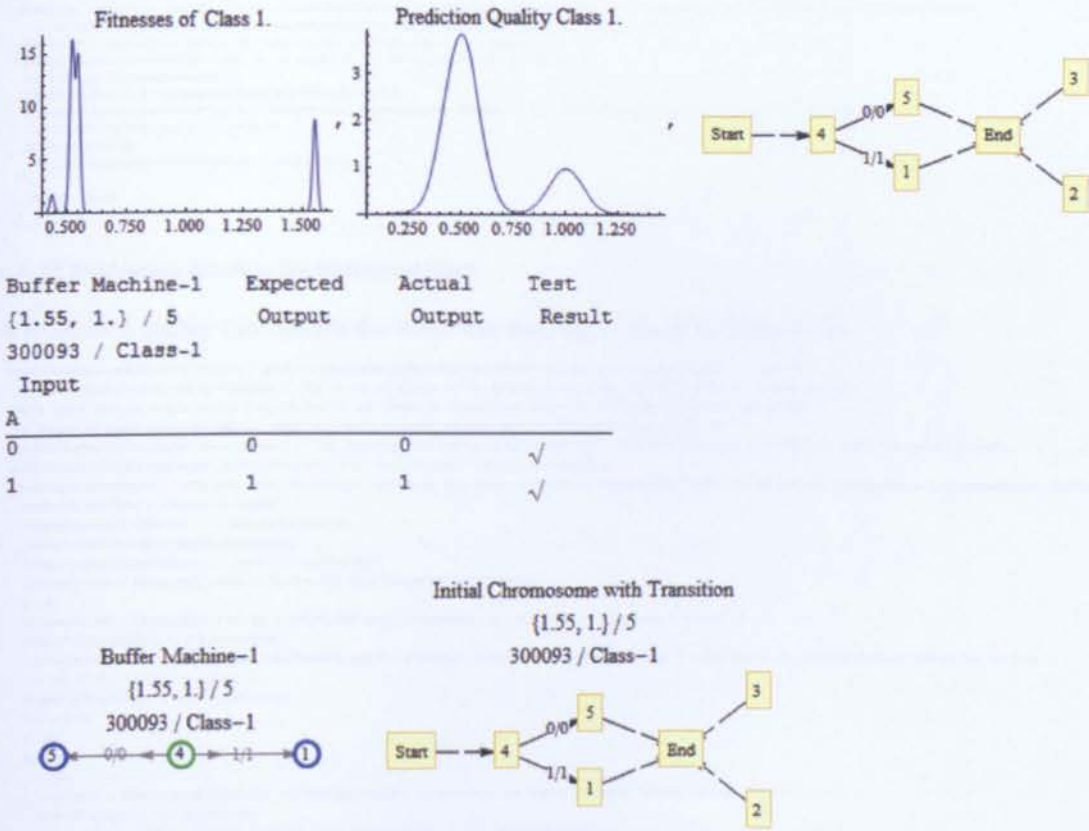


Figure 4–14 The evolution and evaluation of a class (Evclass 1.) of mealy machines, in this example a simple buffer machine from the simple mealy machine.

What figure 4–14 shows is that this evolved machine has 5 states, with only 1, 4 and 5 being valid states. The transition values 0/0 and 1/1 are shown between the initial state 4 (shown in green) and the final states 5 and 1 (shown in blue). The graphs of fitness and prediction quality show a non-normal distribution, which is anticipated due to the evaluation and selection mechanism used. The majority of chromosomes have a prediction quality of 0.5, i.e., they do not correctly predict the next state. Only a small percentage has a prediction quality of 1. Figure 4–15 shows that 604 attempts were made to evolve this initial class, with the top 5 being shown for each generation step.

Chapter 4. Architecture and Design

Prediction machines are essentially of two types, those that generate predictions using the *permanent object invariants* from the ENBF, and those that generate components of a digital circuit. In generating digital circuits, the inputs are all the possible set of internal binary values. Each separate class of schemes has a prediction quality and fitness evaluation functions.

An evaluation function for the buffer not class is listed in figure 4–15. This is typical of the evaluation functions required:

```
evalBasicGate::usage = " evalBasicGate[aut, _inputs, _outputs] returns the fitness of an BASIC GATES automata and the prediction quality to match the Set of input and output signals.
Inputs can be either of the form {{a1,a2,a3,a4},{b1,b2,b3,b4}} or {{a1,a2,a3,a4}}. Note there must be double braces!
Inputs can be any length so long as the length of each input string is equivalent.
The length of input string must be the same length as the output strings. outputs must be the form {a1,a2,a3,a4}.
The maxime value of fitness is 1.50 for a NOT gate and 1.55 for an AND, OR, NAND, XOR, XNOR, XNOR gate.
The maxime value for prediction quality is 1.0.
"

evalBasicGate[aut, _inputs, _outputs] :=
Module[{pp, lastOfFirst, finalSta, fit = 0, fitLastPrediction = 0, fitLastOfFirst = 0, fitFinalSta = 0, fitNumberStates = 0, maxNumberOFStates = 5, numberOFStates},
{pp, lastOfFirst, finalSta} = predictionQualityBasicGate[aut, inputs, outputs, 1]; (* Get the prediction quality *)
If[finalSta, fitFinalSta = .95]; (* Add scalar if the lastPrediction == firstInput *)
If[lastOfFirst, fitLastOfFirst = .95]; (* Add scalar if the FSA is back at its initialState *)
numberOFStates = Length[States[aut]];
If[numberOFStates = 0, fitNumberStates = (numberOFStates+0.01)];
If[numberOFStates = maxNumberOFStates, fitNumberStates = (numberOFStates+0.01)]; (* Add a scalar based on the number of states...it should be 1 for AND,
OR, XOR, XNOR, XNOR and 1 for NOT, Buffer *)
If[pp = 0, fit = 30,
fit = pp + fitFinalSta + fitLastOfFirst + fitNumberStates
];
{fit, fit, #|pp}}
];
```

Figure 4–15 Evaluation function for BufferNot class.

A prediction quality function for the buffer not machine is listed in figure 4–16.

```
predictionQualityBasicGate::usage = " predictionQualityBasicGate[aut Automaton, environ_List, initial_Integer:1] returns
(predictionQuality, True/False, True/False). Inputs can be either of the form {{a1,a2,a3,a4},{b1,b2,b3,b4}} or {{a1,a2,a3,a4}}.
Note there must be double braces ! inputs can be any length so long as the length of each input string is equivalent.
and length of input string is same as length of output strings. outputs must be the form {a1,a2,a3,a4}."
predictionQualityBasicGate::InvalidFormat = " The format of the inputs is not consistent, they should be pairs of input to match the output string";
predictionQualityBasicGate[aut Automaton, inputs_List, outputs_List, initial_Integer:1] :=
Module[{actualResponse, toPredict, bits, lastPredict, lastState, fit, numberOFInputSets, numberOFInputValues, fitnessValues, abOutputList, i, expectedOutput, thisInput},
numberOFInputSets = (inputs // Length);
Print["numberOFInputSets = ", numberOFInputSets];
numberOFInputValues = Length[inputs[[1]]];
Print["numberOFInputValues = ", numberOFInputValues];
fitnessValues = Table[#, {numberOFInputValues, 6}], {numberOFInputValues};
i = 0;
fitnessValues = (Table[#, {i+1, 10, 4}], {numberOFentries}) // Reverse; (* only works if there are 4 values *)
Print["fitnessValues ", fitnessValues];
abOutputList = Partition[Flatten[#, {Thread[inputs, outputs]}], 1 + numberOFInputSets]; (* Step 1 - Get the A and B's and Outputs together by row e.g.,
{a1,b1,a1},{a2,b2,a2} or {a1,a2} *)
Print["abOutputList = ", abOutputList];
fit = 0.0;
i = 0;
Table[
i++;
thisInput = Take[abOutputList[[i]], numberOFInputSets]; (* Take only the number of input values, not the expected output *)
Print["thisInput = ", thisInput];
expectedOutput = Drop[abOutputList[[i]], numberOFInputSets]; (* Get the expectedOutput to compare against the response *)
Print["expectedOutput = ", expectedOutput];
{actualResponse, lastState} = automatonResponse[aut, thisInput]; (* Returns response [outputs] from FSA and last state *)
Print["{actualResponse, lastState} = ", {actualResponse, lastState}];
If[MemberQ[FinalStates[aut], lastState], (If[Last[actualResponse] == expectedOutput[[1]], fit += fitnessValues[[i]], fit == 0.0]), fit += 0.0];
(* Add the fitness if the valid end in final state *)
}, {Length[abOutputList]};
{fit, If[fit = 1., True, False], If[fit = 1., True, False]} (* Return the derived fitness *)
];
```

Figure 4–16 Prediction quality function for BufferNot class.

As in most functional languages, parameters can be passed; in this case, they are used to control evolution through evaluation functions. The definitions of these parameters are included in the function headers in the respective function, including those needed for the evaluation functions. The critical piece for this class is the definition of the scheme class, which provides in the input and output values, as shown in figure 4–17.

```
EVChromosomeClass = {}; (* Initialize as a m = parents, l = children *)
desc = "Buffer Machine-1"; inputs = {{0}, {1}}; outputs = {{0}, {1}}; alphabet = {0, 1};
AddEVClass[inputs, outputs, BUFFERNOT, alphabet, Description -> desc, EvaluationFunction -> evalBufferNot,
  ResponseFunction -> automatonResponse, AvailableStrategy -> {BEST, FITPROP},
  G -> 3, M -> 50, L -> 50];
```

Figure 4–17 The definition of a class of machines (EVClass 1.), in this case a BufferMachine.

The argument that nature does not provide these supervised values, in this case (0, 1), is countered by the argument of cell evolution⁷⁴ (Kauffman, 1969; Kumar and Bentley, 2003).

A simple reading of Piaget on self-organization and self-stabilization (Piaget, 1980a, p24; Piaget, 1980a, p59; Furth, 1969, p209 among others), could assert that this simple machine and the functions that manufacture it follow adaptation. For instance, the process of assimilation builds the structure of the FSA: accommodation outputs the transition values and equilibration, which seeks a balance to the environment, is the evaluation function itself. This research thesis disagrees with this approach and posits that even in the event of a biologically inspired reward function (Schultz, 1997, p2; Cannon and Bseikri, 2004, p742; Sutton and Barto, 1998 and others) and a level of reinforcement that Piaget said occurred⁷⁵ (Piaget, 1954, p5; Piaget, 1964, p13) upon which the evaluation functions are based, that Piaget's work was more concerned with higher level functions. Had he been aware of FSA, he may have reconsidered his approach. In this research, the focus is on the development of higher-level functions as evidenced in his empirical observation of the relationship of language and logics (Furth, 1969, p233; Piaget, 1970a, 1970b, p13; Furth, 1969, p66).

The chromosome class itself is built dynamically from the environmental input. For instance to process a number line, the simulation will construct a prediction machine to read it. Essentially the system builds a machine to understand the external number line world using its *permanent object invariants*.

Argos Scheme: An Example of Equivalence

Argos schemes are HFSA that marshal environmental (external or internal) information across a network of prediction schemes.

Argos schemes are evolved from simpler prediction schemes. Each Prediction scheme has a prediction quality of 1, such that they can transform their inputs to the desired outputs as required.

⁷⁴ Each FSA can be considered as an abstraction of very simple biological cells. The initial state can be compared to the maternal factors for a zygote cell. The inputs are the inter-cellular communication proteins, hormones and environmental factors. The outputs are the physiological properties of the proteins produced by behavioral genes. Each state bit, is a binary abstraction of the concentration of one or more specific proteins, synthesized in the cell (Kauffman, 1969; Kumar and Bentley, 2003).









⁷⁵ Piaget makes it clear in his early work (Piaget, 1954, p5; Piaget, 1964, p13) that reinforcement is responsible for imitation.

EVClassID	Description	EVChromosomeID
1.	Buffer Machine-1	300 089
2.	NOT Machine-2	302 625
3.	Sensory Motor Move 1-3	312 713
4.	Basic Gale AND Machine-4	398 187
5.	Basic Gale OR Machine-5	433 882
6.	Basic Gale XOR Machine-6	450 962
7.	Basic Gale NAND Machine-7	513 953
8.	Basic Gale NOR Machine-8	551 349
9.	Basic Gale XNOR Machine-9	574 876

Figure 4-18 A simple depiction of chromosome (FSAs) used in the random construction of more complex as classes, such as Argos full adder schemes.

When these prediction schemes are randomly combined into a hierarchy, the inputs, outputs, channel capacities and the respective configurations of initial, input, output and final processing needs must be accounted for. Within a HFSA, a consumed FSA essentially becomes a “state” in this larger machine.

Table 4-3 Depiction of the channel capacity of FSAs. This channel capacity is used when automata are randomly combined.

EVClassID	Classic Symbol	Description	Channel Capacity	
			Input	Output
1.	 Buffer	Buffer Machine	1	1
2.	 NOT	Not Machine	1	1
3.	N/A	Sensory Motor Move 1	1	n/a
4.	 AND	AND Machine	2	1
5.	 OR	OR Machine	2	1
6.	 XOR	XOR Machine	2	1
7.	 NAND	NAND Machine	2	1
8.	 NOR	NOR Machine	2	1
9.	 XNOR	XNOR Machine	2	1

Of the possible configurations of HFSA, there is an optimized set of connections between states that can produce the required results. These configurations are 1) initial states, 2) input states, 3) output states and 4)

final states. To be consistent with FSA, there must be a series of transitions (alternatively called, synchronizations) between these states. A HFSA, merely marshal's information between FSAs and other HFSA's. The system randomly evolves these connections based on the input and output channel capacities of the contained machines.

One constraint imposed by the Argos approach, is the necessity to always produce a valid configuration. As such, when the HFSA is mutated, each configuration is valid, with the veracity of its utility and warranty, ascertained through evaluation. To execute the transformations that a contained FSA produces using synchronizations, requires that additional buffer machines be added as the final and output configurations of a HFSA.

Using this process, a 4-bit equivalence machine is classically depicted in figure 4-19.

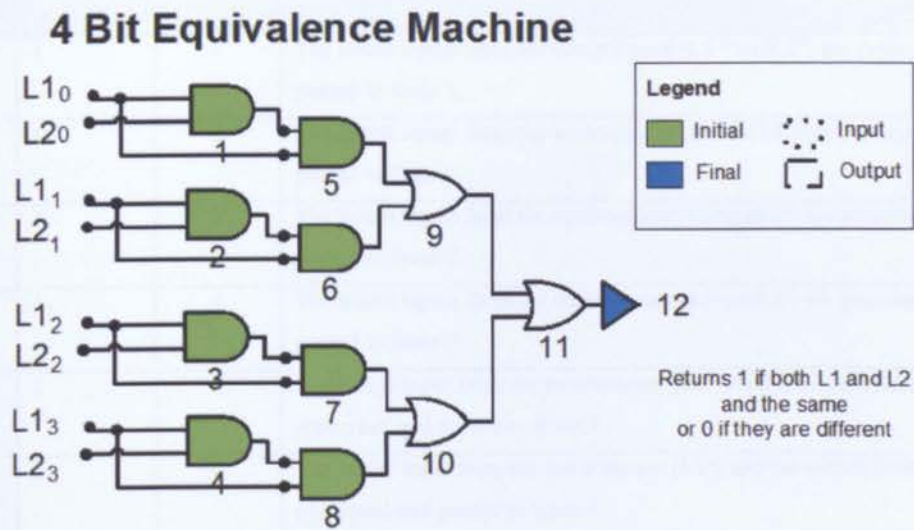


Figure 4-19 The depiction of an Argos Equivalence scheme using initial, final, input, output and ordinary states. This 4th example shows that the process of randomly selection “States” of machines, works.

Table 4.4, describes the states in the Argos equivalence scheme:

Table 4-4 A Table of states in the Argos equivalence scheme

State	EVClassID	Parent ID	Type	Capacity		Initial State	Input State	Output State	Final State
				Input	Output				
1	6.	772354	AND	2	1	Y	-	-	-
2	6.	772354	AND	2	1	Y	-	-	-
3	6.	772354	AND	2	1	Y	-	-	-
4	6.	772354	AND	2	1	Y	-	-	-
5	6.	772354	AND	2	1	Y	-	-	-
6	6.	772354	AND	2	1	Y	-	-	-
7	6.	772354	AND	2	1	Y	-	-	-

Chapter 4. Architecture and Design

State	EVClassID	Parent ID	Type	Capacity		Initial	Input	Output	Final
						State	State	State	State
8	6.	772354	AND	2	1	Y	-	-	-
9	7.	825480	OR	1	1	-	-	-	-
10	7.	825480	OR	1	1	-	-	-	-
11	7.	825480	OR	1	1	-	-	-	-
12	1.	300093	Buffer	1	1	-	-	-	Y

The processing for the Argos equivalence scheme is shown below in table 4-5.

Table 4-5 Equivalence – synchronizations (transitions)

From State	To State	Parallel Seq	Sequence Ordinal	Description
1	5	1	1	The initial inputs from the environment (L1 ⁰ and L2 ⁰) are processed and passed to State 5.
2	6	1	2	The initial inputs from the environment (L1 ¹ and L2 ¹) are processed and passed to State 6.
3	7	1	3	The initial inputs from the environment (L1 ² and L2 ²) are processed and passed to State 7.
4	8	1	4	The initial inputs from the environment (L1 ³ and L2 ³) are processed and passed to State 8.
1	5	2	5	The initial input from the environment (L1 ⁰) and the output from State 1 are processed and passed to State 9.
2	6	2	6	The initial input from the environment (L1 ¹) and the output from State 2 are processed and passed to State 9.
3	7	2	7	The initial input from the environment (L1 ²) and the output from State 3 are processed and passed to State 10
4	8	2	8	The initial input from the environment (L1 ³) and the output from State 4 are processed and passed to State 10
9	11	3	9	The output from state 5 and 6 are processed and passed to state 11.
10	11	3	10	The output from state 7 and 8 are processed and passed to state 11.
11	12	4	11	The outputs from state 9 and 10 are processed and passed to state 12

The passing of values from the environment to the HFSA, especially when inputs are passed to more than one machine is a problem of this approach. The assignment occurs randomly, with valid machines being capable of processing the inputs into the desired outputs, this causes an overhead in evolution, but it works.

The execution of these synchronizations is the processing of inputs provided by the environment and the production of the following example set of values (table 4.6).

Piaget Equivalence

Table 4-6 Equivalence Table – Example Outputs. These outputs, provided by the environment are used in the evaluation of this class of machines.

L1 ⁰	L1 ¹	L1 ²	L1 ³	L2 ⁰	L2 ¹	L2 ²	L2 ³	D ⁰
0	0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	1	1
1	1	0	1	0	1	0	0	0
1	1	1	1	1	1	1	0	0

There are 16 tuples of a 4-bit machine. For every set of L1 inputs, there are 16 possible combinations of L2 tuples, only one of which is equivalent to the L1 inputs. It is suspected that when a child is developing the notion of equivalence, the reward process builds an equivalence table that is used to construct the HFSA. Over time, it gets constructed and reconstructed based on the changes in values. Sometimes it builds valid machines from the start, other times these machines are not built correctly at all. This though is consistent with how children develop number-sense: sometimes it takes them a long time, with lots of small improvements, and then dramatic changes.

Piagetian Scheme: An Example of Equivalence

A Piagetian scheme consists of a set of Argos schemes that can adapt to the environment. For example, the equivalence relation is critical to the development of logical thought and occurs throughout the developmental stages (Copeland, 1974, p84). Often called “one to one correspondence,” it is seen to develop early. To be able to determine equivalence using a Piagetian model is not as simple as using an operator such as “=” or “==,” the operator itself has to be constructed by the system before it can be used, otherwise the system falls foul of Fodor’s LPI argument. The bead problem is evidence of the issues associated with equivalence (Copeland, 1974, p85–87 and p91–92). In this research, equivalence is the appearance of a machine that compares the properties of one machine against another in binary using an AND machine (a FSA) that is wrapped up into an Argos scheme. The internalization is an action of accommodation, which produces the “binary value.” This is to be compared, with different actions for the different combinations penState (penUp, penDown), direction (left, right), stop as well as length. When the values are the same, it returns a 1; when they are different, it returns a zero. In more complex arrangements, the predictive model tests the states, inputs and outputs of various machines to see if they are equivalent. In essence, the equivalence relation determines the truth of validity of a set of machines.

Equivalence is used by other relations, such as ordering and seriation. It is anticipated that equivalence is part of a developmental trend of increasing discretization (differentiation) of the environment. This would suggest that a child is constantly comparing and contrasting elements in the environment, even from the earliest stages. The diagram below is of a hypothetical equivalence machine (figure 4-20).

Piaget Equivalence machine using HFSA

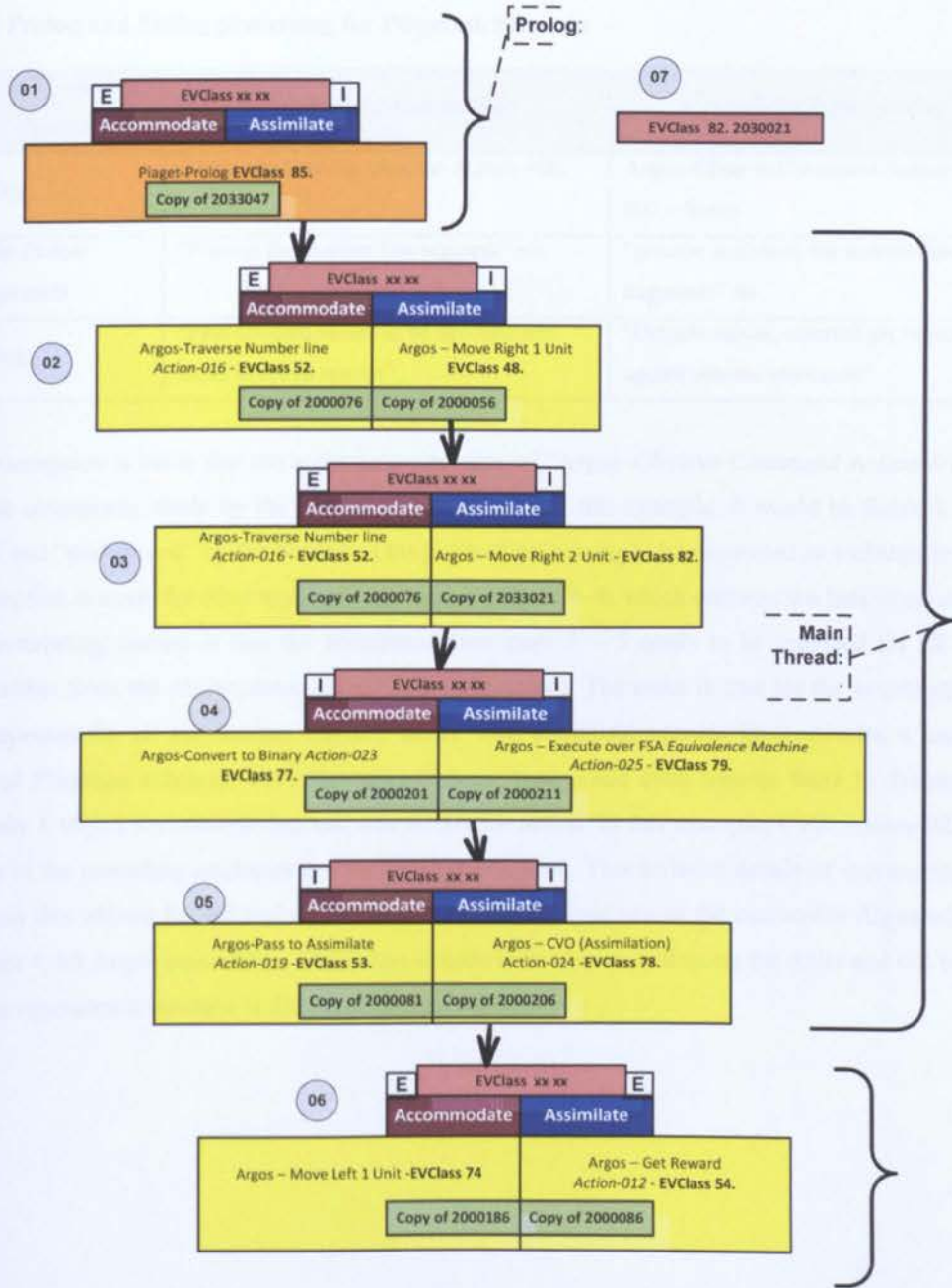


Figure 4–20 Piagetian Equivalence Scheme. In this image, the twin processes of assimilation and accommodation are clearly seen as separate action. This process follows Piaget’s knowing circle and makes use of chromosomes as Argos and Piagetian machines to process the inputs from, and act on the environment. The processing is separated into Prolog, Main Thread and Epilog processing.

The second principle at work here is that for the system to actually work the “states” and synchronizations within the Piagetian scheme have to be tailored, in that they have prolog and epilog

Chapter 4. Architecture and Design

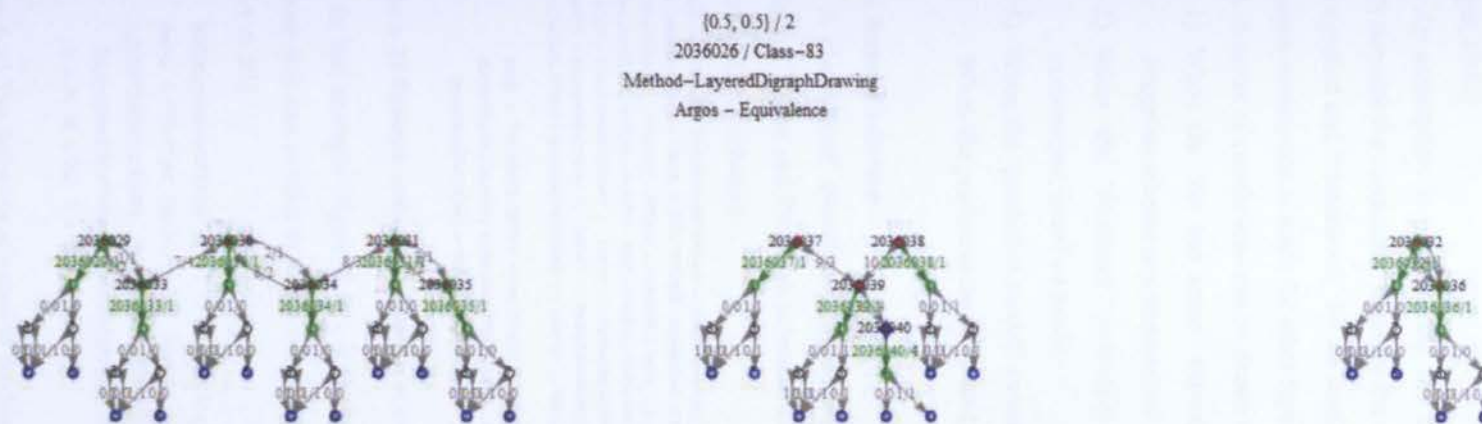
processing. A depiction of an executable Piagetian scheme that can count is shown in the evaluation (§ 5.2.5.1). For operative schemes and if one uses the example of counting this consists of the details in table 4.7.

Table 4–7 Prolog and Epilog processing for Piagetian Schemes

Step	Accommodation Process	Accommodation Action	Assimilation Action
1)	Prolog	Argos–Do Nothing, Student Action–000.	Argos–Observe Command Action–015 e.g., 002 – Solve.
2)	Main Thread (Repeated)	“Process the number line segment” etc.	“process and count the number line Segments” etc.
3)	Epilog	“Pass counted values to be decoded and move decoded results”.	“Decode values, external get reward and update scheme relevance”.

An assumption is made that the assimilation process of “Argos–Observe Command Action–015,” which decodes the commands, made by the teacher to the student (in this example, it would be Solve $1 + 2 = ?$) is recognized and “understood” by the Student. This piece of processing is implemented as a change in state. The same assumption is made for other types of commands (§ figure 4–8, which contains the lists of commands).

The interesting feature is that the accommodation steps 2 – 5 needs to be repeated for all the inputs provided (either from the environment, or internally generated). The same is true for the assimilation step; it must be repeated for all the actions that are taken. This simplistic process thus provides a basis for the execution of Piagetian schemes. Each Piagetian Scheme (one could even rename these to dynamic HFSA) contains only 1 HFSA for construction and one HFSA for action. In this example, CVB Action–023 converts the outputs of the preceding machines to a binary representation. This includes details of movement, penState, direction and thus allows for left and right. Further, the main thread reuses the executable Argos scheme. It is assumed that 4–bit Argos equivalence HFSA has already been constructed using the AND and OR basic gates. A Piagetian equivalence machine is displayed in figure 4–21.



Copied from EVChromosomeID	EVChromosomeID	EVClassID	Description A Root Occurs twice firstly in parent then secondly to demarcate components.	Initial State	Input State	Output State	Final State
0	2036026	83.	Root-Argos - Equivalence	na	na	na	na
772354	2036029	6.	Basic Gale AND Machine-6	✓	-	-	-
772354	2036030	6.	Basic Gale AND Machine-6	✓	-	-	-
772354	2036031	6.	Basic Gale AND Machine-6	✓	-	-	-
772354	2036032	6.	Basic Gale AND Machine-6	✓	-	-	-
772354	2036033	6.	Basic Gale AND Machine-6	✓	-	-	-
772354	2036034	6.	Basic Gale AND Machine-6	✓	-	-	-
772354	2036035	6.	Basic Gale AND Machine-6	✓	-	-	-
772354	2036036	6.	Basic Gale AND Machine-6	✓	-	-	-
825480	2036037	7.	Basic Gale OR Machine-7	-	-	-	-
825480	2036038	7.	Basic Gale OR Machine-7	-	-	-	-
825480	2036039	7.	Basic Gale OR Machine-7	-	-	-	-
300093	2036040	1.	Buffer Machine-1	-	-	-	✓

Figure 4-21 A depiction and composition of a binary equivalence machine that compares the binary values of two machines. From the composition, one can imagine that its simple construction of only and, or and buffer machines, would appear early, which is part explains Piaget's observation of seriation and order, before more complex structures such as counting.

Chapter 4. Architecture and Design

The interesting feature of this equivalence machine is the representation with three different branches. The reason for this is simple, the root node connections to the separate branches is not being shown. This research suggests that the notion of permanent (or “stable”) mathematical equivalence is a notion rooted in perception, trial and error processing which is finally solidified as logical and reversible mental operations (Piaget, 1952).

An assumption is made during the assimilation process of “Argos–Observe Command Action–015,” which decodes the commands, made by the teacher to the student (in this example, it would be Solve $1 + 2 = ?$) is recognized and “understood” by the student. This piece of processing is implemented as a change in state. The same assumption is made for other types of commands (§ 4.2.1.1) such as main control loop, act/sense and learn. A series of conclusions can be drawn about processing by the student:

- 1) When the “act and sense” executive is processed, a search is made for Piagetian Schemes. The Piagetian scheme accommodation step is executed continuously until it determines when to stop.
- 2) When the “reinforce” executive is processed, Piagetian, Argos and Prediction schemes are constructed, based on needs.
- 3) When the “predictive model” executive is processed, only Piagetian schemes are executed in memory. When the predictive model it used, the actions are internalized on a “virtual” number line.

Class Based Evolution

A class–based evolutionary mechanism was recreated from earlier work (Jacob, 2001). This allows prediction, Argos and Piagetian schemes to be evolved. In the figure below, is a definition of a typical class (figure 4–22) of schemes:

```
Options[AddEVClass] = {EvaluationFunction → evalAutomaton, ResponseFunction → automatonResponse, Description → ToString[Datelist[]], M → 50, L → 50,
  AvailableStrategy → {BEST, ELITE, FITPROP, RANDBASED, RANDOM, TOURNAMENT}, G → 3, StopAtPerfection → True, Copies → 50, Vanilla → 10};
AddEVClass[inputs_, outputs_, EVType_, alphabet_, opts_] :=
Module[{newClass, evType, evalFct, desc, respFct, EVInputFlag, EVOutputFlag, availableStrategy, g, m, l, stopAtPerfection, copies, vanilla},
  evalFct = EvaluationFunction /. {opts} /. Options[AddEVClass];
  respFct = ResponseFunction /. {opts} /. Options[AddEVClass];
  availableStrategy = AvailableStrategy /. {opts} /. Options[AddEVClass];

  desc = "Sensory Motor Move Right 1 Unit-5"; inputs = {1, 5, 6, 10403, 4}; outputs = {}; alphabet = {1, 4, 5, 6, 10403};
  AddEVClass[inputs, outputs, PREDICTION, alphabet, Description → desc, EvaluationFunction → evalAutomaton,
    ResponseFunction → automatonResponse, AvailableStrategy → {BEST, FITPROP}, G → 20, M → 100, L → 100];
```

Figure 4–22 Example code of the creation of a class of chromosomes used in class–based evolution.

In this example (figure 4–22), a prediction scheme is created, the aim of which is to randomly create a machine that can predict an input sequence. The example below is the definition of a class of chromosomes (figure 4–23):

```
EVChromosomeClass = {}; (* Initialize as a m = parents, l = children *)
desc = "Buffer Machine-1"; inputs = {{0}, {1}}; outputs = {{0}, {1}}; alphabet = {0, 1};
AddEVClass[inputs, outputs, BUFFERNOT, alphabet, Description → desc, EvaluationFunction → evalBufferNot,
  ResponseFunction → automatonResponse, AvailableStrategy → {BEST, FITPROP},
  G → 3, M → 50, L → 50];
```

Figure 4–23 The definition of a class of machines (EVClass 1.), in this case a Buffer Machine

Chapter 4. Architecture and Design

In this particular case, it is a class of a buffer machine.

Act, Sense and Reinforce: Producing Schemes and Using Them

The approach taken in this research is to use class-based evolution as a process for learning. In this section, the terms used in the thesis to the algorithms employed in the process of learning are related. A description follows about how they are utilized in the design by providing sample code. A detailed definition each of the algorithms used in the Piagetian “act and sense” and “reinforce” processes (§ 2.3.6) is provided.

The Process of Adaption

This section examines how finite state automata (as Prediction schemes) are constructed using the processes of assimilation and accommodation, using a set of algorithms which allow the system to adapt to changing environmental inputs. The basic structure of these algorithms is an extension of the work by Christian Jacob (Jacob, 2001, p327), but in the present approach, when the system determines that learning needs to occur, the system switches into the “reinforce” state (§ 4.2.2.2.). Once a scheme has been constructed, that has learnt how to process the environmental inputs; the system simply switches back to “act and sense.” These algorithms do not consider the executive process as described by Pascual-Leone (§ 2.1.6 and 2.8.1), and as such only represent an intermediate step in a full solution. Using an event process chain diagram, the process of adaptation is described in figure 4-24.

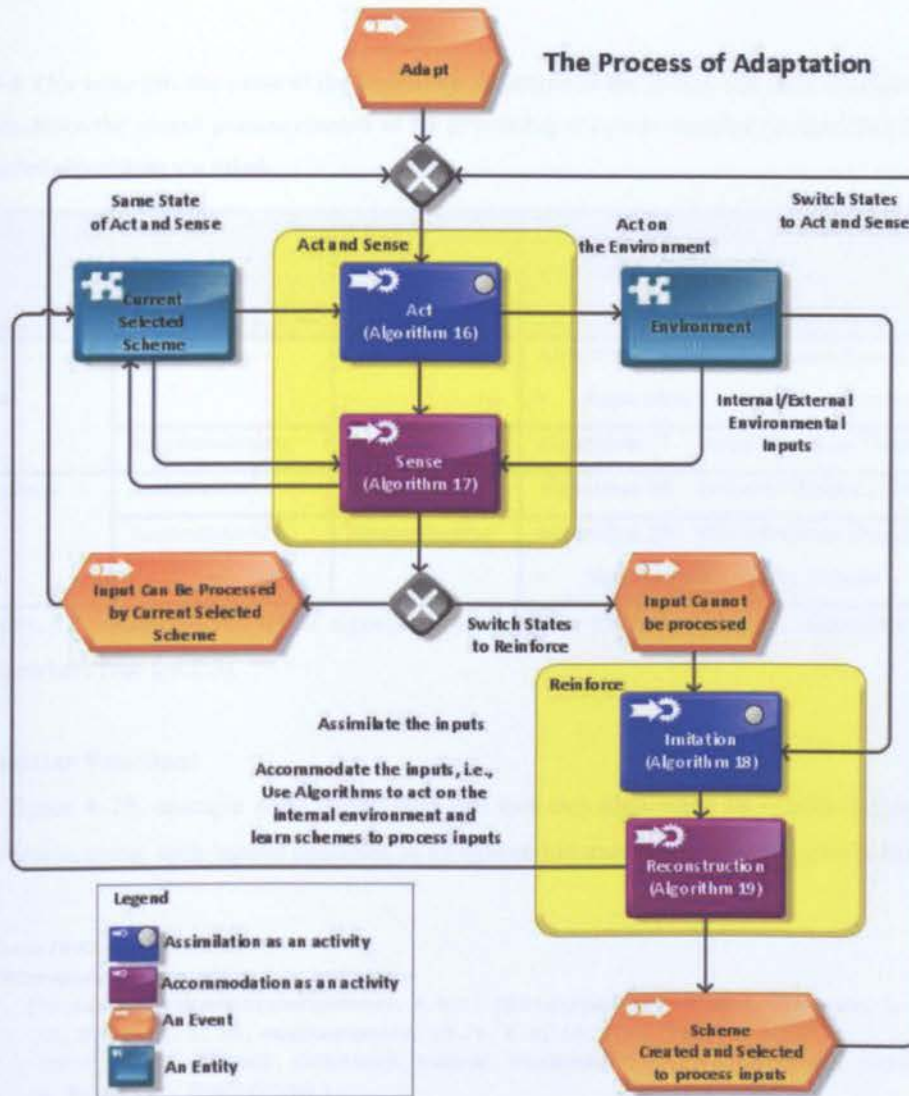


Figure 4–24 The process of adaptation as described using an event process chain diagram, shows the switching from act and sense to reinforce when the inputs from the environment cannot be handled by the sense activity. When the schemes have been constructed to handle the input, the system switches back to using its repertoire of schemes. This two state processing of assimilation and accommodation, is the key to understanding the Piagetian approach used in this research.

In the event process chain diagram (figure 4–24), the Piagetian processes of assimilation and accommodation (§ 2.3.6) are related to the algorithmic approach of this research. Table 4–8 lists those algorithms by the type of Piagetian processing they perform.

Table 4–8 This table lists the name of the algorithms employed in the system and their relationship to the Piagetian processes. Since the overall process consists of the processing of a state machine (as described in figure 4–24), only the included algorithms are listed.

#	State	Abstract Piagetian Process	Piagetian Term	Primary Algorithm Name and list of included algorithms.
2	Act and Sense	Assimilation	Act	Algorithm 16 – Act (Piagetian Recognition) • Algorithm 2 –Automaton Response
3		Accommodation	Sense	Algorithm 17 – Sense (Piagetian Perception)
2	Reinforce	Assimilation	Imitation	Algorithm 18 – Imitation (Piagetian Imitation)
3		Accommodation	Reconstruction	Algorithm 19 – Reconstruction (Piagetian Reconstruction) • Algorithm 5 – Evolve Classes

Table 4.8 introduces the set of algorithms used by the system, with each algorithm described in more detail elsewhere (see § 4.2.3).

The Piagetian Functions

In figure 4–25, example code is provided that executes **algorithm 18** – imitation and **algorithm 19** – reconstruction, using their named functions as an executable trace showing the output of the mutation learning process.

```

Clear[EVChromosomeClass]:
EVChromosomeClass = {
  {"Class-1", {{EVMutationOperator, 0.95}, {EVCopyOperator, 0.05}}, True, {1, 1, 1, 1}, 1., {0, 1},
  10, PLUS, 10, 1, 20, evalAutomaton, 10, 5, 0.5, 10, 0.05, "Class-1-001",
  {BEST, ELITE, FITPROP, RANKBASED, RANDOM, TOURNAMENT}, {}, {}, "Standard Input Prediction Machine",
  0, False, {}, PREDICTION});

getEVClassList[]
EVChromosomeClass[All, 5] (* Field 5 - Get all the EVChromosomeClassIDs *)

{1.}
{1.}

popFSAClass1Gen1 = EVEvolveClasses[pop[]]:

AddTransition::AlreadyUsed: from state and input(alphabet) is already a transition
AddTransition::AlreadyUsed: from state and input(alphabet) is already a transition
AddTransition::AlreadyUsed: from state and input(alphabet) is already a transition
General::stop: Further output of AddTransition::AlreadyUsed will be suppressed during this calculation. =>
Prediction Quality: Class-1.genCount-1-genStep-0 {0.5, 0.25, 0.25, 0.25}

```

Chapter 4. Architecture and Design

```

DeleteFinalState::InvalidState: The Final State to delete, does not exist in the automaton
General::stop: Further output of DeleteFinalState::InvalidState will be suppressed during this calculation. >
ChangeTransitionOutput::DuplicateTransition: The new transition already exists
DeleteState::InitialState: Cannot delete initial state
DeleteState::InitialState: Cannot delete initial state

Prediction Quality: Class-1.genCount-3-genStep-1 {1., 1., 0.25, 0.25, 0.}

pop[chromo[q[{1.75, 1.}], p[- Automaton -], s[{300519, 1., Class-1, 1., 0., {300407.}}], h[undef]],
      chromo[q[{0.0625, 0.}], p[- Automaton -], s[{300548, 1., Class-1, 1., 1., {300410., 300495}}], h[undef]],
      chromo[q[{0.28125, 0.25}], p[- Automaton -], s[{300539, 1., Class-1, 2., 0., {300402., 300472}}], h[undef]],
      chromo[q[{1.75, 1.}], p[- Automaton -], s[{300519, 1., Class-1, 1., 0., {300407.}}], h[undef]],
      chromo[q[{0.28125, 0.25}], p[- Automaton -], s[{300531, 1., Class-1, 2., 0., {300402., 300472}}], h[undef]]]

plotEVClasses[popFSAClass1Gen2] (* Analyze it *)

```

Figure 4–25 This figure provides an executable trace of algorithm 18 (to imitate through building a chromosome class), and algorithm 19 (to reconstruct, through building schemes that can predict the given input sequence, as finite state automata for the chromosome class). In the example code provided, the system passes the inputs from the environment that it cannot process, and creates a scheme that can process this information. Initially the chromosome class and then the chromosomes themselves are constructed. In the example shown, warning messages from the system show the attempted mutation that have taken place. In the final step, chromosome 300519 is generated with a prediction quality of 1.0, having reached perfection, and a fitness of 1.75.

In this example of figure 4–25, the processing for the algorithms is shown to produce valid output, with the resultant machines showing a prediction quality of 1 and a fitness of 1.75.

4.2.3 Algorithms

Several global structures are used by the system, these are defined in table 4–9:

Table 4–9 Global structures used in the research.

#	Name	Description
1	<i>EVChromosomeClass</i>	<i>EVChromosomeClass</i> holds the definitions of all the classes of chromosomes used in the execution of the system. Each class of chromosome may or may not have a set of chromosomes in <i>EVPopulation</i> .
2	<i>EVChromosomeClassID</i>	<i>EVChromosomeClassID</i> holds the ID of the last <i>EVChromosomeClass</i> created in <i>EVChromosomeClass</i> .
3	<i>EVPopulation</i>	<i>EVPopulation</i> holds the population of all the chromosomes generated in the executing system. The population is contained within a <i>Pop[]</i> structure. Each chromosome in the population is constructed within a <i>chromo[]</i> structure. Each chromosome holds the ID of the <i>EVChromosomeClass</i> , from which it was created.
4	<i>EVChromosomeID</i>	Each chromosome in <i>EVPopulation</i> has a unique chromosome ID. The <i>EVChromosomeID</i> holds the ID of the last chromosome created.
5	<i>EVClassOperator</i>	<i>EVClassOperator</i> contains a set of functions that define all the possible mutations for a finite state automata (see § 4.2.3.4 Algorithm 4).

What follows is a definition of each of the key algorithms used in this research.

4.2.3.1 Algorithm 1 – Prediction Quality

We describe the prediction quality algorithm for evaluating mutated finite state automata against a benchmark. Given a list of finite state automata, and a definition of the output requirements of the finite state automata, this algorithm determines the prediction quality of each finite state automaton in turn. Since there can be many types of prediction schemes, the input values and expected output values for each type is stored in a chromosome class, and the values are passed to this algorithm as a list of values.

Algorithm 1 The prediction quality algorithm for evaluating mutated finite state automata.

Require: Given a *finite-state automata*, the environmental *inputs* and the *initial-position*, then:

1. Compute the *response* list and *last-state* for the automaton by applying the environmental *inputs* to the given *finite-state-automata* using **Algorithm 2**.
2. Set the *last-prediction* to be the last element in the *response* list.
3. Set the *to-prediction* list to be the environmental *input* list.
4. Set *hits* to be *response – prediction*.
5. Set the *prediction-quality* to be the numeric count of 0 hits divided by the number of items in the environmental *inputs*. If the count of 0 hits is zero, then *prediction-quality* = 0.
6. **If** *last-state* is a **final state** for the *finite-state automata* and the *prediction-quality* is 1, **then**
 - a. Set *prediction-quality* = 1.
7. **If** *last-state* is **final state** for the *finite-state automata*, **then**
 - a. Set *final-state* = TRUE.
8. **Else**
 - a. Set *final-state* = FALSE.
9. **If** *last-prediction* is equivalent to first element in the environmental *inputs* **then**
 - a. Set *last-eq-first* = TRUE.
10. **Else**
 - a. Set *last-eq-first* = FALSE.
11. Return { *prediction-quality*, *last-eq-first*, *final-state* }

As an example, if the environmental *inputs* are {1,0,1,1,1,0,0,1,1,1,0,1} and the *initial-position* is 1 and the output produced by the automata is {1,0,1,1,1,0,0,1,1,1,0,1} then it will {1., TRUE, TRUE} because the *last-prediction* is the first element in the environmental *inputs* i.e., it is cyclically correct. It will also return TRUE because the *last-state* is a **final state** in the automata. This process follows the work of Christian Jacob (Jacob, 2001, p330). Two variations of this algorithm are generated, one for basic gate finite state automata (which receive an input list and an output list of binary values to generate 2 bit binary processing units include OR, AND etc.) and one for a buffer not machine (which receive only single bit values and allow for the creation of Buffer or Not machines).

Chapter 4. Architecture and Design

The definition of a chromosomes prediction quality (**algorithm 1**) is augmented by the definition of fitness (**algorithm 3**) which determines how well the chromosome is adapted to the environment.

4.2.3.2 Algorithm 2 – Automaton Response

The automaton response algorithm used in this research for determining the response of a given finite state automata to a given set of inputs is described. It is used by both the prediction quality and fitness algorithms (**algorithm 1** and **algorithm 2** respectively), to determine if the mutated automata can be constructed to mirror the given input values. The automaton response is also used to execute the automata against the internal or external environment. From a given input, the automaton will produce a given output, since finite state automata are inherently deterministic.

Algorithm 2 The automaton response algorithm for executing a given automata against a given set of inputs.

Require: Given a *finite state automata* and an list of environmental *inputs*, then:

1. Pass each of the environment *inputs* in turn to the *finite state automata* and collect the *output-values* from the transition.
2. **If** the automata is at the **final state** then
 - a. return {*output values*, TRUE}
3. **If** the automata is not at the **final state** then
 - a. return {*output values*, FALSE}
4. **Else**
 - a. return {*output values*, FALSE}

As an example, if a finite state automata has been mutated to 8 states, whose aim is to reproduce the following input sequence {1,0,1,1,1,0,0,1,1,1,0,1}, and it has a set of transitions defined that mirror these input values to the output values, and state 1 is the initial state and the final state. Then the automaton response algorithm will return {True, 1,0,1,1,1,0,0,1,1,1,0,1}. This process follows the work of Christian Jacob (Jacob, 2001, p328). Several variations of this algorithm are generated, one for automata that process binary information as buffer and not machines, one for basic gate machines (to support the generation of AND, OR type circuit components) and one for Prediction Schemes which process a multitude of inputs value ranges.

4.2.3.3 Algorithm 3 – Evaluate Automaton Fitness

Described is the fitness algorithm for determining how adapted each finite state automata; it is based on its mutations. Given a list of finite state automata, and a definition of the output requirements of the finite state automata, this algorithm determines the fitness of each finite state automaton in turn. Since there can be many types of prediction schemes, the input values and expected output values for each type is stored in a chromosome class, and the values are passed to this algorithm as a list of values.

Algorithm 3 The fitness algorithm for evaluating the adaptation of a finite state automata.

Require: Given a *finite state automata* and a set of environmental *inputs*, then:

1. Compute the *prediction-quality* of the given *finite state automata* using **algorithm 1** (**algorithm 1** will return $\{prediction-quality, last-eq-first, final-state\}$).
2. Set the *fitness* to be the *prediction-quality*.
3. If $last-eq-first == TRUE$ i.e., the first input state is the same as the last input state, **then**
 - a. Add a scalar (0.25) to *fitness*.
4. If $final-state == TRUE$ i.e., the automata is at its final state. This is preferred approach to building automata that mirror the external environment, **then**
 - a. Add a scalar (0.25) to *fitness*.
5. Add a scalar (0.5 / number of usable states) to *fitness*. This effectively reduces the development of a large number of states in an automata.
6. Return $\{fitness, prediction-quality\}$.

Example: If the environmental *inputs* are $\{1,0,1,1,1,0,0,1,1,0,1\}$ and the *initial-position* is 1 and the output produced by the automata is $\{1,0,1,1,1,0,0,1,1,0,1\}$ and the *last-prediction* is the first element in the environmental *inputs* i.e., it is cyclically correct, then the returned *prediction-quality* will be 1.0. If the automata is at its final state, and if it has no extraneous states then the fitness returned will be 1.54 . This process follows the work of Christian Jacob (Jacob, 2001, p329). The definition of fitness uses, and modifies prediction quality (**algorithm 1**). As such, the measure of fitness will always be a numerically greater value, than prediction quality.

Two variations of this algorithm are generated, one for evaluating the fitness of basic gates (AND, OR etc) and one for evaluating buffer not machines.

4.2.3.4 Algorithm 4 – Operators

A description of the operator algorithm used in this research, returns a number of mutated or copied children into a population of chromosomes for the given class. The mutation operators include the full rang of possible point changes for a finite state automata⁷⁶.

Algorithm 4 The Operators algorithm mutates or copies the passed parent population depending on the passed parameters.

⁷⁶ A set of mutation operators are supported, namely: add final state, add state, add transition, change initial state, change final state, change transition input signal, change transition output symbol, change transition source, change transition target, delete final state, delete state and delete transition.

Require: Given a population of chromosomes for a class, the *chromosome class ID*, an optional number of *copies* from their *EVChromosomeClass*, an optional *stopAtPerfection* flag and the available mutation operators in *EVClassOperator*, then:

1. Set *parent*, to be passed set of chromosomes.
 2. **If** *stopAtPerfection* and one of the chromosomes in the *parent* population has a prediction quality ≥ 1 . **then**
 - a. Return *{parent}*, as the population of chromosomes.
 3. Create a table of mutation operators for the class (*class-operators*). Each mutation operator is a function that can be called to mutate a feature of a finite state automata.
 4. For each chromosome in *parent*, then for each set of *copies*:
 - a. Randomly select one operator from *class-operators* as *selected-operator*.
 - b. Get the *prediction quality* of the chromosome.
 - c. **If** the *prediction quality* ≥ 1 ., **then**
 - i. Set the *EVClassOperator* to be COPY.
 - d. **If** the *EVClassOperator* is COPY, **then**
 - i. Copy the chromosome.
 - e. **Else**
 - i. Mutate this chromosome using the *selected-operator*, which will provide a random mutation on the selected chromosome (*newChromo*).
 - f. Append the new chromosome (*newChromo*) to the *children* population.
 5. Return *{children}*
-

As an example, if the number of generations is set to 5, and the selection strategy is COMMA (to use only children), and the number of parents is set to 50, the number of children is set to 50, the number of copies is set to 5, and the environmental input values are {1,0,1,1,1,0,0,1,1,0,1}. Then the system will generate an initial set of parents by using **algorithm 8** (create chromosomes), then it will add the strategy parameters to the chromosome *s[]* structure, then evaluate the fitness and prediction quality of the chromosomes by using **algorithm 3** (Evaluate Automaton), then **algorithm 4** (Operators) is used to evolve 5 copies of each of the chromosomes in the population. This process is a significant departure from the work of Christian Jacob (Jacob, 2001), and enables the system to work more autonomously, by cycling through various evolutions to enable the system to adapt to its environment.

4.2.3.5 Algorithm 5 – Evolve Classes

A description of the algorithm for evolving classes of chromosomes and selectively, the chromosomes within this class follows. This is a modification of the process described by Christian Jacob, for evolving finite state automata that can successfully predict the next input sequence (Jacob, 2001, p288). Each class of automata has an entry in the global structure *EVChromosomeClass*, which holds various properties of the class.

Chapter 4. Architecture and Design

These properties include: the name of the evaluation function that computes prediction quality (**algorithm 1**), fitness (**algorithm 3**); the response function of the automata (**algorithm 2**); the search strategy to use in the evolutionary process (§ 4.2.3.7 on strategy); the population of parents to keep in each generation; the population of children to propagate through mutation; the number of copies of the mutation operators to use per evolutionary step; various flags to determine when to stop evolving for instance to stop evolving at perfection; the number of vanilla entries to add at each generation; and the value of inputs and the expected outputs of the automata. The evolutionary classes themselves are never removed from the system, thus the system has a record of every type of class generated. There is no upper bound limit to the number of classes that can be evolved; however, the only evolutionary reward is adaptability, and there may be some value in having this evolutionary record. Specifically, **algorithm 5** (Evolve Classes), is a 12 step process, that drives the performance of the whole system by adding newly mutated chromosomes to the population of chromosomes (in the structure `pop []`) and removes selected chromosomes from this population based on the properties of the class of chromosomes. This is a general-purpose algorithm which supports evolution through mutation from an initial population of two-bit chromosomes through to mature populations, which have been evolved through successive generations. The information in the chromosome class may prevent the further mutation, of the chromosomes themselves.

Algorithm 5 The evolve classes algorithm for evolving classes of chromosomes using the class based definitions for these chromosomes held in *EVChromosomeClass*.

Require: Given the population of chromosomes *popFSA*, the optional class ID of chromosomes to evolve through mutation, the flag *save-top-chromosome*, the population of all chromosomes stored in the *parents* structure, then:

1. For every class passed:
 - a. Get all the evolution strategy properties for the chromosomes for this class, from the `S []` structure.
 - b. Index all the strategy parameters for each chromosome, so one then has a position/in matrix from which to process.
 - c. Get and store all the chromosome IDs for this class in *thisClassIDList*.
 - d. Get all the IDs of the chromosome for this class and store in *statclassChromosomesList*.
 - e. Save the IDs of the chromosomes, since some will need to be deleted, in *deleteIDList*.
 - f. Get the positions of all chromosomes for this class in *parents*, and store in *thisClassPosList*, so one then has a position/in matrix.
 - g. Build the index of all chromosomes to be deleted in *thisClassPosDeleteList*, based on their individual strategy values.
 - h. Copy all the chromosomes for this class into *popClass* and then call **algorithm 6**

(Evolve Chromosome For Class) with *popClass* to evolve chromosomes for this class.

- i. **If *save-top-chromosome*, then**
 - i. Create a population of chromosomes using only those chromosomes with the highest fitness and store in *popClass*.
 - j. Get the positions in the new population (*popClass*) of all the chromosomes that were selected for deletion.
 - k. Delete the selected chromosomes from the original population, *parents*.
 - l. Join the new evolved class population *popClass* with *parents*.
2. Return {*parents*}

As an example, if the initial population of chromosomes is null, and a class definition has been created which can draw a value on a number-line, then **algorithm 5** (Evolve Classes) will generate a population of chromosomes that have been mutated from the simple 2-bit machine that can perform this function. The resultant population will be stored into a global structure, which is used through the execution of the system. This process follows the work of Christian Jacob (Jacob, 2001, p246), with the addition of new strategy parameters to monitor and randomly control, the mutation rates of individual chromosomes.

4.2.3.6 Algorithm 6 – Evolve Chromosome for Class

A description of the evolve chromosome for class algorithm, which mutates chromosomes in the passed population (*EVParents*) for the given chromosome class ID using the chromosome class parameters follows. These parameters dictate how the operators are randomly chosen to mutate each of the chromosomes.

Algorithm 6 The evolve chromosome for class algorithm mutates a population of chromosomes using the parameters specified in the class.

Require: Given a population of chromosomes (*parents*) and a *Class ID*, and the parameters in *EVChromosomeClass*, then:

1. Get the counts of *tournament-competitors*, *bestCount*, *input-values*, *stopAtPerfection* from the *EVChromosomeClass* using the *ClassID*.
2. Increment and store the *generation-count* for this class. The *generation-count* is used in the evolutionary analysis.
3. **If** there are no chromosomes in the passed population (*parents*), **then**:
 - a. Use **algorithm 8** (Create Chromosomes) to create the initial population of chromosomes for this class.
 - b. For each chromosome, use **algorithm 3** (Evaluate Automaton) to determine the prediction quality and fitness of each chromosome. A different evaluation function exists for each class of scheme (Prediction, Argos or Piagetian).
 - c. Store the population in *evaluatedInitialParents*.
4. **Else**
 - a. Store the current population (*parents*) in *evaluatedInitialParents*.
5. **If** *stopAtPerfection* and a chromosome has a perfect prediction quality of ≥ 1.0 , **then**
 - a. Return {*evaluatedInitialParents*}, since there is no need for further mutation.
6. Initialize the *children* chromosome population to be pop [].
7. For every *g* generation and *i* independent runs defined in the class:
 - a. Increment the count of generations *gen* by 1.
 - b. Set *parents* to be the *initialParents*, and then after each generation, assign *parents* to be the pruned population in *parents*.
 - c. **If** the class requires the creation of vanilla children, rather than mutating the chromosomes from the population, **then**:
 - i. Use **algorithm 8** (Create Chromosomes), and generate an initial population of chromosomes into *initialParents*.
 - ii. Add the strategy parameters to each chromosome in the population.
 - iii. Use **algorithm 3** (Evaluate Automaton) and evaluate the prediction quality and fitness of each chromosome in the population and store in *evaluatedVanillaParents*. This process is varied based on the type of scheme (Prediction, Argos or Piagetian) being mutated.
 - iv. Join the *evaluatedVanillaparents* to the *parent's* population of chromosomes.
 - d. Set *children* to be the result of the mutation of the chromosomes in *parents* using

algorithm 4 (Operators). This is fundamentally, the main evolutionary process for the system.

- e. Use **algorithm 3** (Evaluate Automaton) and evaluate the fitness of each chromosome in the *children* population and store in *children*.
 - f. Based on the class selection pool mechanism either a PLUS strategy, or a COMMA strategy, set *selPool*) to be either the *parents* plus the *children*, or children only.
 - g. Use **algorithm 7** (Evolution Strategy) to choose the selection strategy to use from the class of chromosome.
 - h. Execute the selected strategy to return the set of chromosomes into the *parents* structure, one of:
 - i. **algorithm 9** (Best Selection);
 - ii. **algorithm 10** (Random Selection);
 - iii. **algorithm 11** (Fitness Proportionate Selection);
 - iv. **algorithm 12** (Rank based Selection);
 - v. **algorithm 13** (Elite Selection) or
 - vi. **algorithm 14** (Tournament Selection).
8. Return *{parents}*, which contains the evolved population of chromosomes.

Example: If the initial population of chromosomes (*EVPopulation*) is null, and a class definition (in *EVChromosomeClass*) has been created which can draw a value on a number-line, then **algorithm 5** (Evolve Classes) will process each of the classes and will call **algorithm 6** (Evolve Chromosome For Class) to evolve the chromosomes for this class. This will generate a population of chromosomes that have been mutated from the simple 2-bit machine that can perform this function. The resultant population will be stored into a global structure (*EVPopulation*), which is used through the execution of the system.

This process follows the work of Christian Jacob (Jacob, 2001, p247), with modification to support classes of chromosomes, rather than a single chromosome class.

4.2.3.7 Algorithm 7 – Evolution Strategy

A description of the evolutionary strategy algorithm for increasing the speed of evolution for a class of chromosomes, by using a selection mechanism based on historical analysis of the trend of the evolution, is given. Each chromosome stores in it's "s []" structure, its own evolutionary path. The algorithm can use these structures to development a coefficient assuming that the final destination of the automata can be determined. In the initial tests of the system, it was determined that a simple random selection mechanism be used, and this is described below. The selection strategies available are best, elite, fitness proportionate selection, random, rank based and tournament. These selection strategies are described in **algorithm 9 – algorithm 14**.

Algorithm 7 The evolutionary catch-up strategy algorithm for varies the selection mechanism

for a class to improve the evolution of the class based on the trend of the evolution.

Require: Given a class ID, then:

1. For each chromosome:
 - a. Determine the list of *unused-strategies* from the information stored in the chromosome.
 - b. **If** there are no *unused-strategies*, **then**
 - i. Randomly select from the list of *all-selection-strategies* for the class.
 - c. **Else**
 - i. Randomly select one from the *unused-strategies*.
2. Return *{selection-strategy}*

As an example, if a class has been defined to use only best, elite and tournament, then the system will randomly select from this list of selection strategies e.g., tournament, elite or best then start again, and randomly select one from the list. This process was not included in the work by Christian Jacob (Jacob, 2001).

4.2.3.8 Algorithm 8 – Create Chromosomes

A description of the create chromosome algorithm, which creates a population of random automata based on the passed number, the maximum number of states and an alphabet is given. This algorithm is used to seed the population within the system.

Algorithm 8 The create chromosome algorithm builds a random set of finite state automata.

Require: Given a number of chromosomes to build (*how-many*), the maximum number of states (*maxStates*) and an *alphabet*, then:

1. The system formats a population structure as *pop* [chromo [q [undef], p [], s [], p []] where a random finite state automata has been constructed and stored in *p []* for each *how-many* chromosomes.
2. Return *{pop}*, the population of finite state automata.

As an example, if **algorithm 8** is called with {50,10,{0,1,2,3,4}} then it will randomly create a population of 50 finite state automata with a maximum number of 10 states using an alphabet of {0,1,2,3,4}. This process extends the work of Christian Jacob (Jacob, 2001, p246), with additional strategy parameters. This population structure, and its embedded “chromo” structure, is the basis of all the schemes in the system.

4.2.3.9 Algorithm 9 – Best Selection

A description of the best selection algorithm for selecting chromosomes for mutation is provided. Like all selection mechanisms, the selected chromosomes pass their characteristics onto the next generation.

Algorithm 9 The best selection algorithm for selection of chromosomes.

Require: Given a population of chromosomes (*parents*) and the *best-count* of chromosomes required, then:

1. Sort the chromosomes in *parents*, into descending order of fitness.
2. Return {*best-count* set of chromosomes from *parents*}

As an example, if there are 100 chromosomes in a population, and *best-count* is 5, then **algorithm 9**, will return 5 chromosomes with the highest fitness. This process follows the work of Christian Jacob on genetic algorithm selection mechanisms (Jacob, 2001, p161).

4.2.3.10 Algorithm 10 – Elite Selection

A description of the elite selection algorithm for selecting chromosomes for mutation is given. Elite selection is similar to rank based selection. Each individual is given a ranking based on their fitness, and a set of individuals are chosen at random. The aim is to get a uniform distribution of individuals, and so increase the variability of the population. This is achieved by selecting a range of individuals e.g., 10, or selecting a % of those individuals by rank and then randomly choosing within that rank, the best individuals to form the next parent population.

Algorithm 10 The elite selection algorithm for selecting chromosomes for mutation.

Require: Given a *population* of chromosomes and the *elite-count* and *best-count* of chromosomes required, then:

1. Sort the *population* of chromosomes by fitness in descending order.
2. **If** *elite-count* is Integer, **then**
 - a. Set *eliteSet* to be the top *elite-count* chromosomes in *population*.
3. **Else**
 - a. Set *eliteSet* to be the top *elite-count* % of chromosomes in *population*.
4. Return {*best-count* randomly selected chromosomes from *eliteSet*}

As an example, if the *elite-count* is .32 and *best-count* is 5, then algorithm 10 will return 5 chromosomes from the top 32% of the population. This process follows the work of Christian Jacob (Jacob, 2001, p166).

4.2.3.11 Algorithm 11 – Fitness Proportionate Selection

A description of the fitness proportionate selection algorithm for selecting chromosomes is given. In the fitness proportionate selection, a balance between exploitation (previously explored regions of the search space i.e., existing schemes) against exploration (new organizations of scheme which might only be accessible via low fitness areas) is undertaken. The probability of an individual being reproduced into the next generation is

Chapter 4. Architecture and Design

proportionate to its fitness. These fitness values can be arranged as segments on a roulette wheel, with the size of each segment being directly proportionate to the corresponding individual's fitness (Jacob, 2001, p162).

Algorithm 11 The fitness proportionate selection algorithm for selecting chromosomes for mutation.

Require: Given a *population* of chromosomes and *best-count* of chromosomes required, then:

1. Create *fitpropSet* as the set of cumulative fitness values for the *population* of chromosomes.
2. Return {*best-count* randomly selected chromosomes from *fitpropSet*}.

The standard problem of convergence to a local optima (rather than global optima) occurs in fitness proportionate selection when there are a few individuals in a population with high fitness values compared to the rest of the population (Jacob, 2001, p164). Rank based selection overcomes this issue.

4.2.3.12 Algorithm 12 – Rank Based Selection

A description of the rank based selection algorithm for selecting chromosomes for mutation is given. Rank Based selection avoids the undesirable convergence of fitness proportionate selection, by normalizing differences in fitness values. In this approach, the individuals are sorted then selected not based on the value of their fitness, but on an individual's position within a fitness rank scale.

Algorithm 12 The rank based selection algorithm for selecting chromosomes for mutation.

Require: Given a *population* of chromosomes and the *best-count* of chromosomes required, then:

1. Sort the *population* into descending order of fitness of the chromosomes.
2. Set *rank-positions* to be rank of the fitness values of the chromosomes.
3. Return {*best-count* chromosomes with the highest *rank-positions*}

As an example, given a population of 100 chromosomes with the sorted fitness values of {0.88, 0.88, 0.84, 0.84, 0.7, 0.6...}, and a *best-count* of 2, then **algorithm 12** will return 4 chromosomes. This process follows the work of Christian Jacob (Jacob, 2001, p164).

4.2.3.13 Algorithm 13 – Random Selection

A description is given of the random selection algorithm for selecting chromosomes for mutation. Given a population of chromosomes, a random selection is returned.

Algorithm 13 The random selection algorithm for selecting chromosomes for mutation.

Require: Given a *population* of chromosomes and the *random-count* of chromosomes required, then:

1. Return {a *random-count* set of randomly selected chromosomes from *population*}

This process follows the work of Christian Jacob, which contains further discussions on deterministic and stochastic selection methods (Jacob, 2001, p167).

4.2.3.14 Algorithm 14 – Tournament Selection

A description is provided for the tournament selection algorithm for selecting chromosomes for mutation. Each individual is randomly selected from the population to compete in a tournament of size *tournament-competitors-count*, against a set of similarly randomly selected chromosomes. Each of the competitor's fitness values are compared to a randomly selected set of individual. The number of the times the competitors fitness are worse than the randomly selected individuals, are counted. This number is denoted as the individuals score. After these tournaments, all individuals are ranked in descending order of score value. The *best-count* of individuals with the highest score are randomly selected to form the next parent population.

Algorithm 14 The tournament selection algorithm for selecting chromosomes for mutation.

Require: Given a *population* of chromosomes and the *tournament-competitors-count* and *best-count* of chromosomes required, then:

1. Set *random-Population* to be a random selection of *tournament-competitors-count* chromosomes from *population*.
2. Set *competitors* to be a random selection of *tournament-competitors-count* chromosomes from *population*.
3. For each chromosome in *random-population*
 - a. Set *wins* to be the count of the number of times the chromosome fitness in *competitors* is < chromosome fitness for each chromosome in the *random-population*.
 - b. Store *wins* into the *random-population* by chromosome.
4. Sort into descending order, the *random-population*, by *wins*.
5. Return {*best-count* set of randomly selected chromosomes from the *random-population*}.

Chapter 4. Architecture and Design

The maintenance of diversity in a population is often the key to success in a given task, since diversity prevents premature convergence to a local optima. In many such systems, diversity is maintained through selection schemes, which encourage the population to break into separate sub-populations that occupy distinct niches of the search space. These “niche operators” maintain diversity in the population by fitness sharing. Theoretically, fitness sharing should distribute the number of individuals in various areas proportionally to the height of peaks in those areas. With a limited population size, only the “highest” regions will be covered. Fitness sharing works well with fitness proportional selection schemes, whereas tournament selection decreases the stability of the fitness sharing algorithm (Jacob, 2001).

As an example, if there are 11 fitness values with *tournament-competitors-count* of 5 and *best-count* of 5, then if fitness values as {0.31, 0.43, 0.52, 0.20, 0.17, 0.78, 0.99, 1.02, 0.35, 0.52, 0.61}, then the result of the tournament selection would be the chromosomes at positions {6, 7, 8, 10, 11}. This process follows the work of Christian Jacob (Jacob, 2001, p324).

4.2.3.15 Algorithm 15 – Add Class

A description is given of the add class algorithm which creates a descriptor of a new class of chromosomes (schemes) which will be stored in the global structure, *EVChromosomeClass*. Given a set of environmental inputs (from either an external or internal source), this process will generate a descriptor for the class. This descriptor is then used by the system to build a scheme that can process the environmental inputs. These inputs may come from both the external or internal environment.

Algorithm 15 The add class algorithm for adding a new class of chromosomes.

Require: Given a set of environmental *inputs*, *outputs*, the *type* of class (Prediction, Argos or Piagetian), the alphabet and an optional evaluation *function*, *response* function, set of *available-strategy* (*best*, *elite*, *fitness proportionate*, *rank*, *random*, *tournament*), *number-of-generations*, *description*, *number-of-parents*, *number-of-children*, *stop-at-perfection*, *number-of-copies*, *count-of-vanilla*.

1. The system generates an entry into the *EVChromosomeClass*, giving it a unique *chromosome class ID*, as well as the passed parameters.
2. Increment the *chromosome class ID* by 1.
3. Return {TRUE, *chromosome-class-ID*}

As an example, if **algorithm 15** is passed {{1,0,0,1,1,2},{1,0,0,1,1,2}, Prediction, {0,1}}, then the algorithm will generate an entry into *EVChromosomeClass* as a prediction scheme using the standard prediction quality function (**algorithm 1**), the standard automaton response (**algorithm 2**), as well as the standard evaluate automaton (**algorithm 3**).

4.2.3.16 Algorithm 16 – Act on Environment

A description is given of the act algorithm as the implementation of the Piagetian term “recognition,” which enables the execution of a selected scheme against the environment. The environment may be either the internal or the external environment.

Algorithm 16 The act algorithm for executing against the environment.

Require: Given the currently selected *scheme*, then:

1. Use **algorithm 2** (Automaton Response) to generate the outputs from the automata and then pass these to the environment. The result of the execution is that the environment is changed.
2. Pass control to **algorithm 17**.

As an example, if the scheme 10115 sets the penState to penDown, sets the direction to right, and then sets the duration to 3, then when this scheme is executed, a line is drawn on the external number–line. A direct relationship to the Piagetian processes can clearly be seen (§ 2.3.6.1).

4.2.3.17 Algorithm 17 – Sense Environment

A description is given of the sense environment algorithm, as the implementation of the Piagetian term “perception,” which observes the environment and then determines how that information can be used. The focus of sensing is the assimilation of inputs from the environment. The environment in this context can be either the internal or the external environment.

Algorithm 17 The sense algorithm for assimilating the environmental input.

Require: Given the *environment* in a specific state, the environment provides a set of environmental *inputs*. These inputs are passed to the algorithm along with the *EVChromosomeClass*, *EVChromosomeID* and *EVPopulation* then:

1. Assimilate the environmental *inputs* by determining if a *scheme* exists, that can process the *inputs* by comparing the *inputs* to the *inputs* in the set of *chromosome classes*.
2. **If a *scheme* has been found that has a prediction quality of ≥ 1 . (perfection), then**
 - a. Pass control to **algorithm 16**, to process the inputs using the selected *scheme*.
3. **If a *scheme* has not been found that can process the *inputs*, then**
 - a. Change state to “reinforce” and pass control to **algorithm 18** and, by consequence, **algorithm 19**, to enable imitation and reconstruction on an existing or new *scheme*.
4. **Else**
 - a. Pass control to **algorithm 16**.

Chapter 4. Architecture and Design

As an example, if the environmental input is {1,0,1,0,1} and chromosome Class 12 can process this input, and scheme 12345 has a prediction quality of 1., then the system will use continue in the state of “acting and sensing” to process this input. A direct relationship to the Piagetian processes can clearly be seen (§ 2.3.6.1).

4.2.3.18 Algorithm 18 – Imitation

A description is provided of the imitation algorithm (used in this study), as the implementation of the Piagetian term “imitation,” which describes how the system imitates the environmental inputs and potentially creates a new class of scheme to process those inputs.

Algorithm 19 The imitation algorithm assimilates the environmental inputs and imitates them.

Require: Given a set of environmental *inputs*, the *EVChromosomeClass*, *EVChromosomeID* and *EVPopulation* then:

1. If a *scheme* is close enough to be evolved to handle the *inputs* (the prediction quality < 1.0, so more learning through evolution is required), then
 - a. Pass control to **algorithm 19** to enable reconstruction (continued learning through evolution) on the existing *scheme*.
2. If a *scheme* has not been found that can process the *inputs*, then
 - a. Use **algorithm 15** to create a chromosome class. In this case, there will be no chromosomes in the population, and as such, they will be built from the 2-bit machines (the simplest of all finite state automata).
 - b. Pass control to **algorithm 19** to enable reconstruction (creation through continued learning through evolution) of the new *scheme*.
3. Else
 - a. Pass control to **algorithm 19**.

As an example, if the environmental input is {1,0,1,1,1} and there are no schemes to process this input, then the system will use **algorithm 15** to create a chromosome class and then pass control to **algorithm 19** to build a prediction scheme that can process this input. A direct relationship to the Piagetian processes can clearly be seen (§ figure 4–3).

4.2.3.19 Algorithm 19 – Reconstruction

A description follows of the reconstruction algorithm (used in this study), as the implementation of the Piagetian term “reconstruction,” which creates a new class of scheme or continues to modify an existing scheme if the prediction quality of the scheme is less than 1.0 (perfection). This algorithm can be used for a particular class or for multiple classes of chromosomes. In most cases, it will be for a single class.

Algorithm 19 The reconstruction algorithm for creating new scheme or modifying an existing scheme.

Require: Given a set of environmental *inputs*, *EVChromosomeClass*, *EVChromosomeID* and *EVPopulation* and an optional *chromosome-class-ID*, **then**

1. Use **Algorithm 5** to evolve classes of chromosomes.
2. Select a scheme from the *population* as the selected *scheme*.
3. Change state to “act and sense” and pass control to **algorithm 16** to enable action on the environment by the selected *scheme*.

As an example, if the system was in the state of “act and sense,” and **algorithm 17** – sense environment, could not process the environmental input of {1, 0, 1, 1, 1}, then **algorithm 17** – sense environment, would have changed the state of the system into “reinforce” and passed control to **algorithm 18**. **Algorithm 18** would have created chromosome class ID 1234567 to imitate the input in *EVChromosomeClass*, and then passed control to **algorithm 19**. **Algorithm 19** then uses **algorithm 5** to build a prediction scheme that can process this input. For example, chromosome class 1567 could have been built to process the inputs {1,0,1,1,1}. In building the new scheme, the following algorithms will be used in turn **algorithm 5**, **algorithm 6**, **algorithm 8**, **algorithm 3**, **algorithm 8**, **algorithm 3**, **algorithm 4**, **algorithm 3**, **algorithm 7** and **algorithm 9**. When the evolutionary cycle has completed, and a *scheme* with chromosome ID 1871787 (it may be a partial scheme) has been built, then **algorithm 19** will change the state to be “act and sense” and then pass scheme 1871787 and control to **algorithm 16** to act on the environment. From this example, it is hoped that the relationship between the Piagetian processes of assimilation and accommodation and the algorithmic solution are clearly seen (§ 2.3.6.2).

4.2.4 Example Execution of Dialectic System

An example execution of the *dialectic system* is provided in table 4-10. It specifically lays out the required set of evolutions required to meet the needs of the *worked example* (§ 2.3.14), and occurs over a set of generations of schemes. This also provides a model of the evaluation that adheres to the research design (§ 3.4). The algorithms defined in (§ 4.2.3) describe how the Prediction schemes are constructed. The Prediction schemes are directly related to Level 1 – regularity, Level 2 – Coordinated Action, Level 3 – Internalized Structure, and partially Level 4 – Symbolic Functions. Each of the levels is described in summary, below.

4.2.4.1 Level 1 – Regularity

Starting from the simplest FSA, prediction schemes are evolved through mutation that can detect regularity in the environment, including point, line, movement and penState.

Chapter 4. Architecture and Design

4.2.4.2 Level 2 – Coordinated Action

Reusing the FSA of level 1, networks of FSA are produced that move discrete units along a number line (Furth, 1969, p125).

4.2.4.3 Level 3 – Internalized Structure

FSA are evolved that internalize external values. The simplest structures are components of digital circuits. These networks reuse the FSA of level 1 and 2.

4.2.4.4 Level 4 – Symbolic Functions

HFSA as Argos schemes are evolved together to produce circuitry such as equivalence machines, encoders, decoders, half-adders, full-adders. HFSA as Piagetian Schemes machine are evolved together to produce interaction with the environment to support the processes of assimilation and accommodation, to support more complex processing. The HFSA reuse and schemes of level 1, 2 and 3 to produce schemes that for instance can count using a process of assimilation and accommodation.

The normal operation of this simulation consists of a set of steps. Though the steps themselves are not programmed into the system, there is a natural sequence of operations, which occur. A simplistic sequence is given in table 4-10:

Table 4-10 Normal Operation of the system.

#	Description
1)	The simulation executes and the teacher creates a worksheet.
2)	The teacher interacts using the worksheet with a student, responding with rewards.
3)	From its primary reactions and innate schemes, the student interacts with the worksheet, plays and develops its sensory motor model of the world and in doing so works through WE1 – WE5 .
4)	The student continues work on its worksheet adapting its sensory motor schemes and developing its processes of assimilation and accommodation, using its hierarchical machines to expand its repertoire of actions and constructions – its figurative and operative schemes of perception–recognition, imitation–reconstruction and mental image–evocation.
5)	In experiencing the number line world, over time, there are emergent schemes that enable a student to attempt to resolve the presented problems by enabling the agent to control the interaction.
6)	Typically, the environment will evolve basic FSA of basic gates, buffers, not machines and prediction machines and from these construct hierarchical machines for interaction and marshaling the processing of information.

4.3 Summary

This chapter described the architecture and the design of the *artificial neural network implementation* and the *dialectic system* solution to resolve (partially) the learning paradox. It included definitions of the major algorithms in the system that are used to construct Prediction schemes. For the *dialectic system*, it shows that an evolutionary path of at least 4 levels is required. In **Level-1**, simple mealy machines are evolved to detect and predict regularities in the external environment using the number line. In **Level 2**, these prediction schemes are to make more complex predictions with the environment, to coordinate actions. In **Level-3**, prediction machines are evolved that can internalize values from the environment so as to form digital circuits. In **Level-4**, the schemes from Level 1, 2 and 3 are combined into hierarchies to develop Argos schemes, which can process inputs from the environment, and Piagetian Schemes that can fully interact with the environment using a process of assimilation and accommodation. In this it can be seen, that a Piagetian simulation is only possible, after the evolution of the preceding levels has occurred.

The architecture sidesteps Fodor's arguments for **LP1** by having a propositional layer (of Prediction schemes) and a predicate layer of Argos and Piagetian Schemes. Potentially the system can operate autonomously (**LP3**) and has the capacity for **LP5**, with some potential for development (**LP6**). It is yet to be seen if it can resolve the issues of emergence of hierarchical concepts using evolutionary process (**LP2**), or act in novel, opportunistic and noisy situations (**LP7**) or more importantly mirrors the real world behavior of children developing number-sense (**LP4**). It has been shown that it has the potential for developing machines that can count. It is suspected however, that the machines would have to go through wholesale reorganization and make use of many of the mathematical relations, which Piaget observed during child hood development (Copeland, 1974) to develop schemes that can develop number-sense.

Chapter 5

5. Evaluation

In § 1.2 the aims of this project were defined as: (i) *evaluate epistemological solutions to the learning paradox* (§ 2.1), (ii) *evaluate sources of emergence* (§ 2.2), (iii) *provide a computational reading of Piaget's theory* (§ 2.3), (iv) *clarify emergence in a real-world worked example* (§ 2.3.14 and 2.4), (v) *compare and contrast different notions of mathematical concept formation* (§ 2.3, 2.4, 2.5, 2.6 and 2.7), (vi) *provide arguments to support a separation into an artificial neural network implementation and a dialectical system* (§ 2.10 and 3.4), (vii) *evaluate the artificial neural network implementation*, and (viii) *evaluate the dialectical system*.

The aim of this chapter is to evaluate the research solution to determine if project aims 7 and 8 (§ 1.2) are met. To do this, requires a series of evaluation criteria; namely, a set of constraints that, any solution that attempts to resolve the learning paradox would have to resolve (§ 1.2.7 for a definition of **LP1 – LP7**). Next, an example problem of numerosity in a number–line world (§ 2.3.14 for a definition of **WE1 – WE5**) is defined that was used to evaluate candidate concept formation solutions during the literature review. Next, Crutchfield's intuitive emergence approach (Crutchfield, 1994a, p2) is identified as a mechanism to detect emergence in the behavior of a system (§ 2.2.1). The review of the literature asserted that a Piagetian model of genetic epistemology (Furth, 1969 and Copeland, 1974), which is mirrored in the model of *Drosophila* (Miesenböck, 2008, p52; Shang, Claridge–Chang, Sjulson, Pypaert and Miesenböck, 2007, p601) could be capable of providing an archetype for emergent behavior, and identified this as the primary research goal (§ 3.2.1). The execution of this implemented architecture and design is evaluated in this chapter, against these evaluation criteria.

This chapter is organized as follows: in § 5.1 is provided an evaluation of the biologically inspired *artificial neural network implementation* of a Piagetian / *Drosophila* model using reinforcement learning, against the requirements of the research experiments (§ 3.5). In § 5.2 the biologically inspired Piagetian / *Drosophila* modeled *dialectic system* is assessed in relationship to the requirements of the research experiments (§ 3.5) and the needs of the worked example (**WE1 – WE5**). This approach is justified based on the evaluation of results that suggest a Piagetian model of cognition (Modha and Singh, 2010, p13488; and Albus, 2000; 2008 and 2010b, p193). A summary is provided in § 5.3.

5.1 Evaluation of Artificial Neural Network Implementation

This section evaluates the biologically inspired *artificial neural network implementation* of a Piagetian / *Drosophila* model using reinforcement learning against the requirements of the research experiments (§ 3.5). Specifically, there are two levels of tests (i) the identification of the permanent objects in the environment, and (ii) coordinated movement. Each set of experiments is described separately.

5.1.1 Level 1 – Identification of Permanent Objects Invariants

Within the permanent object tests there are 3 discrete levels: detect regularities in the environment, coordinate movements in the number line world and plan actions. Each is covered in turn.

5.1.1.1 Detecting Regularities in the environment

The aim is to determine the conditions under which the system can identify regularities in the environment using different configuration values for discrete and continuous sensors and various reinforcement learning rates (§ 4.1). Specifically, to determine if there is emergence of the *permanent object invariants* of: point, line, direction, penState (penUp, penDown, Stop) and movement.

The task for the Verve Agent was to move from a starting location to a goal location within the problem space and to appropriately make use of its penState (penUp, penDown) to draw lines along its path. The agent could sense its environment using both discrete and continuous sensors. As the agent explores different parts of the state space, it adds new radial basis functions (RBFs). Eventually, all the states that are actually experienced, get covered in RBFs, as shown below, figure 5–1:

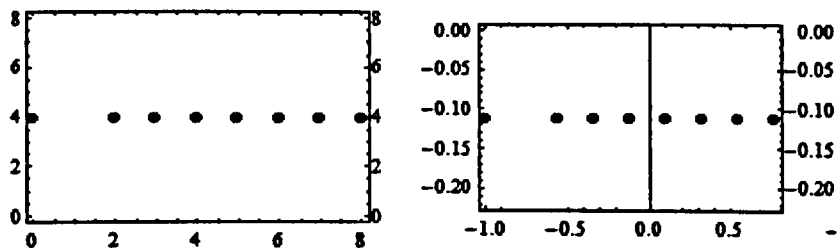


Figure 5–1 A comparative state space representation of a radial basis function neural network with discrete and continuous sensors – used in the detection of regularities in the environment. In this case, RBFs detect the point “1” in the environment.

In figure 5–1, the circle diameters are meaningless: the RBFs overlap significantly. In the example from test–27, the starting location is never evaluated; this is true for both discrete and continuous sensors. As the agent solves more complex problems, these RBFs will cover more of the state space. What figure 5–1 shows is that the system can identify regularities in the environment.

Chapter 5. Evaluation

The purpose of the state value function is as a “critic” that criticizes the actions made by the agent and reinforces actions using prediction errors (Sutton and Barto, 1998, p151). The evaluation of the discrete state value function is represented in the following 3D contour maps (figure 5–2):

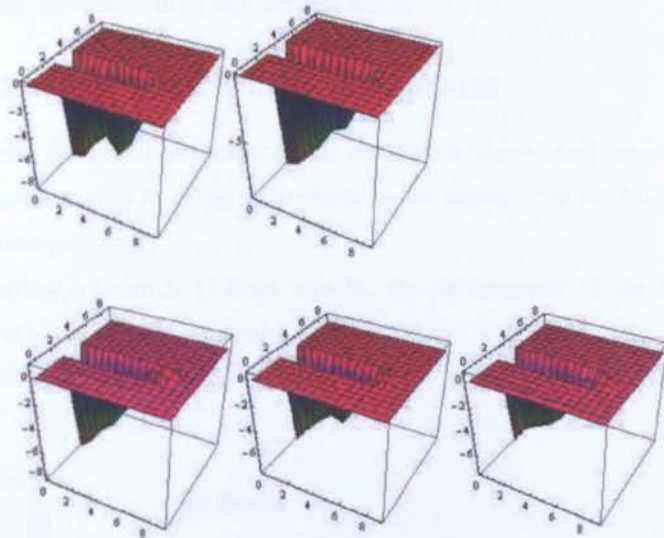


Figure 5–2 A depiction of a state value function using discrete sensors (from test 26), that show that the agent using a linear neural network and reinforcement learning can draw a line and identify points in a number line world and so achieve WE1 – WE3 of the worked example.

In figure 5–2, it is clear that the agent progressively develops a negative reward gradient that lead towards a localized positive maxima, the goal point. This is understood, when one realizes that the reward function drives the development of the state value function. An alternate view of the state value function for discrete sensors for test–26 is depicted in the following heat maps, figure 5–3:

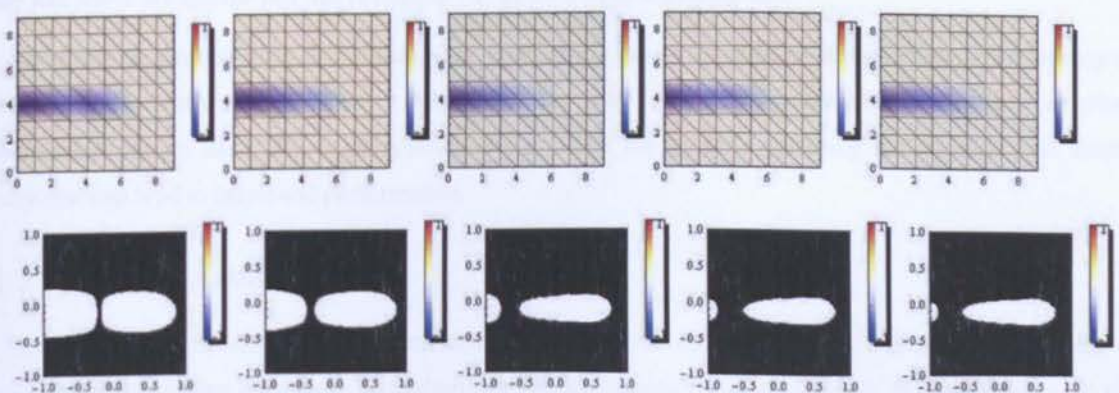


Figure 5–3 A heap map of a state value function using discrete sensors and continuous sensors (from test 26), that show that the agent using a linear neural network and reinforcement learning can draw a line and identify points in a number line world and so achieve WE1 – WE3 of the worked example.

Chapter 5. Evaluation

In figure 5-3, the state value function is shown as a progressive development of the awareness of finite points along the number line. It can be seen that the agent develops a state value function that accurately depicts the landscape in which it inhabits. As a result, it identifies regularities in that environment, including, as depicted, a point, line and movement in its number line world.

5.1.1.2 Coordinated Actions in the Number Line World

The aim is to determine the conditions under which the system can act and sense using the regularities it has identified in the environment, and in doing so navigate in the number line world as a form of coordinated action from relative and static positions.

In the comparison below, (figure 5-4) from test-26, the performance of the agent using continuous sensors is better than that of using discrete sensors:

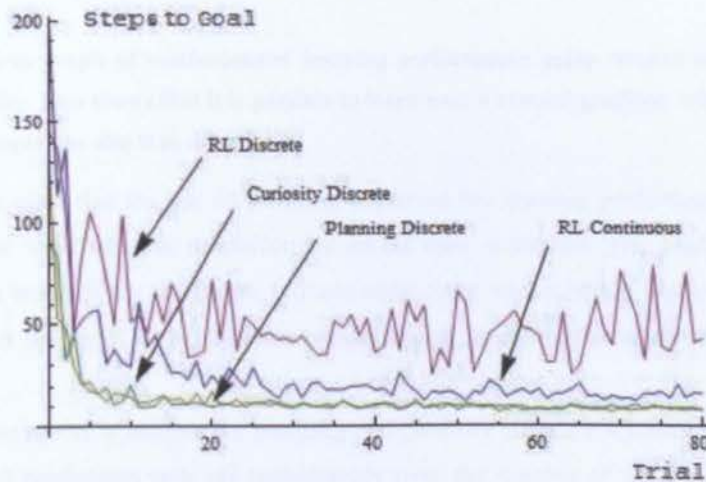


Figure 5-4 Comparison of learning performance – continuous vs. discrete sensors for a verve agent navigating an a number line world the specific test was to identify “3” as $1 + 1 + 1$ as a number line.

What can be gleaned from this is that when continuous values are used as real numbers, the accuracy of the movement is clearly shown. When a reward for curiosity is used, the processing is the same as when continuous sensors are used. The implications are that when the agent is operating in a number line world, curiosity rewards lead to improved performance.

5.1.1.3 Planning Action Tests

This aim of the planning test is to determine the conditions under which the system can plan a set of actions rather than relying on merely acting/sensing the environment, and so improve its performance. In the comparison below (figure 5-5), from test-26, test-27 and test-28, the reward sum for reinforcement learning is compared against planning and curiosity.

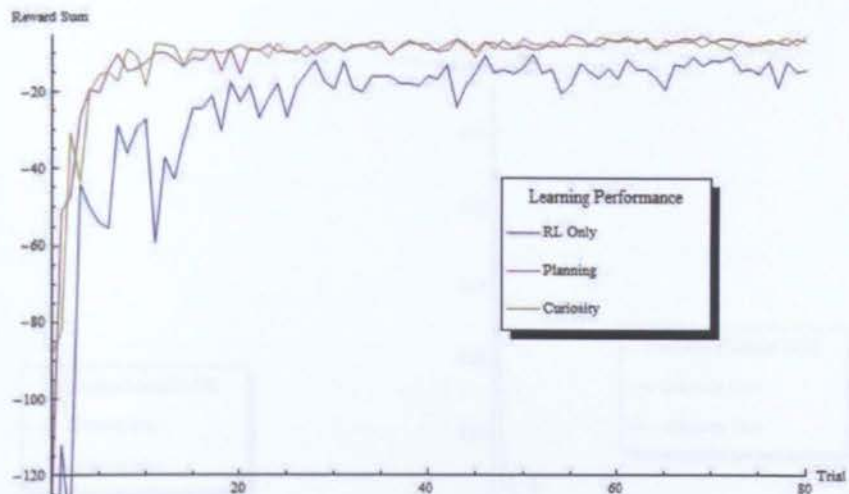


Figure 5-5 A comparative graph of reinforcement learning performance using reward sum for learning only, planning and with curiosity. This shows that it is possible to learn over a reward gradient, with curiosity improving performance. In this example, the aim is to draw 1 + 2.

In figure 5-5, it is clear that the use of planning improves the learning performance by increasing the average reward over the trial, whereas reinforcement on its own is limited. The similarity of planning to curiosity is explained in terms of the conditions of these tests; there was only one numeric reward in the test. Curiosity, which enables the agent to explore unexplored states, is ineffective where the reward gradient a single maxima.

When the predictive model is utilized for planning and curiosity (figure 5-6), the mean squared error for observations and reward predictions tails off considerably over the number of trails and typically becomes optimal after five trials when using discrete sensors. For continuous sensors, it takes 20 trials to reach optima.

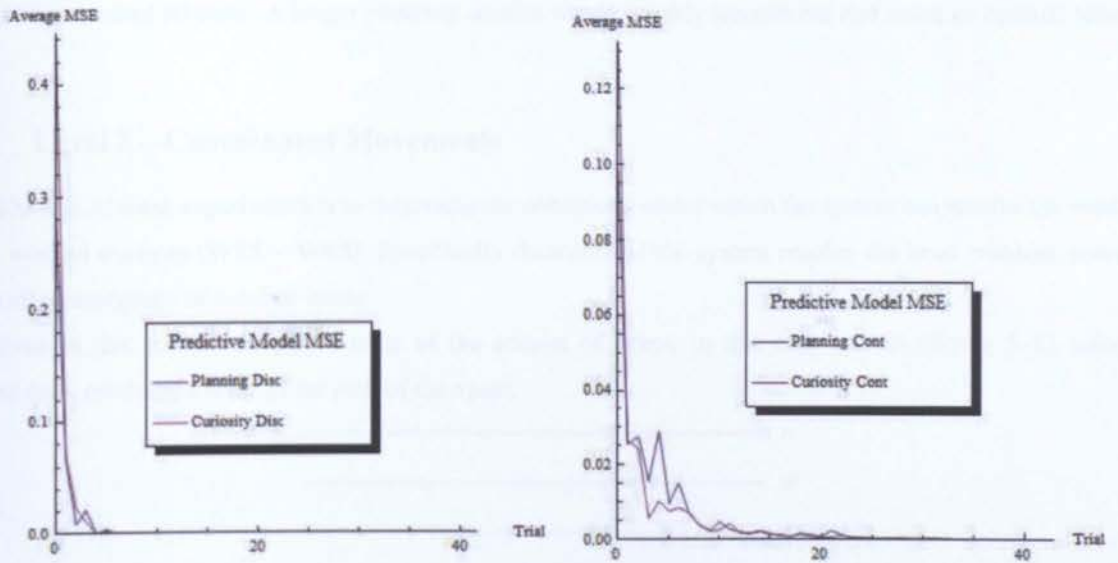


Figure 5-6 A comparative analysis of mean squared error (MSE) using a predictive model to detect objects in the environment with discrete and continuous sensors. It shows that planning with continuous sensors provides benefits over other models. In this test, the aim was to receive a reward after drawing 1 + 1 then 2.

There is processing overhead (as shown in the right hand diagram: the planning cost is greater than the curiosity cost) when using a predictive model. This overhead is caused by the generation of the predictive model. This is countered by the improvement in the number of planning steps taken over time. This is simply displayed in the diagram below (figure 5-7):

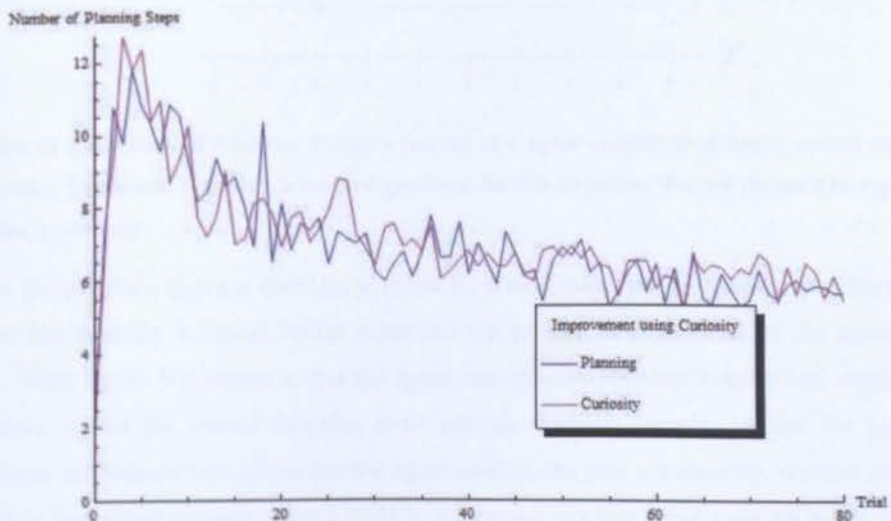


Figure 5-7 A comparative model of the number of steps to reach the maximum reward, showing the improvement when curiosity is embedded into the reward gradient. Thus an agent which explores achieves greater reward. These processes provide the basis for sensory motor schemes in a Piagetian model. In this test, the aim was to receive a reward after drawing 2 + 2 then 4.

Chapter 5. Evaluation

One noticeable point in figure 5–7 is the trending of planning and curiosity at 6 steps. Given that test 26, 27 and 28 only used 80 trials. A longer planning session would quickly smooth out and reach an optimal value of 6.

5.1.2 Level 2 – Coordinated Movements

The aim of these experiments is to determine the conditions under which the system can resolve the needs of the worked example (WE1 – WE5). Specifically determine if the system resolve the bead problem, count and exhibit emergence of number–sense.

Even in this trivial example, a trace of the actions of agent, in this case test–26 (figure 5–8), using discrete data, produces a trace of the path of the agent.

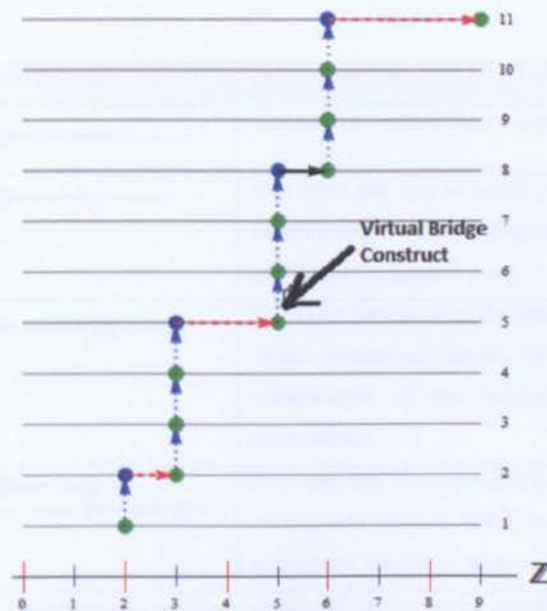


Figure 5–8 A plot of the actions of trial run 4 trial 9 test–26 of a agent composed of linear neural networks, as the operate in a number line world receiving a reward gradient. In this example, the red dotted line represents penup and the black line, pendown.

Whenever the penState changes (penUp, penDown), a new number line segment is constructed and for the purposes of intelligibility a virtual bridge construct not present in the actions of the agent is added for visual contrast. What figure 5–8 shows is that the agent can traverse the search space and reach its goal! The interesting feature is that the reward function must provide distinct rewards, so that the agent can draw uninterrupted lines. Subsequent tests show that the agent reaches the goal consistently, without interruption.

This result is interesting because when a child tackles a number line using a pen on paper, he/she initially draws lines that are not straight, nor parallel to the worksheet number line. The child draws small disconnected stages; his/her pen does not always touch the page. In the examples presented (figure 5–8), the agent plainly gets the reward and mirrors the child like interplay.





Chapter 5. Evaluation

Over time with repeated performances and with a reward gradient that rewards the agent for drawing a line with the correct pen state, as one continuous line, the agent will mirror the more accomplished child.

5.1.3 Summary

From the results presented it is clear that Verve, through its use of actor-critic reinforcement learning, with radial basis functions and linear neural networks, provides a mechanism for defining an alphabet of interaction, the *permanent object invariants*. A summary is presented in table.5-1, below:

Table 5-1 A summary evaluation of the sensory motor development (actions in the external world) of an agent composed of linear neural network operating with continuous and discrete sensors (as radial basis function neural network) as they explore their number line world and develops an internal network that corresponds to objects in the external world.

#	Need	Visualization	Analysis
1	WE1 – Base Level		The agent can draw lines using reinforcement learning.
2	WE2 – Constrained Level		The agent can stop at points along a continuous line. This is Piaget's conservation of length and distance (Copeland, 1974, p253) and the implied point.
3	WE3 – Differentiated Level – Object		The agent has developed a notion of an object; this is Piaget's object permanence (Furth, 1969, p125). This is clear in the visualization of the reward gradient and its repeated performance.
4	WE4 – Hierarchical Level – Segregation into Units		It is doubtful if a convincing argument could be made that these result support WE4, because the agent does not yet understand the notion of the application of unit.

The results confirm that a linear neural network implementation using reinforcement learning (Streeter, Oliver and Sannier, 2006); can build the primary reactions and innate schemes (Furth, 1969 p162, p186, p197, p224 and p231) – the alphabet of interaction (point, line, direction) – that the dialectic solution needs to use. This alphabet is the set of *permanent object invariants*, predicted by Piaget. This agrees with results of other Piagetian researchers (Chaput, 2004) and confirms the appearance of conservation of line, distance, measurement upon which the development of Piagetian mathematical ability is based. It also appears that the solution can perform Piagetian action-coordination, and the identification of the *permanent object invariant* (Furth, 1969, p248).

A simple assumption is made at this point, that symbolic external environmental values are substituted for these sub-symbolic sensory values (Mayo, 2003, p57). This allows for the development of *dialectic system* to manipulate it. One of the reasons for not developing further a hierarchical linear neural network solution was the time it took to build the neural network. Even for trivial problems like this, it took 2 weeks running at 100% CPU on a quad core machine. It seems that this process is very inefficient method of building a network.

5.2 Evaluation of Dialectic System

This section evaluates the biologically inspired Piagetian / *Drosophila* modeled *dialectic system* against the requirements of the research experiments (§ 3.5) and the needs of the worked example (WE1 – WE5). Specifically, there are four levels of tests of emergence (i) regularities in the environment; (ii) coordinated action; (iii) internalized structure, and (iv) symbolic functions, which will make use of the tests for intuitive emergence (§ 2.2, A.9 and Crutchfield, 1994a; 1994b).

5.2.1 Level 1 – Regularities in the Environment

There are two discrete aims. First, detect regularities in the environment. Second, perform the processes of acting and sensing using those identified regularities.

5.2.1.1 Detecting Regularities in the environment

The aim is to determine the conditions under which the simulation can learn regularities in the environment using the *permanent object invariants* provided by the *artificial neural network implementation*. Specifically, to determine if there is emergence of the permanent objects invariants of point, line, direction, penState (penUp, penDown, Stop) and movement as required by the *worked example* (WE1 – WE5).

Further, to determine if the simulation can generate Prediction schemes from the EBNF grammar that will allow the simulation to control its external and internal the environment. These Prediction schemes include the penStates of penUp and penDown.

Initially, an evaluation of the evolutionary process used to construct Prediction schemes from simplest type of mealy machine is described.

Learning as Evolution

The evolutionary process used to construct Prediction schemes takes as its input the simple mealy machine described in the architecture and design (§ figure 4–13). It also includes, the class definition, which provides the required input and output values using the selected evolutionary strategy, which evolves the required prediction scheme.

Figure 5.9 is the result of the evolution of a class (EVClass 1.). Copies of the simple Mealy machine, that were described in the design are randomly mutated, so that given an 0 as input, they produce a zero as output and given an input value of 1, produce a 1 as output. Over 2 generation steps, they generate chromosome 300093, with a prediction quality of 1.0 and a fitness of 1.55.

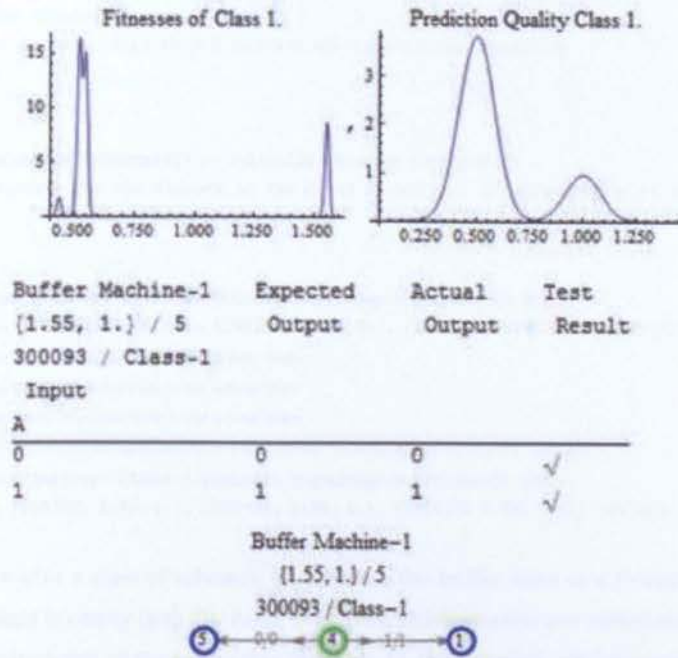


Figure 5-9 The evolution and evaluation of a class (Evclass 1.) Of a prediction buffer scheme. The top two graphs describe the fitness and prediction quality (x axis) against population (y axis). In both the distributions of the measures are non-random, as would be expected. The central image describes the input and output values of the prediction scheme. The bottom images provides a representation of the prediction scheme with the starting states in green (state 4) and the 2 final states (states 1 and 5) in blue. The transition values are also shown.

What Figure 5-9 demonstrates is that this evolved scheme has 5 states, with only states 1, 4 and 5 being valid states. The transition values 0/0 and 1/1 are shown between the initial state 4 (shown in green) and the final states 5 and 1 (shown in blue). The graphs of fitness and prediction quality show a non-normal distribution, which is anticipated due to the evaluation and selection mechanism used. In the following diagram (figure 5-10), it shows that 604 attempts were made to evolve this initial class, with the top 5 being shown for each generation step. The majority of chromosomes have a prediction quality of 0.5, i.e., they do not correctly predict the next state and only 3 of the 604 have a prediction quality of 1.

A simple step mechanism, as a set of commands, is used to evaluate all the classes of Prediction schemes that have not reached a prediction quality of 1. An example is presented in figure 5-10:

Chapter 5. Evaluation

```
(* Step 3 - Evolve the classes *)
EVParents (* Show the parents, this should include any constructed values *)
EVChromosomeClassID;
EVChromosomeID;
EVChromosomeClass;
EVParents = EVEvolveClasses[EVParents]; (* Initially this is just pop [] ,
and it should only execute for the classes in the global structure. If this changes it should not evolve any
other classes. *)

pop[]

IDs-Fitness-Prediction Quality: Class-1.genCount-1-genStep-0 EVClassID: ALL
{{300049, 0.55, 0.5}, {300050, 0.04, 0.}, {300048, 0.05, 0.}, {300047, 0.04, 0.}, {300046, 0.03, 0.}}
ChangeFinalState::NotAFinalState: The fromState is not a final State
ChangeFinalState::NotAFinalState: The fromState is not a final State
ChangeFinalState::NotAFinalState: The fromState is not a final State
General::stop: Further output of ChangeFinalState::NotAFinalState will be suppressed during this calculation. >
IDs-Fitness-Prediction Quality: Class-1.genCount-2-genStep-1 EVClassID: ALL
{{300604, 1.55, 1.}, {300093, 1.55, 1.}, {300604, 1.55, 1.}, {300110, 0.55, 0.5}, {300080, 0.55, 0.5}}
```

Figure 5–10 Commands to evolve a class of schemes, in this case the buffer class as a Prediction scheme. Initially, the population of chromosomes is empty (pop []). After evolution, chromosomes are added and mutated to produce machines that meet the requirements of the evaluation function. In this example, chromosome 300604 has a fitness of 1.55 and a prediction quality of 1.

The commands call the `EVEvolveClasses` function with a null population of chromosomes (pop []). The `EVEvolveClasses` function uses the values in `EVChromosomeClass` and evolves each class in turn (§ 4.2.2.5 and figure 4–22 on class structure and class based evolution). The first few lines of output of this function show the Chromosome IDs, fitness and prediction quality of the top 5 chromosomes for the class. After generation count 2, the fitness and prediction quality of chromosome 30604 reached 1.55 and 1.0 respectively. This simulation has correctly produced a prediction scheme as a FSA that can correctly predict the input sequence. This prediction scheme can be used to act in the world and sense in the world. However, this is a toy example.

Each class has a selection mechanism of processes that can be used and varied. The class-based evolution function processes each population of schemes for the class, as a list and each defined class has a specific evaluation function. Typically, all processing in Mathematica is list based. The chromosome itself “chromo [q [], p [– Automaton –, s [], h []]” have a number of parameters that are used both to capture its construction and its processing: q holds the fitness and prediction quality, p is the automaton s holds the strategy parameters, including all the details of the class. The h [] structure contains all the details about the hierarchy. An analysis of all the evolved classes for generation 1 produces a list of classes with the top ranked chromosomes and their byte count being shown below (figure 5–11):

EVClassID	Description	EVChromosomeID	EVClassName	ByteCount	Fitness
1.	Buffer Machine-1	300 093	BUFFERNOT	774 392	1.55
2.	NOT Machine-2	303 178	BUFFERNOT	774 360	1.55
3.	Binary Output 1-3	306 204	BUFFERNOT	774 504	1.55
4.	Binary Output 0-4	306 379	BUFFERNOT	774 200	1.52
5.	Sensory Motor Move Right 1 Unit-5	327 264	PREDICTION	775 112	1.5
6.	Basic Gate AND Machine-6	772 354	BASIGATE	777 880	1.67
7.	Basic Gate OR Machine-7	825 480	BASIGATE	775 928	1.68
8.	Basic Gate XOR Machine-8	1 559 639	BASIGATE	775 960	1.68
9.	Basic Gate NAND Machine-9	1 616 465	BASIGATE	783 944	1.32
10.	Basic Gate NOR Machine-10	1 623 088	BASIGATE	775 968	1.66
11.	Basic Gate XNOR Machine-11	1 337 873	BASIGATE	775 648	1.68
12.	Action-000 Do Nothing-12	1 671 354	PREDICTION	789 008	1.5625
13.	Action-001 penDown-13	1 350 070	PREDICTION	783 760	1.625
14.	Action-002 penUp-14	1 354 679	PREDICTION	777 248	1.66667
15.	Action-003 Left-15	1 356 139	PREDICTION	780 304	1.6
16.	Action-004 Stop-16	1 360 594	PREDICTION	781 152	1.625
17.	Action-005 Right-17	1 363 337	PREDICTION	777 536	1.75
18.	Action-006 Move <duration>-18	1 365 466	PREDICTION	782 176	1.625
19.	Action-007 Student Request, Whats the solution-19	1 368 519	PREDICTION	782 880	1.625
20.	Action-008 Student Request, I give up-20	1 375 069	PREDICTION	779 888	1.66667
21.	Action-009 Student Request, Explain Solution-21	1 379 899	PREDICTION	777 312	1.625
22.	Action-010 Add Segment at end-22	1 676 911	PREDICTION	776 528	1.75
23.	Action-011 Delete Last Segment-23	1 393 955	PREDICTION	778 096	1.75
24.	Action-012 Get Reward (Assimilation)-24	1 395 849	PREDICTION	781 680	1.6
25.	Action-013 Reset-25	1 684 712	PREDICTION	784 672	1.55556

Figure 5-11 An example execution of class-based evolution showing the top ranked chromosomes from generation 1.

Prediction Scheme: penUp

The aim of this Prediction scheme (figure 5-12) is to be able to sense a change in its penState of penUp, which corresponds to action 2 being provided by the environment. Table 5-2 describes the action in terms of the EBNF grammar (§ figure 4-1).

Table 5-2 Parameters to change the penState to penUp in the environment.

#	Action	Description
1	2	penUp

Once built, the Prediction Scheme penUp can be used to act on the environment and change the penState, and even detect the penUp penState in the current environment.

The same process is followed for all Prediction schemes and for all unary values in the EBNF grammar (§ 4.2.1.1)

Chapter 5. Evaluation

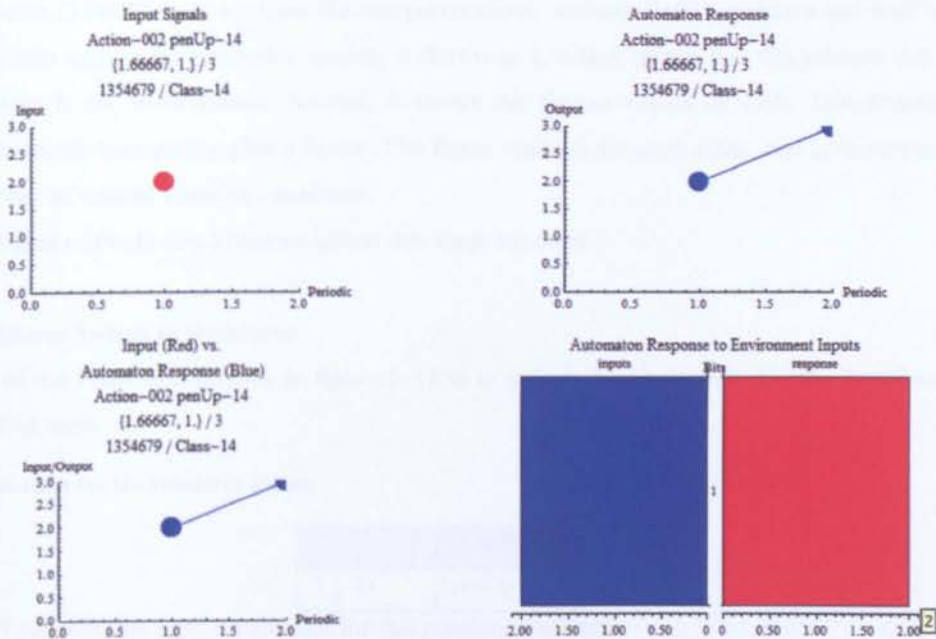


Figure 5-12 An example of a prediction scheme that correctly detects regularities in the environment, in this case the penState of penUp as action 2. The graphs in red show the value that needs to be detected. The graphs in blue show the values the scheme predicts. The bottom right graph shows the prediction scheme has correctly predicted the input values. Since this is only a single value, the bottom right image is a perfect 1:1 match on input values.

Figure 5-12 shows that the simulation can build Prediction schemes that correctly predict the environment values. A depiction of this Prediction Scheme is shown in figure 5-13.

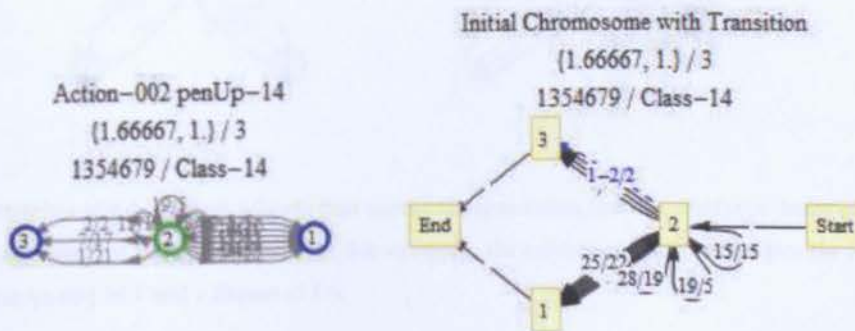


Figure 5-13 A depiction of chromosome 1354679 as Class-14 as a prediction scheme, that updates penState. This scheme has a prediction quality of 1 and a fitness value of 1.66. In the image on the left, the start state is shown in green, and the two end states in blue. In image on the right shows the same chromosome with the values of the transitions shown.

Chapter 5. Evaluation

Chromosome 1354679 is shown from the two perspectives: without (left) transitions and with transitions (right side). In both images the prediction quality is shown as 1, which means that this scheme can accurately detect the feature in the environment. Second, it shows the fitness values of 1.66. This fitness value is computed as the prediction quality plus a factor. The factor changes for each class, and goes down that there are a large number of unused states or transitions.

All the images of Prediction Schemes follow this same structure.

Prediction Scheme: Switch to Reinforce

The aim of the Prediction scheme in figure 5–14 is to switch the simulation into the Reinforce process i.e., to enable it to learn.

Table 5–3 Parameters for the reinforce action.

#	Action	Description
1	21	penUp

Table 5–3 provides the input parameters for this prediction scheme.

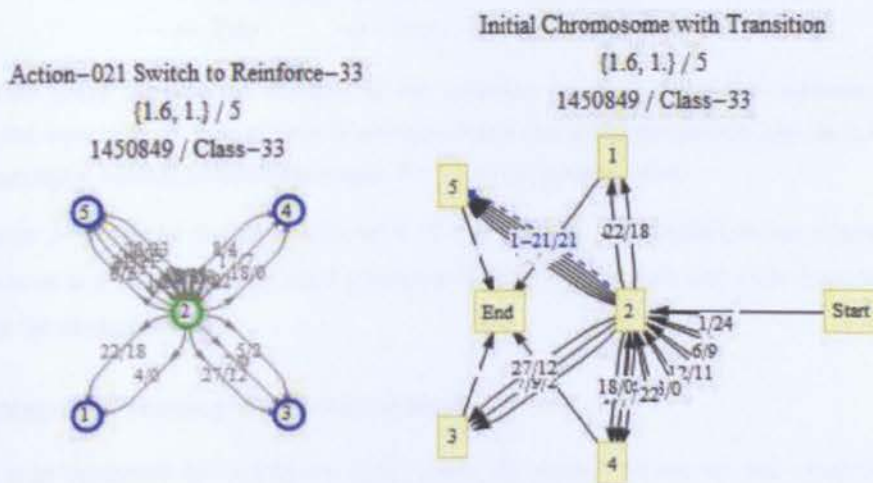


Figure 5–14 Evolution of a prediction scheme that moves the simulation into the reinforce–learn process. The switch is provided as a permanent object invariant. In this example, the scheme correctly generates the required structure with a prediction quality of 1 and a fitness of 1.6.

In this example of chromosome 1450849 in class–33, the simulation has correctly built the scheme to switch to a learning state. The Prediction scheme can be used to read a command from a teacher, and to be used internally to switch states. Thus allowing the student some measure of autonomy to respond, and randomly change its behavior.

Prediction Scheme Learning Rates using Mutation

Figure 5–15 below provides an evaluation of the number of mutations required to detect regularities in the environment using Prediction schemes.

Example Mutation of Prediction Machines to Detect Regularities in the Environment

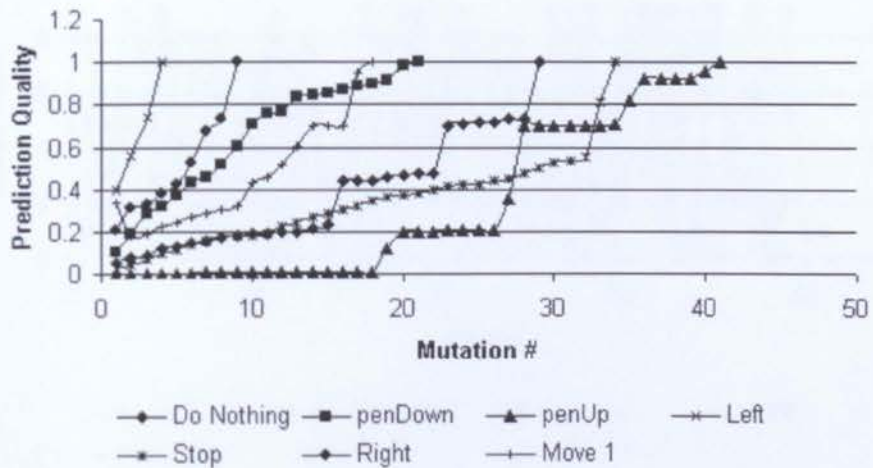


Figure 5–15 This graph provides an example of the mutation rates for Prediction schemes to detection of regularities in the environment. This process is complete when the prediction quality gets to 1. In the examples provided, the maximum number of mutations required is 42 for the penUp action.

What figure 5–15 shows is that finally after 42 generations, the simulation has constructed a set of Prediction Schemes to accurately detect these regularities in the environment, and more importantly, can make these actions in the environment.

5.2.1.2 Acting and Sensing the Environment

The aim is to determine the conditions under which the simulation can act and sense the environment using its Prediction schemes.

Acting in the Environment

Figure 5–16 shows how the regularities in the environment are used by the simulation to act in the environment.

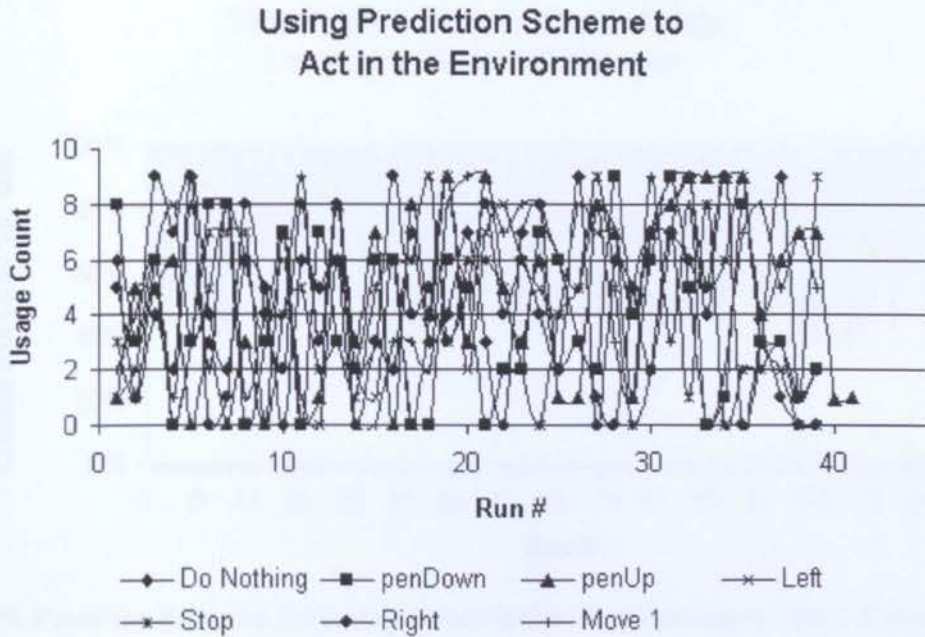


Figure 5–16 This diagram shows the number of times a particular prediction scheme is used over a number of executions. It shows that simulation can use its prediction schemes to act in the environment. There is however, no discernible pattern, these are random actions.

The results in figure 5–16 show that the simulation randomly selects actions and applies them to the environment. There is no discernible pattern to these actions.

Chapter 5. Evaluation

Sensing the Environment

The simulation uses its learnt Prediction schemes to sense the environment as represented in figure 5–17, which presents a sample form 129 separate executions (runs) of the simulation.

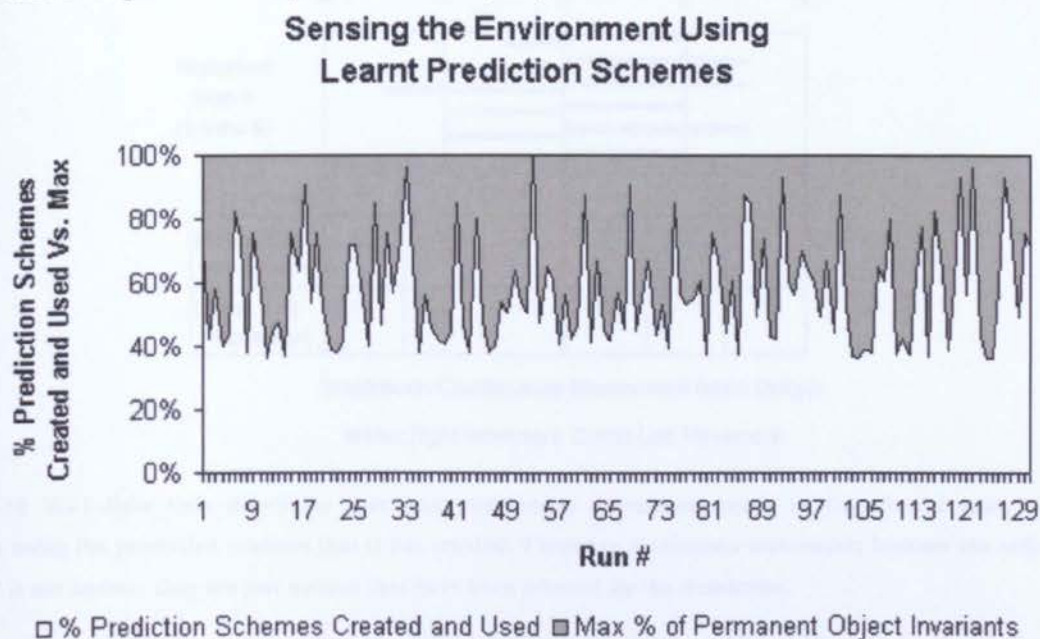


Figure 5–17 Sensing the environment using learn prediction schemes. This diagram compares the number of prediction schemes created and used against the maximum number of permanent object invariants in the EBNF grammar.

These results shows that the simulation can build schemes to sense the environment, but that there are always some *permanent object invariants* that were not created. This implies that the simulation will always need the capability to build prediction schemes.

More importantly, since this research is interested in building internal structures, the variances and nuances of perception such as shadows, perspective and obliqueness have not been covered. For instance, when a teacher draws a mark on a number line, that boundary needs to be determined by the student. This issue has been sidestepped in this research.

Support for the Worked Example

The aim of these tests is to confirm how far the simulation can support the worked example (WE1 – WE5).

WE1–Base Analysis

The following diagram (figure 5–18) describes the range of motions observed through a set of separate executions (runs) of the simulation.

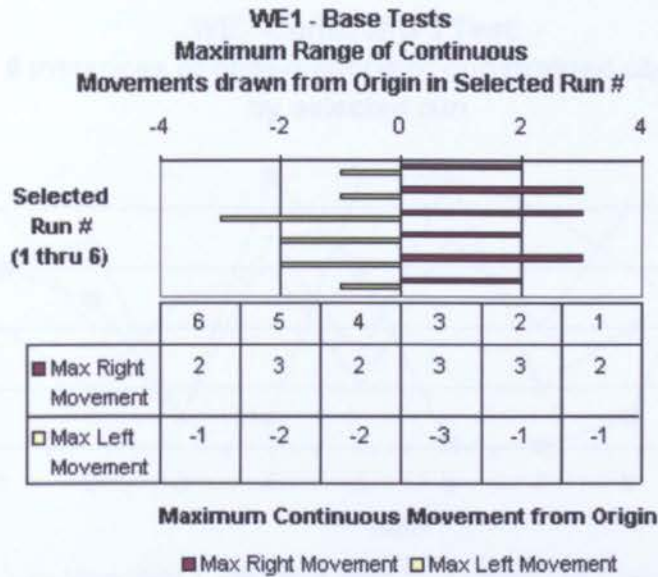


Figure 5-18 WE1-Base tests shows the maximum continuous movements made in six selected runs by the simulation using the prediction schemes that it has created. These are continuous movements because the notion of “number” is not known; they are just actions that have been selected by the simulation.

What figure 5-18 illustrates is that the simulation can randomly select actions to apply, and make movements on a number line. These movements are “continuous” in the sense that they are unconstrained; there is no notion of “number.” These results confirm that the simulation at this level can meet Level’s 1 of the worked example (WE1).

WE2-Constrained Analysis

In figure 5-19, an analysis of the constrained objects tests which counts the number of times that a particular movement occurs is provided.

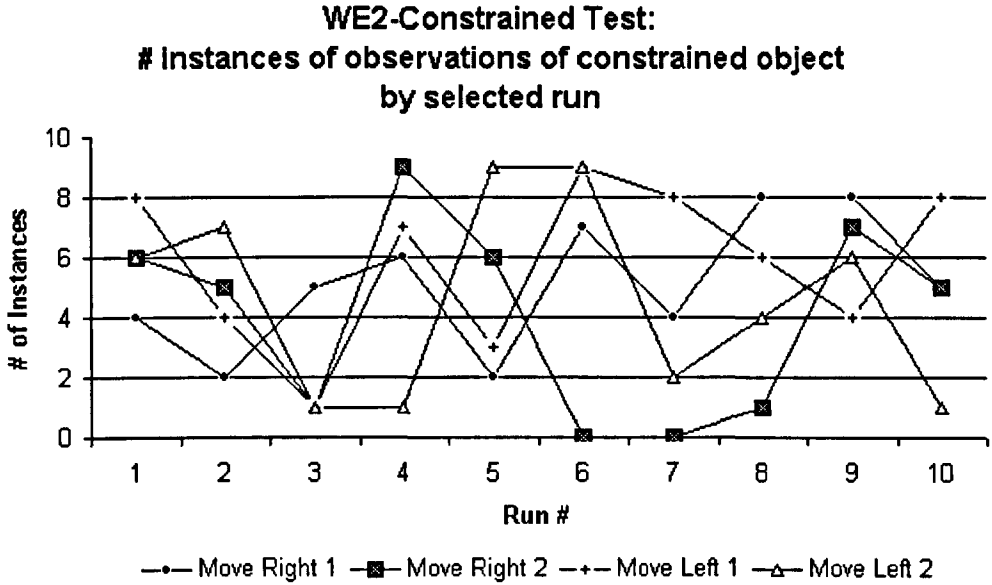


Figure 5–19 This figure illustrates the WE2–Constrained Tests. It compares the number of instances of constrained movements by the selected run. It illustrates a random pattern of movements.

What figure 5–19 confirms is that the simulation can make repeated constrained movements. These movements exhibit no patterns, and are senseless movements along a number line. This implies that only until discrete Prediction schemes can be constructed where they control a number of *permanent object invariants* such as penState, direction, movement and stop, is there the notion of the differentiated object. As such, the simulation at this level can only meet level 2 of the worked example (WE2).

5.2.1.3 Summary

Level–1 starts from the evolution of simple schemes to identify objects in the number line world. Level–1 results confirm that the simulation can act/sense, learn in a number line world. Specifically, the *dialectic system* can detect regularities in the environment of point, line, direction, and penState using a Prediction Scheme. Results show that it can only support tasks WE1 and WE2 in the *worked example*. Observations suggest the emergence of structure in the simulation as a set of Prediction schemes, which are used to act in and sense the environment.

5.2.2 Level 2 – Coordinated Action

The aim of these tests is to determine the conditions under which the simulation can develop coordinated action; for instance, to learn Prediction schemes that have discrete actions to draw number lines and sense number lines in the environment. The secondary aim is to determine if these coordinated actions can resolve the needs of the worked example (WE1 – WE5), the bead problem, count and exhibit emergence of number–sense.

5.2.2.1 Prediction Scheme: Move Right

To be able to create Prediction Schemes, a class definition needs to be provided. In figure 5–20 a definition is provided that will enable the simulation to move right one unit in a coordinated fashion.

```
Options[AddEVClass] = (EvaluationFunction → evalAutomaton, ResponseFunction → automatonResponse, Description → ToString[DateList[]], M → 50, L → 50,
  AvailableStrategy → {BEST, ELITE, FITPROP, RANKBASED, RANDOM, TOURNAMENT}, G → 3, StopAtPerfection → True, Copies → 50, Vanilla → 10);
AddEVClass[inputs_, outputs_, EVType_, alphabet_, opts_] :=
Module[{newClass, evType, evalFct, desc, respFct, EVInputFlag, EVOutputFlag, availableStrategy, g, m, l, stopAtPerfection, copies, vanilla},
  evalFct = EvaluationFunction /. {opts} /. Options[AddEVClass];
  respFct = ResponseFunction /. {opts} /. Options[AddEVClass];
  availableStrategy = AvailableStrategy /. {opts} /. Options[AddEVClass];
  desc = "Sensory Motor Move Right 1 Unit-5": inputs = {1, 5, 6, 10403, 4}; outputs = {}; alphabet = {1, 4, 5, 6, 10403};
  AddEVClass[inputs, outputs, PREDICTION, alphabet, Description → desc, EvaluationFunction → evalAutomaton,
    ResponseFunction → automatonResponse, AvailableStrategy → {BEST, FITPROP}, G → 20, M → 100, L → 100];
```

Figure 5–20 Example chromosome class code used in the evolution of a set of chromosomes to move right 1 unit on a number line.

This definition is typical for Prediction schemes and includes the name of the evaluation functions, “evalAutomaton,” along with the set of selection strategies to use (best and fitness proportionate selection), the number of generations (20), the number of members of the population (100), and the number of children (100). It also provides the alphabet of actions that the Prediction scheme can use, along with the set of inputs that it needs to predict. The set of inputs are the *permanent object invariants* from the EBNF grammar (§ 4.2.1.1). Table 5–4 is a representation of these selected values:

Table 5–4 Parameters for Move Right 1 Unit in the environment.

#	Action	Description
1	1	penDown
2	5	Right
3	6	Move <duration>
4	10403	The duration to Move
5	4	Stop

A set of chromosomes is evolved to predict the input sequence correctly. An example of one such Chromosome (327264), as a Prediction scheme is displayed in figure 5–21.

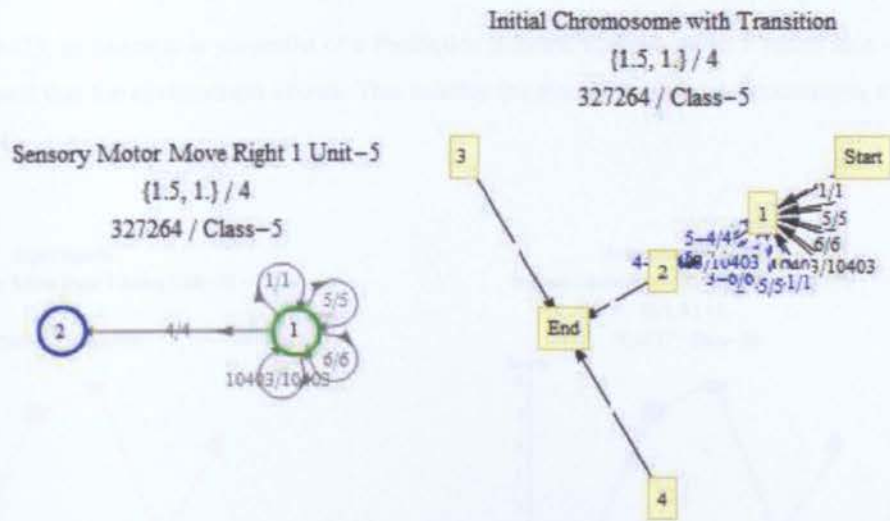


Figure 5-21 The representation of a Prediction scheme, chromosome 327264 in class 5, that can move 1 unit on a number line. The image on the right shows the Prediction schemes with the transition values. The image on the left is the base scheme.

Chromosome 354393 is another example of the class that can perform the draw a number line as well as sensing it (figure 5-22).

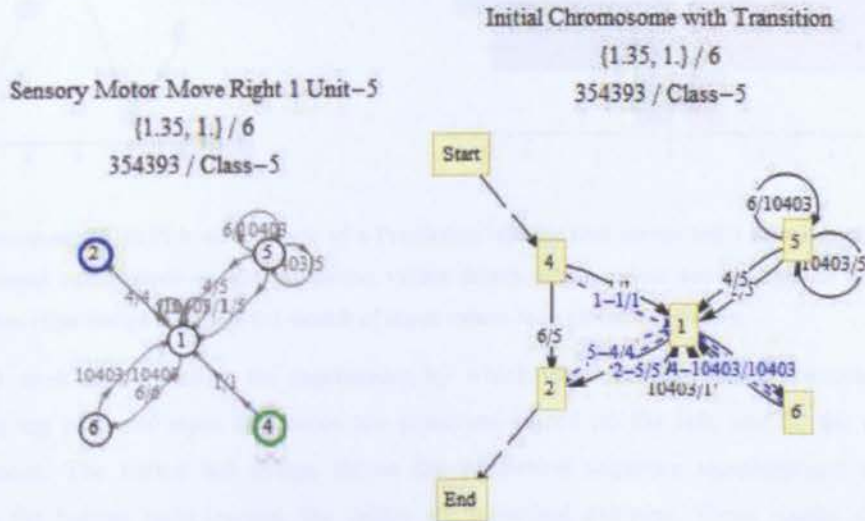


Figure 5-22 An alternate example of a class 5 chromosome. This Prediction scheme can move right 1 unit on the number line, as well as sense it. In the image on the left four states are shown, with state 4 being the start state and state 2 the end state. The same chromosome is shown on the right hand side with named transitions and transition values.

The process of building Prediction schemes to match an input sequence has a mirror in cell biology, where the DNA is read to produce for instance, proteins. The interesting feature of figure 5-22 is that some schemes have a greater fitness 1.35 vs 1.5.

5.2.2.2 Prediction Scheme: Move Right 1 Micro Unit

In figure 5–23, an example is presented of a Prediction Scheme that can move 1 micro unit – the smallest sensory movement that the environment allows. This enables the simulation to make continuous movements in the environment.

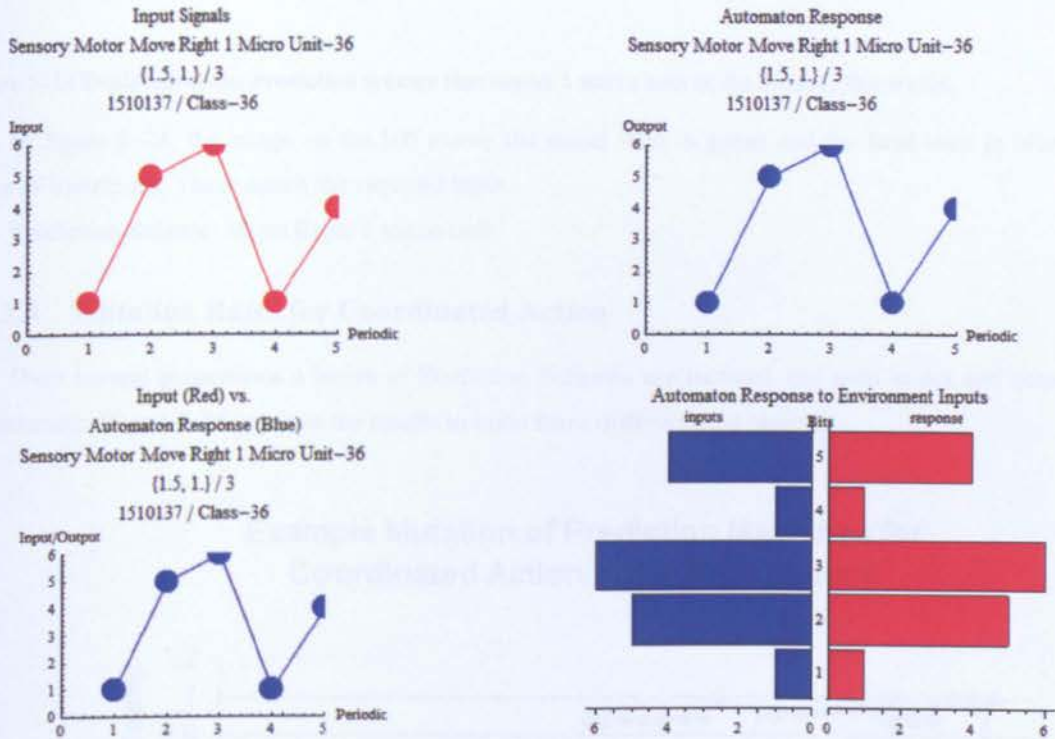


Figure 5–23 Chromosome 1510137 is an example of a Prediction scheme that moves left 1 micro unit. The top row images show the input values (red) vs. the prediction values (blue). These values are overlaid in the bottom left image. In the bottom right image a perfect 1:1 match of input values to prediction is shown.

Figure 5–23 most clearly shows the mechanism by which the Prediction scheme matches the input sequences. In the top row, the input sequences are presented in red on the left, and on the right are the prediction sequences. The bottom left image shows the prediction sequence superimposed on the input sequence, and in the bottom right images, the values are matched pairwise. These results show that the prediction schemes can be evolved correctly. A depiction of the chromosome 1510137 for class 36 is shown in figure 5–24.

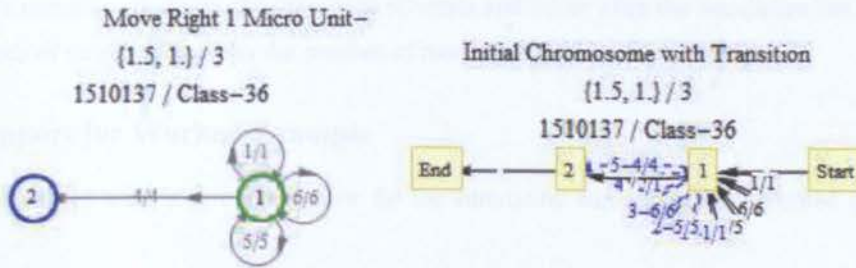


Figure 5-24 Depiction of the Prediction scheme that moves 1 micro unit in the number line world.

In figure 5-24, the image on the left shows the initial state in green and the final state in blue, with a series of transitions. These match the required input.

Prediction Scheme: Move Right 1 micro unit.

5.2.2.3 Mutation Rates for Coordinated Action

Over several generations a series of Prediction Schemes are mutated and used to act and sense in the environment. Figure 5-25 presents the results to build these differentiated objects.

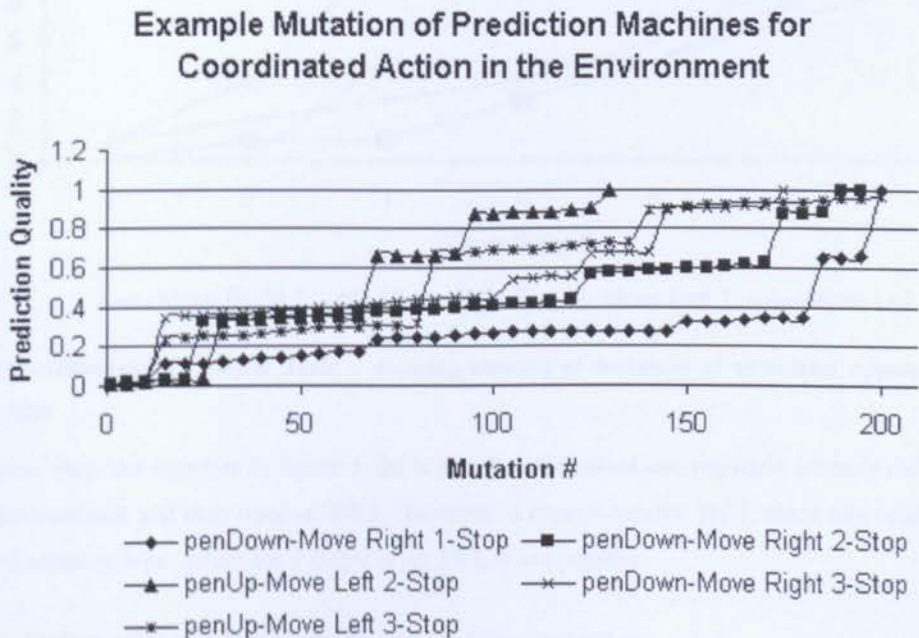


Figure 5-25 Example mutation of Prediction scheme that enables coordinated action and sensing in the environment.

What figure 5-25 shows is that mutation provides a suitable mechanism to combine, in particular sequences, separate *permanent object invariants* together to form discrete objects using coordinated action.

Chapter 5. Evaluation

These are more complex structures than previous schemes and occur after the simulation has been running for extended periods of time, as shown by the number of mutations (202 in this example).

5.2.2.4 Support for Worked Example

The aim of these tests is to confirm how far the simulation can support the worked example (WE1 – WE5).

WE3–Differentiated Object Analysis

The following diagram (figure 5–26) describes the range of differentiated objects that were sensed and acted (drawn) by the simulation over a set of generations.

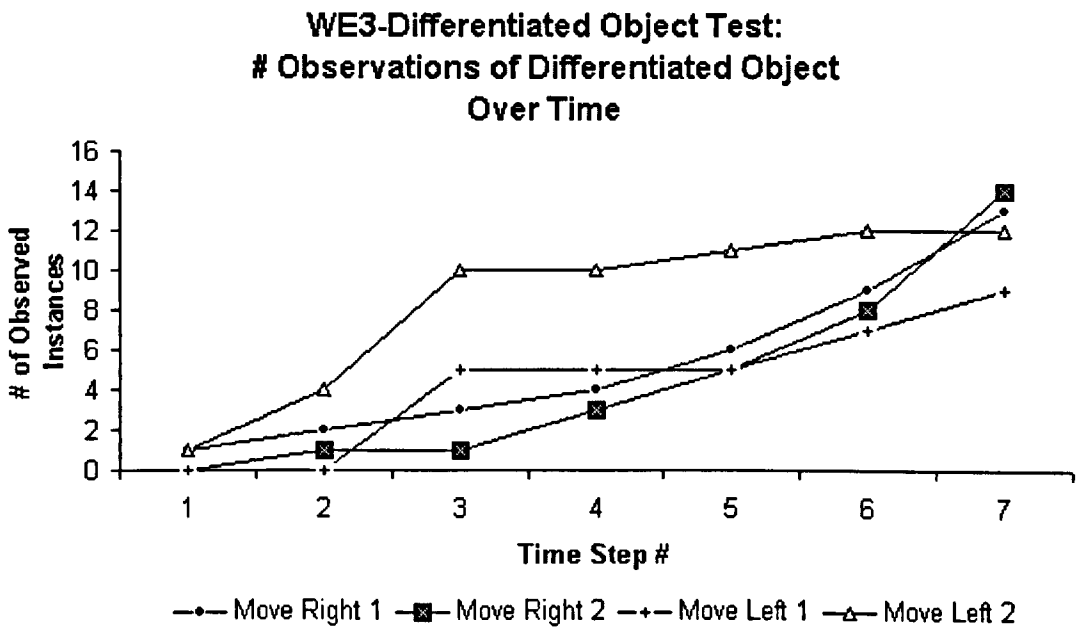


Figure 5–26 WE3–Differentiated Object Tests – showing number of instances of structures appearing in the simulation over time

What is clear from the example in figure 5–26 is that the simulation can regularly identify differentiated objects in the environment and thus resolve WE3. However, it cannot resolve WE4, since this requires that it have hierarchical relationships, which for a single layer FSA is not possible.

5.2.2.5 Resolution of Bead Problem through Conservation

If one considers how the child approaches the bead problem, there is a large degree of dialectical exchange, for instance in the teacher asking the child “which one is greater?” The child in this respect already has the “description” of the concept of greater, but not the mechanism by which to control or act on it. The approach of this study is to look at the control mechanisms that enable the child to experience such constructions as greater, less then etc.

Chapter 5. Evaluation

The bead problem of spatial awareness conflicting with number-sense in early childhood (Piaget, 1952, p36 and Copeland, 1974, p83) and latterly being resolved by the child can be explained using the coordinated action Prediction Schemes that support WE3-Differentiated Objects. The diagram below (figure 5-27), shows the classic dispersal of beads (left image) with an arrangement of number lines segments in the number line world.

A Reformulation of the Bead Problem using Number-Line Segments

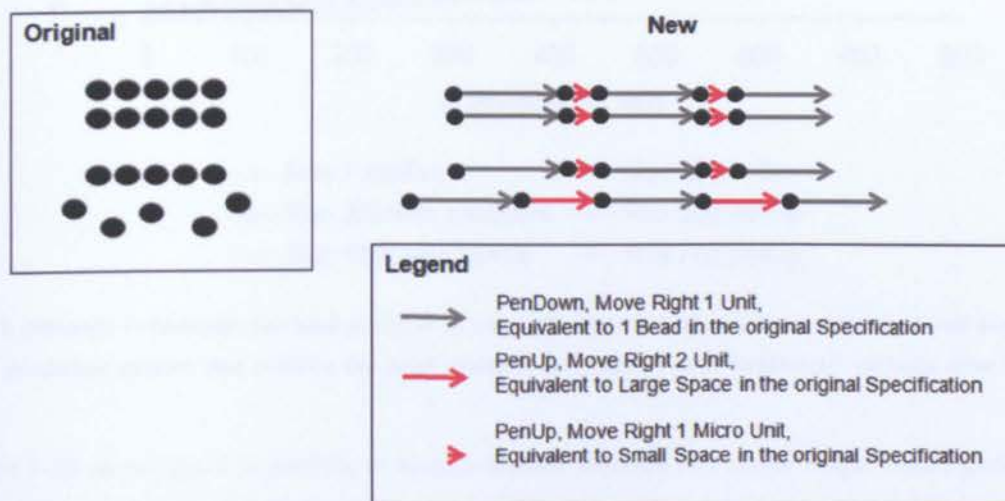


Figure 5-27 A depiction of Piaget's bead problem (Figure 1-1) as number line segments with pen-up in red and pen-down in black. The accompanying text indicates that a more primitive simulation would determine that the spread-out line is bigger, because the penUp movements are greater. After conservation of number, this is no longer an issue: space has no value. This figure therefore presents a solution to the bead problem through conservation.

This research demonstrates that the simulation processes inputs from the number line world as set of command sequences. The first set of beads would be repeating sequences of {Action-001, Action-005, Action-006, 10403, Action-004...}. The second set would be repeating sequences of {Action-001, Action-005, Action-006, 10403, Action-004}, as well as repeating sequences of {Action-002, Action-005, Action-006, 10403, Action-004}. The second set includes sequences that define the spaces and in this number line world, the spaces are actual values: the penState in these instances is penUp {Action-002}. This can be related to the depiction of the bead problem and the associated use of reflection (§ figure 1-1 and figure 2-1).

Figure 5-28 provides an analysis of how long it takes to build these Prediction bead schemes:

Attempts to construct the Bead Problem as a penUp and penDown Prediction Schemes

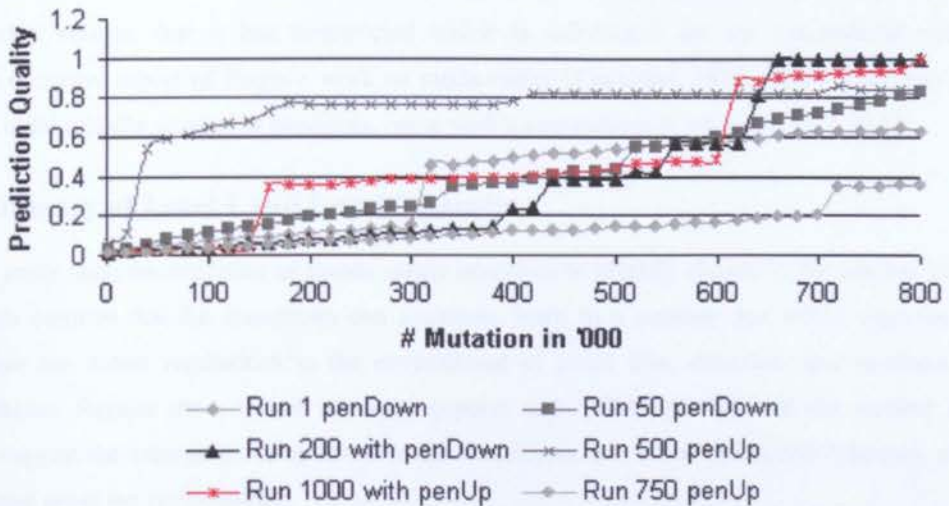


Figure 5–28 Attempts to construct the bead problem as a mutation process, shows that a simulation can accurately produce a prediction scheme that exhibits the bead problem as “penUp” and “penDown” variants after 802,000 mutations.

Figure 5–28 shows that it is possible to build prediction schemes that mirror Piaget’s bead problem. If one were to compare the size of these Prediction schemes, then the one with the red penUp penState would be larger, they are not equivalent schemes.

However, this requires the creation of a mechanism to test for equivalence between prediction schemes. Since Fodor commented: an operator such as “=” or “≡” cannot be used, the operator itself has to be constructed by the simulation beforehand, otherwise this solution falls foul of Fodor’s LP1 argument. In addition, it is believed that this equivalence process requires the interiorization of the external values. The following analysis explains the need for interiorization.

5.2.2.6 The Need for Interiorization to Develop Number–Sense

When does a simulation know the meaning of 1? Given the metaphor of the number line, the key issue is to define the context of the interaction between the simulation and its environment. For the simulation, the context is defined by the relationship of its innate capabilities, its alphabet of changing its pen state (pen–up, pen down), changing direction (left, right), moving duration and stopping its actions as it takes in the external world and by observing changes in the environment.

Using a cognitive model provided by Piaget, if the simulation assimilates to and accommodates actions using this “differentiated object scheme” of what is externally attributed to the value of “1” on a number line, then it has acquired a sensory motor scheme of “1.” This is achieved through the development of a figurative scheme (as a Prediction scheme to hold for instance of the actions {Action–001, Action–005, Action–006, 10403, Action–004}), as well as the operative scheme to process the actions. Further, if the simulation,

Chapter 5. Evaluation

consumes, within a hierarchical structure, this prediction scheme and attributes a binary value of 1, then it is said to have abstracted the notion of the sensory motor scheme into a more complex scheme (more complex, because it is internalized). The interiorization is a random action that the simulation can take, with an internal binary prediction scheme that it has constructed which is substituted for the externalized value. The substitution is a critical aspect of Piaget's work on mathematics (Copeland, 1974, p252). This may begin to explain why a child initially guesses at meanings, using verbal associations (Copeland, 1974, p111).

5.2.2.7 Summary of Level 1 and Level 2 Results

Level-1 starts from the evolution of simple mealy machines to identify objects in the number line world. **Level-1** results confirm that the simulation can act/sense, learn in a number line world. Specifically, the *dialectic system* can detect regularities in the environment of point, line, direction, and penState using a Prediction Scheme. Results show that it can only support task **WE1** and **WE2** in the *worked example*. Observations suggest the emergence of structure in the simulation as a set of Prediction Schemes, which are used to act in and sense the environment.

Level-2 results show that the evolved *dialectical system* exhibits coordinated action using more complex schemes and resolves **WE3**–Differentiated Objects. Using these schemes, it can act and sense the environment and provide an explanation for Piaget's bead problem (conservation of number). However, to produce a mechanism that can support equivalence, as well as the other features of Piagetian mathematics as identified by Furth (Furth, 1969) and Copeland (Copeland, 1974), and this includes the work by Liebeck (Liebeck, 1984), the simulation needs to be able to internalize values (§ 2.3 and C).

5.2.3 Level 3 – Internalized Structure (Interiorization)

The aim of these tests is to determine the conditions under which the simulation can develop internalized structures, for instance, to learn Prediction schemes that can construct digital circuits. The secondary aim is to determine if these coordinated actions can resolve the needs of the worked example (**WE1** – **WE5**), count and exhibit emergence of number–sense.

5.2.3.1 Prediction Scheme: OR Prediction Scheme

Using the class-based evolutionary mechanism described earlier, the example in figure 5–29 of chromosome 626128 shows the result of the mutation of an OR chromosome from the OR class as a Prediction Scheme:

Chapter 5. Evaluation

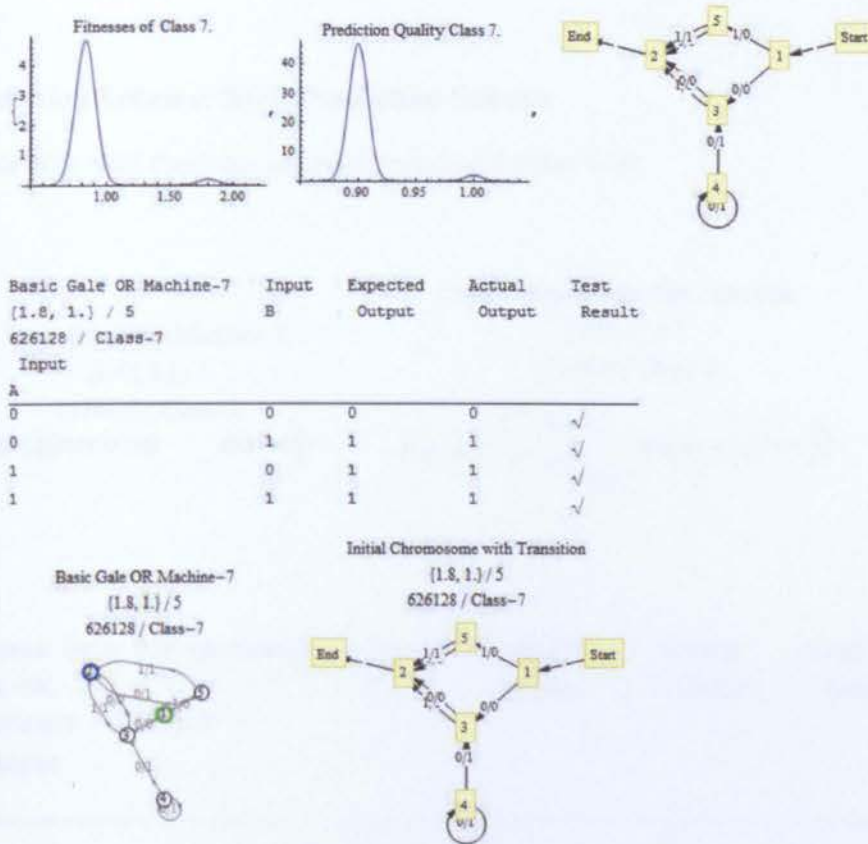


Figure 5-29 A depiction of an evolved prediction scheme that possesses the properties of a logical OR. Chromosome 626128 has a prediction quality of 1, and the results show that it accurately performs its logical or function. What this diagram shows is that mutation can easily produce these logical components.

The interesting feature of these Prediction schemes is that they use the same mutation process as the Prediction schemes that sense and act in the environmental. All of 2-bit propositional logic components can be built in this way.

5.2.3.2 Prediction Scheme: XOR Prediction Scheme

An example of an XOR Prediction scheme is presented in figure 5–30.

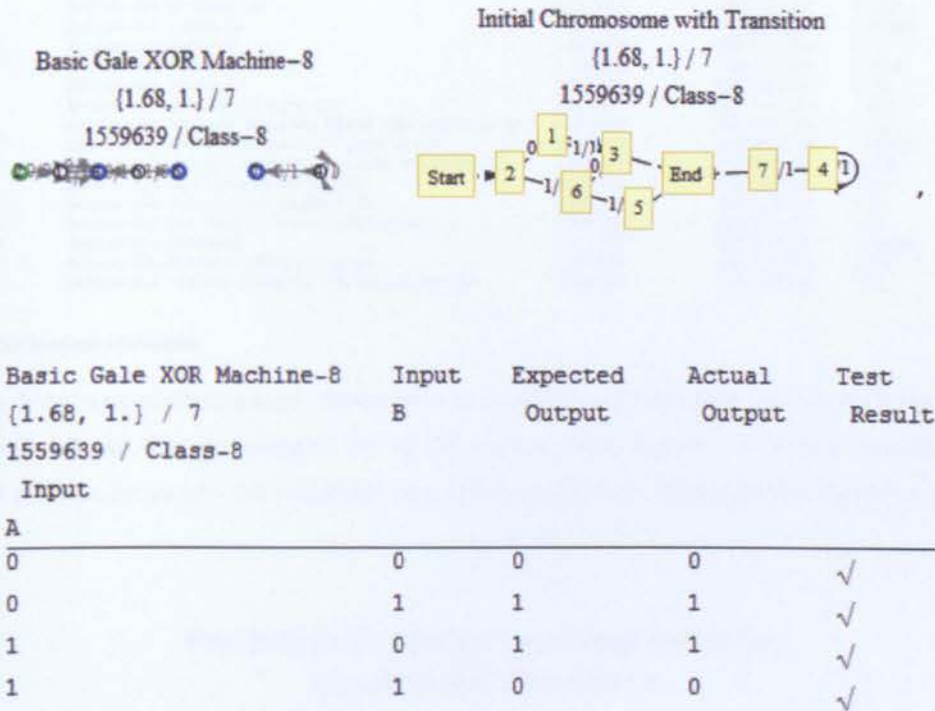


Figure 5–30 An example of an XOR Prediction scheme that has been mutated from the simple mealy scheme. Chromosome 1559639 has a prediction quality of 1 and a fitness of 1.68, which suggests that it is an efficient processor.

The interesting feature of these Prediction Schemes is that they merely process information; they need a mechanism to pass them information, to process.

5.2.3.3 Final Evolution of Classes

The progressive mutation of classes of chromosomes, ultimately lead to the instances of chromosomes that each have a prediction quality of 1. This is described in figure 5–31.

EVClassID	Description	EVChromosomeID	EVClassName	Fitness	Pred-Qual
1.	Buffer Machine-1	300 093	BUFFERNOT	1.55	1.
2.	NOT Machine-2	303 178	BUFFERNOT	1.55	1.
3.	Binary Output 1-3	306 204	BUFFERNOT	1.55	1.
4.	Binary Output 0-4	306 379	BUFFERNOT	1.52	1.
5.	Sensory Motor Move Right 1 Unit-5	327 264	PREDICTION	1.5	1.
6.	Basic Gate AND Machine-6	772 354	BASICGATE	1.67	1.
7.	Basic Gate OR Machine-7	825 480	BASICGATE	1.68	1.
8.	Basic Gate XOR Machine-8	1 559 639	BASICGATE	1.68	1.
9.	Basic Gate NAND Machine-9	1 616 465	BASICGATE	1.32	1.
10.	Basic Gate NOR Machine-10	1 623 088	BASICGATE	1.66	1.
11.	Basic Gate XNOR Machine-11	1 337 873	BASICGATE	1.68	1.
12.	Action-000 Do Nothing-12	1 671 354	PREDICTION	1.5625	1.
13.	Action-001 penDown-13	1 350 070	PREDICTION	1.625	1.
14.	Action-002 penUp-14	1 354 679	PREDICTION	1.66667	1.
15.	Action-003 Left-15	1 356 139	PREDICTION	1.6	1.
16.	Action-004 Stop-16	1 360 594	PREDICTION	1.625	1.
17.	Action-005 Right-17	1 363 337	PREDICTION	1.75	1.
18.	Action-006 Move <duration>-18	1 365 466	PREDICTION	1.625	1.
19.	Action-007 Student Request, Whats the solution-19	1 368 519	PREDICTION	1.625	1.
20.	Action-008 Student Request, I give up-20	1 375 069	PREDICTION	1.66667	1.
21.	Action-009 Student Request, Explain Solution-21	1 379 899	PREDICTION	1.625	1.
22.	Action-010 Add Segment at end-22	1 676 911	PREDICTION	1.75	1.
23.	Action-011 Delete Last Segment-23	1 393 955	PREDICTION	1.75	1.
24.	Action-012 Get Reward (Assimilation)-24	1 395 849	PREDICTION	1.6	1.
25.	Action-013 Reset-25	1 684 712	PREDICTION	1.55556	1.
26.	Action-014 Memorize Numberline-26	1 405 050	PREDICTION	1.75	1.
27.	Action-015 Observe Command (Assimilation)-27	1 655 530	PREDICTION	1.6	1.

Figure 5-31 Evaluation of classes

In figure 5-31, a set of chromosome classes have been selectively evolved to reach a prediction quality of 1. This example shows that the mutation of all 27 classes takes approx 1.6 million mutations. These chromosomes can then be used by the simulation, to perform useful work. This is further explained in figure 5-32 below.

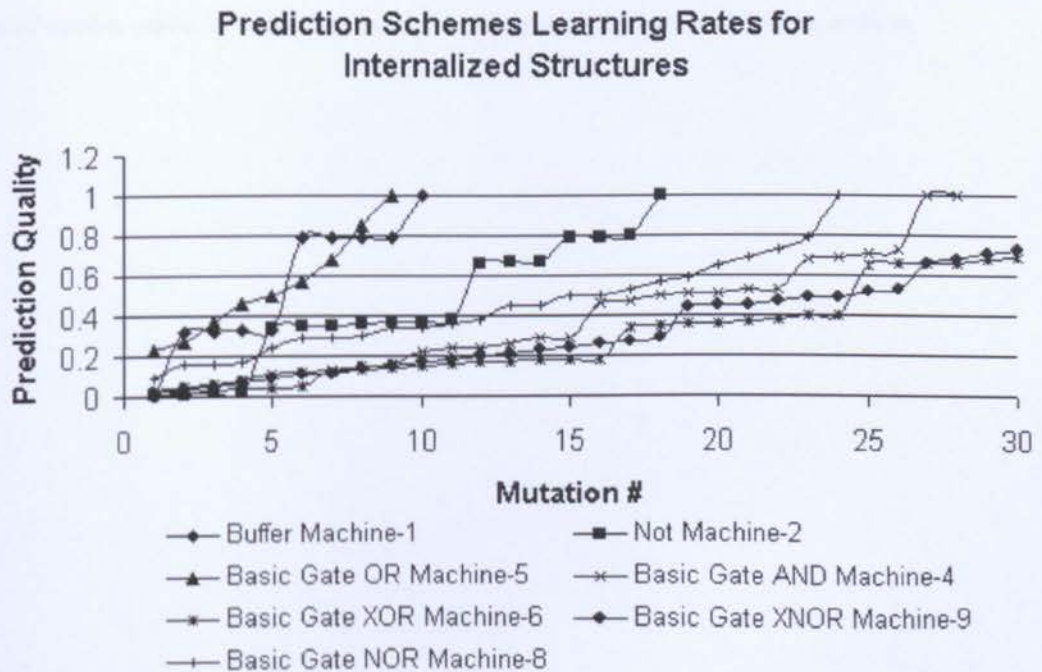


Figure 5-32 Learning rates for Prediction schemes to develop 2 bit propositional logic components.

Chapter 5. Evaluation

What is clear from figure 5–32 is that it is a simple process to mutate the simple mealy machines into the propositional logic components.

5.2.3.4 Support for Worked Example

The creation of Prediction Schemes to internalize structures adds no additional value to resolving the worked example.

5.2.3.5 Summary

Additional evolution of the *dialectical system* at **Level 3**, shows that the *dialectical system* can develop internalized structure as propositional logic components in digital circuits along with dialectical student actions from a teacher. **Level 3** results suggest that it can work autonomously; yet even this is not the appearance of number. Further evolution of the *dialectical system* to **Level 4** is necessary to enable the *dialectical system* to generate hierarchical schemes to reuse existing prediction schemes, and with evolutionary computing, internally construct digital circuits.

5.2.4 Level 4a– Argos Schemes for Symbolic Functions

The aim of these tests is to determine the conditions under which the simulation can develop hierarchical structures (Argos Schemes) as digital circuits that include the prediction schemes. The secondary aim is to determine if these schemes can resolve the needs of the worked example (**WE1 – WE5**), count and exhibit emergence of number–sense. A series of Argos schemes are described in the following sections.

Chapter 5. Evaluation

Argos Scheme: Move Right

Figure 5–33 presents an Argos scheme that was mutated to include a chromosome from class 5, to move 1 unit along a number line.

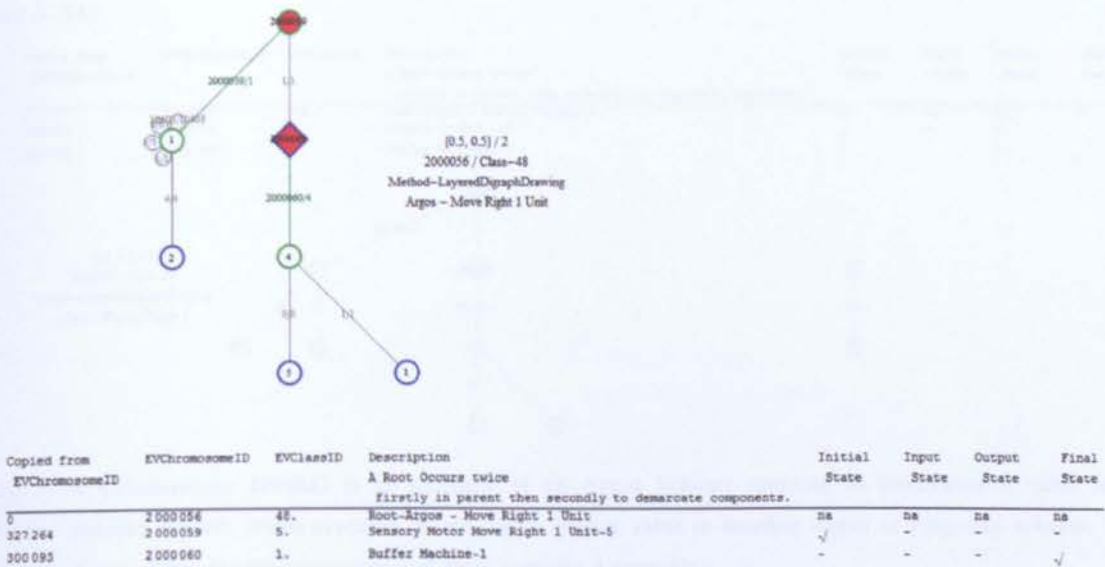


Figure 5–33 The depiction of an Argos Scheme (chromosome 2000056, class 48) that includes a copy of chromosome 327264 and 300093 that allows the Argos scheme to draw a number line segment. The prediction quality and fitness are set on these schemes to be 0.5.

The depiction of Argos Scheme 2000056, presented in figure 5–33 illustrates the execution of the sensory motor Prediction Scheme 2000059 against the environment, drawing a number line. A similar process is followed by all the Argos Schemes that marshals information across their embedded prediction schemes based on the synchronization tables that have themselves been mutated.

5.2.4.1 Argos Scheme: Internalize Values

In a similar vein, Argos schemes can be evolved to make use of prediction schemes that internalize values (figure 5–34).

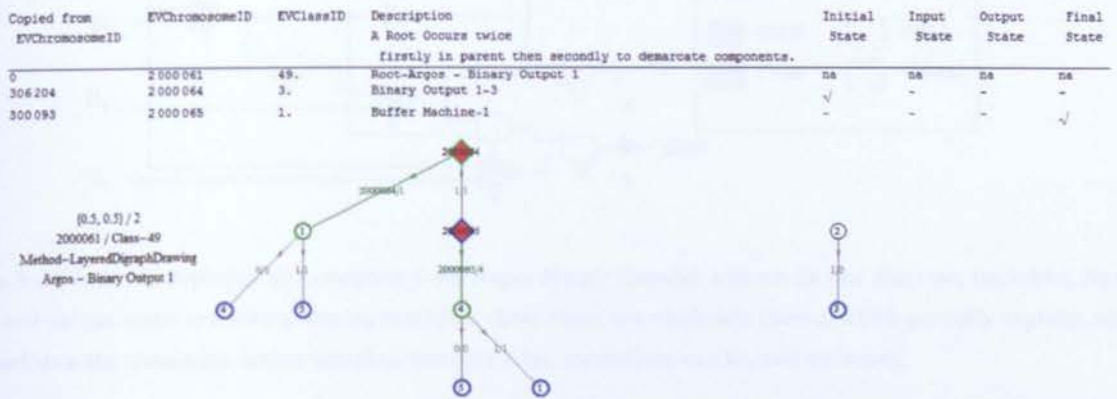


Figure 5–34 Chromosome 2000061 is an example of an Argos Scheme mutated to internalize a value using Prediction Scheme 306204. When executed, it can pass a binary value to another Argos or Piagetian scheme. This low–level scheme allows for the construction of more complex hierarchies.

In the example presented in figure 5–34, the Argos schemes are shown with the initial states, input states, output states and final states. These follow the process defined in the architecture and design (§ 4.2.1.1 analysis of design of HFSA).

What is clear from this example is that Argos schemes provide a mechanism for coordinating processing. Other examples of Argos schemes are included in appendix § D.

Being able to internalize external values will not provide the capability to develop number–sense. There must be a way of encoding the information and then using it. In the design, is discussed the need to generate digital circuits. What follows are example of digital circuits that can be mutated as Argos schemes (§ 4.2 on the design of digital circuits using HFSA).

5.2.4.2 Argos Scheme: Binary Encoder

On their own, sequential gates cannot count sequential values; they need binary coded information. Encoders and composite decoders are required to provide the low–level support for sensing and acting. This implies a progression of schemes from simple to more complex varieties that must persist in memory to allow the simulation to use them. What follows is a description of these binary encoders, using information from the analysis of the architecture and design (§ 4.2).

The binary encoder converts a set of serial inputs, provided by the environment, into a form usable by an Argos scheme. A classic implementation of a 4 bit encoder is given below (figure 5–35), although there are several possible configurations that can produce the same result.

Encoder

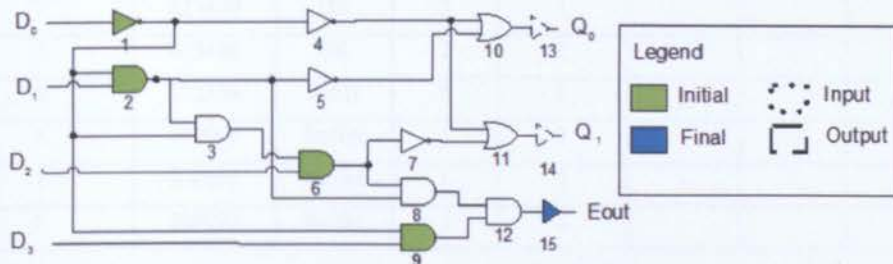


Figure 5–35 A classic depiction of a complete 4-bit Argos Binary Encoder scheme. In this diagram, the initial, final, input and output states are shown. During mutation, these states are randomly chosen, which partially explains why now and then the simulation makes mistakes, but over time, corrections can be, and are made.

In the depiction, states (as Prediction schemes) receive input from the environment at their initial state; produce outputs at the designated states and final values at their final states. They also receive inputs from previous schemes at their input states. The encoder does not make use of these inputs, though other Argos schemes do.

This last transition is critical to processing since it signals to the controlling Argos scheme that it has finished process. To ensure that the contained Prediction schemes are executed, a Prediction Scheme buffer is added as the actual last output and final state. This ensures that the information can be marshaled through the constrained Argos schemes. Thus, the depiction of the encoder is different to the classic interpretation. The following table (table 5–5) describes the encoder in these terms.

Table 5–5 This table describes the states in a Argos binary encoder scheme. Since these Argos schemes are randomly evolved, this is the depiction of viable schemes.

State	EVClassID	Parent ID	Type	Capacity		Initial State	Input State	Output State	Final State
				Input	Output				
1	2.	303178	NOT	1	1	Y	–	–	–
2	6.	772354	AND	2	1	Y	–	–	–
3	6.	772354	AND	2	1	–	–	–	–
4	2.	303178	NOT	1	1	–	–	–	–
5	2.	303178	NOT	1	1	–	–	–	–
6	6.	772354	AND	2	1	Y	–	–	–
7	2.	303178	NOT	1	1	–	–	–	–
8	6.	772354	AND	2	1	–	–	–	–

State	EVClassID	Parent ID	Type	Capacity		Initial State	Input State	Output State	Final State
				Input	Output				
9	6.	772354	AND	2	1	Y	–	–	–
10	7.	825480	OR	2	1	–	–	–	–
11	7.	825480	OR	2	1	–	–	–	–
12	6.	772354	AND	2	1	–	–	–	–
13	1.	300093	Buffer	1	1	–	–	Y	–
14	1.	300093	Buffer	1	1	–	–	Y	–
15	1.	300093	Buffer	1	1	–	–	–	Y

Some conclusions can be drawn: First, the final and output states of an Argos scheme are always single bit buffer machines. Second, the inputs from the environment are always passed to the initial states of an Argos scheme. Finally, through not directly inferable from this example, the outputs from proceeding Argos scheme always pass to their values to input states in subsequent schemes.

The key, as can easily be seen, is that the processing takes place on the transition to the next state. Table 5–6 provides the set of synchronizations of a fully evaluated scheme. When these synchronizations are executed, the value of the Prediction scheme must be saved. In this way each prediction scheme, becomes a push down automata. Memory, when seen in this context, is the storage of the execution of these schemes at particular states. The initial values provided to the environment are consumed by those schemes that are initial states.

Table 5–6 Synchronizations (Transitions) for Argos binary encoder scheme.

From State	To State	Parallel Seq	Sequence Ordinal	Description
1	4	1	1	Initial 1 bit value (D^0), provided by the environment is processed.
1	2	1	2	Initial 1 bit value (D^0), provided by the environment is processed.
1	9	1	3	Initial 1 bit value (D^0), provided by the environment is processed.
1	3	2	4	Initial 1 bit value (D^0), provided by the environment is processed.
2	3	2	5	Initial 1 bit value (D^1) provided by the environment, along with the output from state 1 is processed in the same synchronization step and passed to state 3.
2	8	4	6	Initial 1 bit value (D^1) provided by the environment, along with the output from state 1 is processed in the same synchronization step and passed to state 8.
2	5	4	7	Initial 1 bit value (D^1) provided by the environment, along with the output from state 1 is processed in the same synchronization step and passed to state 5.
3	6	3	8	The output from states 1 and 2 are processed and passed to state 6.
4	10	5	9	The output from state 1 is processed and passed to state 10.

Chapter 5. Evaluation

From State	To State	Parallel Seq	Sequence Ordinal	Description
4	11	5	10	The output from state 1 is processed and passed to state 11.
5	10	5	11	The output from state 2 is processed and passed to state 10.
6	7	4	12	Initial 1 bit value (D^2) provided by the environment, along with the output from state 3 is processed in the same synchronization step and passed to 7.
6	8	4	13	Initial 1 bit value (D^2) provided by the environment, along with the output from state 3 is processed in the same synchronization step and passed to 8.
10	13	6	14	The outputs from state 4 and 5 are processed and sent to state 13, which is an output state (Q^0).
7	11	5	15	The outputs from state 6 is processed and sent to state 11.
11	14	6	16	The outputs from state 4 and state 7 are processed and passed to state 14 and output state (Q^1).
8	12	5	17	The outputs of state 6 and state 2 are processed and passed to state 12.
9	12	5	18	Initial 1 bit value (D^3) provided by the environment, along with the output from state 1 is processed in the same synchronization step and passed to state 12.
12	15	6	19	The outputs of state 8 and 9 are processed together and passed to state 15, producing the final value (Eout)

What is interesting is that to build an Argos scheme of this complexity will take a considerable amount of processing power. Perhaps this could explain why so much of early childhood development has no noticeable output in terms of responses to stimuli. In addition, the parallel sequence for synchronization is always more than the maximum number of transitions of any proceeding transition. It seems likely, that the best way to construct these schemes is to assume that the final state (and there is only ever 1) always has the highest parallel sequence, and then work backwards.

The execution of these synchronizations is the production of the following encoding (table 5-7).

Table 5-7 These are the outputs of Argos binary encoder scheme, which are used in the evaluation function to test the random schemes. When schemes scale up in complexity, the reward is an aspect of the environment, which weights the network in the Argos scheme.

D^3	D^2	D^1	D^0	Q^1	Q^0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1
0	0	0	0	X	X

Chapter 5. Evaluation

To build an encoder of more than 2 bits requires the Argos schemes to be strung together. A visualization of a binary encoder as an Argos scheme is provided in figure 5-36.

{0.5, 0.5} / 2
 1009191 / Class-38
 Method-LayeredDigraphDrawing
 Argos - Encoder

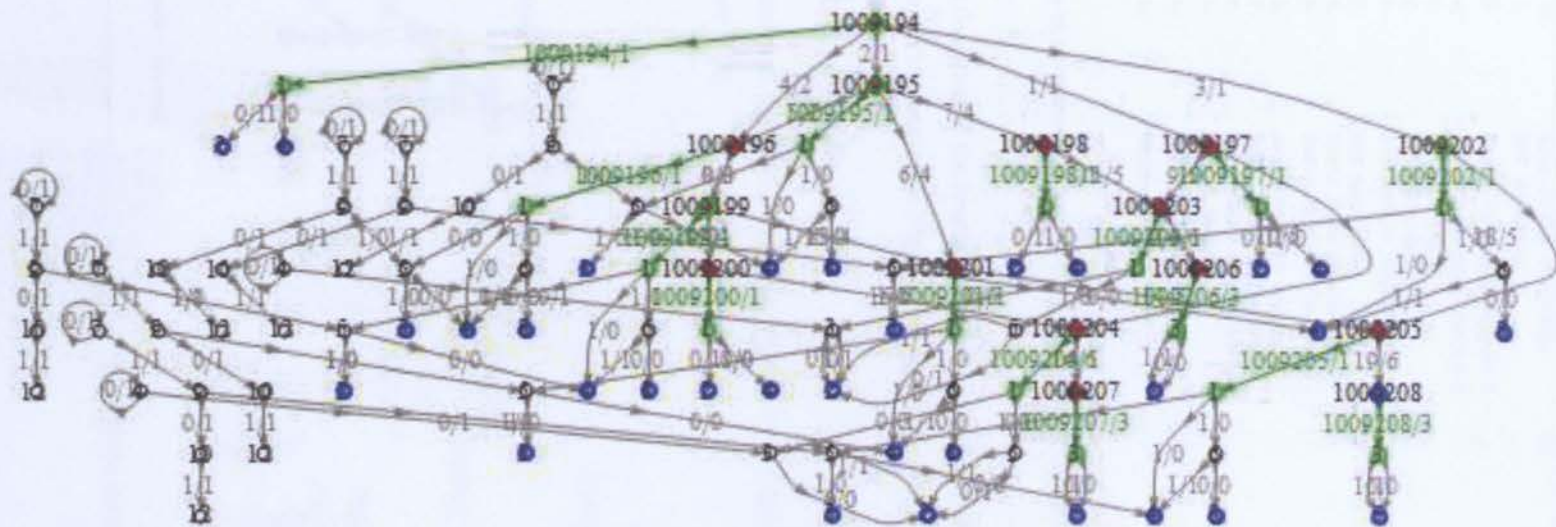


Figure 5-36 A depiction of a binary encoder as Argos schemes that marshal information across prediction schemes to allow them to process the information. The Argos schemes are in red and green. The Prediction schemes are blue and grey. The lines between the points represent the transitions and synchronizations.

The format and function of the Argos Scheme as a binary decoder is described in figure 5-37 below:

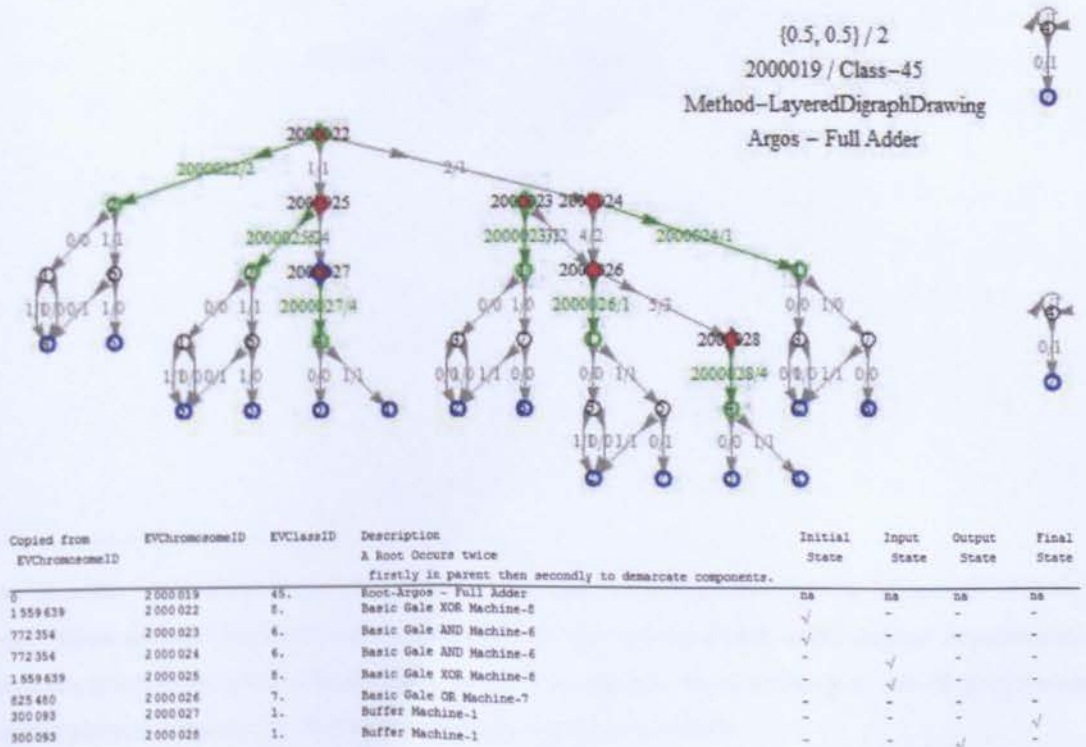
EVChromosomeID	EVClassID	Description	Initial State	Input State	Output State	Final State
1009191	38.	Root-Argos - Encoder	na	na	na	na
1009194	2.	NOT Machine-2	✓	-	-	-
1009195	6.	Basic Gale AND Machine-6	✓	-	-	-
1009196	6.	Basic Gale AND Machine-6	-	-	-	-
1009197	2.	NOT Machine-2	-	-	-	-
1009198	2.	NOT Machine-2	-	-	-	-
1009199	6.	Basic Gale AND Machine-6	✓	-	-	-
1009200	2.	NOT Machine-2	-	-	-	-
1009201	6.	Basic Gale AND Machine-6	-	-	-	-
1009202	6.	Basic Gale AND Machine-6	✓	-	-	-
1009203	7.	Basic Gale OR Machine-7	-	-	-	-
1009204	7.	Basic Gale OR Machine-7	-	-	-	-
1009205	6.	Basic Gale AND Machine-6	-	-	-	-
1009206	1.	Buffer Machine-1	-	-	✓	-
1009207	1.	Buffer Machine-1	-	-	✓	-
1009208	1.	Buffer Machine-1	-	-	-	✓

Figure 5-37 The composition of chromosomes that make use up an Argos binary encoder scheme as chromosome 1009191

What is clear from the depiction of the binary encoder is that the architecture and design approach work. It provides a simpler mechanism to mutation of Prediction schemes to evolve Argos Schemes.

5.2.4.3 Argos Scheme: Full-Adder

Using the same mechanism as Encoders, an Argos scheme for a single full adder can be constructed. A depiction is provided in figure 5-38.



Chapter 5. Evaluation

Figure 5–38 The depiction of a full adder as an Argos scheme as chromosome 2000019, class 45. It includes a set Prediction schemes and Argos schemes. This is only a single full–adder. It can be connected in series to other full–adders using the Argos mutation process.

In this, the Argos scheme, chromosome⁷⁷ 2000019, EVClass 45, is composed of a set of schemes. The interesting point is that these schemes are not perfect. They have outliers, as in the far right of figure 5–38 that are not part of the main thread of processing, but at a future mutation point, may become significant. The Argos Scheme synchronizes the processing of inputs from the environment across the contained prediction schemes using the information in its synchronization table, which allows it to count 2 bits. It works as a digital circuit (§ D.5 for a fuller description of a full adder).

The second interesting point is that the Argos schemes must remain active over a period of time, such that the simulation can count a range of values provided through the interaction with the environment. In supporting the work of Piaget, Pascual–Leone imagines a model of processing where schemes can remain active, in equilibration, until the system itself is forced into a process of disequilibration and other schemes become dominant. For instance, if the system is counting and it cannot count anymore because it has run out of processing units, it regains equilibrium by adding more full–adders and continues counting until either the input from the environment stops and it finishes or some other scheme becomes active and dominant e.g., boredom. This processing is essential to the overall utility and warranty of the system. An alternate depiction of an Argos scheme full adder is described in figure 5–39.

⁷⁷ As an aside, in this architecture, the evolution of chromosomes occurs within the lifetime of Student, in this respect “chromosomes” are more akin to “neurons” and this is a throwback to the original implementation from which this solution has grown (Jacob, 2001, p213). Thus the process of evolving a class of chromosomes, is the learning process employed by this system and is the resolution of LP5.

{0.5, 0.5} / 2
 1009209 / Class-39
 Method-RadialDrawing
 Argos - Full Adder

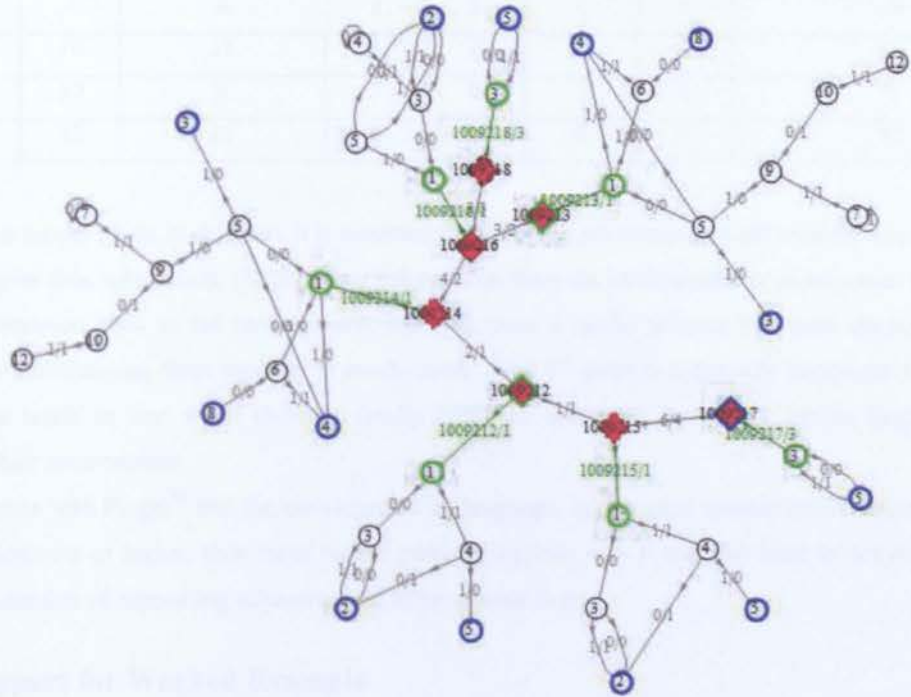


Figure 5-39 An alternate depiction of a full adder as a set of Argos schemes (red node) and prediction schemes (blue and green nodes). The Argos schemes marshal information across their prediction scheme network using a series of numbered transitions in grey. It is an example of a binary executable HFSA machine that is mutated to interact with the environment.

It is through these highly structured combinations of full adders and ripple carry adders that the simulation has the propensity to fully “count” in a similar way to a child. For instance, an improperly built ripple carry adder will sometimes get the right values and sometimes not. There is a clear correspondence with Piaget’s observations of children (Copeland, 1974).

5.2.4.4 Argos Scheme: Full-Adder Evaluation of Resource Usage

Assuming a resource count of 1 for each random mutation, to achieve the desired configuration of Argos encoders, full adders, full subtractors and decoders requires the following resource demand⁷⁸ (table 5-8):

⁷⁸ It would be an interesting project to construct values for M-Demand and M-Capacity based on the machine size, and build time, to see how they compare to the published results of Pascual-Leone.

Table 5–8 Summary of resource usage in Argos schemes

Type of Argos Scheme	# States	# Synchronization	# Initial States	# Input States	# Output States	# Final States	Measure of Resource Cost / Complexity Value
Encoder	15	19	4	0	2	1	41
Full Adder	7	6	2	2	1	1	19
Full Subtractor	10	11	4	1	2	1	29
Decoder	10	8	5	0	3	1	9
Evaluation	12	11	8	0	0	1	32

If a similar model exists in children, it is assumed that (i) there are computational reasons why adding, for children is simpler than subtraction, (ii) decoding information from the environment is much easier than adding structured information back to the environment, and (iii) once a useful scheme has been constructed, then creating copies and chaining them together is much easier. This 3rd point is especially important. Observation and experience teach us that when children finally begin to count on their own, he/she begin to count everything in their environment.

If one agrees with Piaget⁷⁹ that the development of language, in this case mathematical language, begins from the development of logics, then these results partially explain why it takes so long to develop number-sense. A great number of supporting schemes need to be created first.

5.2.4.5 Support for Worked Example

The aim of these tests is to confirm how far the simulation can support the worked example (WE1 – WE5).

⁷⁹ “Before the operations are formulated in language, there is a kind of logic of action coordination in the sensory-motor stage there is a sort of “generalizable action that prefigures classes and relations,” examples include seriation of blocks of decreasing size or number on a number line, but these structures are not capable of being represented, until processing has reached symbolic functioning. These structures (schemes) are pre-configurations of the later notions of conservation and reversibility. These operations are independent of language during the *concrete operations stage* and are tied as actions related to objects (Piaget, 1963 cited in Furth, 1969, p125). For Piaget, “Logic is but the formalization of an equilibrated structure” (Furth, 1969, p216). With logical necessity being in the structure of adaptation that has been developed and where this logical necessity is a logical coherence, an implication that one part stands and falls with the whole (Furth, 1969, p233).

Chapter 5. Evaluation

WE4-Hierarchical Units Analysis

To support hierarchical units, the simulation needs to be able to internalize external values and then manipulate them through encoding, counting, decoding their values. It is assumed that since hierarchical Argos schemes can be mutated, and can process these binary values, that a future implementation would support **WE4**. It has been shown that these Argos schemes can process hierarchical information across their embedded network to encode, decode, count and encode.

WE5-General Relationships Analysis

Since the assumption is that hierarchical unit analysis (HE4) is a prerequisite for general relationships (WE5), the simulation at this point cannot support WE5.

5.2.4.6 Summary

Using the mutation process discussed in the architecture and design (§ 4.2.1.1), it has been shown that Argos schemes can internalize external values, and be mutated into encoders, full-adders, subtractors and decoders. This, it has been demonstrated that the propensity to develop digital circuits in a form similar to evolvable hardware (Greenwood and Tyrrell, 2006, p12). In the examples presented, the Prediction schemes are the propositional components and the Argos schemes are the “predicate.” In this way, they follow the Esterel like formalism (Maraninchi and Remond, 2001, p65) and sidestep LP1.

That the simulation has the propensity to resolve **WE4** has been demonstrated, however, to develop number-sense in a Piagetian model requires that the processes of assimilation and accommodation be blended into the Argos scheme process. This is covered in Level-4 Piagetian Schemes.

5.2.5 Level 4b- Piagetian Schemes for Symbolic Functions

The aim of these tests is to determine the conditions under which the simulation can develop hierarchical structures (Piagetian schemes in a Piagetian Model) and include the Argos Schemes as the marshaling component of digital circuits with the Prediction schemes as the propositional logic components of these circuits. These Prediction schemes also form the sensory motor capabilities to sense and act in the environment.

The secondary aim is to determine if these Piagetian schemes can resolve the needs of the worked example (**WE1 – WE5**), count and exhibit emergence of number-sense.

5.2.5.1 Piagetian Scheme: Counting as a Worked Example

In this example, there is adherence to Piaget’s knowing circle (§ 2.3.5 on Knowing Circle), which assumes a process of accommodation then assimilation through every interaction.

The initial image (figure 5-40) provides a legend to the process in figure 5-41. This legend is used in the other presented examples.

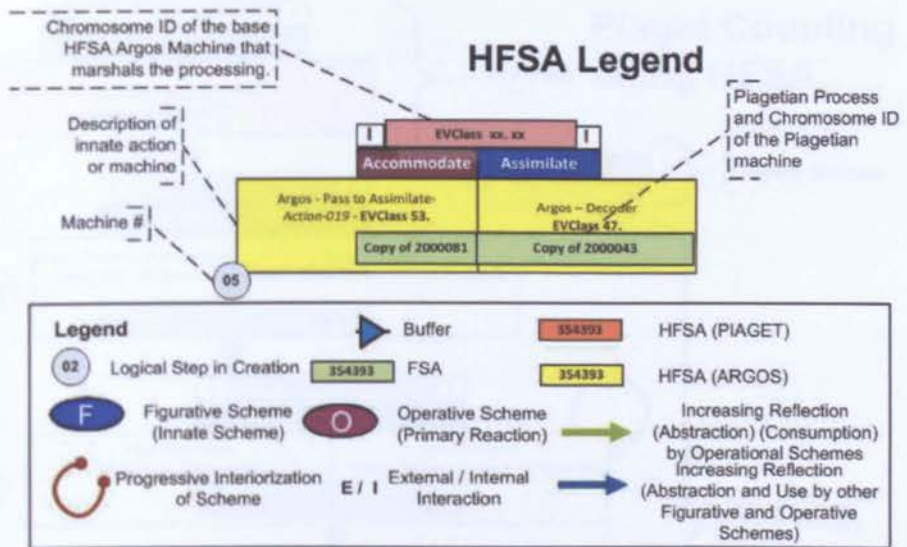


Figure 5-40 This is the legend for examples of Piagetian Schemes implementations.

Figure 5-40 explains the processing of this Piagetian Counting Scheme, which would be mutated in the similar way to the Argos schemes using the same synchronization table structures.

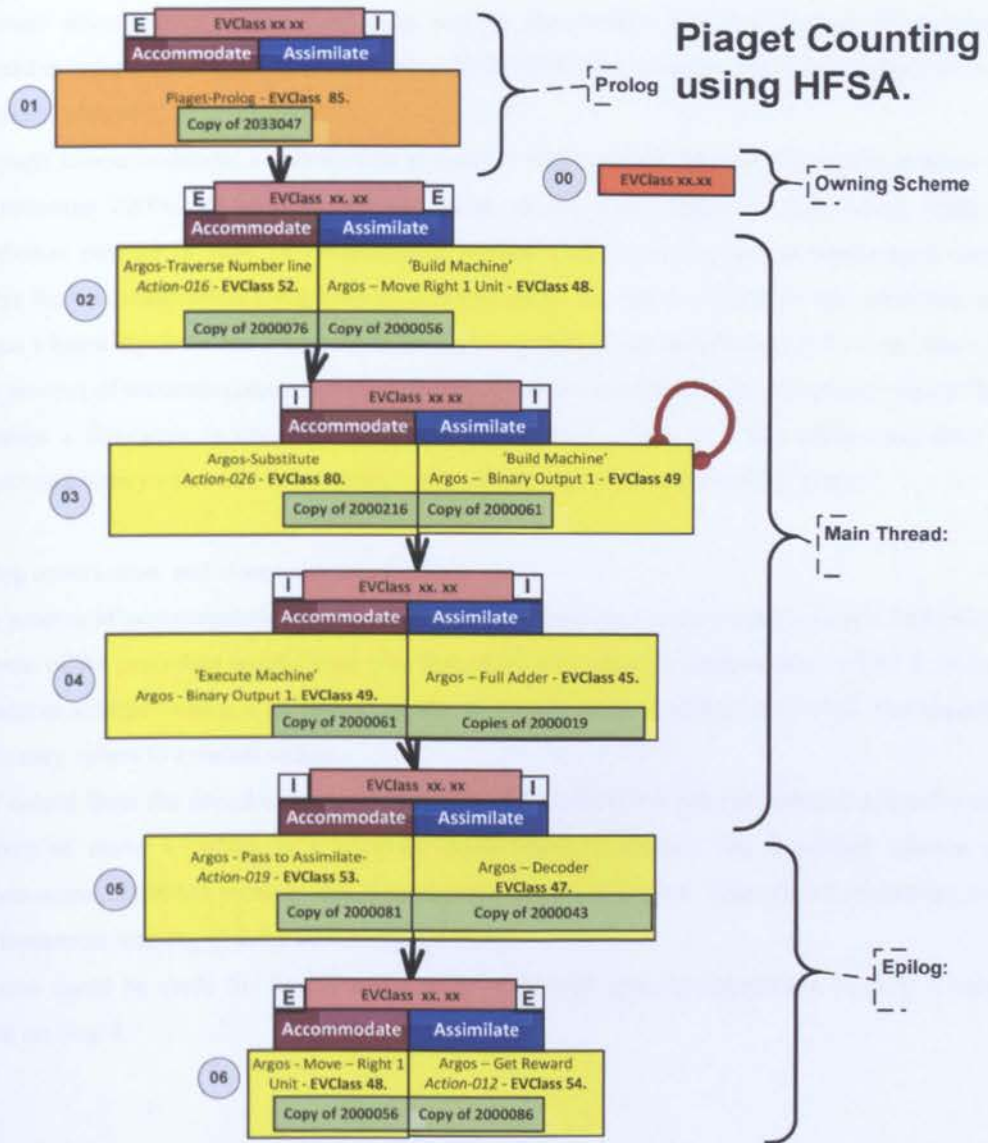


Figure 5-41 A worked example of counting using Piagetian schemes showing assimilation, accommodation and the embedding of Argos schemes which enable interaction with the environment. The interesting feature is the alternate processing of accommodation and assimilation and the use of named chromosome – the inclusion of new copies of chromosomes is seen as critical to the development of a simulation that can adapt to the environment through mutation of its internal structure.

The process of counting depicted in figure 5-41 using Piagetian Schemes is described below:

Prolog

- 1) This is the prolog scheme. It handles the initial invocation and is an existing Piagetian scheme (a copy of chromosome 2033047).

Main Thread

The main thread executes until the end of the environment input, or an interrupt occurs. In this example, there is only 1 item on the number line, so, it counts as only 1 value.

Chapter 5. Evaluation

- 2) Through accommodation, the simulation acts on the number line and through assimilation builds a figurative scheme (a copy of chromosome 20000056) that predicts the environment, in this case an external value of 1.
- 3) Through accommodation, a substitution is used to replace the “Move-1” figurative scheme (a copy of chromosome 20000216) with an internal value, in this case, binary 1. This binary value is itself a Prediction scheme (a copy of chromosome 2000061). It is this process of progressive interiorization, where there is substitution along with an internalized action that is critical for the simulation to adhere to Piaget’s knowing circle and switch between accommodation and assimilation (§ 2.3.5 on Knowing Circle).
- 4) The process of accommodation then executes an operative scheme (a copy of chromosome 20000061) that contains a figurative (a copy of chromosome 2000019), which is a full-adder, and then passes the substituted binary value of 1. The binary value of 1 is then counted by the full adder.

Epilog

The epilog occurs once, and closes any processing.

- 5) The process of accommodation using an operative scheme (as a copy of chromosome 2000081) passes the outputs of the preceding assimilation (the full adder as a copy of chromosome 2000019) to its contained figurative scheme (which is an Argos decode, as a copy of chromosome 2000043). The decoder converts the binary values to external values.
- 6) The output from the decoder (a copy of chromosome 200043) is the externalized action by an operative scheme of move 1 (which is a copy of chromosome 2000056). The figurative scheme (a copy of chromosome 2000086) receives the reward from the environment. This reward processing is similar to reinforcement learning (§ 2.3.5 on Knowing Circle).

A case could be made for Step 3 and 4 to be combined with Accommodate on Step 3, connecting to assimilate on Step 4.

UML Sequence Diagram for Counting

Figure 5-42, is a visualization of the sequence of processing for the Piagetian Scheme:

Execution of HFSA (Schemes) in the process of counting Using Act & Sense, Reinforce

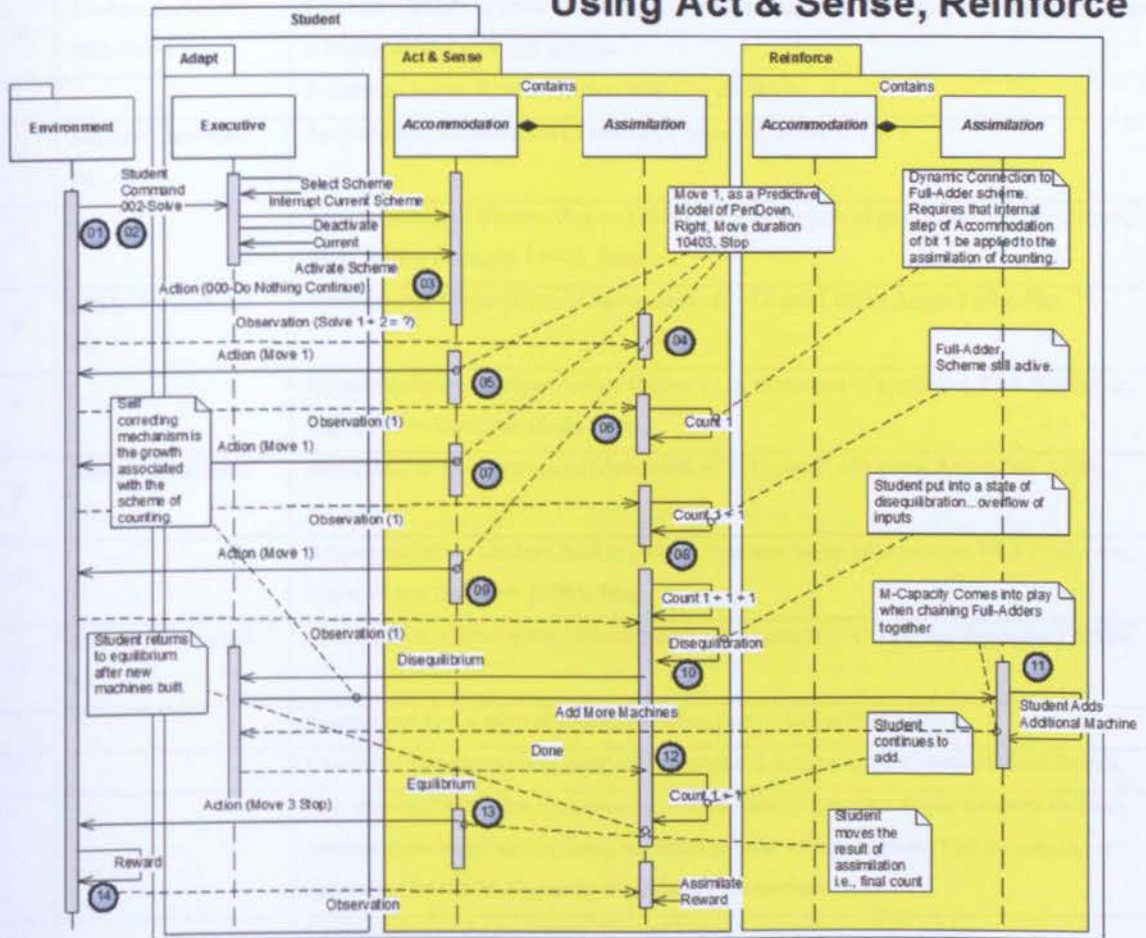


Figure 5-42 A UML sequence diagram of the Piagetian counting scheme. It shows how disequilibrium and the stabilizing processes of equilibration occur to enable the system to count larger values. This is important because it shows how the system can regain stability using the internally constructed schemes and confirms Piaget’s approach to adaptation.

To adhere to the principle of cognizant failure the Piagetian schemes themselves have to adjust to errors and interrupts. The UML sequence diagram shows this process as an interrupt, which causes the simulation to mutate, adding additional Argos adder schemes in order to process the input.

The steps in figure 5-42 are described in table 5-9.

Table 5-9 The steps in the UML sequence for counting.

Step	Environment	Student
1.	Student command 001–Wake–Up	Executive of Student Wakes Up and Student responds ok (Action 000). This ok scheme is now the current active scheme.
2	Student command 002–Solve ?	Executive selects scheme to respond, interrupts currently executing scheme and deactivates it then activates selected scheme.
3		Accommodation: Student Action 000 (Do nothing).
4	Student Command ok	Assimilation: Observation of student Command Solve $1+1 = ?$
5		Accommodation: Student Action (Move 1) as execution of predictive FSA PenDown, Right, Move Duration 10403, Stop.
6	Student Command ok	Assimilation: Observation of Movement of 1, Count 1 using Argos Full Adder. ⁸⁰
7		Accommodation: Student Action (Move 1) as execution of predictive FSA PenDown, Right, Move Duration 10403, Stop.
8	Student Command ok	Assimilation: Observation of Movement of 1, Count (1+1) using Argos Full Adder.
9		Accommodation: Student Action (Move 1) as execution of predictive FSA PenDown, Right, Move Duration 10403, Stop.
10	Student Command ok	Assimilation: Observation of Movement of 1, Count (1+1+1) using Argos Full Adder.
		Student put into a state of disequilibrium (overflow on Full Adder).
11		Executive Interrupts processing and attempts to put student into state of Equilibrium.
		Executive calls reinforce processing to dynamically generate Argos schemes that can process more input and in doing so return to state of equilibrium. This processing is dependent on the M–Capacity available to the student.
12		Continuation of Assimilation, Count (1+1+1) using Argos Full Adder.
13		Accommodation: Student Action (Move 3) as execution of predictive FSA PenDown, Right, Move Duration 10403, Stop 3 times.
14	Teacher Marks Student response	Assimilation of reward and strengthening of Argos Scheme Interaction stored in memory.

⁸⁰ Like all counting step, this step assumes that a prior step has evolved a Figurative Scheme (a Piagetian Scheme as a HFSA) that has converted an observation of unit movement of 1 unit, to a internal action of binary value of 1. Initially this binary value can be seriated and eventually is reused in the counting process. Without this, the process of counting is meaningless. The development of an internal binary of 1 for an external movement of 1 underpins the development of seriation. This implies that the Piagetian scheme of counting requires the additional levels of external conversion of representation.

Chapter 5. Evaluation

The diagram shows how the self correcting mechanism of the Piagetian scheme of counting (currently a full adder) is re-executed to bring the system back into equilibrium through the addition of a new full-adder. The presupposition that the addition of a new full-adder is the self-correcting mechanism is explained in terms of the initial creation of the full-adder. Initially, the student cannot count. In response to this, the reinforce process is used to generate Piagetian schemes of assimilation and accommodation. The reward for assembling such a scheme is provided by the Teacher. The utility of this process is the reuse of this scheme to enable the system to count ever-larger numbers. These steps show that the design can resolve the counting issue using a Piagetian model albeit, with the Student tracing out all the requests from the student command. This also presupposes that the Student has already constructed schemes that can relate the actions on the number line. However, this process does not make use of the predictive model. A worked example of the predictive model is included (§ appendix D.6).

Piagetian Scheme: Prolog

Because of the two-fold process of assimilation and accommodation, each Piagetian scheme is a complete process, which can be executed separately. The prolog and epilog processing in the design is one such separation. A depiction and analysis of its process is provided in figure 5-43.

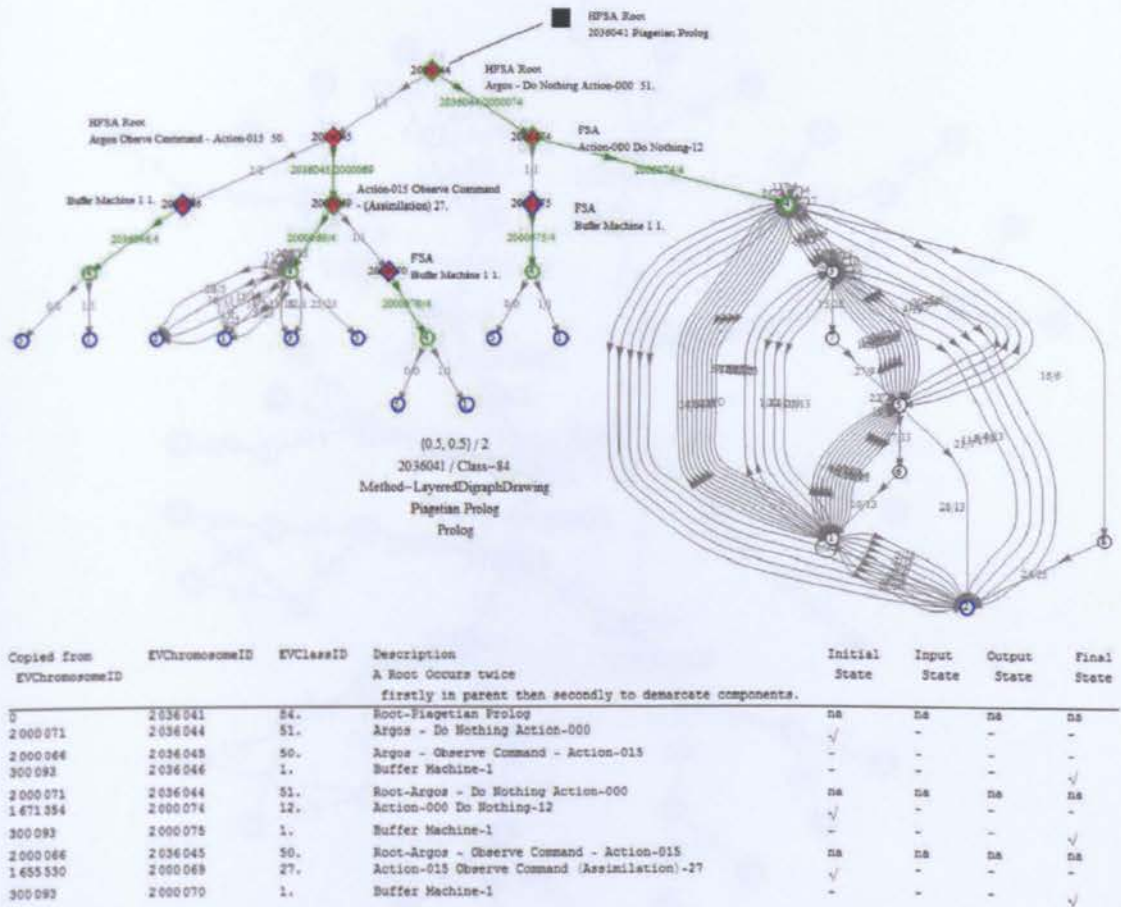


Figure 5-43 A depiction of a Piagetian prolog scheme with Argos schemes (red nodes) and contained Prediction schemes (blue and green nodes) that form part of the basic process of assimilation and accommodation.

The interesting point is that the prolog processing is a reasonably self-contained process that works within the Argos reactive system framework. It is imagined to be included into each Piagetian Scheme and is an example of the hierarchical nature of the Piagetian Schemes. The epilog processing, should work in a similar way.

Piagetian Scheme: Counting Example

Figure 5-44 is a depiction of a Piagetian Scheme that attempts to count using a network of Argos and Piagetian Schemes.

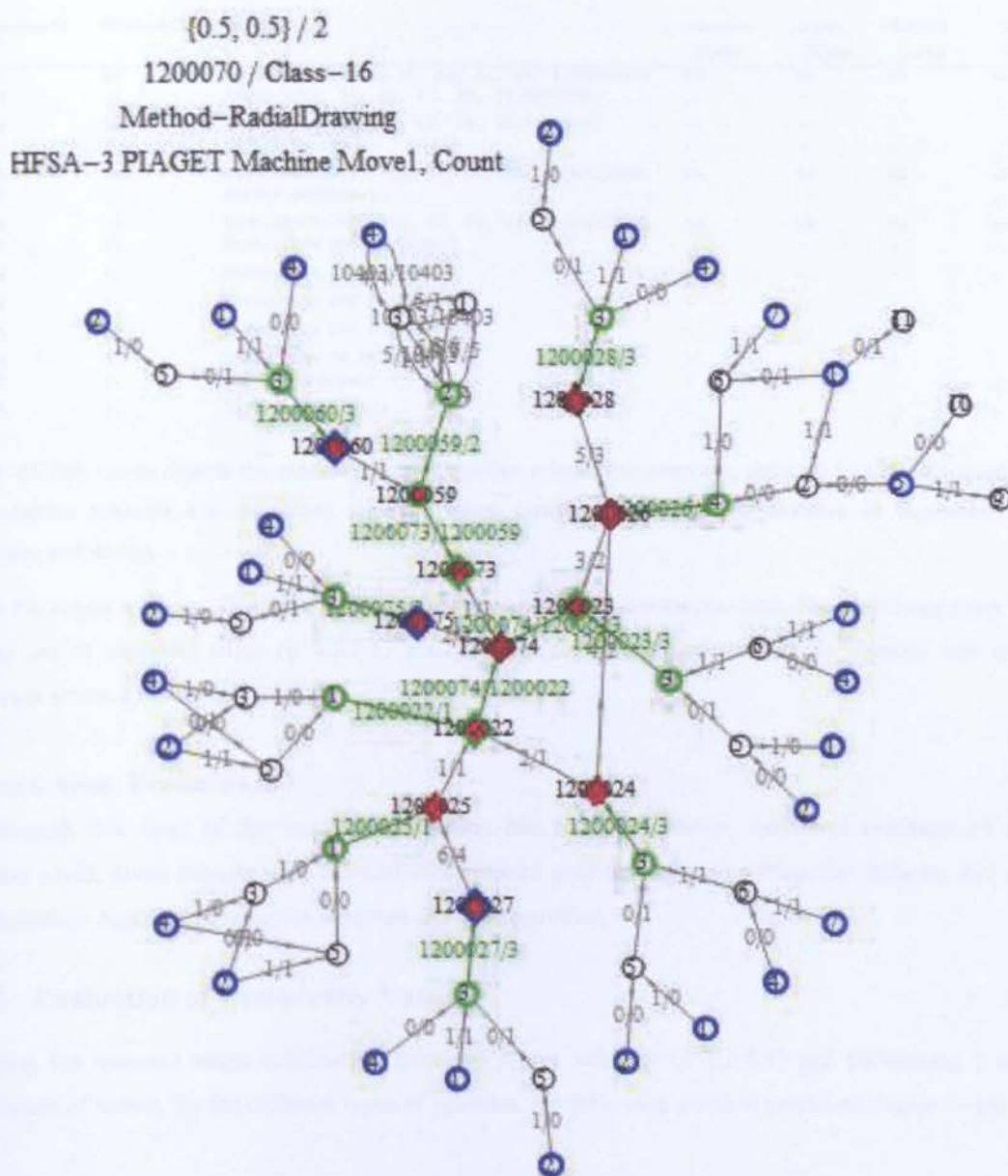


Figure 5-44 A depiction of a Piagetian scheme that moves (reads) along a number line, internalizes each of the values and then counts them producing the resultant counted value. The Argos schemes (red nodes) and the Prediction scheme (green and blue nodes) are connected through synchronization and transitions to enable adaptation with the environment.

The interesting feature is that all the processing takes place at the edges of the Scheme, since these contain the Prediction Schemes. The central channel contains the set of Argos schemes that marshal the information to the Argos and Prediction Schemes. The second interesting feature is its simplicity; it represents an executing scheme, but all it contains is hierarchies of finite state and HFSA, held together by an Argos framework.

The composition of the Piagetian Scheme is listed (figure 5-45):

Chapter 5. Evaluation

EVChromosomeID	EVClassID	Description	Initial State	Input State	Output State	Final State
1200 070	16.	Root-PIAGET{2011, 11, 27, 22, 40, 1.4900574}	na	na	na	na
1200 073	14.	ARGOS{2011, 11, 27, 12, 50, 34.5411836}	√	-	-	-
1200 074	11.	ARGOS{2011, 11, 27, 12, 49, 58.6421303}	-	-	-	-
1200 075	1.	Buffer Machine-1	-	-	-	√
1200 073	14.	Root-ARGOS{2011, 11, 27, 12, 50, 34.5411836}	na	na	na	na
1200 060	1.	Buffer Machine-1	-	-	-	√
1200 074	11.	Root-ARGOS{2011, 11, 27, 12, 49, 58.6421303}	na	na	na	na
1200 022	6.	Basic Gale XOR Machine-6	√	-	-	-
1200 023	4.	Basic Gale AND Machine-4	√	-	-	-
1200 024	4.	Basic Gale AND Machine-4	-	√	-	-
1200 025	6.	Basic Gale XOR Machine-6	-	√	-	-
1200 026	5.	Basic Gale OR Machine-5	-	-	-	-
1200 027	1.	Buffer Machine-1	-	-	-	√
1200 028	1.	Buffer Machine-1	-	-	√	-

Figure 5–45 This image depicts the contents of the Piagetian scheme for counting, showing how the different Argos and Prediction schemes are connected together using synchronization and transitions as described in the architecture and design.

As for Argos schemes, Piagetian Schemes use the same mutation mechanism. The surprising thing is that a simple set of mutation rules (§ 4.2.1.1) can produce unexpected complexity to process and marshal information across a network.

Piagetian Scheme: Evaluation

Although this facet of the working simulation has not been shown, sufficient evidence of how a simulation could, given enough time and sufficient reward gradients, mutate a Piagetian Scheme that counts using embedded Argos and Prediction schemes has been provided.

5.2.5.2 Evaluation of Complexity Values

Using the resource usage information from the Argos schemes (§ 5.2.4.4) and performing a straight accumulation of values, for the different types of schemes, the following graph is produced (figure 5–46):

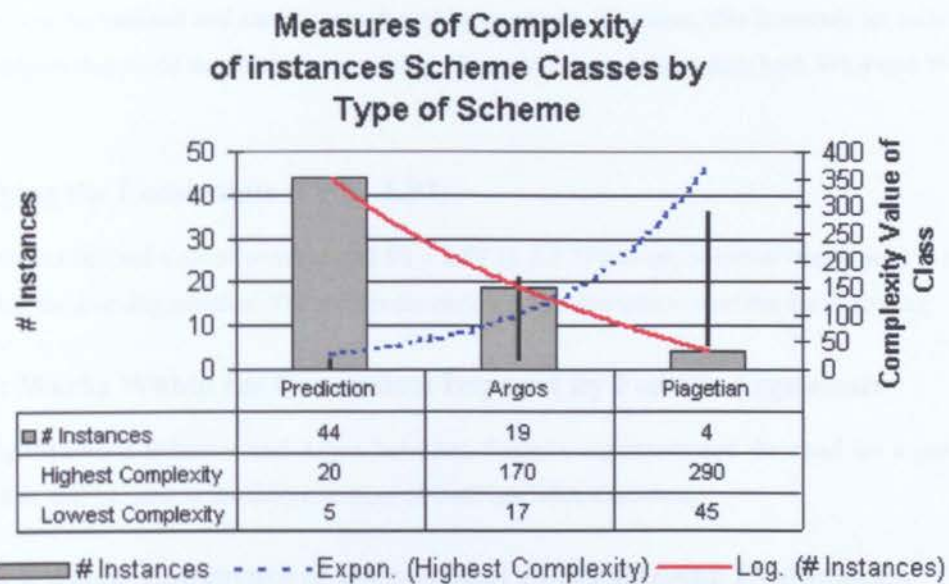


Figure 5-46 Measures of complexity of instances of scheme classes by type of scheme. Here the complexity is a simple measure of machine (scheme) size.

When one compares the complexity of Prediction, Argos and Piagetian Schemes one can get a sense of what Piaget referred to as maturation process (§ 2.1.7.5, 4.1, 2.3.5 and 2.3.11). The number of instances of different classes of Prediction Schemes and their low level of complexity is balanced by the complexity of the Piagetian Schemes and their relative complexity, with the Argos schemes sitting in between. The element of time can also be introduced on this graph with the appearance in **Level 1** of the Prediction Schemes. They coordinated action in **Level-2** and finally the interiorization of action in **Level-3**, then the appearance of Argos schemes to marshal the information across the prediction schemes in **Level-4**, and finally, the appearance of Piagetian schemes to be able to make use of these Argos and Prediction Schemes through the internalization of the hierarchical processes.

Once can also draw parallels to the observation of children, with their frequented repeated patterns of behavior and movement, with the examples of the repeated movements on the number line which give way in later maturity to more complex behavior.

5.2.5.3 Support for Worked Example

The aim of these tests is to confirm how far the simulation can support the worked example (**WE1 – WE5**).

WE4–Hierarchical Units and WE5 General Relationships

To support hierarchical units, the simulation needs to be able to internalize external values and then manipulate them through encoding, counting and decoding their values. Evidence has been presented where a

Chapter 5. Evaluation

Piagetian scheme can be mutated and can process these binary values. However, this is merely an architecture with worked examples that could support this processing. As such, the simulation fails both WE4 and WE5.

5.2.6 Satisfying the Constraints (LP1 – LP7)

The research has defined a set of constraints LP1 – LP7 (§ 2.1.7) that any solution would need to resolve to be able to tackle the learning paradox. The results presented in this evaluation confirm the following.

5.2.6.1 LP1: Works Within the Constraints Imposed By Fodor’s Arguments

By utilizing Prediction Schemes and Argos Schemes, Fodor’s arguments and the need for a predictive model and how that will be used in the determination of truth has been discussed.

5.2.6.2 LP2: Exhibits Emergence of Hierarchical Concepts using Evolutionary Process

In chapter’s 3 and 4, was developed an argument that concluded that the Piagetian model of learning and development used in this research is mirrored in the model of *Drosophila*. An analysis of how binary FSA and HFSA have parallels in developmental biology was then presented. Further, it was demonstrated that evaluation functions could be formulated in a similar fashion to TD Learning to form networks. From this, it was assumed that the Piagetian model was biologically plausible. In using mutation as an evolutionary process, it was shown how there is emergence of structure from a simple mealy machine to Prediction Schemes using the innate structures and reflexes (Pascual–Leone, 1980, p263) through Argos Schemes and to Piagetian Schemes. The development of symbolic forms through counting is an example which exhibits the properties defined by Piagetian schemes as “coordinated action” (Furth, 1969, p125). The simulation exhibits the emergence of hierarchical concepts, including what would be considered as the simulations “symbolic forms,” which themselves are more complex than its provided innate scheme and primary reactions (Pascual–Leone, 1980, p263). It was demonstrated how Crutchfield’s *intuitive emergence* (§ 2.2 on Crutchfield and A.9) can be used to detect these structures through visualization of the resultant forms.

By utilizing the mutation mechanism, the capacity to develop increasingly projectible behavior, resulting in “concepts” such as “number” through the appearance of digital circuits as evidence of partially resolving the symbol grounding problem has been demonstrated.

5.2.6.3 LP3: Operates Autonomously

Aspects of the simulation where it works autonomously with minimal innate knowledge and can overcome internal problem situations by being able to process using earlier forms, for instance through the inclusion of prediction schemes has also been demonstrated. The process of equilibration and disequilibration, which work to overcome errors, has been described. Described also is how the mutation mechanism always

Chapter 5. Evaluation

produces valid schemes, but these may not necessarily produce the right answer. In doing so, correspondence to Firby (Firby, 1989) and cognizant failure has been explained.

5.2.6.4 LP4: Mirrors Real World Behavior of Children Learning Number–Sense using a Number line

As a real–world problem of developing number–sense, a solution to Piaget’s bead problem (§ figure 1.1) has been produced. Also described are various scenarios where the cumulative behavior is similar to children, namely the development from simpler forms of Prediction Schemes to Piagetian Schemes and the expression of counting using composed digital circuits. The formalization of Piaget’s bead problem (§ figure 2–1) using reflective abstraction is demonstrated as possible using the interiorization mechanism (§ 5.2.6). Though a complete solution that can be fully mapped to childhood development e.g., the development of all the mathematical forms observed in childhood (Copeland, 1974; Liebeck, 1984; Fennell and Landis, 1994) has not been developed, a foundation for further research has been laid, especially when one considers the use of a predictive model to ascertain truth.

Further, it has been shown that Piaget’s notion of genetic epistemology, and the development of the “logical and rational organization of knowledge and the corresponding formative psychological processes” (Piaget, 1970a, p13) is mirrored in the development of digital circuits principally using Prediction and Argos schemes. A workable explanation of the processes of assimilation and accommodation has also been developed using through the development of the Piagetian schemes, which is an improvement on present research (Drescher, 2002; Chaput, 2004).

5.2.6.5 LP5: Learns Incrementally with Minimal Innate Knowledge and Reflexes

It has been shown that the simple model of cognition from *Drosophila* (Miesenböck, 2008, p52; Shang, Claridge–Chang, Sjulson, Pypaert and Miesenböck, 2007, p601) can be mapped to a Piagetian Model (§ 2.3.5 on Piagetian Model in Architecture and Design). The *permanent object invariants* emergent from the *artificial neural network implementation* can be utilized to produce more complex structures that allow the simulation to adapt to its environment using what it has already learnt. It has been demonstrated how the use of evaluation functions mirrors TD learning that allows the system to effectively bootstrap into more complex structures. All these processes use a trial and error approach, which is consonant with Piaget observations of early childhood (Furth, 1969, p191). Because of their “binary nature,” all these schemes are executable. If their “logic” is flawed’, it is evidenced by their interaction, they make mistakes and recover. The solution of this research thus overcomes LP5.

5.2.6.6 LP6: Develops its Learning Process

The simulation uses a mutation prediction mechanism to construct schemes (Prediction, Argos and Piagetian) that exhibit required input sequences. It has been demonstrated that the system can learn to predict input sequences, based on what it has already learnt, by allowing the system to recombine existing schemes.

Chapter 5. Evaluation

For example, a Prediction OR scheme, or Argos Full-Adder or Piagetian Prolog Scheme can be reused, so the system does not need to learn this process again. In doing so, the simulation supports LP6.

5.2.6.7 LP7: Acts in Novel, Opportunistic and Noisy Situations

For example, in showing the evolution through mutation of prediction schemes, it has been illustrated how the system acts in situations with noisy data. Though examples of opportunistic and novel situations have not been specifically shown, a sound basis on which to conduct further research has been established.

5.2.7 Scalability

This research has principally focused on the development of number-sense on a number line using a Piagetian / *Drosophila* (Miesenböck, 2008) model. Though not complete, results have been provided from an *artificial neural network implementation* in which has been identified the emergence of permanent object invariants in the environment (§5.1.1) and coordinated action using the identified permanent object invariants (§5.1.1). This research has shown how a *dialectic system* identifies regularities in the environment e.g., pen state, point, line, distance, using predictive FSA (§5.2.1) and exhibits coordinated action using combinations of sets of identified regularities, in the environment e.g., drawing a line (§5.2.1). It has also been illustrated how the Piagetian process of interiorization (§5.2.1) can occur through the development of propositional logic components using internalized binary values (§5.2.3). It described how Argos machines (hierarchical finite state machines that include embedded propositional logic components) can be evolved into digital circuits e.g., full adders, encoders and decoders (§5.2.4). It also explained how counting using Piaget's scheme process could work by reusing existing Argos machines (§5.2.5). Further, these results have described how Piagetian processes⁸¹ can be understood in terms of a simulated student that uses evolution as its learning process (§4.2.2.5) using FSA and HFSA that manipulate digital circuits. This approach is valid because the process of evolving circuits has been shown by other researchers to be universal (Koza, 1992, p647; Greenwood and Tyrrell, 2006). This research further concludes that the internalization and development of digital circuits using a Piagetian model (§2.3.6 and 5.2.4) where schemes are assimilated and accommodated using a simple 6 algorithm process (algorithm 16, 17, 18 and 19, see §4.2.3) provides a constrained mechanism for interaction with the environment. This process, where the simulated student uses schemes to act / sense, learn and plan in a number-line world with external rewards, provides a structure for evolving machines for basic arithmetic on "small numbers."

Though this "small numbers" approach has not been specifically proven in this research, a simulation could be scaled to include other Piagetian forms, such as logical connectives, spatial concepts, separation and order which are used in more complex mathematics (§C.1). Since this research has shown how the developed solution resolves the Piagetian bead problem (§ figure 1-1, figure 5-27 and 5.2.2.5) through conservation, it

⁸¹ Piagetian processes, for example: knowing circle, assimilation, accommodation, equilibration and developmental trend (§2.3).

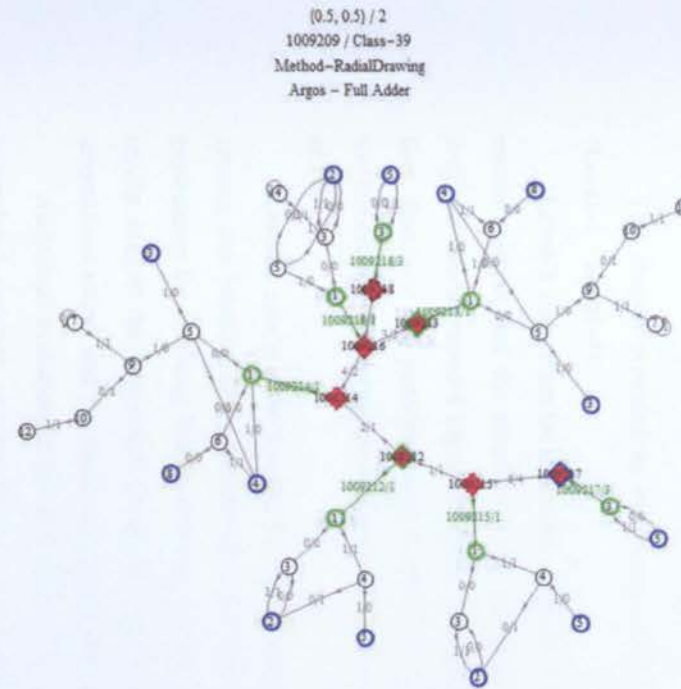
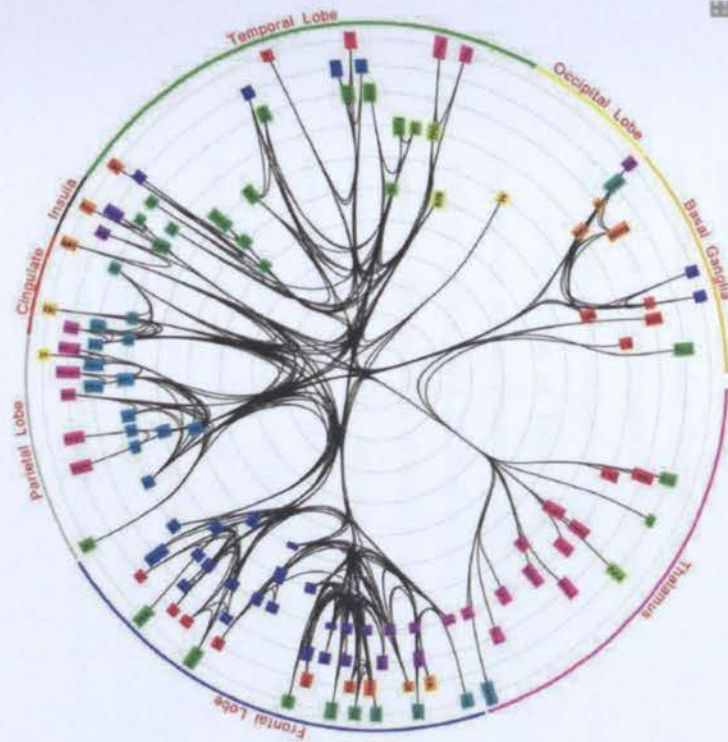
Chapter 5. Evaluation

can be further predicted, that a suitable solution could be extended to include other mathematical forms, such as those observed in childhood (Copeland, 1974; Liebeck, 1984; Fennell and Landis, 1994). These could include the following: place matching, sorting, pairing, multiplication and division, once the fundamental construction of Piagetian schemes emerges. Using this approach, it is believed to be possible to address the development of rational number-sense as originally postulated by Bereiter (Bereiter, 1985, p202) and described elsewhere (§C.2). Since the process in the system developed in this thesis essentially bootstraps from simpler forms – in this paper, the simple mealy machine (§ figure 4–13), using evolutionary computing – it is suspected that it could perform more complex processes, once the simulation matures.

It could be argued that all this research creates is what Dehaene refers to as the underlying “innate” ability for small number (Dehaene , 1997; Dehaene, Izard, Spelke and Pica, 2008; Fuhrman and Boroditsky, 2010; de Hevia, Girelli and Macchi–Cassia, 2012). On reflection, this may well be true, yet Piaget had already observed this aspect of childhood development and associated it with features of assimilation and accommodation (Piaget, 1952, p154). In terms of scalability, this project has merely provided a base for further analysis of his theories, which, it is suspected, will be relatable to the work by other researchers.

5.2.8 Resemblance of Argos Schemes to Macaque Brains

An interesting parallel can be drawn between recent research on the processing pathways in Macaque brains (Modha and Singh, 2010, p13488) and Argos schemes (figure 5–47)



a) Network architecture of the long-distance pathways in the macaque brain
 b) Example of Argos Scheme as a full-adder.

Figure 5-47 Comparison of processing in Macaque (left Side) and Piagetian schemes (Right Side), (Modha and Singh, 2010, p13485).

In their directed network “Hubs distribute information, whereas authorities aggregate information” (Modha and Singh, 2010, p13488), one can tangibly correlate hubs with Argos Schemes that marshal information over Prediction Schemes, which is how authorities process information as a digital circuit.

Chapter 5. Evaluation

Further, when one reviews the work of Albus, one can draw parallels to the Piagetian architecture of figurative schemes and assimilation to *receptive field hierarchy* and the *event hierarchy* to accommodation and operative schemes (Albus, 2000; 2008 and 2010b, p193). Only future work will be able to evaluate this claim.

5.2.9 Summary of Dialectic Evaluation

This chapter provided an evaluation of the *dialectical system* with a series of results being produced (Level-1 – Level-4).

Level-1 starts from the evolution of simple schemes to identify objects in the number line world. **Level-1** results confirm that the simulation can act/sense and learn in a number line world. Specifically, like the *artificial neural network implementation*, the simulation can detect regularities in the environment of point, line, direction, and penState using a prediction mechanism. In also supports tasks in **WE1** and **WE2** the *worked example*. Observations suggest conservation and the development of structure in the simulation as a set of FSA.

Level-2 results show that the evolved *dialectical system* exhibits coordinated action including acting and sensing and learning using movements from fixed and relative positions. These **Level-2** results confirm an explanation for resolving **WE3** and Piaget's bead problem (conservation of number). It would seem that these results suggest the emergence of number, but Piaget would attest that number can only be experienced as internalized action, and that these are intermediate results.

Additional evolution of the *dialectical system* at **Level-3**, show that the *dialectical system* can develop internalized structure as propositional logic components in digital circuits, along with dialectical student actions from a teacher. **Level-3** results suggest that it can work autonomously: yet even this is not the appearance of number. Further evolution of the *dialectical system* to **Level-4** is necessary to enable the *dialectical system* to generate Piagetian schemes, which can reuse existing Argos and Prediction schemes and with evolutionary computing, internally construct digital circuits.

Observations of **Level-4** results suggest the emergence of structure, specifically number through internalized action. Presented examples include the Piagetian mathematical operations of equivalence (§ 4.2.2.5), counting (§ 5.2.5.1) along with encoders (§ 5.2.4.2) and adders (§ 5.2.4.3). It is at **Level 4** that support for all the defined tasks of *worked example* could be achieved. These **Level-4** results show that equivalence requires interiorization, and that disequilibrium/equilibrium is a stabilizing mechanism for Piaget's developmental trend. It is also at **Level 4**, that Piaget's need for a parallel in the development of learning and logic is realized (Piaget, 1970a, p13); since the digital circuits in the Piagetian and Argos Schemes, form their own "logic of co-ordination" (Furth, 1969, p125).

A discussion on the scalability of the approach considered how a series of tests could be performed to support more complex mathematical forms including multiplication, measuring (Copeland, 1974) and other methods (Liebeck, 1984), as well support the development of rational number-sense.

5.3 Summary

In this chapter was described an evaluation of an *artificial neural network* and a *dialectical system* implementation of a Piagetian model of cognition. Its purpose is to determine if project aims 7 and 8 (§ 1.2) have been met. The main evaluation criteria used were those constraints any solution that attempts to resolve the learning paradox will need to resolve (§ 1.2.7 for a definition of **LP1 – LP7**). The solution was evaluated against an example problem of numerosity in a number–line world (§ 2.3.14 for a definition of **WE1 – WE5**), which were included in these research experiments (§ 3.5). Finally, this project was discussed in terms of scalability and considered related work.

Chapter 6

6. Further Work

This project is an evaluation of the epistemological implications of the learning paradox, as it applies to the childhood development of number–sense using a number line. This thesis has argued that Piaget’s genetic epistemology (Piaget, 1970a; Furth, 1969; Copeland, 1974) provides a framework for a solution that explains childhood development, but not its biological basis. An analysis of biology and evolution provided a plausible model of emergence of structure using a model of *Drosophila* (Miesenböck, 2008). A case is made that FSA and reinforcement learning provided a plausible mechanism to build a simulation for emergence. In reevaluating the roots of Piaget’s genetic epistemology, it is demonstrated that his cognitive processing of assimilation and accommodation is mirrored in the model of *Drosophila*. A review of the observations of the development of the mental number line, provided arguments to support using a number line in the development of number–sense by the simulation. By contrasting this approach to emergence against other mathematical concept formation models (Colton, 2000; Colton, 2002), this simplistic model is at a much lower level. Similarly, in evaluating models of metaphorical concept formation (Lakoff, 1992; Lakoff and Johnson, 1980; Lakoff and Núñez, 2001; Fauconnier, 1985 and Fauconnier and Turner, 2002) this approach provided a model of emergence as suggested by Fauconnier (Fauconnier and Turner, 2002, p146; Guhe et al., 2011, p250). By comparing and contrasting this method to models of cognitive development, issues that would need to be addressed including the symbol grounding problem were identified, and it was concluded that this approach is an emergent model. By comparing and contrasting this reevaluated model of Piaget to existing Piagetian implementations, uncovered the fact that they were not solving the same problem. Finally, in reviewing other machine learning frameworks, the conclusion was reached that they were not specifically addressing the learning paradox, only providing interesting areas for further work.

There are many ways this research thesis could be extended. These include improving the implementation of Piagetian methods described in § 6.1, deepening the model of cognition, described in §6.2 and extending this model to fully resolve the learning paradox, described in § 6.3 and § 6.4. These improvements and an estimation as to whether each improvement would be a long or short–term project will be discussed in this chapter.

6.1 Improving the Implementation of Piagetian Methods

6.1.1 Singular Evolution for Finite and Hierarchical Automata

For expediency, the architecture of the system was separated into FSA and HFSA. A more appropriate solution would be to assume an initial hierarchical perspective and then separately evolve Prediction, Argos and Piagetian Schemes. This would further provide evidence to support Piaget's notion that a "genetic epistemology" (Piaget, 1970a) is a mechanism to partially resolve Fodor's learning paradox (Fodor, 1980). This is a small project, but the development of a set of libraries would significantly add to discussions on evolutionary development.

6.1.2 Developing the Predictive Model

Though not implemented, the design suggests that making use of a predictive model, one that consumes schemes, would be a useful avenue for future research for two reasons. First, it is expected, like planning in reinforcement⁸², that using a predictive model would speed up the cognitive process. Second, Piaget anticipated that abstract forms only occur through play and the use of the predictive model is indicative of play (Furth, 1969, p95–98; Furth, 1969, p156). It is believed that planning is an evolutionary process to conserve resource usage, which would add greatly to the range of activities and problems that could be tackled. Appendix D – Further Examples of Dialectic Evaluation (§ D) provides a simple model of how the predictive model would work. The development of the predictive model is believed to be a substantial project.

6.1.3 Using Intrinsic Rewards

The *dialectic system's* implementation could be improved by adding intrinsic motivations; the current reward process is built using evaluation functions. This could be completed by fully implementing the rewards process used in TD learning (§ A.6.2 on TD Learning and § 4.2.1.1 in Architecture and design). This would make the system more autonomous by allowing it to make use of curiosity rewards (as occurs in the *artificial neural network implementation*) and support the development of the predictive model.

6.1.4 Incorporating Emergent Schemes

Changes to the mutation mechanism may allow the use of emergent schemes based on observed regularity in the environment. This would, enable the simulation to "fully reuse" what it has already learned and more easily adapt to its environment. These emergent schemes effectively give the simulation, the capacity that Bereiter referred to as bootstrapping (Bereiter, 1985, p202). Ideally, this would be combined with the use of intrinsic motivations. This will be a medium size project.

⁸² Reference is made here to the evaluation of reinforcement planning stages.

Chapter 6. Further Work

6.1.5 Expanded Use of Memories

Since this research builds memories as Prediction, Argos and Piagetian schemes (as FSA and HFSA), it is quite possible that memories could be added to the system. Since memories are executable machines, these memories would provide the system with new capabilities. For instance, a scheme could be hand crafted, or have a specific evaluation function to classify objects, or it could have new senses that the system could utilize. Similarly, sets of executing systems in different domains could share “schemes.” A problem that would need to be overcome is what constitutes a shared ontology. This is important for multiple reasons, not least of which is that the majority of what schemes do is classify, but if this classification is provided externally, it could provide significant benefits for the simulations predictive capabilities. This is thought to be a medium sized project that would benefit to the overall approach.

6.1.6 Using Play to Develop Metaphors

Consideration was given to the possibility that Piagetian play might support the development of metaphors (Fauconnier and Turner, 2002). For instance, Piaget considered play essential to the formation of language (Furth, 1969, p156; Furth, 1969, p96). Play in the Piagetian sense is a form of planning. Therefore, the development of a predictive model is thought of as an essential step to expand the capabilities of this model. This will be a medium sized project.

6.1.7 Constraining Resources to Force Planning

By building a model of resource constraints (Pascual–Leone and Goodman, 1979; Pascual–Leone, 1980) insights will be gained into the formation of concepts in a Piagetian model. Consider how the limit of 1m CCUs (Albus, 2008; 2010) would affect the building of the *receptive field hierarchy* and the *event hierarchy*. It is anticipated that the simulated model of *Drosophila* would rely on planning (mental image and evocation) to make more effective use of available resources and so be more adaptive to its environment. Further, it would be interesting to correlate this research to Pascual–Leone’s notions of M–Capacity and M–Demand (Pascual–Leone, 1976, 1980 cited by Bereiter, 1985, p201). The limit of the availability of mental attentional energy (M–Capacity), as well as the resources required to complete the assigned tasks (M–Demand), could be identified in the simulation and compared to published results for students in the *concrete operations stage* of development (Bereiter, 1985, p221). If improvements could be identified in the simulation and transposed to changes in human performance, then the educational validity of the approach could be established. Similarly, it may be possible to formally relate the work of Pascual–Leone strategy to the approach used in this research to form a bridge between the two Piagetian views. This will perhaps be a medium project, since the development of the planning process would need to be completed first.

6.2 Deepening the Model of Cognition

There are several ways to improve upon this research's understanding of cognitive models. The coordinated evolution from simplistic schemes to more complex hierarchical schemes may provide interesting insights to the interaction of acting/sense, learning and planning in humans.

6.2.1 Extended Evaluation for Rational–Number–Sense

An active area for further development would be to complete the implementation of HFSA and then run the simulation against the *worked example* WE1 – WE5, including the definitions for rational number–sense (§ C.2.1). The WE6 results could then be compared to the results from the rational number project (Cramer, Behr, Post and Lesh, 1997a and 1997b). This would provide further evidence that the solution approach was valid or conversely was not valid. A useful process would be to provide executable traces of the system and relate these to the assumed Piagetian cognitive processes, the design features and a trace of a child performing similar “novel” tasks. This could provide deeper insights into the processes of learning and development, and of emergence of structure itself. It is suggested that this is a large project which would extend the range of features of the system and include aspects of § 6.1.1 singular evolution and § 6.1.2 using the predictive model.

6.2.2 Comparisons to Models of the Mental Number line

An interesting area for future research would be to evaluate Siegler's number line task (Nuerk, Helmreich, Zuber, Moeller, Pixner, Kaufmann, 2011, p598) against benchmarks for students and this research. In a similar vein, it may also be possible to determine the Piagetian Stage of the student. This would provide an effective way of grading the systems capabilities. It may also be able to analyze the logarithmic to linear shift as observed in the use of the MNL (Dehaene, Izard, Spelke and Pica, 2008) against this research to see if bias could be added. It may be possible to determine the specifics of a mechanism for spatial representation of time and number using a MNL (Fuhrman and Boroditsky, 2010, p1432). Similarly, further work could determine the mechanics of left right bias using a MNL (Opfer, Thompson and Furlong, 2010, p761). This would be a very difficult project, for it relies on the complete development of a solution and the construction of sensory–motor bias mechanisms.

6.2.3 Comparison to Other models of Mathematical Cognition

With a complete solution it may be possible and interesting to make comparisons to other models of mathematical cognition; for instance, to determine how the appearance of structure compares to formal models of mathematical discovery (Colton, 2002), as well to compare the Piagetian model of the childhood mathematics development (Copeland, 1974; Furth, 1969) to the more formal model of mathematicians (Lakatos, 1976 cited in Pease, 2007, p218). Similarly, it may be interesting to compare the algorithmic approach to conceptual blending (Guhe et al., 2011, p253) against an emergent structure approach as used in

Chapter 6. Further Work

this research. This will be a difficult project, for it requires the development of a full solution, as well as the integration of different conceptual models.

6.3 Extending the Model

6.3.1 Using Vision to see Beyond a Number line

It may be possible to extend the *dialectic system* using techniques from machine vision. For example, the eigen-background approach (Rymel Renno, Greenhill, Orwell and Jones, 2004) could be used to detect moving objects on a number line. Similarly, line trajectories (Ren, Orwell, Jones and Xu, 2004) could be used to predict ending movements on a number line. This research could use these approaches to “see” statistical regularity and “geometry” which would extend processing into three dimensions. It may also be possible to relate the notion of concept formation to concepts in visual systems (Le, Ranzato, Monga, Devin, Chen, Corrado, Dean and Ng, 2012). This would be a small project, which could be undertaken in a reasonable amount of time.

6.3.2 Optimization and Simplification through Formal Methods

It may be possible to simplify the approach by writing out the deterministic equations of assimilation and accommodation and compare them to TD Learning (Streeter, Oliver and Sannier, 2006; Suri and Schultz, 1998, p350; Schultz, 2000, p201; Schultz and Dickinson, 2000, p475; Fiorillo, Tobler and Schultz, 2003, p1898). This may provide insights into the process of emergence, which has been shown to occur in the use of the Verve toolkit (§ 5.1 on evaluation of *artificial neural network implementation*). It may also be then possible to compare the model developed in this thesis the formal model provided by Pascual-Leone (Pascual-Leone and Goodman, 1979; Pascual-Leone, 1980; Pascual-Leone and Johnson, 1999). This would be a small project, which could be undertaken immediately, but would be tricky because it would need to account for development of the learning process.

6.4 To Fully Resolve the Learning Paradox

It may be possible to fully resolve the learning paradox. This research provides an initial bridge between multiple disciplines by showing how a cognitive model originally proposed by Piaget (Piaget, 1952) in the early part of the 20th century still has validity in the current age of neurophysiology and machine learning. This can be accomplished in three ways (1) By correlating the Piagetian model of genetic epistemology to a model of *Drosophila* (Miesenböck, 2008, p52; Shang, Claridge-Chang, Sjulson, Pypaert and Miesenböck, 2007, p601). (2) By identifying the similarities of the outputs of HFSA / FSA solution developed in this thesis to recent representations of brain scans (Modha and Singh, 2010, p13488) - in which there is a separation of processing. (3) By showing the similarities to the cognitive processing models provided by Albus and Grainger (Albus, 2000; 2008 and 2010b; Grainger, 2006a, p18 Granger, 2006b) to the model of Piaget. There are remaining issues that need to be resolved. Each of these issues has been discussed in the preceding section,

Chapter 6. Further Work

though only those critical areas are listed: expanded use of memories (§ 6.1.5); developing the predictive model (§ 6.1.2); using intrinsic rewards (§ 6.1.3); incorporating emergent schemes (§ 6.1.4); using play to develop metaphors (§ 6.1.6); constraining resources to force planning (§ 6.1.7) and using singular evolution (§ 6.1.6). This would be a complete research project requiring a great deal of skill as well as resourcefulness to accomplish.

6.5 Applications of this Research System

Four key applications of the techniques used in the research solution are identified.

6.5.1 Classification Systems

It was suggested that it is possible to build a Piagetian model of concept formation. This may provide new techniques in the development of classification systems.

6.5.2 Concept Formation using a Piagetian Model

This research suggests that it is possible to build a simulated Piagetian model using mutated binary HFSA. This simulation builds digital circuits to count using a process of assimilation and accommodation. This is an interesting area for concept formation, since Piaget's notion of developmental action coordination is significantly different from other surveyed methods of concept formation. This is notable for two reasons: (1) It sidesteps the existing limitations of cognitive development models (§ 2.1.7 Fodor on LP1 – LP7) and provides an action graph of the concept, through the processes of assimilation and accommodation. (2) The basis of the model is the development of a reactive system (Maraninchi and Remond, 2001; Maraninchi, 1992) as a form of evolvable hardware (Greenwood and Tyrrell, 2006). Given the significant amounts of research on both these topics, it seems feasible that it provides a fertile ground for further research. If the system can generate novel concepts, and can be given sufficient sensory motor capabilities, then it may be possible for text classification to emerge in a similar way to its emergence of number-sense.

6.5.3 Piagetian Studies

It may be possible to apply this approach to the continuing study of Piaget's work, since it provides an interesting view of interiorization, assimilation, accommodation and the abstract process of the knowing circle (§ 2.3 on The Roots of Genetic Epistemology).

6.5.4 Conceptual Blending

As an evolvable mechanism, this research provides an interesting area for further analysis of concept formation, especially when one realizes the need for a model of emergence of concepts in conceptual blending theory (Fauconnier and Turner, 2002, p146; Guhe et al., 2011, p250).

6.6 Conclusion

This project evaluated the learning paradox from a number of research perspectives and developed a set of constraints (LP1 – LP7) and a worked example (WE1 – WE5) by which to grade potential solutions. In this way, it produced a biologically plausible model of Piaget’s genetic epistemology and implemented it using a mutation model of binary HFSA. Though not complete, it provides a sound basis to conduct more research on the childhood development of number–sense. The hope was to demonstrate that, it is both possible and useful to provide a computational reading of Piaget’s ideas. This work is only a preliminary reading and there are many future directions, which the project could take. These directions have been discussed in this chapter.

Further possibilities, which could be valuable and interesting endeavors to pursue, include the following:

(i) Improving the implementation model through the singular evolution of finite automata and the implementation of a predictive model to support the notion of play and metaphors. The use of intrinsic rewards and emergent schemes would support behavior that is more autonomous. A more complete model could be supported by expanding the use of memories and play by constraining resources to force planning.

(ii) By extending the model to support rational numbers along with comparisons to early childhood mathematics research and concept formation would deepen the model of cognition.

(iii) By extending the model through the use of machine vision would allow the system to explore other spaces, and provide interesting areas for concept formation.

Chapter 7

7. Conclusions

Pascual–Leone defines Fodor’s problem as the learning paradox (Pascual–Leone, 1976, 1980 cited by Bereiter, 1985, p202) which is a meta–theoretical problem and is defined by a meta–theoretical question: “How can a structure generate another structure more complex than itself?” Piaget’s position on the learning paradox is that new concepts can be learned through the normal learning process, without requiring that additional, but unused capabilities exist (Fodor, 1969). Piaget’s work is still relevant today, as evidenced by the continued work on his theories (Drescher, 2001 and Chaput, 2004).

Piaget’s work is of interest to researchers in cognitive psychology and artificial intelligence, since it provides an account of the emergence of structure that is biologically plausible, which is also rich in detail and structure. The field artificial intelligence and neurophysiology provides the tools to gain a novel perspective on Piaget’s work. This thesis is the story of the present exploration of the relationships between these three fields.

The main hypothesis of his research study is that a Piagetian model of human cognition exists that explains emergence and so resolves the learning paradox. A secondary hypothesis is that a biologically plausible model of mathematical cognition exists. Finally, a hypothesis is proposed that hypothesis 1 and 2 can be modeled and implemented in a simulation with number–sense as an emergent property of the executing system. In § 7.1 strong evidence has been provided for these positions. In § 7.2, the contributions of this thesis is discussed and brought to a conclusion in § 7.3.

7.1 Have the Aims Been Achieved?

In §1.2 the aims of this project were to:

- 1) *evaluate epistemological solutions to the learning paradox;*
- 2) *evaluate sources of emergence;*
- 3) *provide a computational reading of Piaget’s theory;*
- 4) *clarify emergence in a real–world worked example using a number–line;*
- 5) *compare and contrast different notions of mathematical concept formation;*
- 6) *provide arguments to support a separation into an artificial neural network implementation and a dialectical system;*
- 7) *evaluate the evaluate the artificial neural network implementation; and*

Chapter 7. Conclusions

8) *evaluate the dialectical system implementation.*

Achieving the first of these aims provides evidence for the primary thesis. It was shown that a possible epistemological resolution to the learning paradox occurs through normal development, as formalized by Piaget's theory of genetic epistemology (Furth, 1969; Copeland, 1974). The remaining seven aims suggested ways in which providing such a reading would be a useful endeavor to undertake.

The first aim was achieved by reviewing the epistemological issues associated with the learning paradox in which the need for emergence was identified (§ 2.1). This included identifying a set of constraints (LP1 – LP7) any solution would need to overcome to resolve the learning paradox. Candidate solutions were graded using these constraints.

The second aim was accomplished by reviewing the biological basis of emergence where it was determined that a simple model of emergence in *Drosophila* was probably preserved in evolution. Also confirmed was that actor–critic temporal difference learning provides mechanisms for emergence (§ 2.2).

The third aim was fulfilled developing a computational model of emergence of Piaget's theory through a close reading of Furth (Furth, 1969) and Copeland (Copeland, 1974), in (§ 2.3). Parallels were drawn between Piaget's theory and a model of *Drosophila* that inferred a biological basis for Piaget's theory. A worked example was also developed (WE1 – WE5) based on an analysis of Piaget's theory that was used to evaluate the research solution.

By comparing and contrasting the worked example of Piagetian development (WE1 – WE5) to research on concept formation using the mental number line, the fourth aim was met (§ 2.4).

The fifth aim was accomplished by using the constraints (LP1 – LP7 and the worked example WE1 – WE5) to review and determine if existing solutions have emergent concept formation without side stepping the issues associated with the learning paradox. The following solutions were considered: automated reasoning and theory formation in pure mathematics (§ 2.5), metaphors and conceptual blending theory (§ 2.6), cognitive development models (§ 2.7), existing Piagetian implementations (§ 2.8) other machine learning frameworks not covered by cognitive development models (§ 2.9).

In performing the literature review, a set of principles under which any solution would need to work were identified, and then, through determining a limitation of existing artificial neural network solutions, the need for a two–part solution as an *artificial neural network implementation* and a *dialectical system* was identified (§ 2.10), thus achieving the sixth aim.

This research system has enabled the evaluation of the methods used, thus achieving the seventh and eighth aim, by allowing the conduction of empirical experiments and their evaluations. Specifically, a worked example was defined (WE1 – WE5) based on Piagetian theory (§ 2.3.14) demonstrating how a simulation working only in a number–line world could have emergent number–sense using a biologically inspired Piagetian model (§ 2.3). The architecture and design was constrained to a biologically plausible model of Piaget that also worked within the constraints that were defined in this study (§ 4). Further, the experiments that were defined identified the emergent behavior that was anticipated would be observable in the executing system (§ 2.3).

Chapter 7. Conclusions

These results are described in chapter 5. In particular, three hypotheses have been tested:

- 1) Is it possible for an artificial neural network implementation to show emergence of *permanent object invariant* as predicted by Piaget?
- 2) Is it possible to build a computational model of Piaget (the *dialectic system*) that adheres to the constraints imposed (LP1 – LP7)?
- 3) Is it possible to build a full computational model of Piaget (the *dialectic system*) that adheres to worked example (WE1 – WE5)?

Results from this evaluation (§ 5.1) show that the *artificial neural network implementation* has met point 1 (the seventh aim). Further results (§ 5.2) show that point 2 has been met, but not point 3 (the eighth aim). Further work is required to develop a full solution that can resolve the learning paradox with a biologically plausible model of Piaget’s theory that shows how a simulation can develop number–sense.

7.2 Contributions

This is not the first, but is a thorough investigation of Piagetian theory as it applies to the emergence of number–sense. A contribution is made in the following areas: computational cognitive psychology, evolutionary computing and concept formation systems.

7.2.1 Computational Cognitive Psychology

A contribution to computational cognitive psychology has been made by providing a novel perspective on Piagetian theory. The computational model of Piaget’s genetic epistemology draws parallels to a model of *Drosophila* (Miesenböck, 2008, p52; Shang, Claridge–Chang, Sjulson, Pypaert and Miesenböck, 2007, p601) and actor–critic temporal difference learning. This computational similarity has suggested new ways to view the Piagetian processes of assimilation, accommodation, reflective abstraction and interiorization as facets of biological processes, which have been related to other sources (Albus, 2008; Albus, 2010). The emergence of these “digital circuits” as Piagetian schemes provides an argument for reexamining Piaget’s “logic of development” (Piaget, 1970a, p13). It also provides the “emergent logic,” as required for the emergence of concepts in conceptual blending theory (Fauconnier and Turner, 2002, p44 and p146).

7.2.2 Evolutionary Computing

A contribution to evolutionary computing has been made by extending the capabilities of a computational model of evolution (Jacob, 2001) and binary FSA (Levy, 2002) with classed based evolution. This improves upon existing methods and builds digital circuits by providing a separation of concerns for mutated propositional and predicate components that use a Piagetian model of emergence.

Chapter 7. Conclusions

7.2.3 Concept Formation Systems

A contribution to concept formation systems has been made by defining a set of constraints (**LP1 – LP7** and a worked example **WE1 – WE5**) has been used to grade existing candidate solutions. This improves upon existing evaluation criteria by providing a benchmark of early childhood development of number–sense, which can be used to determine if there is the emergence of concepts, and so resolve the learning paradox.

7.3 Conclusions

Understanding how to overcome the learning paradox (creating a structure more complex than itself) is a challenge to researchers in the field of artificial intelligence, as well as to cognitive psychologists and educationalists. Piaget has shed some light on the problem and provides a model of how children naturally overcome it as they mature. Piaget’s insights and theory has been used to develop a model of evolved cognition that is mirrored in a model of *Drosophila*. This thesis has demonstrated that an *artificial neural network implementation* shows emergence of structure, but cannot develop a Piagetian form of number–sense. It also shows that a *dialectic system* can implement a fuller Piagetian model of development that exhibits emergence of aspects of number–sense. In doing so, this thesis improves upon the state of the art models of emergence and Piagetian models in particular.

Appendix A

A. Evolution and Emergence

In § 2.1.7, a set of constraints (LP1 – LP7) is defined that would need to be met by any simulation that attempted to tackle the learning paradox using a Piagetian model. Since evolution has already resolved the learning paradox, in this appendix some key aspects of biology and evolution are reviewed, and then discussed with the purpose of discerning how they can be simulated and tested for emergence without contradicting the identified constraints. Specifically, simple models of emergence of structure in ant colonies and fruit flies are discussed. These are related to techniques to model neural networks, and the nature of hierarchical models and the need to validate the approach of using a number line world are evaluated. Also evaluated are the mechanisms that simulate evolution as well as how the storage and concurrency issues can be resolved by using FSA within a reactive systems framework.

A.1 Simple Models of Emergence of Structure

“How does evolution produce increasingly fit organisms in environments which are highly uncertain for individual organisms? How does an organism use its experience to modify its behavior in beneficial ways (i.e., how does it learn or adapt under sensory guidance)? How can computers be programmed so that problem-solving capabilities are built up by specifying what is to be done rather than how to do it?” (Holland, 1975, p1)

There is emergence of structure in the collective behavior of ant colonies, for instance in searching for food (Johnson, 2001, p45). This is evidence of them solving LP5 and LP7, as an optimization problem. This process can be simulated using swarm techniques where simple rules, embodied in agents, using vast populations, are shown to exhibit novelty in opportunistic and noisy situations. They can solve problems of routing, assignment, scheduling and subsets (Dorigo, Birattari and Stutzle, 2006, p34). Wolfram sees this emergence as order from randomness and uses simple cellular automata to simulate these characteristics (Wolfram, 2003, p223). These processes can be mapped to biological processes (Wolfram, 2003, p383 and p398). The current swarm models are not applicable to this research, since their simulations do not produce the more complex ant social model, and it is necessary to model aspects of human cognition. These swarm rules do not support a Piagetian model, since they do not have the capability to develop their learning process, nor planning and so fail LP6.

A.2 Fruit Fly Models of Emergence

A model of *Drosophila* provides an internal engine that is more complex than a simple set of ant colony rules, and explains how their brains switch between acting/sensing, learning, planning (Miesenböck, 2008, p52; Shang, Claridge-Chang, Sjulson, Pypaert and Miesenböck, 2007, p601). Their research engineered the 12 neurons in *Drosophila* so that these could be switched on artificially by the use of an external light source (blue/red). Experiments showed that this switch was a determining factor in whether the fly incorporated experiences which were “learned,” “remembered/performing” or whether they became negative experiences, even when the experiences would not normally lead to any change in behavior (Miesenböck, 2007). Since many aspects of the basic pathway are probably conserved evolutionarily, the fly studies research provides a simple model for cognition that can provide a mechanism to resolve LP6 and this requires implementation as a neural model. It is assumed that this model of *Drosophila* has similarities to a Piagetian model of learning and development.

A.3 Techniques to Model Neural Networks

Given that evolution has produced the neural networks in fruit fly and human brains, and also cell networks, it is important to use a biologically plausible simulation mechanism. Two approaches are suggested, artificial neural networks and FSA.

A.3.1 Artificial Neural Networks

Many researchers have used artificial neural networks with supervised, unsupervised learning and self-organizing methods to address problems such as pattern recognition and prediction (Carpenter, Grossberg, Rosen, 1991; O'Reilly and Munakata, 2000; Sun, 2004; Drescher, 2002; Sutton and Barto, 1998 and Streeter, Oliver and Sannier, 2006). These researchers make use of back-propagation, linear and radial basis neural networks amongst others. The approach used in this research are linear and radial basis neural networks, principally because they were available in the Verve toolkit (Streeter, Oliver and Sannier, 2006).

A.3.1.1 ANNs: Where is the Problem?

A lack of hierarchies in current artificial neural networks means that they cannot resolve LP2. In addition, neural networks are seen to be difficult to implement. When this is combined with a need for a “full” Piagetian model, the decision was made not to pursue a pure artificial neural network implementation.

A.3.2 Finite State Automata

Minsky proved that every “finite-state machine is equivalent to, and can be simulated by some neural net” (Minsky, 1967). In this, Minsky showed that McCulloch and Pitts neurons (McCulloch and Pitts, 1943, p115) could be modeled, by Mealy machines. A FSA can also be considered as an abstraction of very simple biological cells (Kauffman, 1969). The initial state can be compared to the maternal factors for a zygote cell.

Appendix A. Evolution and Emergence

The inputs are the inter-cellular communication proteins, hormones and environmental factors. The outputs are the physiological properties of the proteins produced by behavioral genes. Each state bit is a binary abstraction of the concentration of one or more specific proteins synthesized in the cell (Kauffman, 1969; Kumar and Bentley, 2003). More recent research has shown that even chemical reactions can implement FSA, with some even being able to count (Hjelmfelt, Weinberger & Ross, 1992, p383).

A FSA can be implemented as a synchronous sequential circuit consisting of a sequential logic section and a combinational logic section (Minsky, 1967). The outputs and internal flip flops (FF) progress through a predictable sequence of states in response to a clock and other control inputs. Each FSA contains a state variable which holds its present state and a memory section and a control section that controls the next state of the machine (by clocks, inputs and present state). FSA read their input once and are deterministic. They seem trivial, yet they play a major role in the modern analysis of hardware and are used to prove the decidability of complex logical theories, one example being Presburger's Arithmetic. Two main methods exist: Moore machine and Mealy machines. A Moore machine outputs are generated as products of the states i.e.; the output values are determined solely by its current state. A Mealy machine output values are determined both by its current state and by the values of its inputs i.e.; the outputs are generated as products of the transition between states (Mealy, 1955, p1045). Mealy machines have been more closely related to biological process (Yang, 2006) than Moore machines.

FSAs have been utilized in evolutionary computation (Rechenberg, 1965 and 1973; Fogel, Owens and Walsh, 1966, cited in Koza, 1999, p386). Other researchers have produced self-reproducing FSAs (Ray, 1991 cited in Koza, 1992, p647) that incorporate aspects of introspection and others that use the mechanism for ant foraging (Jefferson et al., 1991 cited in Koza, 1992, p54).

To mirror biology, Boolean mealy machines would need to be used rather than the traditional symbolic approaches.

A.3.2.1 FSA: Where is the Problem?

FSAs have trouble dealing with external resource conflicts and storing information. Solutions to make FSAs concurrent are typically external to the FSA itself, either as support logic or as a set of tools to make them concurrency safe. FSAs provide no easy way to synchronize multiple modular behaviors together, unless reactive system or a synchronous language, such as Argos is used. Mealy machines on their own cannot resolve LP2. Koza points to the need for self-organization of hierarchies (Koza, 1994, p93), but this can be achieved using HFSA.

A.3.3 Conclusions

Radial basis neural networks and linear neural networks mirror neuronal processing and are good examples of artificial neural networks (Sutton and Barto, 1998 and Streeter, Oliver and Sannier, 2006). FSA are useful because they mirror neuronal and cell processing and are simpler to implement (Minsky, 1967; Rechenberg, 1965 and 1973; Fogel, Owens and Walsh, 1966, cited in Koza, 1999, p386; Ray, 1991 cited in

Appendix A. Evolution and Emergence

Koza, 1992, p647). A lack of hierarchies in current limitations of artificial neural networks means that they cannot resolve LP2. A lack of hierarchies in FSA means that they cannot solely be used to resolve LP2 and must be augmented with a hierarchy.

A.4 Hierarchical Nature of Brain Networks

Albus provides a model of the brain as a hierarchical signal processing and control system, with distinct, interconnected networks: the *receptive field hierarchy* and the *event hierarchy* (Albus, 2000; 2008 and 2010b, p193).

In the model (Albus, 2008; Albus, 2010 and Sherman and Guillery, 2001) a synapse is an electronic gate with complex biochemistry and the site for long-term memory. A *neuron* is a computational element with non-linear processes on many inputs and decisions. An *axon* is an active fiber connecting one *neuron* to others (it transmits a scalar variable on a publish-subscribe network with bandwidth ~ 500 Hz). There are two kinds of *axons*, a *driver* and a *modulator*. A *driver neuron* preserves topology and local sign data vectors or arrays of attributes and state-variables i.e., images, objects, events, attributes and state which provide color, shape, size, position, orientation and motion. A *modulator* does not preserve topology nor local sign. It consists of context and broad-cast variables, addresses and pointers. These modulators select and modify algorithms, set parameters and define relationships. *Axons* work in *receptive field hierarchies*, which are defined by *driver neurons* flowing up the processing hierarchy and are relatively fixed (as will be seen, *receptive field hierarchies* can be considered to be Piagetian figurative schemes). *Event hierarchies* are defined by modulator activity that can establish or break networks of “belongs-to” and “has-part pointers,” which typically occur in ~ 10 ms (as will be seen, these *event hierarchies* can be considered to be Piagetian operative schemes).

Neurons are clustered into functional units, which perform arithmetic or logical operations, correlations, convolutions through which they coordinate transformations (Albus, 2008; 2010). These *neuron clusters* can be modeled as FSA, and are constructed into hierarchies with set of rules, a grammar using direct and indirect addressing (Albus, 2010). The cortical structure (cortex) consists of a 2d cortical sheet approx. 2000 cm^2 and is partitioned into functional *regions*. Each *region* is segmented into arrays of columns are arranged into hierarchical layers. Each of these *cortical hyper-column* and the *thalamic loop* (which provides the raw processing power) is a *cortical computational unit* (CCU). Each CCU can be implemented as FSA with memory, with the cortex containing approx. 1 million CCUs, which provides the resource constraint for a simulation.

Research by Grainger (Grainger, 2006a, p18 Granger, 2006b) confirms a HFSA approach to model brain structures as well as providing evidence of the correlation to the simple model of Drosophila (Miesenböck, 2008, p52; Shang, Claridge-Chang, Sjulson, Pypaert and Miesenböck, 2007, p601). For Granger, the entire striatal system can be understood in terms of four subassemblies (Granger, 2006a, p18):

- 1) Cortex \rightarrow matrisome projections (acting)
- 2) Cortex \rightarrow striosome projections (evaluation/sensing)
- 3) SNc dopamine (DA) projections to both matrisomes and striosomes (learning through reinforcement)

Appendix A. Evolution and Emergence

- 4) TAN projections to matrisomes (exploration) which relates to predictive modeling (planning)

A.4.1 Conclusions

Recent brain research confirms the approach of using HFSA to model the brain structures (Albus, 2008; Grainger, 2006a; Granger, 2006b) with twin pillar hierarchies of perception and executive (action), and using a processing model that mirrors *Drosophila* (Miesenböck, 2008, p52; Shang, Claridge-Chang, Sjulson, Pypaert and Miesenböck, 2007, p601; Granger, 2006a; Granger, 2006b) that supports acting/sensing, learning and planning.

A.5 The Need to Validate a Biologically Plausible Model

Since the set of constraints (LP1 – LP7) dictate a biologically plausible model, there is a need to confirm that it is possible to use the research model in a number line world. It is therefore necessary to confirm that an artificial neural network implementation can identify regularities in the environment e.g., point, line, direction, penState, and so be capable of addressing LP4.

A.6 Mechanisms for Structuring Networks

Human behavior is determined by the way in which humans assign values to different situations and it is suggested that reward based reinforcement learning facilitates this behavior (Sutton and Barto, 1998). The benefit of reinforcement is that it enables autonomy in a system (Sutton and Barto, 1998, p16), and provides a mechanism to resolve LP3 and LP5. Reinforcement learning also enables resolution of the temporal credit assignment problem (Streeter, Oliver and Sannier, 2006, p2) structural credit assignment problem (Streeter, Oliver and Sannier, 2006, p4) as well as resolve the problem of exploration vs. exploitation (Streeter, Oliver and Sannier, 2006, p4). Although identified as problems, they will not be tackled in this research. Any future research which seeks to fully resolve the learning paradox will need to account for them.

A.6.1 Biological Basis of Reinforcement Learning

Neurophysiology experiments have revealed that neurons in the mammalian midbrain use the neurotransmitter dopamine to process information about rewards and reward-predicting stimuli (Schultz, 1997, p2). Cannon and Bseikri (Cannon and Bseikri, 2004, p742), and classify these rewards into 2 categories, though their research questions if dopamine is actually required for brain reward processing. The categories are (1) hedonistic, and (2) sensory cues, which have an associated incentive value. This second category of association must be learned since most sensory cues are initially neutral and the incentives associated with the cue are initially not known.

For rewards to affect future behavior, sensory events must first be detected. Research by Schultz confirms this to be true: dopamine neurons in certain brain areas, including the substantia nigra in the midbrain, are

Appendix A. Evolution and Emergence

sensitive to these sensory events (Schultz, 1997, p2; Schultz, 2000, p201). These rewarding events are sensed through the same sensory modalities as all other sensory events; there are no special “reward sensors.”

The capability to predict sensory information is a fundamental brain function (Schultz and Dickinson, 2000, p475–495), with the purpose being: given a set of sensory inputs (event occurrences) an association is learned that maps sensory events to rewards, enabling the prediction of future rewards. The temporal gap between event occurrence and future consequences allows the brain to predict what the world will be like after some time. Sensory prediction entails calculating the discrepancy between actual and predicted sensory inputs (Rao and Ballard, 1999, p79). In Rao and Ballard’s model, only unexpected sensory information is passed to higher levels of sensory processing. This is ideal from an information theory standpoint since only uncertain events contain information. In the model of *Drosophila*, this is the predictive model (Miesenböck, 2008; Shang, Claridge–Chang, Sjulson, Pypaert and Miesenböck, 2007).

Research shows that norepinephrine neurons respond to reward–predicting stimuli i.e., the events signal that a reward is imminent, but not to primary rewards. This suggests that brains are able to form chains of reward–predicting stimuli that eventually lead to primary rewards. Climbing fibers, stretching from the inferior olive to Purkinje neurons in the cerebellum, output error–driven teaching signals (Schultz, 1997, p26). This activity constitutes a prediction error in motor performance: at first dopamine is released only when the reward is received; however, given some reward–predicting stimulus, over time the dopamine neurons respond to the reward–predicting stimulus rather than the reward itself. If the reward is then omitted, the dopamine activity drops below baseline at the time of the predicted reward.

It has been shown that attention is proportional to prediction errors, that is if an event is surprising (surprising event), it elicits more attention from an individual (Tobler, O’Doherty, Dolan and Schultz, 2006, p302). If reward occurrences always match predicted rewards, then nothing is learned. During exploratory learning, these surprising events guide individuals to spend more time focused on those situations where their predictions are inaccurate and where there is more information.

By evaluating outcomes (by comparing the actual state with the predicted state) learning can be improved. In effect, learning from errors increases the learning rate. In the biological system, this occurs through the dopamine learning rate of the association between sensory events and rewards which depends on the dopamine prediction error (Schultz and Dickinson, 2000, p473). Prediction enables the imagining/planning features that allow individuals to “adapt to the predictive and causal structure of the environment” (Schultz and Dickinson, 2000, p474). Further, the generation and use of prediction errors “may contribute to the self–organization of goal–directed behavior” (Schultz and Dickinson, 2000, p495). Prediction–based learning and behavior can take place over large time scales: events that occur now may affect behavior in the distant future. These are the structural credit assignment problem and the temporal credit assignment problems discussed earlier (§ A.6).

Research on human motor control provides evidence that humans learn to predict 7 ½ times faster⁸³ than to being able to control the outcome of their actions⁸⁴ (Flanagan, Vetter, Johansson and Wolpert, 2003, p147). Actually, this capability to predict becomes critical to the development of an autonomous system, since prediction of performance enables the system to construct a model of the world and leads to differentiation of constructs such as the “truth of some of its assumptions about the world.” What is needed is a mechanism to implement this reward processing within a simulation.

A.6.2 Actor–Critic Temporal Difference Learning as an Implementation Mechanism

Biological reward signals in the mammalian mid–brain are very similar to modern computational learning models (Schultz, 2000, p201; Montague, Hyman and Cohen, 2004, p760). Montague, Hyman and Cohen suggest that behavioral control is inseparable from the valuation of objects and circumstances. In effect, value functions are necessary for behavior. In addition, TD Learning “has informed both the design and interpretation of experiments that probe how the dopamine system influences sequences of choices made about rewards” (Montague, Hyman and Cohen, 2004, p760).

Table A–1 below summarizes the relationships of the biological basis for rewards to TD Learning:

Table A–1 Relationships of temporal difference learning (TD Learning) to biology

#	Behavior Based Rewards	Mapping to TD Learning	Source
1	Hedonic experiences.	direct external rewards.	Cannon and Bseikri, 2004, p742.
2	Sensory cues that have an associated incentive value.	Matches states with high estimated value.	Cannon and Bseikri, 2004, p742.
3	Midbrain dopamine neuron reward, dopamine learning rate depends on the dopamine prediction error.	Matches TD Learning and the notion of learning from errors/uncertainty.	Suri and Schultz, 1998, p350; Schultz, 2000, p201; Schultz and Dickinson, 2000, p475; Fiorillo, Tobler and Schultz, 2003, p1898.
4	Prediction response processing (Predict sensory information) – used in imagining and planning and self–organization of goal–directed behavior.	TD Learning.	Schultz and Dickinson, 2000, p474, 495.

⁸³ Predict: mapping motor commands to expected sensory inputs, a forward model.

⁸⁴ Control: mapping desired outcomes to the motor commands required to produce those outcomes, an inverse model.

Appendix A. Evolution and Emergence

#	Behavior Based Rewards	Mapping to TD Learning	Source
5	The structure of the basal ganglia: the striatum corresponds to the actor and the nigrostriatal dopamine neurons correspond to the critic.	Resembles the actor critic model in TD Learning. Multiple architectures have been developed to support different approaches for support in unpredictable environments.	Suri and Schultz, 1998, 351; Khamassi, Lachèze, Girard, Berthoz, Guillot, 2005, p131.
6	Behavioral control is inseparable from the valuation of objects and circumstances.	State value functions are necessary for behavior.	Montague, Hyman and Cohen, 2004, p760.
7	Dopamine neurons in certain brain areas, including the substantia nigra, are sensitive to these sensory events – –but there are no special reward processors.	Sensory stimuli are sensed as events.	Schultz, 2000, p201.
8	Sensory prediction entails calculating the discrepancy between actual and predicted sensory inputs (Rao and Ballard, 1999).	Only unexpected sensory information is passed to higher levels of sensory processing.	Rao and Ballard, 1999, p79.
9	Humans learn to predict the outcome of actions before they learn to control the outcome itself.	Prediction maps motor commands to expected sensory inputs, (forward model) whereas control maps desired outcomes to the motor commands required to produce those outcomes (an inverse model).	Flanagan, Vetter, Johansson and Wolpert, 2003, p147.

From this, it can be seen that the process of TD Learning closely matches research results on midbrain dopamine neuron reward processing and prediction response processing in animals (Suri and Schultz, 1998, p350; Schultz, 2000; Schultz and Dickinson, 2000, p474; Fiorillo, Tobler and Schultz, 2003, p1898). More specific research by Suri and Schultz (Suri and Schultz, 1998, p353) present results from a computational model of TD Learning of dopamine neurons that encode reward prediction errors. Their research suggests that after conditioning, reward-predicting stimuli essentially become rewards themselves.

A.6.3 Conclusions

The dopamine–norepinephrine reward process is a mechanism for autonomously building internal neural structures based on external interaction. Actor–critic temporal difference learning is described as a simulation mechanism for reward processing that can be used to build artificial neural networks that learn incrementally. In doing so, LP3 can be resolved. It is also possible to use reinforcement learning with FSA and HFSA to build their network, though this research will use a more constrained version.

A.7 Mechanisms to Model Evolution

The underlying idea of computational evolutionary strategies (ES) is the separation of solutions for a particular problem e.g., a machine from descriptions of those solutions in memory (De Jong, 2006). Evolutionary strategies (ES) such as evolutionary computation (EC), explore this multi–parameter space of solution alternatives for a particular problem, by means of a population of encoded strings (where each program stands as an alternative) which undergo variation (crossover and mutation) and are reproduced in a way as to lead the population to ever more promising regions of this search space (selection) (De Jong, 2006, p27). ECs work on the descriptions and not on the solutions themselves, that is, variation is applied to descriptions, while the respective solutions are evaluated and the whole (description–solution) is selected according to this evaluation.

If it is assumed that evolution tangled with the learning paradox, then there may be some interesting areas where evolutionary strategies have provided solutions that can be appropriate. Table A–2 summarizes the basic differences between EC (Holland, 1975) and the capabilities found in nature:

Table A–2 Differences between early evolutionary computing and biology (Holland, 1975)

#	Capability of nature	Natural Mechanism	Evolutionary Computation Strategy
1	Self Organization.	No separation of description/solution: attractor behavior of state determined dynamic systems are memory–less, self organization based on internal dynamics that model abstract, internal, characteristics of matter.	Description/Solutions are separated into description (memory) and solution (individual in a population) using aspects of von Neumann’s non–trivial self–reproducing scheme ⁸⁵ – increasing complexity is emergent (Mitchell, 1992, p7) but they do not self organize, since they rely on selective external pressure (memory based selective organization scheme) and are based on stochastic (population) variation.

⁸⁵ In von Neumann’s non-trivial self-reproduction, reproduction is through a complicated process of self-inspection.

#	Capability of nature	Natural Mechanism	Evolutionary Computation Strategy
2	Variation.	Variation is non stochastic: transition rules are state determined.	Variation is stochastic: transition rules are not state determined.
3	Selection.	Selection is internal and external	Selection is external to the population
4	Fitness.	Environment-organism coupling (problem) dictates selective pressures and the solutions are the organisms themselves.	ECs reduce the problem to the optimization of a multi-parameter function (DeJong, 2006, p27).

From table A-2 one realizes that early ECs were modeling very different design principles. As a variety of EC, genetic programming (GPs) formulates a search mechanism that maps an input-output structure to solve problems in symbolic processing, optimal control, automatic programming, empirical discovery and machine learning using a computer program (Haynes, Sen, Schoenefeld and Wainwright, 1995, p3). Evolution uses chance (randomness) but chance has no memory. However, organisms pass on their particular capacity to acquire certain characteristics, rather than any of the characteristics they actually acquire i.e., the GP algorithm produces a skill, but the skill is projectible (Dennet, 1995, p323). GPs could potentially solve the problem of developing number-sense and have two main advantages. First, the programs are understandable by humans who can make predictions about the future behavior of the system (Koza, 1992). This is because the GP produces a program parse tree that represents a syntactically correct computer program where branches in the parse tree represent functions which take argument's; the leaves represent zero argument functions, variables or constants. Second, GPs have been shown to be very general and successfully applied to a wide variety of problems. Examples include emergent behavior, subsumption, evolution of building blocks and hierarchies (Koza, 1992, p527-553).

The emergent behavior in ECs and specifically GPs is from the repetitive application of seemingly simply, self-governing rules which can lead to complex behavior in cellular automata, fractals and Lindenmayer systems (Koza, 1992, p329). Using subsumption, the collective effect of the asynchronous local interactions of the primitive task-achieving behaviors, provide an alternative mechanism for the control of robots (Brooks, 1991). In the emergence of building blocks, the primary strategy is to use existing symbolic function sets including "AND, OR, NAND and NOR" (Koza, 1992, p529) to facilitate the emergence of automated function definitions. When these emergent function definitions, are enabled with the capacity to call existing functions, hierarchies emerge (Koza, 1992, p529). Overall, these strategies improve the ruggedness and performance of ECs.

Appendix A. Evolution and Emergence

A.7.1 Evolutionary Strategies: Where is the Problem?

Like other evolutionary strategies, GPs fall foul of having too much built in, i.e., they are purely symbolic in nature and do not work within the limits of Fodor's argument. First, many solutions use production systems to govern selection. Second, the search for the optimal individual of the population is a search across of all individual members of a population, hence GP's fail **LP1**. Further, since "learning" occurs at the selection of members of a population, that exhibit the correct behavior, learning does not occur incrementally, hence GP's fail **LP5**.

However, Jacob presents a mechanism by which ECs can build binary mealy machines (Jacob, 2001). In this evolutionary approach, the evaluation functions could be constructed using techniques similar to TD-Learning to construct their networks. Collectively these could resolve **LP3**, **LP5** and **LP7** as well as incorporate a model of *Drosophila*.

A.7.2 Conclusions

Evolutionary computing, as a software simulation mechanism, can be used to evolve software in similar ways to evolution, and when used with binary mealy machines resolve **LP3**, **LP5** and **LP7**. There are two constraints: (1) the FSA (as propositional components) must be separated from their hierarchies to adhere to **LP1**, and (2) the storage and concurrency issues must be resolved.

A.8 Resolving Issues with FSA

A.8.1 HFSA

Given the biological basis of FSA, the usefulness of FSAs can be dramatically increased by extending them with structuring and communicating mechanisms. Through the introduction of the "Statecharts" (Harel and Pnueli, 1985), nested state machines using an implicit hierarchy. Orthogonality (which induced parallelism between state machines) and re-usability of components in different contexts popularized their use. HFSA tend to be extremely useful, primarily because they capture the links between the states and can hide complexity through nesting. A hierarchy allows a state of an FSA to be refined into another FSA as a set of sub-states. Concurrency allows multiple simultaneously active states, each refined as an FSA with the capability to communicate through a common messaging mechanism.

One way to implement FSA and HFSA is using a reactive system framework. As a primitive synchronous language for reactive systems⁸⁶, Argos is primarily a set of Boolean mealy machines which follow the synchrony hypothesis. Argos provides a set of operators that allow for parallel combination of these

⁸⁶ Reactive systems are deterministic whereas interactive systems are not. Most reactive systems employ two types of activities: data handling and control. At one extreme are the signal processing systems for data intensive systems and at the control end are pipelines and buses of operators. Argos, like Esterel are tailored to control-intensive applications (Berry, 2000, p4).

Appendix A. Evolution and Emergence

hierarchical automata (Maraninchi and Remond, 2001, p61; Maraninchi, 1992, p553) and specify the syntax and correctness of programs. These operators are the Cartesian product operator (AND)^{vii}, the refinement operator^{viii}, the inhibiting operator^{ix} as well as the combination of operators that produce temporized states^x and incorrect compositions^{xi}.

The proofs of Argos semantics have been defined, but not specifically included here (Refer to Maraninchi, 1992, p560 and Maraninchi and Remond, 2001, p75). What is clear is the need to support a robust “error handling” process within the reactive system, to prevent “inappropriate programs” from developing too far, as well as a process to check “loops.”

By following the control constructs of Esterel, Argos controls the logical behavior of what the program must do and cannot do. Included within this range of behavior are the Boolean operators of AND, NOT, NOR, NAND, XOR, which are constructed as an arrangement of a hierarchical Mealy machines. Included within this is a Boolean algebra, as a mathematical system for the manipulation of variables that have the values of true or false, as well as the orders of precedence. Also included are the processes of the Boolean identities and DeMorgans laws. Though predicates have been integrated with FSA (Van Noord and Gerdemann, 2001, p263), it is certain mealy machine implemented in Argos cannot generate a form of predicate logic.

Argos, like Esterel allows incorrect compositions of FSA. Through the use of competitive processes, these “incorrect programs” can be routed out, but not completely. A suitable selection/detection process is given by Maraninchi and Remond, “The quality of the detection mechanism is good, if it does not reject correct programs, too often” (Maraninchi and Remond, 2001, p65). Argos also contains mechanisms to optimize hierarchical FSA into single sequential FSA as is shown with their Modulo-8 counter example (Maraninchi and Remond, 2001, p66-68). The approach taken in this research is to evolve hierarchical FSA and not optimize the resulting FSA.

A parallel can be drawn between this research approach and that of evolvable hardware (Greenwood and Tyrrell, 2006, p12). The use of evolution in this approach uses a variety of stochastic search algorithms using evolutionary computing. Self-adaptation is used to compensate for failures or anticipated changes in the operational environment, and “As complexity increases, so do faults and errors in these systems” (Greenwood and Tyrrell, 2006, p1). In this respect, the simulation is evolved, with the solution fit for its purpose of adapting to a number line world but not necessarily an optimal design. The simulation is tolerant of faults, i.e., it can identify errors and adapt by creating new networks of hierarchical and FSA to process information. Ultimately, the systems produced are unverifiable. Like most current, circuit adaptation systems it could lack have issues with scalability, risk mitigation, fault detection and isolation (FDI), fault recovery and fitness evaluation (Greenwood and Tyrrell, 2006, p97). Fortunately, because of the way the simulation is generated (it does not include loops), the solutions it produces are autonomous and their adaptation are based on external reward, it is an open system.

Appendix A. Evolution and Emergence

A.8.2 Further Problems

A further problem associated with FSA and this is true of any system that attempts to resolve the learning paradox is the set of criteria defined by Fodor (Fodor, 1980), namely, that the system cannot introduce a concept that the system does not already know. In this situation, the notion of a variable e.g., point, line etc., must be provided in the system as an innate scheme (§ 2.3.5 Piaget on Innate Scheme). Thus, to confirm the Piagetian model and working within Fodor's constraints as well as adhere to the reactive notion of Argos, a set of additional characteristics must be added. These include the innate schemes of variable, inhibiting operator, the refinement operator—which is seen as the capability to create a hierarchy, the Cartesian product operator and the temporized states operator. It is assumed that these operators are part of the Piagetian process.

However, given the set of operators, Argos combines FSAs with a synchronous/reactive model to deliver a perfectly synchronous semantics in the sense of Esterel. Maraninchi and Remond show that Argos is compositional, with respect to the trace equivalence of Boolean Mealy machines and FSAs in Argos have been shown to count (Maraninchi and Remond, 2001, p68). As a mechanism, Argos provides the grounding of a reactive system from which FSA, which has already been shown to exist in nature and it provides a way of evolving functional programs that control the behavior of the simulation and develop, as a reactive system, concepts, in a very Piagetian sense.

A.8.3 Conclusions

In developing a simulation of *Drosophila* using binary FSA, a mechanism to resolve some of the core issues, such as storage and concurrency, is needed. A reactive systems approach using Argos, provides this mechanism, which, when combined with evolutionary computing (Jacob, 2001) and a form of reinforcement (Streeter, Oliver and Sannier, 2006), provides a way of resolving LP6. The combined structure can address LP1, LP2, LP3, LP5, LP6 and LP7.

A.9 The Need for a Model of To Test for Emergence

A model of emergence by which to evaluate this approach is needed. Crutchfield distinguishes 3 notions of emergence: intuitive emergence; pattern formation and intrinsic emergence (Crutchfield, 1994a; 1994b).

A.9.1 Intuitive Emergence

In *intuitive emergence* something new appears, but the “pattern is always referred outside the system to some observer that anticipates the structure via a fixed plate of possible regularities” (Crutchfield, 1994a, p2).

Since the solution presented in this research is a closed multi-agent system, the emergence is of coordinated behavior that develops to control the internal processes. These internal processes, as HFSA would be observable. This is the approach taken in this research.

Appendix A. Evolution and Emergence

A.9.2 Pattern Formation

In *pattern formation*, an observer identifies “organization” in a dynamical system. In this approach, the “relationship between novelty and its evaluation can be made explicit by thinking always of some observer that builds a model of a process from a series of measurements” (Crutchfield, 1994a, p3). The basic problems are these: (i) how does one understand the process; (ii) how does one measure structure; (iii) when does one attribute order to chaos (information or error) and (iv) when does the datum confirm or deny the hypothesis i.e., how can assumptions be changed? Crutchfield (Crutchfield, 1994a, p5) suggests that a balance between information and error can be obtained by using measures of information processing structure from computational theory (Crutchfield and Young, 1989, p68 cited in Crutchfield, 1994a, p7) by building a model of the process as a discrete time series using the language defined for the process enabling the creation of hierarchical models of the environment. The tools available are dynamical systems theory, the mechanisms and structures inherent in computation theory, and inductive inference as a statistical framework.

A series of innate structures and a set of primary reactions as an implementation model of *Drosophila* using HFSA have been identified in this thesis. These are the structures and processes which will be used to act/sense, learn and plan in a number line world. The construction of a statistical framework for analysis of emergence of pattern formation (Crutchfield, 1994a; 1994b), this research believes is beyond its scope.

A.9.3 Intrinsic Emergence

In *intrinsic emergence*, the system itself capitalizes on patterns that appear i.e., to innovate using inductive inference on the existing models. Crutchfield implements these calculi of emergence as a set of processes that:

- 1) Measures the structure of the minimal machine reconstructed from a given process in terms of the machines size;
- 2) An algorithm for reconstructing the machine given an assumed model class; and
- 3) An algorithm for reconstructing the hierarchical machine, which uses inference to detect regularity in a series of increasingly accurate models. These regularities are part of the new representation.

In Crutchfield’s model, it seems the arguments turn on distinguishing several levels of interpretation:

- 1) A system behaves;
- 2) That behavior is modeled;
- 3) An observer (in the system) detects regularities and builds a model based on prior knowledge;
- 4) A collection of agents model each other and their environment;
- 5) Internal observers as agents themselves create models and try to detect changes, using a method such as the scientific method.

In this framework, emergence is the process of discovery, which can be considered as an underlying model for biological cognition. This was highlighted by Chomsky’s as the “deep structures” and as linguistic universals (§ 2.1.2 See Chomsky on universals). When one views the neural and HFSA model of *Drosophila*.

Appendix A. Evolution and Emergence

One can see aspects of planning that can meet Crutchfield's comment that there is an overwhelming need "for a theory of biological structure and a qualitative dynamical theory of its emergence" (Crutchfield, 1994b, p47).

It is assumed in this research, that it is possible for a simulation using a model of *Drosophila*, to measure its predictive model of the environment and from this to hierarchically reconstruct it (Crutchfield, 1994a, p49; Crutchfield, 1994b, p47). It is through this reconstruction that a new class of structure emerges. The construction of a statistical framework for analysis of emergence of pattern formation (Crutchfield, 1994a; 1994b) is beyond the scope of this research.

A.9.4 Conclusions

Anticipated is that the model of intuitive emergence provided by Crutchfield provides a mechanism by which the proposed model in this thesis can be evaluated. A strong foundation for a solution to attempt to resolve the learning paradox has thus been built.

A.10 Conclusions

There is emergence of structure in the collective behavior of ant colonies, for instance in searching for food (Johnson, 2001, p45). This is evidence of them solving **LP5** and **LP7** as an optimization problem. This process can be simulated using swarm techniques where simple rules, embodied in agents using vast populations are shown to exhibit novelty in opportunistic and noisy situations (Dorigo, Birattari and Stutzle, 2006, p34). The current models are limited, since their simulations do not produce the more complex "ant social model," and there is a need to model aspects of human cognition. In addition, these swarm rules do not support a Piagetian model, since they do not have the capability to develop their learning process and so fail **LP6**.

A model of *Drosophila* provides an internal engine that is more complex than a simple set of ant rules (Miesenböck, 2008, p52; Shang, Claridge-Chang, Sjulson, Pypaert and Miesenböck, 2007, p601). Since many aspects of the basic pathway are probably conserved evolutionarily, the fly studies research provides a simple model for cognition that can provide a mechanism to resolve **LP6**. This model needs to be implemented as a neural model, and it is further assumed that the *Drosophila* model has similarities to a Piagetian model of learning and development.

Given that evolution has a solution to the learning paradox in the neural networks in fruit-fly and human brains and cell networks, a simulation mechanism is needed. Two approaches are suggested, artificial neural networks and FSA. Radial basis neural networks and linear neural networks mirror neuronal processing are good examples, but a lack of hierarchies in current implementations means that they cannot resolve **LP2**. FSA are useful solutions because they mirror neuronal and cell processing and simpler to implement. Recent brain research confirms the approach of using HFSA to model the neural processing and brain structures (Albus, 2008; Grainger, 2006a; Granger, 2006b).

Appendix A. Evolution and Emergence

Since the set of constraints (**LP1 – LP7**) dictates a biologically plausible model be used in a number line world, this is a need to confirm that this approach can show emergence of structure by identifying regularities in the environment e.g., point, line, direction, penState, and so be capable of addressing **LP4**.

The simulation in this research needs a mechanism for structuring its internal network to resolve **LP3**. The dopamine–norepinephrine reward process (implemented as actor–critic temporal difference learning) is one such method. It is also possible to use reinforcement learning with binary FSA and HFSA to build their network, though a more constrained version will be used.

Since evolutionary computing is a software simulation mechanism that parallels evolution. It is assumed that this is an appropriate method to use with HFSA to resolve **LP2**. By separating out the propositional components (FSA) from the hierarchical components (HFSA) the simulation sidesteps **LP1**. Crutchfield's model of intuitive emergence provides a mechanism for detection, and thus a strong foundation for a solution to attempt to resolve the learning paradox with a biologically plausible model.

Appendix B

B. Models of Development

In this appendix, a more detailed analysis of hybrid models of cognitive development.

B.1 Hybrid Model

B.1.1 SOAR

“Most of the agent’s competence arises from the encoded knowledge representations (i.e., the set of rules) that SOAR’s mechanisms operate on. Thus, agent knowledge representations must be created to realize any high-level intelligent capability” (Wray and Jones, 2005, p80).

From its beginnings in the study of human problem solving, the SOAR cognitive architecture was developed as an implementation platform to explore the requirements for general intelligence and to demonstrate general intelligent behavior (Laird, Newell and Rosenbloom, 1987; Laird and Rosenbloom, 1995; Newell, 1990; Rosenbloom 1995 cited in Wray and Jones, 2005, p56).

Despite eight major revisions between 1982 and 2007, SOAR provided a multi-agent framework for cognitive modeling that was limited to symbolic processing (Wray and Jones, 2005). In response to observations of human reasoning, additions were made to SOAR in two major areas which are described in the following architecture diagram (figure B-1) and in the accompanying text.

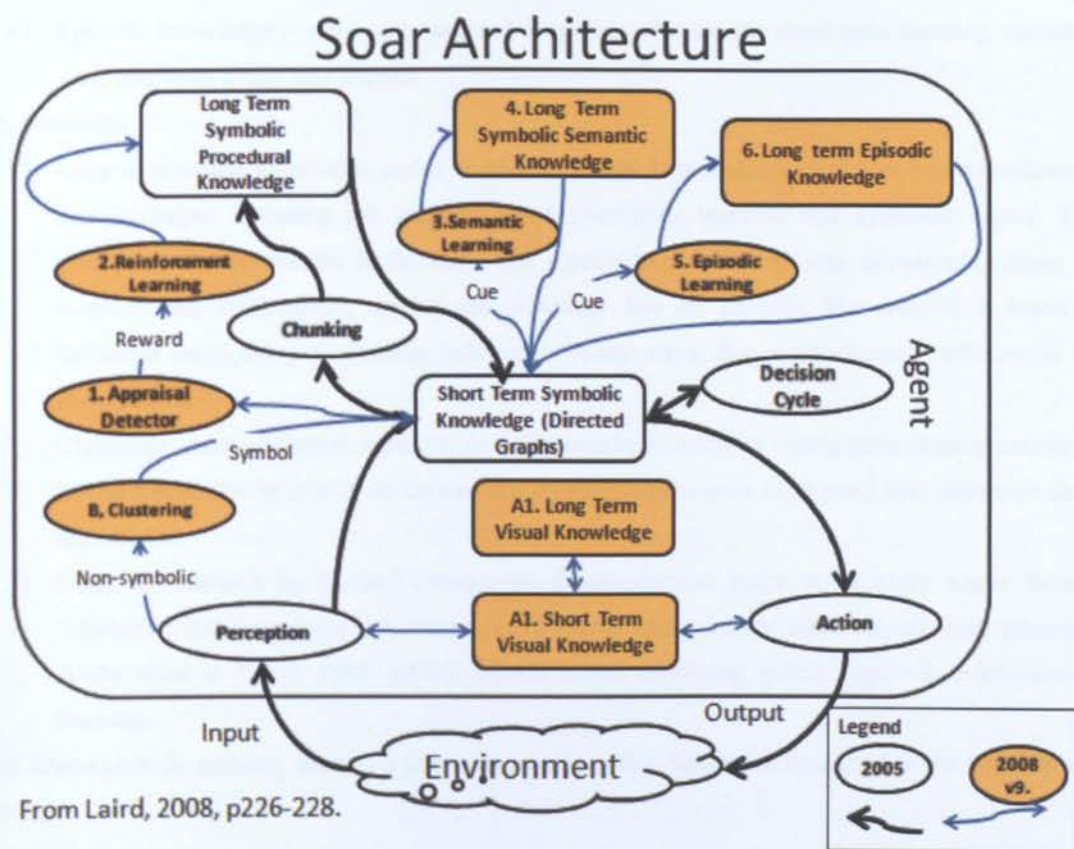


Figure B-1 A depiction of the SOAR Architecture (Laird, 2008, p225-228).

Symbolic:

- 1) Appraisal detector – generates emotions and internal reward for reinforcement using a computational implementation of an appraisal theory. In SOAR, appraisals lead to emotions, emotions influence mood and mood with emotion determine feelings. Individual appraisals produce either categorical or numeric values, which combine to form the intensity of the current feeling. The intensity becomes the numeric internal reward for reinforcement.
- 2) Reinforcement learning is invoked when a new operator is selected and adjusts the selection of actions in an attempt to maximize rewards using numerical values of the operator evaluations. RL applies to every operator selection, on every decision, even when there is no impasse.
- 3) Semantic learning – Uses cues in short term memory to search for best partial match. This knowledge is independent of when and where it was learned. The retrieval of semantic memories occurs on the input phase, so that “the results are available on the next input cycle” (Laird, 2008, p232).
- 4) Semantic knowledge – symbolic facts as declarative knowledge.
- 5) Episodic Learning as working memory activation is updated when rules fire and holds the recent and relevance of short term memory elements. The retrieval of episodic memories occurs on the input phase, so that “the results are available on the next input cycle” (Laird, 2008, p232).

- 6) Episodic knowledge – temporally ordered snapshots of symbolic short term memory, including what and when a rule was learned.

Sub Symbolic

- 1) Long term visual knowledge and a short term visual knowledge – depictive representations of visual images, including the inherent space combining numeric and symbolic values. This processing is co-symbolic, in the sense that it provides an alternate way of reasoning about the world (Laird, 2008, p232), making use of spatial data for analysis. The retrieval is based on deliberate recall using a symbolic referent. In many ways, this is significantly different to the Piagetian view.
- 2) Clustering – uses statistical regularity in a winner take all retrieval (using classification networks) in the perception of inputs to create new symbolic structures (symbols) that represent these regularities.
- 3) Using the research by Richard Granger on thalamocortical loops in the brain where there is “clustering and successive sub-clustering of inputs using winner takes all, circuits” (Granger, 2006a cited in Laird, 2008, p232), SOAR added clustering which improves reinforcement learning.

Like other symbolic systems, there is a processing cycle within SOAR, as described in the figure below (figure B–2):

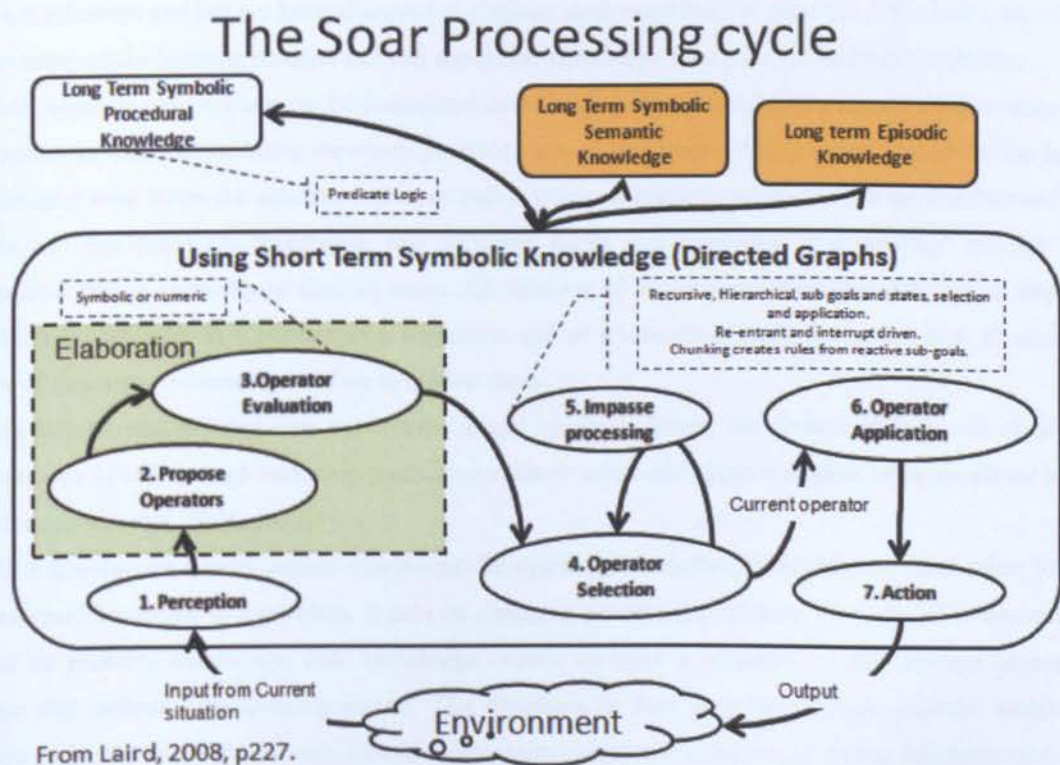


Figure B–2 The SOAR decision cycle – SOAR v9 2008 (Laird, 2008, p227)

Appendix B. Models of Development

- 1) Perception / Input: Changes to perception are processed as inputs and sent to short term memory.
- 2) Elaboration: Rules compute entailment on short term memory objects including the proposal of operators that are appropriate to the input perception.
- 3) Elaboration: Rules are evaluated and preferences are created for the proposed operators using numeric or symbolic values.
- 4) Operator Selection: A fixed decision cycle uses the preferences of the generated operators and selects the current operator.
- 5) Impasse Processing: If the preferences are insufficient for making a decision, SOAR automatically creates sub-states and recursively applies and selects them to resolve the impasse. This impasse processing allows the Agent to deliberate within the same processing cycle. Chunking, the learning mechanism in SOAR creates rules from sub-goals in impasse processing. This meta-reasoning provides the common strength in the architecture. Chunking is based on traces of rule firings in sub goals.
- 6) The actions of the current operator are performed by rules that match the current situation and the current operator structure. By supporting multiple rules firing in parallel and in sequence supports an expressive mechanism for encoding operator actions.
- 7) Actions are passed to the motor system.

There are obvious limitations in this approach. All the knowledge is encoded in predicate logic, which is too rich a mechanism and has not been observed in children until much later in their life. This is at a time after they have resolved the learning paradox and can apply that knowledge to actively solve novel problems.

SOAR assumes any problem can be formulated as a problem space containing a set of possible states and a set of operators which individually transform possible states to goal states. The series of steps from the initial state to the goal state forms the solution (behavior path). When an impasse occurs (within the PSCM module), sub-goals and sub-states are determined and decisions made, this automatic “sub goaling” enables task decomposition from a hierarchy of distinct states. All versions of SOAR implement this ontology of impasse types, which is seen both as a performance advantage and as a limitation, since it assumes that all decision making is of this form, it cannot generalize to hidden states.

Even through the addition new sub symbolic and symbolic forms, the control of behavior of SOAR agents (v9 circa 2008), through encoding production rules to select and apply operators using predicate logic, remains the same (Laird, 2008, p232).

SOAR implements a maintenance component for agents to remember historical states and when SOAR finds a successful solution to a problem, it uses its chunking process (its learning mechanism) to convert the results of its problem search into new knowledge, which includes a summary of the impasse process, a preference that defines why learning occurs. The drawback is that learning must be realized within the constraints imposed by its multi-step inductive reasoning algorithm (chunking) which has been carefully constructed to provide the desired result (Young and Lewis, 1999, cited in Wray and Jones, 2005, p75).

Appendix B. Models of Development

Chunking as an inductive learning mechanism is seen as a severe limitation since it does not allow for generalizations to hidden states. Learning in SOAR models occurs by repeatedly considering whether to attempt to retrieve an object from declarative memory (knowledge search) or to construct the desired object (problem search). Similar to JESS, SOAR performs knowledge searching using the Rete algorithm that is limited because unless one chooses object representations carefully, chunking can result in rules that are computationally expensive to match (Tambe, Newell and Rosenbloom, 1990 cited in Wray and Jones, 2005, p76). As a consequence of the parsimonious nature of SOAR research, to find the minimal set of mechanisms that can be used to realize the full range of intelligent behavior, SOAR defines a low-level machine for implementing algorithms (Wray and Jones, 2005, p60). In this way, representations are built up (using chunking, as the main learning mechanism) from low-level components.

By integrating a number of different algorithms into comprehensive cognitive models, SOAR has been used for developing intelligent systems in wide areas of domains with albeit with accurate but biased solutions. These fine grained encoded models allow for the depiction and prediction of human behavior from an agent simulation in the areas of expert systems, intelligent control; natural language; executable models of human behavior for simulation systems; as well as exploration into the integration of learning and performance; concept learning in conjunction with performance; learning by instruction; learning to correct errors in performance knowledge as well as episodic learning (Wray and Jones, 2005, p53).

A SOAR agent can be described through descriptive observations of its behavior (at its “knowledge level”). Assumptions can be made about the representation of its knowledge, which exist as a set of symbols (at its “symbol level”), while the fixed mechanisms and representations from the architecture, which are used to realize the symbol system (at its “architecture level”). These layers allow distinct separation of processing and as a consequence of its symbol system, universal computation is possible. Thus, “it should be sufficient for any application requiring intelligent behavior (assuming intelligence can be captured in computational terms)” (Wray and Jones, 2005, p56).

The core problem is that the symbol system is encoded into the system - there is no mechanism in SOAR (pre 2005) to develop symbols. In fact, few SOAR models (excepting SOAR-Teamwork, NL-SOAR) reuse any existing knowledge or symbols of prior implementations. Even the current enhancements to SOAR including the development of more efficient methods of capturing, encoding and reusing agent knowledge; through neural networks; encoding and reusing agent knowledge through the use of language extensions; design patterns or through the use of more scalable knowledge bases do not resolve the symbol grounding problem. This failure to account for symbol development is explained away as the fundamental limitation of symbol systems (Wray and Jones, 2005, p58).

Developments of SOAR including the use of decay, is seen as critical, since it tends to reduce interference (Altmann and Gray, 2002, Cited in Wray and Jones, 2005). There is an obvious parallel to the use of decay constants in reinforcement learning. SOAR supports productions (in long-term memory as a series of procedural rules), assorted memory objects (in a blackboard style) in highly structured short term/working memory, stored as a directed graph, where each object is a triple of identifier, attribute and value. The memory

Appendix B. Models of Development

graph is structured into state partitions where every state has a pointer to its super state, as the root node in the graph and preferences. In preference memory, candidate operators compete for selection e.g., acceptable, best using various semantic procedures. SOAR operators are composed from all of these and consequently span all three memories (Wray and Jones, 2005, p62). There is separation between these layers: productions cannot access preferences directly to determine, for instance if one set of operators is better than another (Wray and Jones, 2005, p64); all that the production can do is to test the preference for acceptable use.

SOAR operators are conceptually equivalent to operators in state based planning systems, such as STRIPS (Fikes and Nilsson, 1993 cited in Wray and Jones, 2005, p65). When an operator is selected during the execution of a decision procedure, SOAR creates an operator object in working memory, which can then trigger productions that produce post-conditions (as actions). Like most agent systems, SOAR acts in a sense-decide-act (SDA) cycle. During the input phase, SOAR invokes the input function using its blackboard (short-term memory). The reasoning within the decision cycle is focused on the selection and application of operators, which contribute new commands to execute in the output phase (Wray and Jones, 2005, p67). Each decision consists of three phases: elaboration (the iterative firing using SOAR truth maintenance system to compute “all available logical entailments”⁸⁷); the decision phase (where preference semantics are used to determine the operator to apply) and finally the application phase (where operator application production rules fire, resulting in the creation of output commands). The decision phase maps directly to the BDI loop, common to agents (Wooldridge, 2000) with some differences. SOAR does not support plans and its control process (decision and reconsideration) is fixed, unlike most BDI implementations.

Whereas Belief Desire Intention (BDI) agent architectures do not make explicit distinctions between justified assertions and persistent assertions, they usually use a form of belief revision (Gardenfors, 1988 cited in Wray and Jones, 2005, p76). SOAR makes these distinctions explicit, using its operator construct. In using this uniform approach to reconsideration, SOAR systems must be designed to be reentrant at every step of execution – the reconsideration (reason maintenance) algorithm assumes that it is cheaper to retract and repeat some problem search, rather than attempting to decide whether some deliberate section should continue to be supported (Wray and Laird, 2003, p362). BDI systems in contrast allow execution of plans of arbitrary length or even traditional procedures within a single pass of the control loop, with the immediate benefit of supporting complex plans and procedures. The lack of a plan representation in SOAR means that the SOAR developer must design a representation for the problem space that supports inherent interruption and reentrancy.

Given the research problem, it seems possible to implement the problem within the SOAR landscape, however, the child when faced with learning fraction problems obviously does not have this luxury.

A problem associated with the uniform representation within SOAR is that any new problem representation must be implemented as a symbolic representation, rather than simply reusing the architecture - the architecture does not readily support decision and conflict resolution that does not map directly to current decision process. Further, it is inefficient if not impossible to split the execution of a procedure over multiple

⁸⁷ The operator object itself fails Fodor's arguments for LP1, since it goes against all members of a set.

Appendix B. Models of Development

deliberation steps. Which leads to compromises to ensuring the problem is constructed to fit possible solution spaces (Wray and Laird, 2003, p79).

SOAR is a production system, where each production is expressed as a set of “If Then” rules using predicate logic rather than the propositional logic used in procedural programming languages and is specified by a series of conditions and a set of actions. Conditions are matched against memory objects and when all conditions are satisfied, the rule actions are executed. This execution usually involves changing and exchanging memory objects. SOAR does not map individual production rules to operators such as occurs in traditional rule based systems such as OPS5 (Forgy, 1981), CLIPS (Giarratano, 1998), JESS (Friedman–Hill, 2003) and production system cognitive architectures such as ACT–R (Anderson and Lebiere, 1998) and EPIC (Meyer and Kieras, 1997; Kieras and Meyer, 1997). This provides an advantage in that SOAR can bring to bear any available knowledge relevant to the current problem i.e., conflict resolution occurs by choosing between candidate operators, allowing the decision to be mediated by the preferences, rather than relying on the syntactic features of the situation. To do this, SOAR uses associative pattern matching in its parallel match–fire production system (Wray and Jones, 2005, p69).

Typical for rule-based systems, every change to an agent’s context must be the result of a deliberate commitment. This leads rule-based systems to over–commit to courses of actions i.e., it is difficult for them to change their beliefs. This causes the perception that rule-based systems are “brittle.” In deference, SOAR applies reason maintenance to every non-persistent object in its blackboard (short-term memory) and uses this to update its beliefs about the world. This allows the agent to be responsive to their environment (Wray and Jones, 2005, p71) by allowing the objects in memory to decay over time and thus, no longer apply. It also allows the agent to generally recover from interruptions. SOAR uses an activation and decay mechanism, with similar properties that allow the objects in memory to decay over time (Anderson and Lebiere, 1998 cited in Wray and Jones, 2005, p72).

For SOAR, preference and deliberation - which for an agent in a dynamic environment are the capabilities to deliberate and commit to goals from the set of all possible actions and goals - is the selection mechanism it uses as its automatic reasoning system in its decision cycle. The advantage of rule-based systems (RBS) is obvious: an RBS that has a precondition and action in a single rule has to reason across its entire rule base. In contrast, SOAR reasons across options – which include its historical information (Anderson and Lebiere, 1998 cited in Wray and Jones, 2005, p72).

Laird raised the question, “does the addition of neuro architectures such as Neuro–SOAR (Cho, Rosenbloom and Dolan, 1991 cited in Laird, 2008, p225), which is similar to the ACT–R/Liebre hybrid (Taatgen, Juvina, Herd, Jilk and Martens, 2007 in Laird, 2008, p225) improve performance?” Given the nature of the learning paradox and the symbol grounding problem, it is obvious that SOAR is a step in the right direction. Even through the addition of new sub symbolic and symbolic forms, the control of behavior of SOAR agents (v9 circa 2008), through encoding production rules to select and apply operators using predicate logic remains the same (Laird, 2008, p232).

Appendix B. Models of Development

SOAR is a competitor to the research because in using Richard Granger's research on thalamocortical loops, they have understood that the way to resolve the learning paradox is to use what ones has, in this case statistical regularity as the basis of unitizing from continuous values (Granger, 2006a cited in Laird, 2008, p232). However, SOAR falls on the innateness argument, because the system would need to be constructed with the knowledge to solve the problem using a predicate logic - a logic more powerful than the "logic" observed in children at the age when they are observed to develop fraction-sense and so resolve the learning paradox (Wray and Jones, 2005, p62). In doing so, it fails Fodor's argument for LP1.

B.1.1.1 Bottom-Up Learning using SOAR-R

In addition to the framework of the symbol system hypothesis (§ 2.7.2.3), SOAR is based on the problem space hypothesis (Newell, 1980b). The problem space hypothesis assumes that all symbolic cognitive activity can be modeled as heuristic search in a symbolic problem space. For Newell, reasoning, problem solving and decision making can all be captured as searches in these appropriately defined problem spaces.

The aim of bottom-up learning is to reduce the amount of innate knowledge that an agent would need to be constructed with. In this respect, using SOAR as a bottom-up cognitive model (Georgeon, Ritter and Haynes, 2009) is innovative and as a result it can partially and fully support many of the features required to resolve the learning paradox (LP1 – LP5). By grounding symbols in activity, they resolve the symbol grounding problem (Harnad, 1990) by allowing the agent to self-organize its behavior, such that one can infer goals and knowledge in the agent, when, we, observe their activity.

By not using many of the abstract features of SOAR such as: impasse mechanism, chunking, reward mechanism, stochastic exploration policy, and use of operators or the pattern matching principles. It affords the bottom-up learning agent another knowledge representation, one that it has developed for itself (Georgeon, Ritter and Haynes, 2009). It is noted that the implementation lacks the capability to support recursive schemes in SOAR and this may well be a limitation to fully implement a Piagetian model.

Bottom up learning (Georgeon, Ritter and Haynes 2009), assumes a constructivist epistemology using a Piagetian schema mechanism. Extending Drescher's schema model of context, action and result (Drescher, 2002) into context, action, expectation. Bottom up learning using this mechanism to support sequence learning, which is understood to be hierarchical ordering of schemes using hierarchical reinforcement learning (Sun and Sessions, 2000 cited in Georgeon, Ritter and Haynes 2009).

Hierarchical sequence learning transforms {context, action, expectation} into:

context → sub-scheme + status

action as intention → sub-scheme + status

satisfaction as expectation → satisfaction (context) + satisfaction (intention)

weight → number of actions

It is through this process that sequences of schemes emerge. Though useful and including features such as trial and error learning (evolutionary), knowledge is grounded in praxis (pragmatic) and is self-oriented

Appendix B. Models of Development

(teleological); it lacks schema management. Its great benefit is that it allows the development of the Piagetian invariants, in their case, the development of the knowledge that invariant objects exist in the environment.

Bottom Up Learning in SOAR: Where is the Problem?

As discussed, bottom up learning in SOAR is a significant improvement on other cognitive models. However, they miss the mechanism of the knowing circle, the process of evolutionary trend, and the developmental trends (Furth, 1969) and so fail LP6 and LP7.

B.1.2 PolyScheme

PolyScheme is based in part on the belief that multiple computational mechanisms are required for general intelligence (Cassimatis, Trafton, Bugajska and Schultz, 2004, p19). By integrating multiple representations with reasoning methods and problem solving techniques, PolyScheme builds intelligence into physical robots. PolyScheme specifically addresses the problems of data fusion by (1) generating a coherent model of the environment through senses; (2) grounding symbols through integrating perceptual structure to symbolic structures; and (3) flexibly combining reasoning, planning, perception and action using Bayesian networks, Stanford Research Institute Problem Solver operators (STRIPS) and partially observable Markov decision processes (POMDP). PolyScheme uses a propositional logic for communication between its specialists (agents within a multi-agent system). Representation is in a form of directed graphs and a neural language is used by its “specialists” to deal with perception using these networks. Symbol grounding is resolved through associating the perception of objects by its neural specialists to its rule specialists, which contain rules in propositional logic (Cassimatis, Trafton, Bugajska and Schultz, 2004, p19). PolyScheme uses a number of inbuilt algorithms including case based reasoning; prediction; counterfactual reasoning; backtracking; backward chaining; theorem proving; truth maintenance and Bayesian inference. By assigning all structures with a basic set of relations about time, space, events, identity, causality and belief PolyScheme makes a strong semantic commitment.

B.1.2.1 PolyScheme: Where is the Problem?

PolyScheme suffers from using symbolic forms of reasoning using predicate functions, specifically in its inbuilt logic and algorithms. It thus it fails Fodor’s arguments (LP1). It does however address the symbol grounding problem in a realistic way, yet this has not been shown to exhibit emergence (LP2), nor mirror observations of child development (LP4).

B.1.3 Neuro-SOAR

“Most of the agent’s competence arises from the encoded knowledge representations (i.e., the set of rules) that SOAR’s mechanisms operate on. Thus, agent knowledge representations must be created to realize any high-level intelligent capability” (Wray and Jones, 2005, p80).

Appendix B. Models of Development

From its beginnings in the study of human problem solving, the SOAR cognitive architecture was developed as an implementation platform to explore the requirements for general intelligence and to demonstrate general intelligent behavior (Laird, Newell and Rosenbloom, 1987; Laird and Rosenbloom, 1995; Newell, 1990; Rosenbloom 1995 cited in Wray and Jones, 2005, p56).

Despite eight major revisions between 1982 and 2007, SOAR provided a multi-agent framework for cognitive modeling that was limited to symbolic processing (Wray and Jones, 2005). In response to observations of human reasoning, additions were made to SOAR in for symbolic and sub-symbolic processing including Neuro-SOAR (Cho, Rosenbloom and Dolan, 1991 cited in Laird, 2008, p225).

B.1.3.1 Neuro-SOAR: Where is the Problem?

Neuro SOAR (Neuro-SOAR (Cho, Rosenbloom and Dolan, 1991 cited in Laird, 2008, p225) is a significant improvement on other cognitive models. However, they miss the mechanism of the knowing circle and the process of evolutionary trend and the developmental trends (Furth, 1969) and so fail LP6 and LP7.

B.1.4 ACT-R

Even though ACT-R has both sub-symbolic and symbolic layers, new features have only recently been added to ACT-R to support the learning of new rules (Taatgen and Lee, 2003, cited in Anderson and Lebiere, 2003, p600). Yet these rules are produced by compiling production rules. ACT-R has a declarative memory for facts (that is accessible by introspection and represents conscious awareness of facts) and a procedural working memory of production rules that is not available by introspection (Anderson, Bothell, Byrne, Douglass, Lebiere, Qin, 2004). ACT-R theory defines a data structure, the “chunk,” for memory as “... the units in which knowledge is represented in working memory...” (Anderson, 2007). The system performs pattern matching on perceptions and facts, mediated by the real-valued activation levels of objects. Learning in ACT-R involves creating new facts and productions, as well as updating activations and utilities associated with these structures. In effect “Complex cognitive processes are achieved by stringing together a sequence of such (production) rules by appropriate setting of goals and other writing to working memory and by reading from working memory” (Anderson, 2007; Anderson, Bothell, Byrne, Douglass, Lebiere, Qin, 2004). The system executes these rules to affect the environment or alter declarative memory (Anderson, 2007; Anderson, Bothell, Byrne, Douglass, Lebiere, Qin, 2004).

B.1.4.1 ACT-R: Where is the Problem?

Research on ACT-R has led to the comprehensive computational theories of a wide variety of human phenomena including brain activity (Anderson, 2007, cited in Laird, 2008, p233). However, they have ignored many of the cognitive capabilities in SOAR including episodic memory, emotion, and visual imagery. Like other hybrid models, they fail Fodor’s arguments (LP1) and further, are not seen to develop the stage like variation observed in developing intelligence (LP4).

Appendix B. Models of Development

B.1.5 CogAff

Stemming from a need for an ontology of intelligent agents, the cognitive affect (CogAff) architecture is a balance between central, sensory and motor processes and between reactive (alarm driven), deliberate and meta-management processes (Sloman, 2001). By superimposing the two–three way divisions, a sufficiently rich schema can be assembled that exhibits properties of emotions (Sloman, 2001, p42). CogAff is an architectural specification that does not define any requirements for implementation, with individual researchers using a variety of techniques to implement the information subsystems. The hierarchical perception mechanism, long term associative memory, motive activation process and a set of persona is blended with three central processing systems: (1) A reactive mechanism which uses hierarchical condition–action associations to sense internal and external conditions and respond by producing external state changes or internal state changes using a variable threshold attention filter. (2) A deliberation mechanism that supports planning, decision making and “what if” reasoning using a hypothetical representation system. (3) A reflective or meta management mechanism provides the ability to attend to, monitor and evaluate the affect of the operation of the system on the state of the agent i.e., its emotional state (Sloman, 2001, p42).

B.1.5.1 CogAff: Where is the Problem?

CogAff (Sloman, 2001, p42) fails because it does not specify an implementation that could resolve the learning paradox, it also fails Fodor’s arguments (LP1). Further, it has not been shown to develop its learning process (LP6).

B.1.6 RCS

RCS (Albus, Pape, Robinson, Chiueh, McAulay, Pao and Takefuji 1992) is an architectural framework for developing intelligent physical agents. Expertise resides in a hierarchical set of knowledge modules, each with its own long–term and short–term memories. Knowledge representation is heterogeneous, including frames, rules, images and maps. Modules operate in parallel, with a sensory interpreter examining the current state, a world model predicting future states, value judgment selecting among alternatives and behavior generation carrying out tasks. Higher–level modules influence their children in a top–down manner, whereas children pass information back up to their parent modules. Similar to human control systems, there are constraints of hard deadlines for problem solutions; reliability long with limited observability; concurrent operations.

Heavily influenced by biological models of motor control such as the Marr–Albus model of the cerebellum and the cerebellar model articulation computer (Albus, 1975), the real–time control system (RCS) (Albus, 1993) is a reference model architecture. It is not a design nor is it a specification to implement a specific system; what it does provide is a comprehensive six step methodology for engineering solutions as well as a software reference library (RCS, 2009).

RCS partitions the control problem into 4 basic elements: (1) Behavior generation, which is understood to be task decomposition. (2) World modeling, an internal representation of the external world including

Appendix B. Models of Development

predictions, what if scenarios and updates to its world model. (3) Sensory processing including recognition of 1:1 correspondences and the comparison of entities. (4) Value judgments are evaluation functions of perceptions and plans based on cost-benefit, which are an essential part of planning or learning, as well as perception, knowledge, learning, costs, recursive state estimation, planning and execution.

RCS clusters elements into computational nodes within hierarchical layers, with all control nodes sharing a generic node model e.g., a computational hierarchy as a command tree which when executed produces a behavioral hierarchy of state/time trajectories. Based on the underlying biological model, RCS makes use of simultaneous and continual planning, cyclic re-planning as well as execution of tasks at all levels of the hierarchy.

Using an entity frame data structure to represent task knowledge, RCS utilizes a propositional logic to control relations as state variables and to define rules (Albus, 1991). It also has considerable apriori knowledge encoded into the system. RCS can also use and store images and maps across all its modules. An integral part of many RCS implementations is the use of human designers to interact with the system, as well as to provide apriori knowledge for autonomous interaction. RCS uses CMAC as a form of neural network associative memory, which has been utilized in reinforcement learning. Practical use of the model is limited by the amount of memory size required.

RCS utilizes a theory of intelligence in which “the ability to act appropriately in an uncertain environment, where appropriate action is that which increases the probability of success and success is the achievement of behavioral goals” (Albus, 1991, p27). This theory adheres to a number of tenets. First, both goals and success criteria are generated in the environment, external to the agent. Second, higher levels of intelligence require the abilities to recognize objects and events, store and use knowledge about the world and to reason about and plan for the future. Third, advanced forms of intelligence have the abilities to perceive and analyze, to plot and scheme, to choose wisely and plan successfully in a complex, competitive, hostile world. The amount of intelligence can grow through programming, learning and evolution. Intelligence is the product of natural selection wherein more successful behavior is passed on to succeeding generations of intelligent systems and less successful behavior dies out. Natural selection is driven by competition between individuals within a group and groups within the world.

When implemented within a machine, Albus defines a set of criteria that any solution would need to adhere to: the amount of intelligence is determined by the computational power of the computing engine, the sophistication and elegance of algorithms, the amount and quality of information and values and the efficiency and reliability of the system architecture (Albus, 1991).

B.1.6.1 RCS: Where is the Problem?

RCS (Albus, 1975; Albus, Pape, Robinson, Chiueh, McAulay, Pao and Takefuji 1992; Albus, 1991; RCS, 2009) possesses many characteristics that are necessary in any autonomous system, but lacks the capability to generalize and to change the way it learns over time. It also suffers from having too much built in including its abstract symbolic language. By using frame based knowledge structures and inbuilt logics, RCS fails Fodor’s

Appendix B. Models of Development

arguments (LP1). It also has not shown the capability to develop the stage like variation observed in developing intelligence (LP4).

Appendix C

C. Piagetian Models and Mathematics

This appendix describes a Piagetian model of mathematics by Copeland (Copeland, 1974), in § C.1. Some notes on the development of fraction sense – as a classification of WE6, which is an extended part of the *worked example* – in § C.2 as also provided.

C.1 Piagetian Mathematics

To be able to claim adherence to a pure Piagetian approach, it is necessary to first identify and define the relations (concepts) that Piaget observed in children as they develop mathematics (Copeland, 1974). The definition of the key relations and dependencies are listed in this appendix.

Since the implementation affects how these relations are constructed, this analysis could only take place once the implementation strategy (HFSA using a dialectic method) had been selected. Therefore, the following analysis assumes that the reader has an awareness of the implementation strategy. Analysis of the relations suggests that a set of principles need to be taken into account and applied along the anticipated developmental path that the simulated student will take.

First, in early development, the system must treat the number line and its components as continuous values, being based on perceptual shape, not value. The differentiation into units comes later and is used in counting (Piaget, 1965, p80 cited in Copeland, 1974, p91). Second, Piaget's genetic epistemology defines a series of relations that are observed to occur, with stage like variation throughout the life of the child. As the child (simulated student) matures, these relations become increasingly interiorized and generalized. At any point in time, these relations are combined using the processes of assimilation, accommodation and equilibration to produce the behavior. Further, since the implementation develops FSAs, with HFSA to marshal their processing power, it is conceived that the "dominant machines" that are used for adaptation will be composed of the more generalized machines. For example, if Piaget is correct (Copeland, 1974, p79–80), then those machines that produce serialization should evolve into counting machines. This may not be the case; only the execution of the system can determine this. As the student develops inductive and deductive generalizations e.g., a conservation relation or a relation such as seriation, these "rule based mechanisms" are implemented as HFSA which consume more primitive machines, thus continuing the execution of binary machines. Within this approach, deductive reasoning is considered to occur after a generalization has been

Appendix C. Piagetian Models and Mathematics

made e.g., $\frac{1}{a} + \frac{1}{b} = \frac{1}{b} + \frac{1}{a}$ the child then makes use of this induction and without examining the details of a

specific case e.g., $\frac{1}{3} + \frac{1}{4} = \frac{1}{4} + \frac{1}{3}$, and concludes that they are the same. The generation of deductive reasoning allows the student to make further judgments (Copeland, 1974, p203).

The current implementation mechanism of assimilation and accommodation are separated into act and sense (using existing schemes), reinforce (learning new schemes), and predictive model (performs planning, generating more complex/generalized schemes) with equilibration and disequilibration exigent from an overflow or exception handled by an executive. One can ask, “How do the relations generalize?” There are two approaches: (1) In the first approach, each relation exists in a class with its own defined evaluation function, which over time increases its complexity as a function of the increasing “processing time available” by the machine. There seems to be little evidence for this, since, the existence of the evaluation functions would be common across all students and all students would show evidence of all features, all of the time. (2) In a second approach, as the system executes using a common evaluation function, these relations appear in the field of execution. Evidence for this rests with Pascual–Leone theory of constructive operators (Pascual–Leone and Goodman, 1979), which suggests that as a Piagetian system executes, its capacity increases with a determinate factor over time. This increasing capacity allows for more complex structures to be built; these complex structures are the generalizations that appear e.g., conservation of number, but also the appearance of apparent dependencies in the relations. Therefore, if seriation really does occur before counting, then there should be evidence of this in the execution of the machines. Further, all things being equal, it should be possible to assess the M–Demand of these machines as they execute, learn, evolve and be able to relate this to the M–Capacity of the simulation itself. Comparison should also be possible against the benchmark from the rational number project.

The following section defines the necessary relations, which are believed to be required for the development of rational numbers from integers based on observations by Piaget’s on the appearance of certain features. It also includes their anticipated implementation method, as well as the age at which the relation is initially observed e.g., Logical Connectives at age 0.

C.1.1 Logical Connectives

The logical connectives are basic FSA that are evolved as the student executes. These binary machines (OR,AND,NOT,XOR,NAND,NOR and XNOR) are incorporated into Argos style HFSA which marshal the transformation of information from and to the environment. These logical connectives are also utilized by the student themselves, which is an interesting mix of internalized and externalized action.

C.1.2 Spatial Concepts

Piaget asserted that a child’s understanding of spatial concepts is topological (being composed of proximity, enclosure, separation and order) rather than Euclidian, which treats distance, length, angles, lines

Appendix C. Piagetian Models and Mathematics

etc as if they were “real elements in space” (Copeland, 1974, p226; p232) by observing that the child’s pictorial renderings of their world are topological rather than Euclidian. Children of the age of 4 ½ to 5 evaluate discontinuous quantities as if they were continuous values (Piaget, 1965, p80 cited in Copeland, 1974, p91). At age 5, the child is using their perception of the environment and can for instance, estimate the number of “shorter lines” that are required to match “3 larger lines,” however, the space between the arrows is taken into account. In addition, they have discretized the environment (Piaget, 1965, p80 cited in Copeland, 1974, p91). Thus, the system initially can only utilize topological spatial concepts in continuous spaces using proximity, separation, order and surrounding. Through maturation, the environment becomes discretized by the student and so Euclidian awareness comes to the fore that enables them to identify point, length and distance, leading to conservation of numbers.

This conversion between continuous spaces to discrete spaces has been shown to occur through the use of the Verve development toolkit and linear neural network. This research assumes that this switch (to a discrete environment) has already occurred, hence the unitized values passed in the environment to agent interface.

C.1.3 Proximity

Proximity (“nearby-ness”) is the first perception based mechanism that develops in a child, with the nearness of an object being crucial to them (Copeland, 1974, p214).

Proximity is the appearance of a class of machines that can determine the relative distance from a given point. Once these machines with distances emerge, the relative proximity of number line segments based on penDown can be created. Typically, the proximity of machines is reused to form the basis of unitization.

C.1.4 Separation

As a child grows older, they are able to separate one object from another or to distinguish the parts of an object (Copeland, 1974, p214).

Separation is the appearance of a class of machines that can separate out objects in a class of objects (Copeland, 1974, p214). It is the appearance of machines for particular facets of the environment penUp, penDown, Left, Right, and Movement. Overtime, these machines can be combined together in hierarchies and identify new and more complex structures. Typically, the separation of machines is reused.

C.1.5 Order

The spatial relation of order develops in the young child and enables them to, for instance, order a set of beads on a line (Copeland, 1974, p214).

Order is the appearance of a class of machines that can, from a provided set of number line segments, determine which ones precede and come after other ones.

Appendix C. Piagetian Models and Mathematics

C.1.6 Surrounding

The fourth spatial relation that develops in a child is surrounding (otherwise referred to as enclosure) for example ABC, where B is enclosed by or between A and C. (Copeland, 1974, p214).

Surrounding is the appearance of a class of machines that can, from a provided set of number line segments, determine which ones surround other ones.

C.1.7 Conservation of Point

Though not mentioned by Copeland (Copeland, 1974), conservation of point is achieved through the generation of a prediction machine that can match the stop action on a number line. Since it is a point, direction is ignored.

C.1.8 Conservation of Length

Perceiving a straight line and being able to reproduce it, are two separate problems (Copeland, 1974, p248), with both not fully complete until the *concrete operations stage* (age 7 / 8). The basic problem is that the child age 6, when moving along a number line, only considers the end points, thus two lines, one straight and one crooked, which end and start at the same point, are considered the same length. Similarly, adults take it for granted that the ruler maintains its length when being used, but this is not the case with young children.

Conservation of length is achieved through the process of reinforce (Furth, 1969, p154 and p162), which through imitation (as an process of assimilation) and reconstruction (as a process of accommodation) builds a prediction machine that can match the observed input of a number line segment and wraps this within HFSA Argos schemes inside a Piagetian container. Typically, this machine includes the penState (of penDown), direction (left and right) and movement (10403 units) of the number line segment. This generation may reuse existing proximity machines. This research assumes that the student has already switched between topological to Euclidean geometry (Copeland, 1974, p226, p232) which typically occurs at the age of (4/5 years).

Along with the order relation, the preponderance of these unitized machines is critical to the development, since they are reused repeatedly.

C.1.9 Conservation of Distance

When subdividing line segments, young children in Stage 1 are unable to conceive of the length of a journey as an interval (distance) between two points (Copeland, 1974, p267). In stage 2, they use a trial and error approach and finally in stage 3 (age 7 ½), they have developed a generalized procedure for subdivision and substitution and therefore have a generalized procedure for conservation of length and conservation of distance (the two of which, are inseparable).

The establishment of conservation of distance is seen as a critical aspect of the system and occurs when a HFSA can generate machines of unit lengths, e.g., 1, 2 and 3 etc. Conservation of distance is achieved through the generation of a prediction machine that can match the observed input for number line segments that have a

Appendix C. Piagetian Models and Mathematics

penState of penUp, since this is the distance between two points on a number line. This also takes into account the direction taken.

C.1.10 Equivalence

The equivalence relation is critical to the development of logical thought and occurs throughout the developmental stages (Copeland, 1974, p84). Often called “one to one correspondence,” it is seen to develop early. The bead problem is evidence of the issues associated with equivalence (Copeland, 1974, p85–87 and p91–92).

Equivalence is the appearance of a machine that compares the properties of one machine against another in binary using an AND machine (a FSA) that is wrapped up into an Argos scheme. The internalization is an action of accommodation, which produces the “binary value” to be compared, with different actions for the different combinations penState (penUp, penDown), direction (left, right), stop as well as length. When the values are the same, it returns a 1, when they are different it returns a zero. In more complex arrangement, the predictive model tests the states, inputs and outputs of various machines to see if they are equivalent. In essence, the equivalence relation determines the truth of validity of a set of machines.

Equivalence is used by other relations, such as order. I suspect that equivalence is part of a developmental trend of increasing discretization (differentiation) of the environment. This would suggest that a child is constantly comparing and contrasting elements in the environment, even from the earliest stages.

C.1.11 Subdivision

The concept of arithmetical fractions implies that all the parts are equal, yet young children may leave a part of the whole undivided. Piaget suggested that there could be no thought of fractions until subdivision is exhaustive, there can be no remainders, yet young children ignore this (Copeland, 1974, p158). When the concept of subdivision is operationalized, then children realize the dual nature of fractions - that they are part of a “whole” but also a “whole on their own” - which can be further subdivided (as a nested series). Since fractions relate to the whole from which they came, the whole remains invariant (is conserved). Conservation of a whole is an essential condition for operational subdivision (Copeland, 1974, p159) and occurs at 6 to 7 years of age.

Subdivision occurs when the system generates HFSA that can subdivide a given length without any remainders. Two questions remain, “What is the determination of when that point is reached” and “Is this by symmetry or length, or some other factor?” Another significant factor is the determination of when a value cannot be subdivided. Exceptions such as these will need to be detected when encountered. The antithesis of subdivision is continuity and when one is developed, so should the other. A concern is that subdivision is an operation like counting, it arises from number–sense.

Appendix C. Piagetian Models and Mathematics

C.1.12 Substitution

Substitution allows a child to apply or substitute one part (the measuring rod) upon another (the object being measured) an appropriate number of times, thereby building a system of units (Copeland, 1974, p252).

In its earliest manifestation, substitution is aligned with subdivision within PMIAS to enable the generation of machines that can develop to show the relation of seriation as well as intermediate operations such as “greater than,” “less than.” As explained in the following diagram (figure C-1):

A simple 'Less Than' machine build using Subdivision and Substitution which later forms the rational numbers

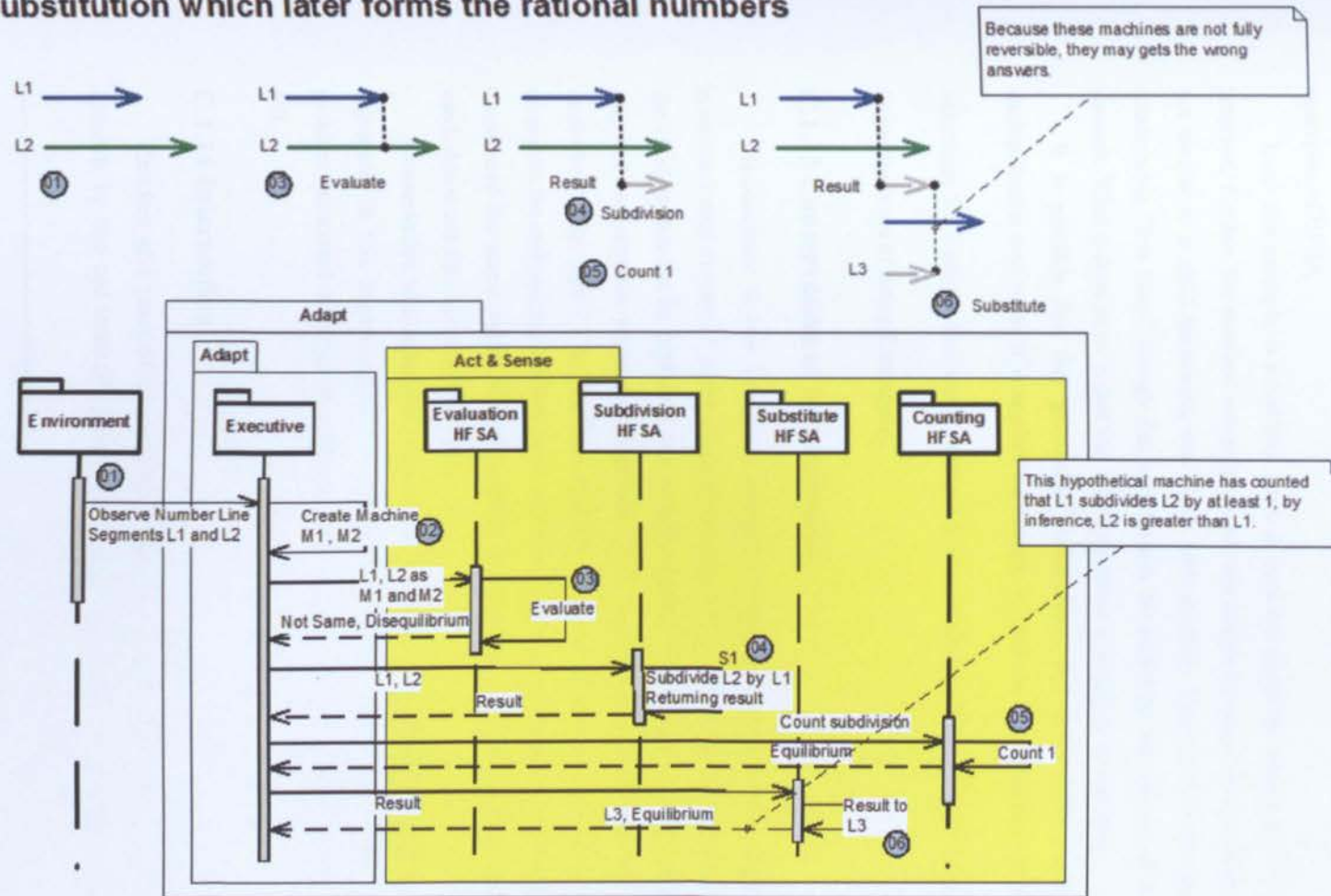


Figure C-1 A UML sequence diagram that shows the Piagetian mathematical relations of substitution and subdivision being used in a less than machine to determine if two counted values are different. It shows the interaction of the executive with the process of act and sense.

Appendix C. Piagetian Models and Mathematics

In this, subdivision and substitution are coordinated together in 6 steps to determine the relationship between L1 and L2. In this, L1 is “less than” L2. This example emphasizes how the system can learn one process (in this case equivalence) and reuse it. The processes of counting, subdivision and substitution are all examples of HFSA.

Even this example is simplistic, since it does not show the length of L3, nor how the counted value is returned. Further, the standard unit is not used; this length L1 could be any value. In this respect, the processes are similar to a child measuring one block with another. There is also no shared moniker that defines the relationship “less than,” though this could easily be added by the teacher, if this behavior is observed long enough. What it does show is that this whole machine is acting as an operator.

It is possible that this geometric⁸⁸ subdivision and substitution could lead to both division and multiplication and when it is applied repeatedly, it leads to the possibility of developing machines that can determine for instance that a unit value of 1, can be divided a number of times as in $1 = \frac{1}{3} + \frac{1}{3} + \frac{1}{3}$, which forms the basis of rational numbers.

C.1.13 Conservation of Measurement

Measurement in one dimension produces length and distance. The capability to measure is not fully completed until at least 11 years of age (Copeland, 1974, p247–70). This capability is based on earlier work by the child, which can be explained in a simple example: perceiving a straight line and being able to reproduce one, are two separate problems (Copeland, 1974, p248) and both are not fully complete until the *concrete operations stage* (age 7 / 8). The basic problem is that the child age 6, when moving along a number line, only considers the end points, thus two lines, one straight and one crooked, which start and end at the same point are considered the same length. Whereas adults take it for granted that a ruler maintains its length when being used, this is not the case with children.

Conservation of measurement appears as machines that can accurately measure the length of number line segments. In this implementation, conservation of length and measurement are commingled. The classic problem associated with this phenomena is the Piaget’s bead problem (§ figure 1–1, figure 2–1 and figure 6–19).

C.1.14 Intersection

Children of 8 years of age, still have difficulty with intersection e.g., the intersection of those numbers divisible by two and three (Copeland, 1974, p59). As a facet of classification, intersection includes disjoint

⁸⁸ Geometric in the sense that the HFSA, built from observations of the number-lines, use the Euclidian length property of the observation – as it is, converted to binary for the equivalence operation and counting.

Appendix C. Piagetian Models and Mathematics

sets, intersecting sets (all or some), one set or subset. Piaget considered that intersection is critical for the conservation of the whole. This is a tentative conclusion.

This process would be available through the appearance of machines that could link sets of characteristics between sets of machines. It is conceivable, that such a machine would appear.

C.1.15 Reversibility

It is the freeing from perception that marks the beginning of an operation on the intellectual rather than perceptual level and it is a lack of reversibility that limits the development of a child that merely uses perception (Copeland, 1974, p90–92). Examples of reversibility relation include: $1 + 2 = 3$ is reversible to $2 + 1 = 3$ (Copeland, 1974, p 90–92) which is observed at the age of 7 years (Copeland, 1974, p124). This particular example is dependent on the conservation of number.

Reversibility is achieved through the appearance of a machine that can fully reverse the operation (action) of existing machines e.g., reverse the order of a given set of number line segments, reverse the order of a set of number line segments, reverse the set of actions. It finally appears in the predictive model and enables the simulated student to test various scenarios. As such, it is a machine, which consumes the outputs and inputs of other machines. It can be chained together into a new HFSA.

C.1.16 Inclusion Relation

The inclusion relation in the addition of numbers is concerned with ensuring conservation of the whole (Copeland, 1974, p112, p117 and p120). As an operation of classification, the inclusion relation provides a mechanism to use an existing classification and determine if a set of input / outputs / machines / monikers fit within that classification.

Inclusion is generated in the predictive model to determine if the characteristics fit within a property of perception (initially) and then within the bounds of executing machine e.g., a machine tests for primes and it determines if the outputs of a machine i.e., a number is a prime or not. As such, this inclusion relation is dependent on conservation of number, reversibility, classification, seriation in varying degrees.

C.1.17 Transitivity

The logic of relations includes the transitive relation as in “If $B > A$ and $C > B$ then $C > A$ ” (Copeland, 1974, p12; p108 and p170). It is this ordering of relations, around 7 years of age that is critical to further development.

For transitivity to occur, there must be a mechanism to internalize a classification value to associate “A something” and “B something”. This may be an internal value associated with the output of surrounding. The system also needs to identify, in this instance, the logical connective “greater than.” It also needs to be able to connect (serialize) the output to another machine. Transitivity is achieved through the appearance of a generalized machine, constructed in the predictive model. It presupposes a number of conditions that can only appear when the internalized structure is validated internally. The predictive model generates random machines

Appendix C. Piagetian Models and Mathematics

that classifying a set of participant values, orders them, provides them with a moniker e.g., A or B, reuses the reversibility machines and makes use of logical connectives to determine a generalization that can be reused by other machines, that always holds true. This generalized machine is used in planning to shortcut the planning process. Transitivity is critically dependent on reversibility.

C.1.18 Seriation

Seriation should be compared to the process of classification within logical relations since it is the “capability to order such as from smallest to largest or to count at the operational level that is with true understanding of the inclusion relations involved” (Copeland, 1974, p79). With this, it is understood that counting (addition and subtraction) as well as multiplication and division are specialization of seriation. It is noted that seriation develops earlier than classification this is because it is easier to use a perception property such as length, than it is to define the properties of a class (Copeland, 1974, p80).

Seriation initially makes its appearance as a machine, created by the predictive model that can order a set of objects based on length. Over time and reuse, seriation develops through specialization into machines of counting and multiplication through the internalization of binary values associated with a perception of external objects. It is this specialization by the predictive model of machines that can count, from those that can serialize, that simplify the Piagetian model. Seriation itself is dependent on the use of both continuous and discrete values, as well as on transitivity, reversibility and equivalence.

An early instance of seriation is understood to be the operators of “less than,” “greater than”, which both reuse equivalence. The Piagetian “less than” machine utilizes equivalence and provides a different representation of the binary output value i.e., it is substituted for something else. The interesting thing to note is that the calling of the Piagetian equivalence machine encompasses both assimilation and accommodation. Later forms of Seriation are the appearance of more ordering and structuring.

A depiction of the “less than” operator as a Piagetian HFSA is provided in the following diagram (figure C-2):

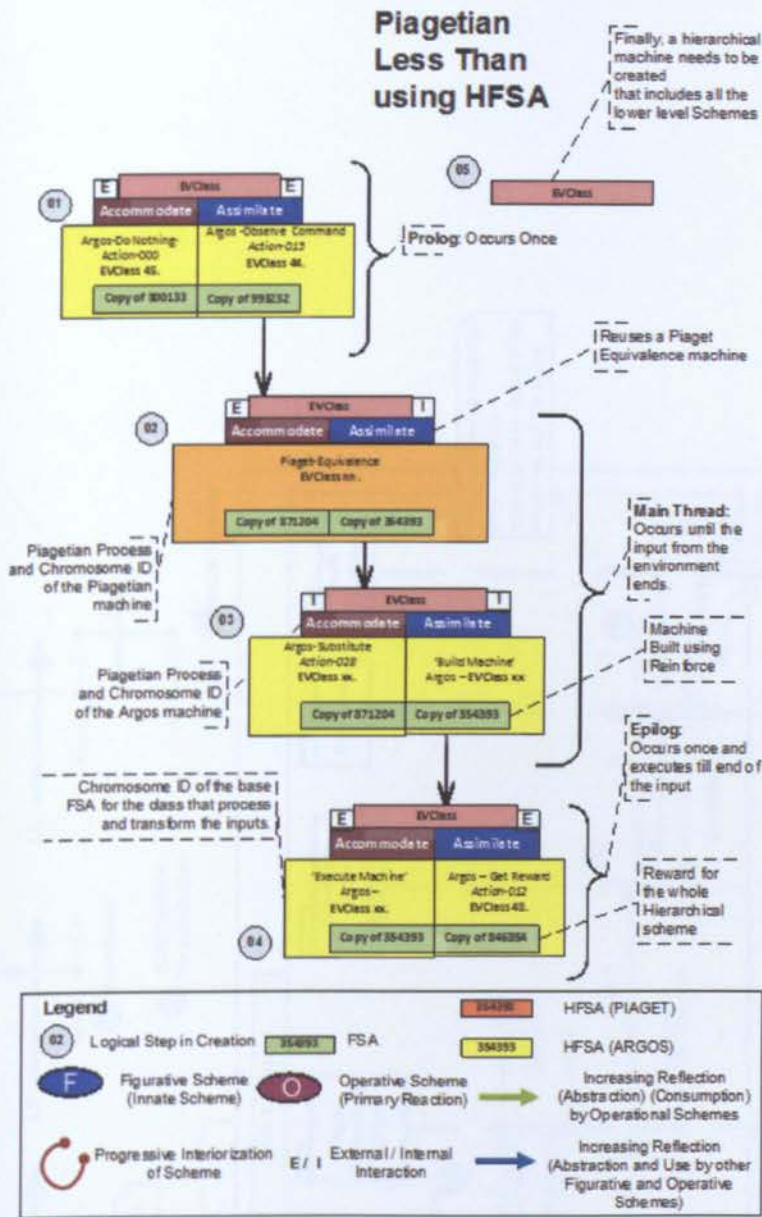


Figure C-2 A worked example of a Piagetian “less than” scheme that includes a Piagetian equivalence scheme. It shows how reversible machines are recombined into hierarchies. In this example, “less than” becomes an operator.

In a similar way, the less than machine can be represented by this sequence diagram (figure C-3):

A simple 'Less Than' machine built using Equivalence and Substitution which later forms the part of the process of rational numbers

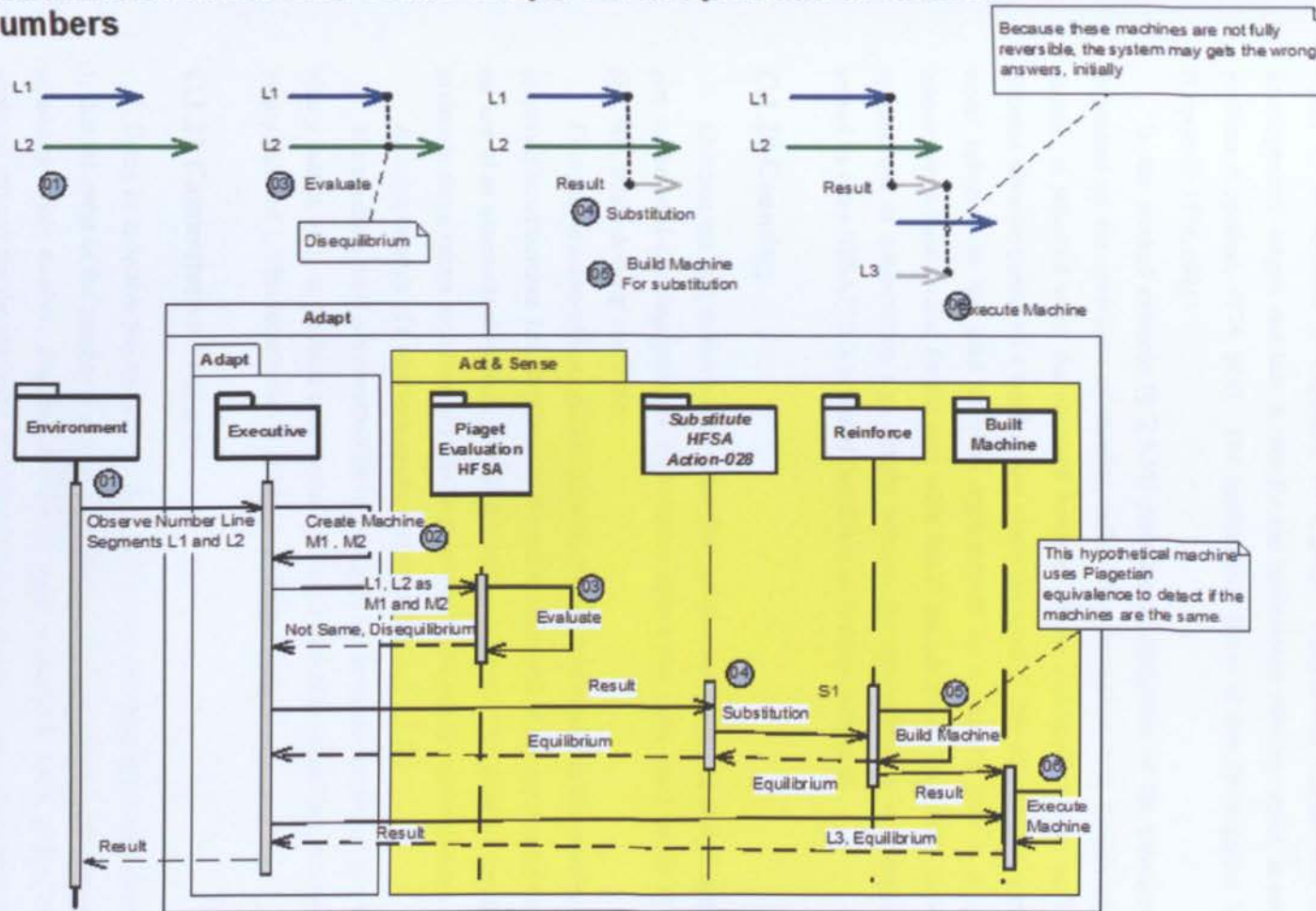


Figure C-3 A UML sequence diagram showing an alternate view of a Piagetian “less than” scheme, which uses more sophisticated subdivision and substitution.

Appendix C – Piagetian Models and Mathematics

In this, the sequence, operations of processing are described as a series of actions against hypothetical machines.

C.1.19 Conservation of Number

Copeland expresses an initial view of conservation of number with a re-iteration of the bead problem (§ figure 1.1 as described in Copeland, 1974, p83). Observations suggest that conservation occurs through developmental stages, and this is true for the equivalence relation, which is required to resolve the bead-problem (Copeland, 1974, p84). The fundamental factor of this development being reversibility of action (Copeland, 1974, p92).

In the worked example (§ 2.3.14) provides a description of the emergence of conservation and is dependent on the invariance of number, rather than perceptual cues (length of the line). Conservation of number is achieved when the observed behavior of the simulated student, no longer takes account of the distance between points on a number line where the number line segment has a penState of penUp. This is a direct reference to the bead problem implemented on a number line. It is achieved when the student internalizes a binary value for the unit value that it has observed. This raises an interesting question “Are all appearances of conservation e.g., length, volume, the appearance of an internalized structure, that can be reused by other HFSA?” It is assumed that this is an accurate statement.

C.1.20 Counting

Children use rote memorization when first introduced to counting (Copeland, 1974, p83), which leads in part to the bead counting problem. Observation shows that there are 3 levels of counting (Copeland, 1974, p89–90), which develop over time.

Counting is achieved through the dynamic evolution of machines that process the internalized structure of conservation of number, i.e., they process the values observed in the environment as binary values and through the use of an externalized reward, provide the correct answers. This requires the machine to add in binary and to decode those values such that they can be presented to the teacher for evaluation.

An interpretation of a counting machine is described in § 5.2.5.1.

The machine itself is a construction of multiple sub-machines organized in a hierarchy. In this machine, a binary output of 1, is machine constructed by the system that substitutes the movement of 1, with a single binary value of 1. This substitution is an aspect of interiorization.

C.1.21 Commutative Property

Using an inductive procedure, the child realizes that by trying other sets of numbers such as $4 + 4$ and $7 + 1$, that the order of the number does not alter the sum. The child then concludes or generalizes that “the order of adding whole numbers does not change the sum” (Copeland, 1974, p123). The commutative property is achieved through the development of a HFSA that is a generalization, developed by the predictive model, that indicates that the way information is processed through the machine can be reversed e.g., $1+3 = 3+1$ and still

Appendix C – Piagetian Models and Mathematics

achieve the same result. It makes use of the equivalence relation to test the outputs and reuses the reversibility machine to reverse the order of its processing. In this way, the commutative property can hold for different types of information being processed.

C.1.22 Multiplication and Division

The basic notion of multiplication being continued addition, division as continued subtraction is necessary for the part to whole evaluation of fractions and proportions (Copeland, 1974, p146), and it is realized that multiplication and division are dependent on reversibility and equivalence.

Multiplication and division is achieved through reusing the counting HFSA, in series. The machine itself is generated through the predictive model, which allows it to test various configurations of machines and to make use of counting, classification, transitivity, and reversibility and equivalence machines.

C.1.23 Introspection

A lack of consciousness of one's own thoughts occurs up to the age of seven, after which there is more effort to become conscious (introspective) of one's own thinking process (Copeland, 1974, p184). Introspection is implemented as the mechanism of planning (using a predictive model) which interrogates chains of HFSA to improve their processing and use them to solve problems that are more complex.

C.1.24 Classification

The classification relation is the appearance of class using groupings (Copeland, 1974, p51 and p60). An object may have multiple classifications (Copeland, 1974, p51 and p60). Some of which may be hierarchical and some simple. They may also be logically incorrect. It is noted that seriation develops earlier than classification, because it is easier to use a perception property such as length, than it is to define the properties of a class (Copeland, 1974, p80); as such classification is not specifically tackled, even though it will allow the use of a internal monikers as "variables" and support more complex processing such as transitivity and logical relations, it is deemed beyond the scope of this research. Classification is probably implemented as a HFSA, which contains references to other machines, in this way the system can generate its own classification hierarchy, based on what it perceives as useful to solve given problems. These classification mechanisms are seeded based on the topological and Euclidian properties available e.g., length, distance etc.

C.1.25 Ordinality and Cardinality

Like most other relations, Piaget identified 3 states of the development of ordinality and cardinality (Copeland, 1974, p97). The feature of ordinality and cardinality is implemented within the system as the construction of a machine that can numerically identify the positions within a set of values.

Appendix C – Piagetian Models and Mathematics

C.1.26 Fractions and Proportions

The emergence of the capability to process fractions and proportions (rational numbers) is seen as a specialization of number and is based on seven characteristics (Copeland, 1974, p158–159) which include the following: a divisible whole, determinate number of parts, no remainders, fixed relationships, all parts are equal, each part is a separate whole and the whole is conserved, i.e., the reversible inclusion relation is used.

The divisible whole is implemented as a HFSA that can successfully partition, based on length a number line segment into a unitary set of parts and then realize that the set of partitions is equivalent to the whole. A thirds machine can then be utilized to partition another segment into thirds. The property “No remainder” is the specific case where there is no remainder from the intended subdivision. This processing probably takes place in the predictive model, which determines potential machines.

The determinate number of parts is an implementation of subdivision (Copeland, 1974, p252). Subdivision is a generalized HFSA that determine how many parts a whole can be produced through by division. Since “numbers” are implemented from conservation of length using binary values and division is a mechanism to divide a numerical value (in this implementation at least), subdivision is seen as division “with no remainders.” This processing must occur within the predictive model, because this process allows the simulated student the capability of making numerous trial and error approaches, the results of which are tested in the environment. This predictive model produces machines that can divide and strings them together in composite chains of processes. If the processing is successful, then a reward is provided which strengths the machine, enabling it to be reused.

The property of “fixed relationships” is an altogether different class of problem and requires that the machine used in building subdivisions, maintains the set of relations. This is perhaps the reuse of the equivalence machine, which compares the movements of each of the individual machines.

In a similar way, the property “all parts are equal” is an implementation of the equivalence relation. The interesting feature is the “each part is a separate whole,” since this will need to be a machine that is recursive and finds all the potential subdivisions that are possible for a machine.

The property of conservation of the whole is a critical machine that must evolve to reject subdivisions if they (their collective length) do not equal the full length. There must be some mechanism that allows these collections of machines to execute together, i.e., such that the machines are self-correcting.

C.1.27 Generalization

The process of transduction, which is understood to be a lack of generalization, occurs up until the age of 7 years (Copeland, 1974, p186). After this age two classes of generalization, inductive and deductive appear regularly. An inductive generalization (basically a rule) about rational numbers is commonly determined by children who realize, for example, when presented with $\frac{1}{3} + \frac{1}{4} = \frac{7}{12}$ also realize that multiplying the denominators of the addends 3 and 4 produces the denominator of the sum, i.e., 12 and that adding the

Appendix C – Piagetian Models and Mathematics

denominators of the addends produces the numerator of the sum, i.e., 7, thus producing $\frac{7}{12}$. If the child can apply this to other unit fractions, then they have made an inductive generalization (Copeland, 1974, p195). In comparison, Deductive reasoning is considered to occur after a generalization has been made e.g., $\frac{1}{a} + \frac{1}{b} = \frac{1}{b} + \frac{1}{a}$, the child then makes use of this induction and without examining the details of a specific case e.g., $\frac{1}{3} + \frac{1}{4} = \frac{1}{4} + \frac{1}{3}$ and concludes that they are the same. The generation of deductive reasoning allows the student to make further judgments (Copeland, 1974, p203). Generalization is imagined to be the result of HFSA that consume other HFSA and FSA. In doing so they encompass the behavior of those consumed machines using an optimized structure. In part this would explain Piaget's assertion of reflective abstraction (§ 2.3.9 on Piaget and Reflective Abstraction).

C.1.28 Other Conditions

Piaget identified several other relations that were required for the development of logical thought. These included egotism, which is seen as the attention focused solely on the individual's perspective, and transduction, which is the lack of available generalization that is not typically observed in children until they reach concrete operations and almost always by formal operations (Copeland, 1984, p183). Whereas egoism cannot be reigned in, in this research, transduction is considered to be ameliorated through the use of the predictive model. The planning performed by the student, produces generalized schemes through the schemes consuming previously produced HFSA. Several other conditions were defined by Furth (Furth, 1969) that has previously been discussed in this thesis.

Several features of Piagetian development are not required for the development of rational numbers; these include the generalized adaptations of conservation of measurement, area, volume, singular class, weight, quantity as well as angle. More complex features such as subtraction are not part of the research. One limitation on the research approach, is the lack of social interaction, which Piaget indicated was a necessity developing a coherent whole (Piaget, 1952, p46 Cited in Copeland, 1974, p46), though this is important, the research focus is on the development an internal model of concept acquisition and so is left out. An interesting aspect of Copeland's review of Piaget is the linking of seriation to the development of memory (Copeland, 1974, p48 and p97).

Last, substitution, which takes an external value and replaces it with an internal one and an internal value with an external 1, is the basis of interiorization.

C.2 Rational Number–Sense

C.2.1 WE6 – Rational Level – Relationships of Unit Values to Continuous Values

This definition of **WE6** – Rational Level is included for completeness, since it is believed that it is required for the development of rational number–sense. It is as an extension of the worked example.

At this level, there should be significant differences between the processing for units and for rational values. There should also be the appearance of schemes that exhibit the behavior of control and prediction of points and planes to the transformation itself. This is shown metaphorically in figure C–4 as the overlaid representational comparison of rational values to the unit values. This representation also exhibits the use of prediction to determine that “double the rational structure,” is equivalent to the unit structure.

Developmentally, if there is appearance of a concept of a fraction as a single entity, it is the conclusion that conceptual knowledge has been developed since a rational number is a more complex symbol representation than for a whole number (Cramer, Behr, Post and Lesh, 1997b, p3). Likewise, ordering fractions is more complex than ordering whole numbers (Cramer, Behr, Post and Lesh, 1997b, p3). However, this brings to the forefront the need to develop symbols. “Comparing $1/4$ and $1/6$ conflicts with children’s whole number ideas. Six is greater than four, but $1/4$ is greater than $1/6$. With fractions, more can mean less. The more equal parts you partition a unit into, the smaller each part becomes.” At issue is understanding how external values (as symbols) can be passed to a student, without encoding too much into the student. It is suspected that the student would have to go through all the developmental stages.

In the image below (figure C–4), the use of relations is described.

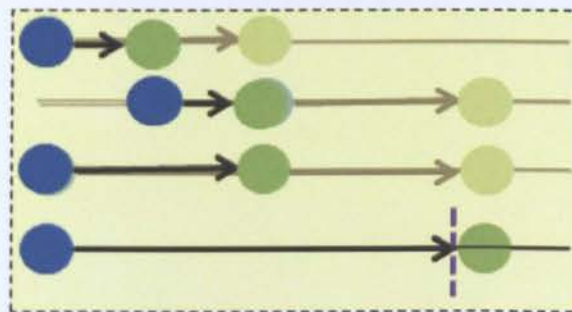


Figure C–4 WE6 – Rational – Level – the sixth action to appear in a number line world, where there are sets of repeated relations. This designates that the processing by the student is making more complex arrangements on their number line world. Though listed, the rational feature will not be evaluated in the research and is included only for completeness.

- 1) For Piaget, the issues with numerators and denominators (Nason and Martin, 1990), will not be a problem in this representation. However, the use of unit (and two sub–units making up this unit) is termed equivalent to the whole number interference problem (Behr, Wachsmuth, Post and Lesh, 1984; Post, Wachsmuth, Lesh and Behr, 1985; Post and Cramer, 1987). Similarly, the use of mixed fractions such as $2 - \frac{1}{8}$ may be problematic (Mack, 1990, p22) and will require that

Appendix C – Piagetian Models and Mathematics

the simulation differentiate the rules at two different levels, once for the whole and one for the part. Using estimation, a student may be able to compensate for potential errors that they discover. The core issue here is that estimation must be part of a developmental or evolutionary trend that exists within the student itself. For Piaget, reflective abstraction, which is understood to be the transformation from imaging points and planes to the transformation itself, is the core principal at work. This reflective abstraction is seen in the backdrop of the evolutionary and developmental trends. At this level, there is the first appearance and development of ordinality and cardinality, as well as fractions and generalization (Copeland, 1974, p97, p158–159, p186 and 195).

- 2) For Fodor, at the rational level “rational laws” need to be learnt in terms of the lower levels (**Base Level, Constrained Level, Differentiated Level, Hierarchical Level, General Level**) that allow for the resolution of the paradox i.e., the unit processing on the number line, being transposed to the rational process (of parts of units) on the number line.

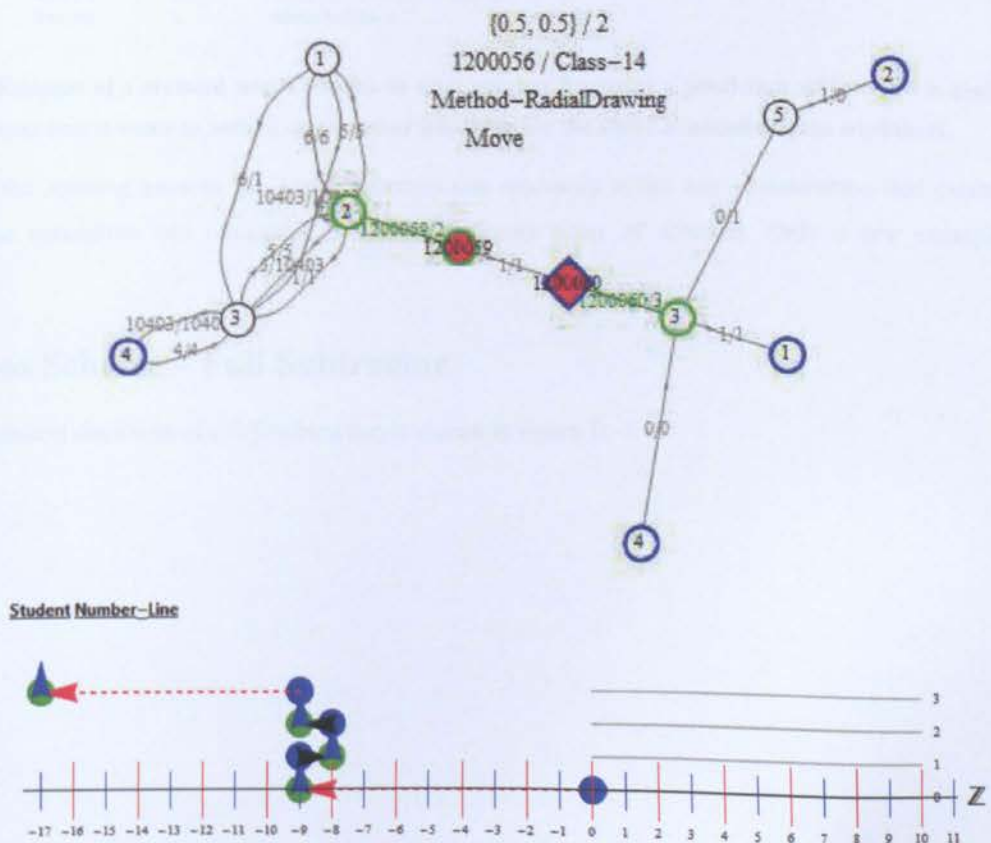
Appendix D

D. Further Examples of Dialectic Evaluation

This appendix contains further examples from the evaluation of the *dialectic system*. Specifically it contains the following examples: Argos Scheme – Movement, in § D.1, Argos Scheme – Full Subtractor, in § D.2, Argos Scheme – Decoder, in § D.3, Argos Scheme – Counter, in § D.4, Argos Scheme – Full-Adder, in § D.5 and a Piagetian Scheme – Using Planning to Count, in § D.6.

D.1 Argos Scheme – Movement

In the example below (figure D-1), Argos scheme 1200056 was evolved to move along a number line using various Prediction schemes that control penState, directions and movements.



Appendix D. Further Examples of Dialectic Evaluation

Figure D-1 A depiction of an Argos Scheme that exhibits coordinated movement on number lines. In this depiction, the movements on the number line are random, there is no coordinated control.

What is clear from figure D-1 is that the simulation can mutate Argos schemes to move in the environment, receiving rewards for accurate movement.

D.1.1 Argos Scheme – Start Number line

In figure D-2, an Argos scheme was mutated to include a prediction scheme to start a number line.

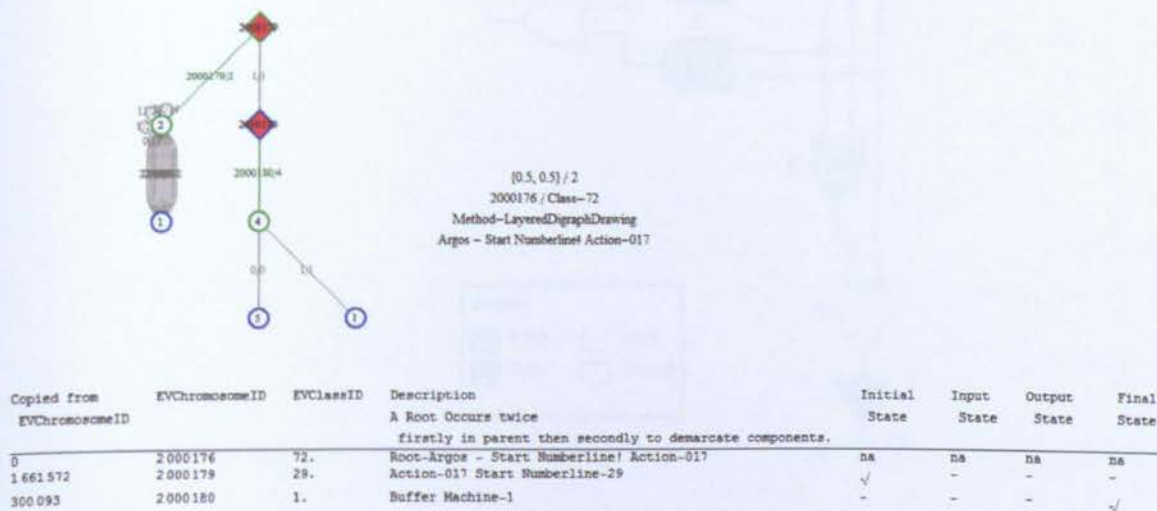


Figure D-2 Example of a mutated Argos Scheme to start number line using a prediction scheme that is used to tell the environment that it wants to build a new number line. This has the effect of initializing the worksheet.

Since the learning process for Argos schemes can randomly select any chromosome that exists in its memory, the simulation can randomly construct numerous types of schemes. Only a few examples are presented.

D.2 Argos Scheme – Full Subtractor

The classical depiction of a full subtractor, is shown in figure D-3.

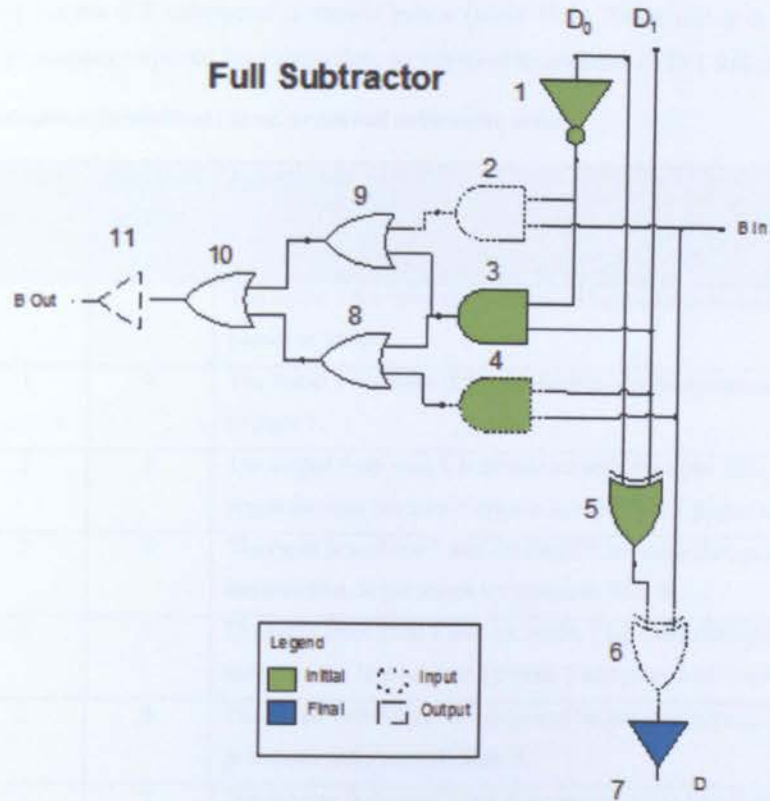


Figure D-3 The classic depiction of an Argos full subtractor scheme using randomly selected initial, final, input, output and ordinary states.

The following table (table D-1) describes the states in the full subtractor:

Table D-1 States in an Argos Full Subtractor Scheme

State	EVClassID	Parent ID	Type	Capacity		Initial State	Input State	Output State	Final State
				Input	Output				
1	2.	303178	NOT	1	1	Y	-	-	-
2	6.	772354	AND	2	1	-	Y	-	-
3	6.	772354	AND	2	1	Y	-	-	-
4	6.	772354	AND	2	1	Y	-	-	-
5	8.	1559639	XOR	2	1	Y	-	-	-
6	8.	1559639	XOR	2	1	-	-	-	-
7	1.	300093	Buffer	1	1	-	Y	-	Y
8	7.	825480	OR	2	1	-	-	-	-
9	7.	825480	OR	2	1	-	-	-	-
10	7.	825480	OR	2	1	-	-	-	-
11	1.	300093	Buffer	1	1	-	-	Y	-

Appendix D. Further Examples of Dialectic Evaluation

The processing for the full-subtractor is shown below (table D-2). From this it is realized that there is significantly more processing required for subtraction, as opposed to addition (§ D.5 full-adder).

Table D-2 Synchronizations (transitions) in an Argos full subtractor scheme

From State	To State	Parallel Seq	Sequence Ordinal	Description
1	2	1	1	The Initial 1 bit value (D^0), provided by the environment is processed and passed to State 2.
1	3	1	2	The Initial 1 bit value (D^0), provided by the environment processed and passed to State 3.
2	9	2	3	The output from state 1 is combined with the input (Bin) from a preceding Argos Scheme (or zero if none is available) and passed to state 9.
3	9	2	4	The input from State 1 and the Initial 1 bit value (D^1) provided by the environment, is processed by passed to State 9.
3	8	2	5	The input from State 1 and the Initial 1 bit value (D^1) provided by the environment, is processed by State 3 and passed to State 8.
4	8	2	6	The Initial 1-bit value (D^1) provided by the environment and the input (Bin) is processed and passed to State 8.
8	10	3	7	The outputs from state 3 and 4 are processed and passed to State 10.
9	10	3	8	The outputs from states 2 and 3 are processed and passed to State 10.
10	11	4	9	The outputs from state 8 and 9 are processed and passed to state 11, the output state as a value (Bout).
5	6	5	10	The Initial 1 bit values (D^0 and D^1) provided by the environment is processed and passed to state 6.
6	7	6	11	The output from State 5 and the Initial 1 bit value (D^1) provided by the environment, is processed and passed to state 7, the final state as a value (D). Since the Argos Scheme, is at the final state it stops.

The execution of these synchronizations, is the processing of inputs provided by the environment and the production of the following values (table D-3):

Table D-3 Outputs of an Argos full subtractor scheme.

Borrow (B) In	D^0	D^1	Borrow (B) Out	D ($D^0 - Bin - D^1$)	Comments
0	0	0	0	0	$0-0-0=0$, No borrow.
0	0	1	1	1	$0-0-1 = -1$, borrow 2, so $2-1 = 1$.
0	1	0	0	1	$1-0-0 = 1$, No borrow.

Appendix D. Further Examples of Dialectic Evaluation

Borrow (B) In	D ⁰	D ¹	Borrow (B) Out	D (D ⁰ -Bin- D ¹)	Comments
0	1	1	0	0	1-0-1=0, No borrow.
1	0	0	1	1	0-1-0 = -1 (bin=1), borrow 2, so: 2-1 = 1.
1	0	1	1	0	0-1-1 = -2, borrow 2, so: 2-2=0.
1	1	0	0	0	1-1-0=0, No borrow.
1	1	1	1	1	1-1-1 = -1, borrow 2, So 2-1 = 1.

Like full-adders, full-subtractors can be serialized, as in the figure below (figure D-4) with the ‘bout’ and ‘bin’ values progressing through the scheme.

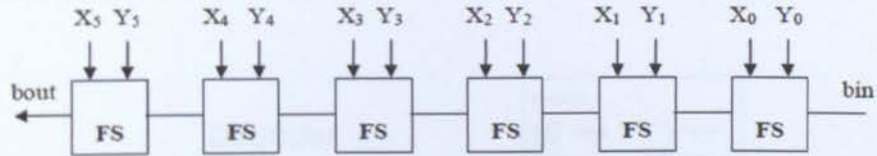
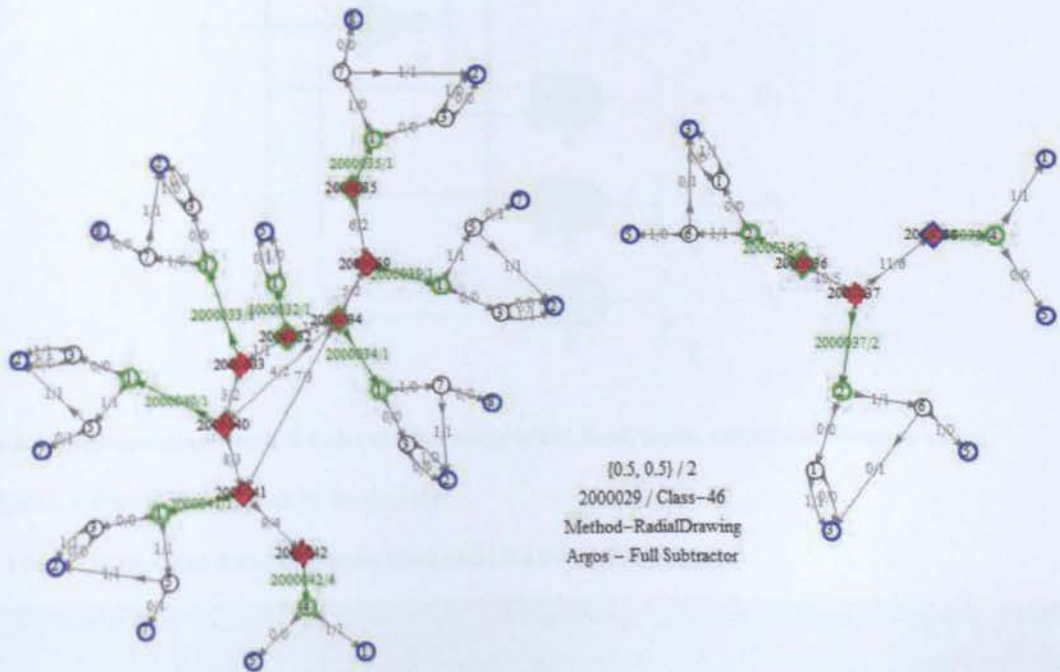


Figure D-4 The depiction of a ripple carry subtractor as a collection of full-subtractors as an Argos scheme.



Appendix D. Further Examples of Dialectic Evaluation

Copied from EVChromosomeID	EVChromosomeID	EVClassID	Description	Initial State	Input State	Output State	Final State
0	2 000 029	46.	A Root Occurs twice firstly in parent then secondly to demarcate components.	na	na	na	na
303178	2 000 032	2.	Root-Argos - Full Subtractor NOT Machine-2	✓	-	-	-
772354	2 000 033	6.	Basic Gale AND Machine-6	-	✓	-	-
772354	2 000 034	6.	Basic Gale AND Machine-6	✓	-	-	-
772354	2 000 035	6.	Basic Gale AND Machine-6	✓	-	-	-
1559439	2 000 036	8.	Basic Gale XOR Machine-8	✓	-	-	-
1559439	2 000 037	8.	Basic Gale XOR Machine-8	-	✓	-	-
300093	2 000 038	1.	Buffer Machine-1	-	-	-	✓
825480	2 000 039	7.	Basic Gale OR Machine-7	-	-	-	-
825480	2 000 040	7.	Basic Gale OR Machine-7	-	-	-	-
825480	2 000 041	7.	Basic Gale OR Machine-7	-	-	-	-
300093	2 000 042	1.	Buffer Machine-1	-	-	✓	-

Figure D-5 An Argos-Scheme that implements a full subtractor.

The depiction follows the same style as other Argos schemes.

D.3 Argos Scheme – Decoder

Using the same mechanisms as the Argos schemes, a 2: 4 bit decoder can be built, as depicted in figure D-6.

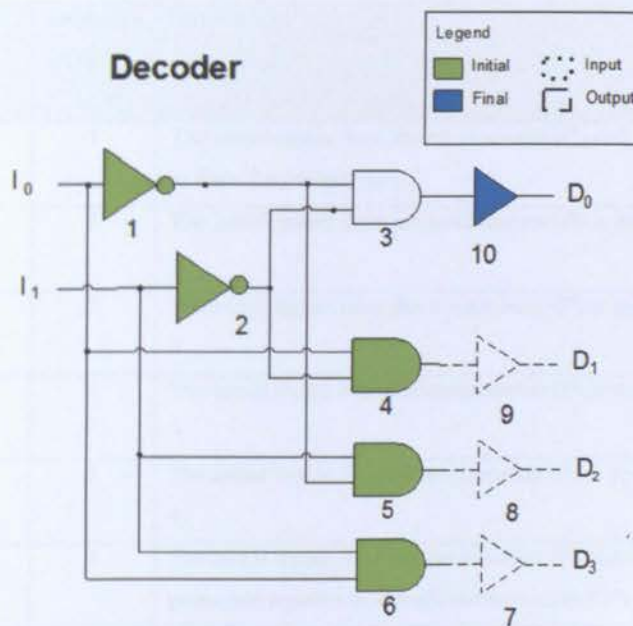


Figure D-6 A Depiction of an Argos decoder scheme using initial, final, input, output and ordinary states.

Table D-4 describes the states in the decoder.

Table D-4 States in an Argos decoder scheme. Each state is a Prediction scheme.

State	EVClassID	Parent ID	Type	Capacity		Initial State	Input State	Output State	Final State
				Input	Output				
1	2.	303178	NOT	1	1	Y	-	-	-
2	2.	303178	NOT	1	1	Y	-	-	-

Appendix D. Further Examples of Dialectic Evaluation

State	EVClassID	Parent ID	Type	Capacity		Initial State	Input State	Output State	Final State
				Input	Output				
3	6.	772354	AND	2	1	–	–	–	–
4	6.	772354	AND	2	1	Y	–	–	–
5	6.	772354	AND	2	1	Y	–	–	–
6	6.	772354	AND	2	1	Y	–	–	–
7	1.	300093	Buffer	1	1	–	–	Y	–
8	1.	300093	Buffer	1	1	–	–	Y	–
9	1.	300093	Buffer	1	1	–	–	Y	–
10	1.	300093	Buffer	1	1	–	–	–	Y

The processing for the decoder is shown in table D–5.

Table D–5 Decoder – Synchronizations (Transitions)

From State	To State	Parallel Seq	Sequence Ordinal	Description
6	7	1	1	The initial inputs from the environment (I^0) and (I^1) is processed and passed to State 7 as output (D^3).
1	3	1	2	The initial inputs from the environment (I^0) is processed and passed to State 3.
1	5	1	3	The initial inputs from the environment (I^0) is processed and passed to State 5.
2	3	1	4	The initial inputs from the environment (I^1) is processed and passed to State 3.
2	4	1	5	The initial inputs from the environment (I^1) is processed and passed to State 4.
5	8	2	6	The initial inputs from the environment (I^1) and the output from state 1 is processed together and produces the output (D^2)
4	9	2	7	The initial inputs from the environment (I^0) is combined with the output of state 2 and processed and passed to state 9 as output (D^1).
3	10	3	8	The output from State 2 and State 1 are combined, processed and passed to State 10 as the final State (D^0). The Scheme stops processing.

What is clear from table D–5, is that the synchronizations are that the output of an Argos scheme always goes to a buffer scheme. Second, the randomly chosen final state must be the last parallel state constructed.

The execution of these synchronizations is the processing of inputs provided by the environment and the production of the following values in table D–6:

Table D–6 Outputs of an Argos decoder scheme

I ⁰	I ¹	D ³	D ²	D ¹	D ⁰
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

The values in this table agree with the classical values.

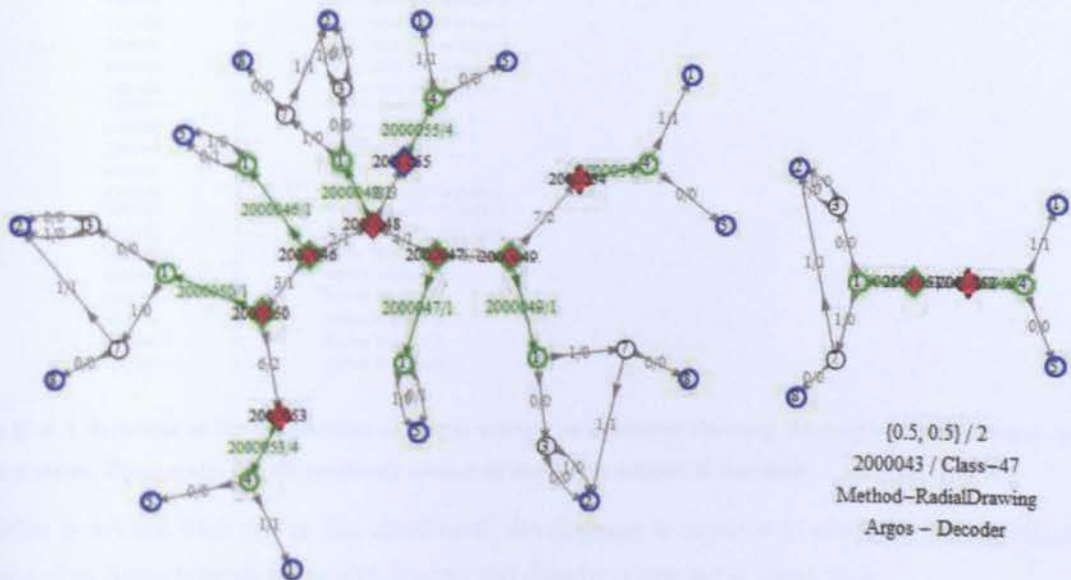


Figure D-7 Depiction of an Argos decoder scheme with Argos schemes in red and Prediction schemes in green and blue. The transitions and synchronizations are shown in grey and green lines with the appropriate transition values.

The construction of these Argos schemes follows the mutation selection mechanism as prediction schemes.

D.4 Argos Scheme – Counter

The example below is of an Argos counting scheme (figure D-8). With its composition:

Appendix D. Further Examples of Dialectic Evaluation

EVChromosomeID	EVClassID	Description	Initial State	Input State	Output State	Final State
1200061	15.	Root-ARGOS(2011, 11, 14, 22, 37, 29.8376311)	na	na	na	na
1200064	14.	ARGOS(2011, 11, 14, 21, 47, 58.6066865)	✓			
1200065	10.	ARGOS(2011, 11, 14, 21, 47, 20.0824830)				
1200066	11.	ARGOS(2011, 11, 14, 21, 47, 29.1760031)				
1200067	13.	ARGOS(2011, 11, 14, 21, 47, 46.5298957)				
1200068	14.	ARGOS(2011, 11, 14, 21, 47, 58.6066865)				
1200069	1.	Buffer Machine-1				✓
1200060	1.	Buffer Machine-1				✓
1200004	2.	NOT Machine-2	✓			
1200005	4.	Basic Gate AND Machine-4	✓			
1200006	4.	Basic Gate AND Machine-4				
1200007	2.	NOT Machine-2				
1200008	2.	NOT Machine-2				
1200009	4.	Basic Gate AND Machine-4	✓			
1200010	2.	NOT Machine-2				
1200011	4.	Basic Gate AND Machine-4				
1200012	4.	Basic Gate AND Machine-4	✓			
1200013	5.	Basic Gate OR Machine-5				
1200014	5.	Basic Gate OR Machine-5				
1200015	4.	Basic Gate AND Machine-4				
1200016	1.	Buffer Machine-1				
1200017	1.	Buffer Machine-1			✓	
1200018	1.	Buffer Machine-1			✓	
1200022	6.	Basic Gate XOR Machine-6	✓			✓
1200023	4.	Basic Gate AND Machine-4	✓			
1200024	4.	Basic Gate AND Machine-4		✓		
1200025	6.	Basic Gate XOR Machine-6		✓		
1200026	5.	Basic Gate OR Machine-5				
1200027	1.	Buffer Machine-1				✓
1200028	1.	Buffer Machine-1			✓	
1200044	2.	NOT Machine-2	✓			
1200047	2.	NOT Machine-2	✓			
1200048	4.	Basic Gate AND Machine-4	✓			
1200049	4.	Basic Gate AND Machine-4	✓			
1200050	4.	Basic Gate AND Machine-4	✓			
1200051	4.	Basic Gate AND Machine-4	✓			
1200052	1.	Buffer Machine-1			✓	
1200053	1.	Buffer Machine-1			✓	
1200054	1.	Buffer Machine-1			✓	
1200055	1.	Buffer Machine-1				✓
1200060	1.	Buffer Machine-1				✓

Figure D-8 A depiction of the composition of Argos scheme as a counter showing the graph of initial, input, output and final states. These states are all randomly chosen as the Argos scheme is mutated.

What is evident from this is that coordinated development is required to construct these machines. A depiction of an Argos Scheme as encoder, counter and decoder is detailed in figure D-9

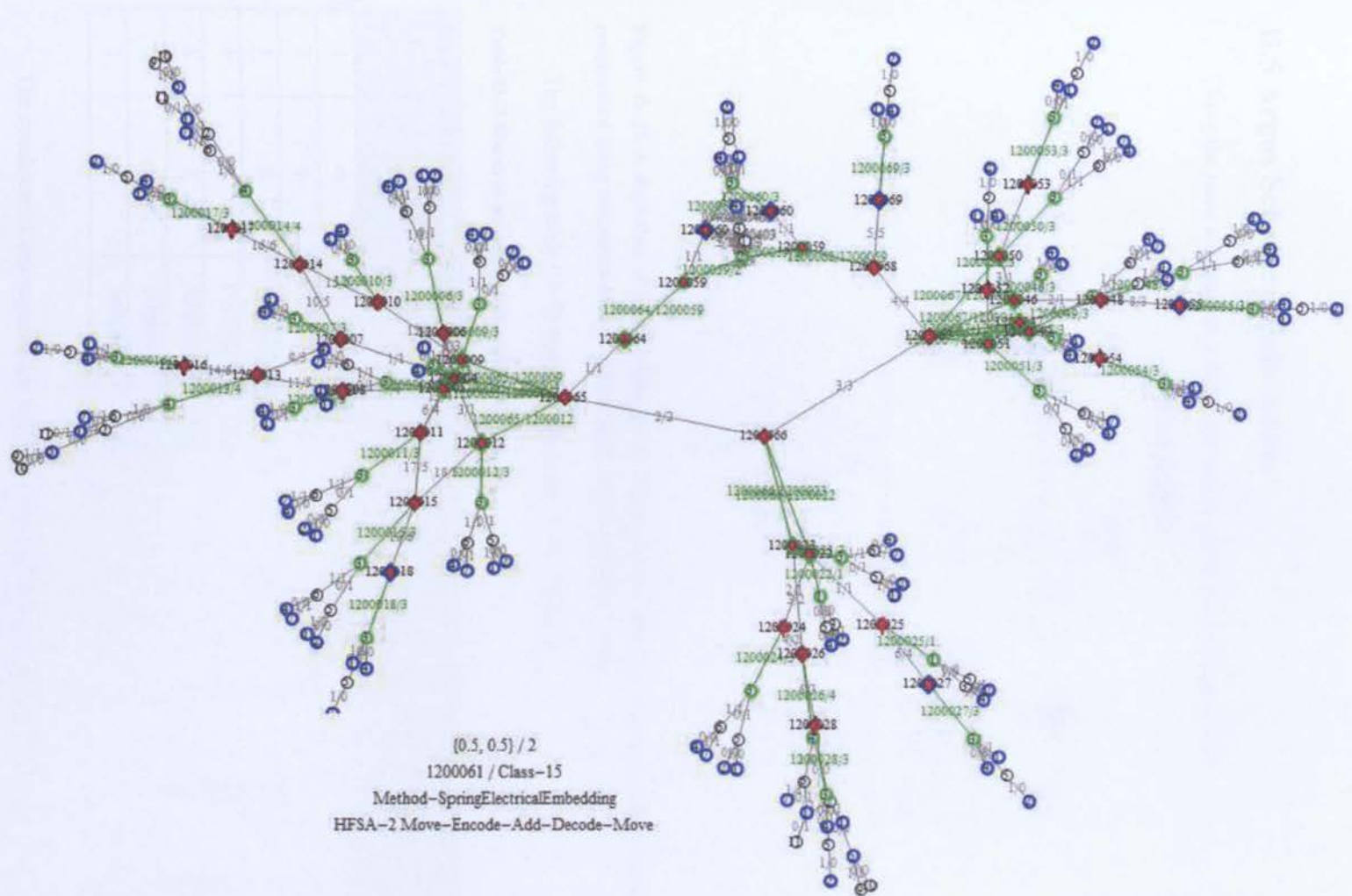


Figure D-9 A depiction of an Argos scheme counter scheme which includes full adders, decoders and internalized actions.

D.5 Argos Scheme – Full-Adder

Using the same mechanism, a full adder can be constructed (figure D-10):

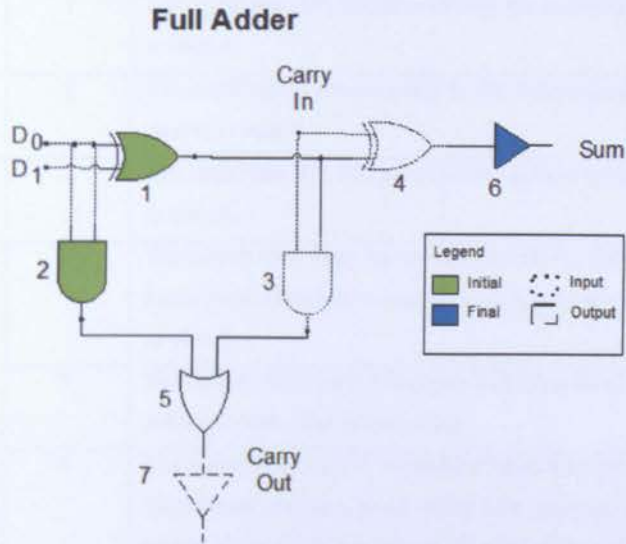


Figure D-10 A depiction of a full adder as an Argos scheme showing that common schemes can be randomly constructed using mutations of their initial, final, input and output states.

The following table (D-7) describes the states in the full adder.

Table D-7 States in an Argos full-adder scheme.

State	EVClassID	Parent ID	Type	Capacity		Initial State	Input State	Output State	Final State
				Input	Output				
1	6.	450962	XOR	2	1	Y	-	-	-
2	6.	772354	AND	2	1	Y	-	-	-
3	6.	772354	AND	2	1	-	Y	-	-
4	8.	1559639	XOR	2	1	-	Y	-	-
5	7.	825480	OR	2	1	-	-	-	-
6	1.	300093	Buffer	1	1	-	-	-	Y
7	1.	300093	Buffer	1	1	-	-	Y	-

The synchronizations required for this full-adder are shown below (table D-8):

Appendix D – Further Examples of Dialectic Evaluation

Table D-8 Synchronizations (Transitions) in an Argos full Adder scheme

From State	To State	Parallel Seq	Sequence Ordinal	Description
1	4	1	1	The initial bits (D^0 , $D1$) provided by the environment are processed and passed to state 4.
1	3	1	2	The initial inputs bits provided by the environment (D^0 , $D1$) are processed and passed to state 3.
2	5	2	3	The initial bits (D^0 , $D1$) provided by the environment are processed and passed to state 5.
3	5	2	4	The output from State 1 is combined with the Carry In (from a previous hierarchical machine, or zero if no such machine exists), is processed and passed to state 5.
5	7	3	5	The outputs from state 2 and state 3 are processed and passed to state 7. The output state as value (Carry Out).
4	6	4	6	The output from State 1 is combined with the Carry In (from a previous hierarchical machine, or zero if no such machine exists), is processed and passed to state 6 the Final State (Sum). This is the final state for the Argos Scheme, so processing stops.

There is no feedback loop for processing as can be seen in this example. Further, all processing is unique; there is no shared processing for these Argos style schemes. To be able to appropriately process the information, it is essential that each instance of a class of chromosome used in an Argos scheme is copied as a unique instance into the population, allowing the Argos schemes to “grow their own” inter-connections. As with encoders, these schemes can be combined in series, with the outputs of one scheme being passed to the inputs of subsequent scheme.

The execution of these synchronizations is the processing of inputs provided by the environment and the production of the following values (table D-9)

Table D-9 Outputs of an Argos full-adder scheme.

D^0	D^1	Carry In	Sum	Carry Out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0

Appendix D – Further Examples of Dialectic Evaluation

D ⁰	D ¹	Carry In	Sum	Carry Out
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

A critical aspect of the outputs is that they provide a mechanism for the system to evaluate the constructed schemes.

Argos Schemes can also be combined in series. For example, a ripple carry adder can be constructed from copies of the simpler Argos Schemes, the full adder.

Ripple Carry Adder as an example of a HFSA

A set of full adder's are combined together into a HFSA, to form a ripple carry adder.

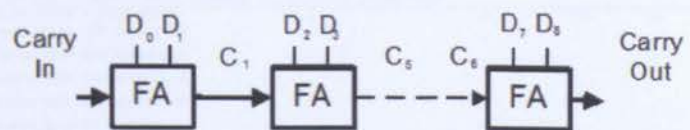


Figure D-11 A depiction of an Argos scheme as a ripple carry adder. Showing the inputs, outputs, initial and final states. This is only one example, others exist.

This raises the issue that the Argos Scheme must remain active over a period of time, such that it can count a range of values provided through the interaction with the environment. In supporting the work of Piaget, Pascual-Leone imagines a model of processing where schemes can remain active, in equilibration, until the system itself is forced into a process of disequilibration and other schemes become dominant. For instance, if the system is counting and it cannot count anymore because it has run out of processing units. It regains equilibrium by adding more full-adders and continues counting until either the input from the environment stops and it finishes or some other scheme becomes active and dominant e.g., boredom. This processing is essential to the overall utility and warranty of the system.

Using simple rules sets of Argos schemes can be randomly evolved together as a more complex Argos Full-Adder scheme as described in figure D-12:

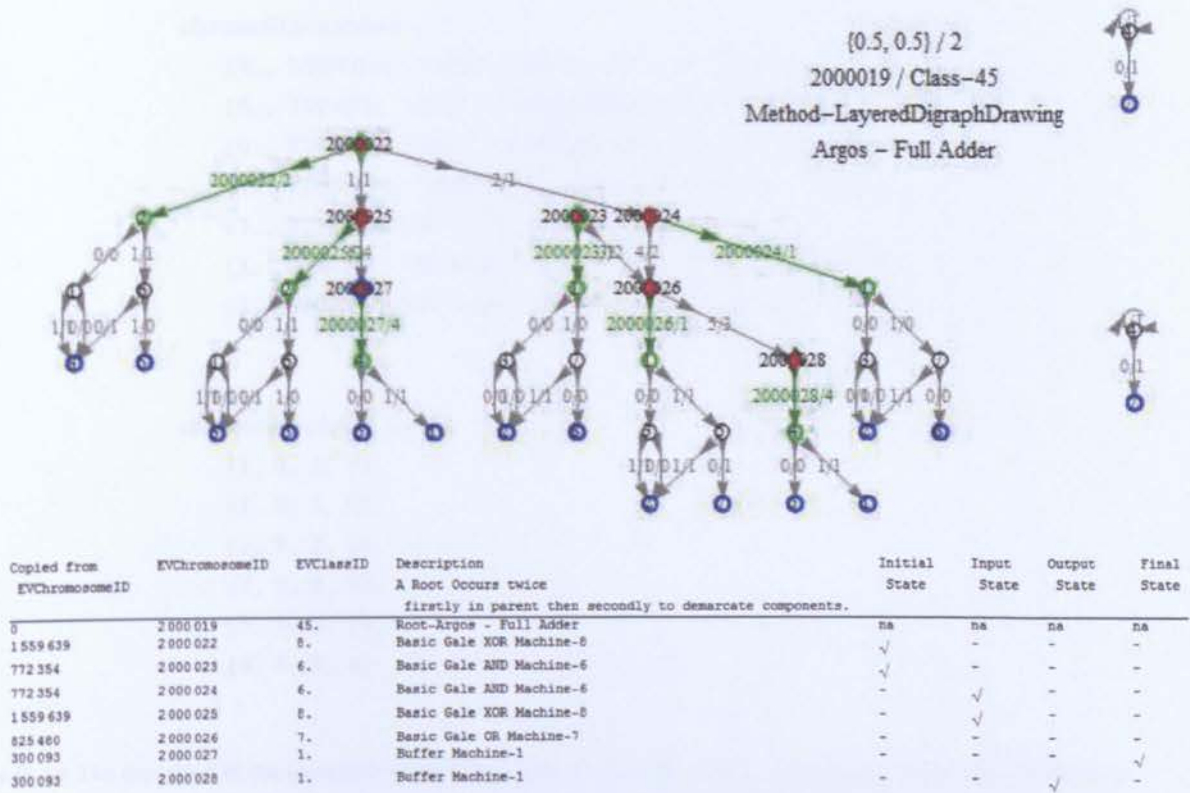


Figure D-12 The depiction of a full adder as a Argos hierarchical finite state machine otherwise called an Argos scheme.

In this, the Argos Scheme, chromosome⁸⁹ 2000019, EVClass 45 is composed of a set of schemes. The interesting point to note is that these schemes are not perfect; they have outliers, as in the far right of figure 12-12. These outliers are not part of the main thread of processing, but at a future mutation, may become significant. The Argos Scheme synchronizes the processing of inputs from the environment across the contained FSA using a set of rules, as shown in the following diagram (figure D-13):



⁸⁹ As an aside, in this architecture, the evolution of chromosomes occurs within the lifetime of Student, in this respect 'chromosomes' are more akin to 'neurons' and this is a throwback to the original implementation from which this solution has grown (Jacob, 2001, p213). Thus the process of evolving a class of chromosomes, is the learning process employed by this system and is the resolution of LP5.

```

chromoStatesList = {
  {8., 1559639, "XOR", "InitialState", "", "", "", 1},
  {6., 772354, "AND", "InitialState", "", "", "", 2},
  {6., 772354, "AND", "", "InputState", "", "", 3},
  {8., 1559639, "XOR", "", "InputState", "", "", 4},
  {7., 825480, "OR", "", "", "", "", 5},
  {1., 300093, "Buffer", "", "", "", "FinalState", 6},
  {1., 300093, "Buffer", "", "", "", "OutputState", 7}
};

chromoSyncList = {
  {1, 4, 1, 1},
  {1, 3, 1, 2},
  {2, 5, 2, 3},
  {3, 5, 2, 4},
  {5, 7, 3, 5},
  {4, 6, 4, 6}
};

```

Figure D-13 The depiction of the structure of an Argos scheme as a full-adder. In execution, these synchronizations are chosen randomly and evaluated using external rewards by the simulation. They are depicted here in a similar fashion to FSA.

The behavior of basic gates can be expressed using finite state machine; for instance, AND, NOT, XOR can all be evolved from minimal single state machines, using appropriate evaluation functions. Combinations of these gates can then implement Boolean expressions, producing a specified output (almost) at the instant when the input values are applied. These combinational circuits (otherwise referred to as sequential circuits or combinational logic circuits) can have multiple inputs and more than one output e.g., a second output can be provided for the complement of the operation. Combinational circuits give us many useful devices with one of the simplest are the “Half Adder,” which finds the sum of two bits. The sum can be found using the XOR operation and the carry using the AND operation. The example full-adder is shown below and is described in figure D-14:

D.4 Dialectic Schemes Using

1009209 / Class-39

Method-RadialDrawing

Argos - Full Adder

{0.5, 0.5} / 2
 1009209 / Class-39
 Method-RadialDrawing
 Argos - Full Adder

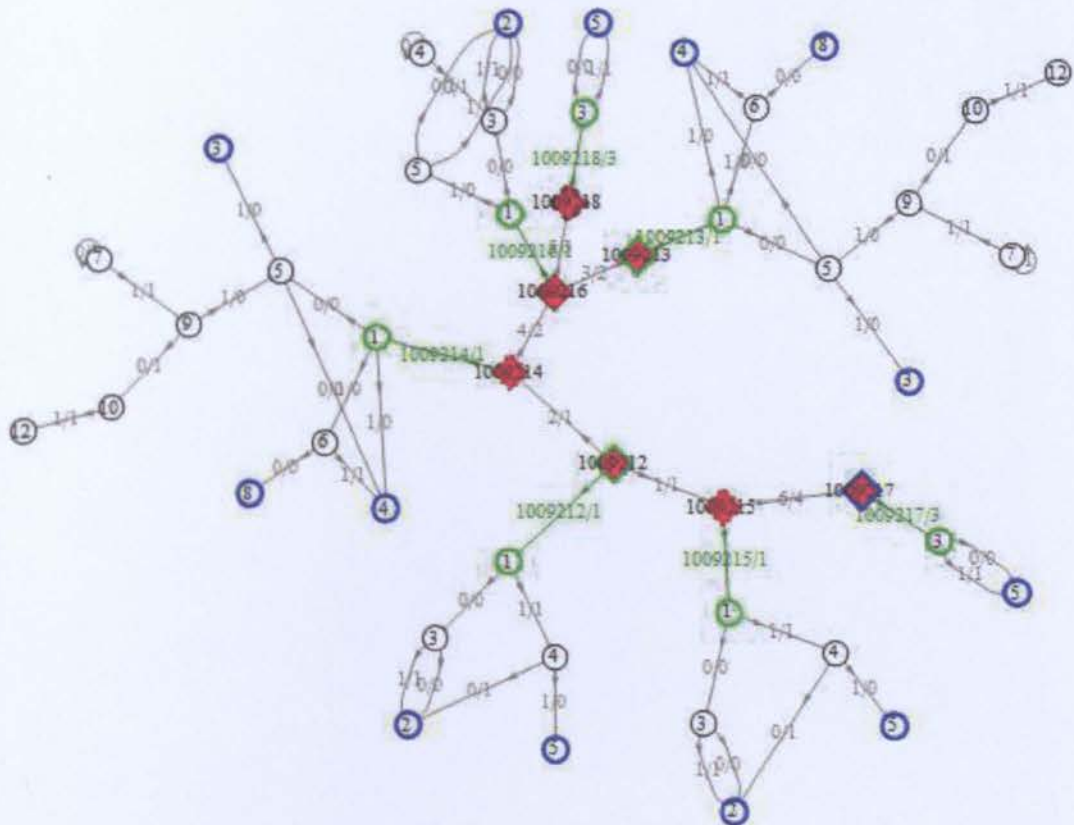


Figure D-14 The depiction of a full adder as a set of Argos schemes (red node) that marshal information using their parallel synchronization tables across prediction schemes (blue and green nodes) that process the information using a series of numbered transitions in grey. It is an example of a binary executable machine that is evolved to interact with the environment.

Through more highly structured combinations, full adders and ripple carry adders can be created. These then have the propensity to fully “count,” in a similar way to a child. For instance, an improperly built ripple carry adder will sometimes get the right values and sometimes not. There is a clear correspondence with Piaget’s observations of children (Copeland, 1974).

D.6 Piagetian Scheme: Using Planning to Count

The example below describes how planning (using a predictive model) would change the processing of assimilation and accommodation and lead to a potentially faster processes of counting. It is included in this appendix for completeness.

Execution of HFSA (Schemes) - Using Predictive Model

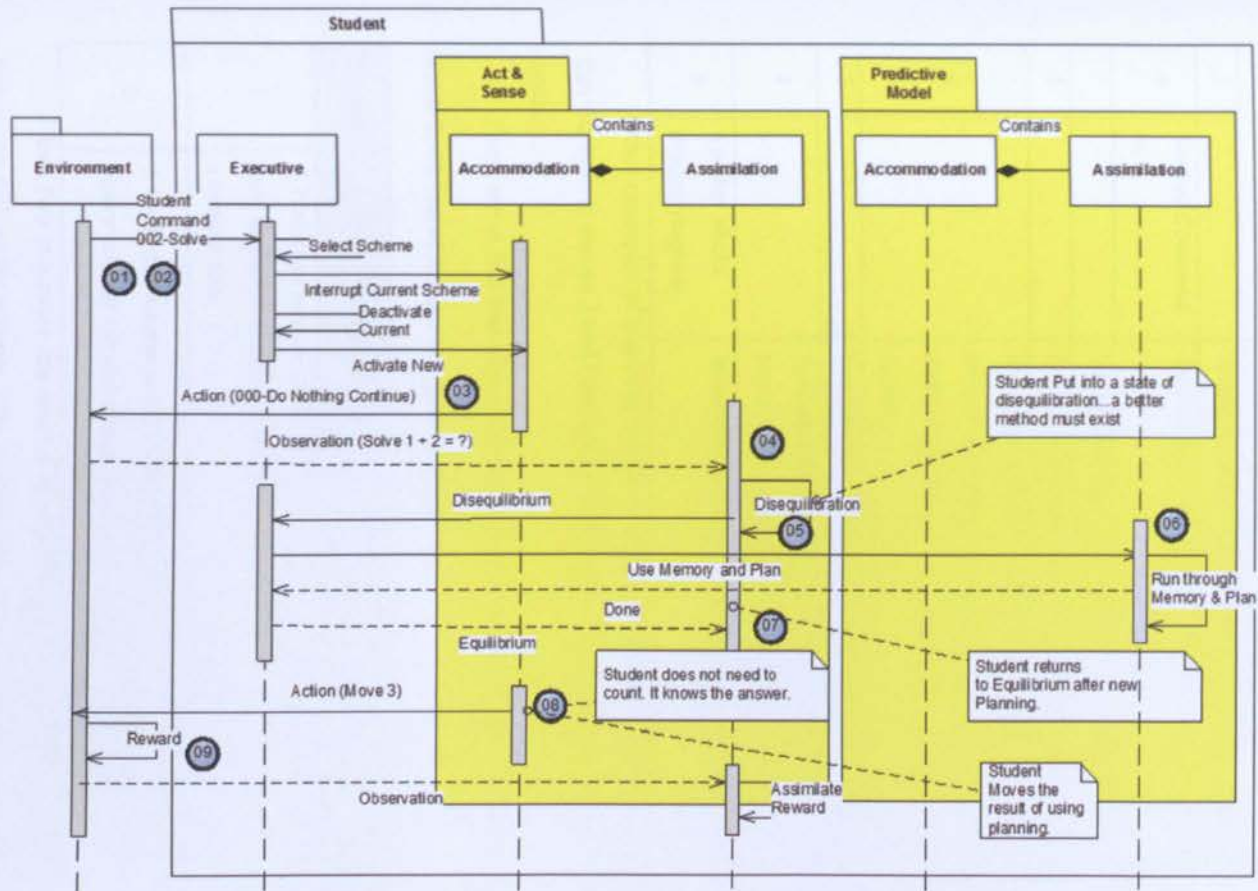


Figure D-15 A UML sequence diagram that shows the use of a predictive model to overcome disequilibrium in a worked example of counting.

Appendix D – Further Examples of Dialectic Evaluation

The steps in the process are described in table D-10.

Table D-10 Steps in the UML sequence for planning

Step	Environment	Student
1.	Student command 001-Wake-Up	Executive of Student Wakes Up and Student responds ok (Action 000). This ok scheme is now the current active scheme.
2	Student command 002-Solve ?	Executive selects scheme to respond, interrupts currently executing scheme and deactivates it then activates selected scheme.
3		Accommodation: Student Action 000 (Do nothing).
4	Student Command ok	Assimilation: Observation of student Command Solve $1+1 = ?$
		Student put into a state of disequilibrium (there must be a better solution).
5		Executive Interrupts processing and attempts to put student into state of Equilibrium.
6		Executive calls the Predictive Model to dynamically generate Argos Scheme that can process more memory and plan a better solution to this problem and in doing so return to state of equilibrium. This processing is dependent on the M-Capacity available to the student.
7		Continuation of Assimilation, Count ($1+1+1$) using Predictive model returns 3.
8		Accommodation: Student Action (Move 3) as execution of predictive FSA PenDown, Right, Move Duration 10403, Stop 3 times.
9	Teacher Marks Student response	Assimilation of reward and strengthening of Argos Scheme. Interaction stored in memory.

Other examples of planning would enable the development of a solution to "Solve $2 + 4$?" It is imagined that the system moves 2 and then adds 4 at the end of the existing movement, rather than going back to the start again.

There are in fact two sets of schemes required to be mutated as described in table D-11:

Table D-11 Sets of Actions to represent 1 bit

Scheme	Assimilation	Accommodation
1.	A Prediction scheme of Move 1 Unit, is used to observe the movement of unit 1, on a number line.	The assimilation of the Prediction Scheme is related to the accommodation (acting internally) of a BUFFERNOT machine producing a binary value of 1.
2.	A Prediction scheme as a BUFFERNOT of binary value of 1 is assimilated internally.	The assimilation of the Prediction Scheme is related to the accommodation (acting externally) of a Prediction scheme of Move 1 Unit.

These two processes are needed as a precursor to counting, scheme 1 allows the bit to be counted by a follow-on full-adder scheme. The second supports serialization i.e., the repeated action of 1-bit, moves the number line units.

Appendix D – Further Examples of Dialectic Evaluation

Bibliography

Bibliography

- Albus, J. S. (1975). A New Approach to Manipulator Control: The Cerebellar Model Articulation Controller (CMAC), *Transactions ASME*, September 1975.
- Albus, J. S. (1991). Outline for a Theory of Intelligence, *IEEE Trans. on Systems, Man and Cybernetics*, Vol. 21, No. 3, May/June 1991.
- Albus, J. S. (1993). A Reference Model Architecture for Intelligent Systems Design, in *An Introduction to Intelligent and Autonomous Control*, Kluwer Academic Publishers, p27–56.
- Albus, J. S., Pape, C. L., Robinson, I. N., Chiueh, T. C., McAulay, A. D., Pao, Y.-H. and Takefuji, Y. (1992). RCS: A reference model architecture for intelligent control. *IEEE Computer*, 25, 56–79.
- Albus, J.S. (2008). Toward a Computational Theory of Mind, *Journal of Mind Theory*, 0, 1, p1–38.
- Albus, J.S. (2010a). A Model of Computation and Representation in the Brain, *Information Sciences*, 180, p1519–1554.
- Albus, J.S. (2010b). Reverse Engineering the Brain, *International Journal of Machine Consciousness*, 2, p193–211.
- Anderson, J. R. (2007). *How can the human mind exist in the physical universe?* New York: Oxford University Press.
- Anderson, J. R. and Lebiere, C. (1998). *The Atomic Components of Thought*, Mahwah, NJ: Lawrence Erlbaum.
- Anderson, J. R. and Lebiere, C. (2003). The Newell test for a theory of cognition, *Behavioral and Brain Sciences*, 26:5, p578–601.
- Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C. and Qin, Y. (2004). An integrated theory of the mind. *Psychological Review*, 111, 1036–1060.
- Anderson, J.R. (2005). *Cognitive Psychology and its implications*. New York: Worth Publishers.
- Baer, R. M. and Martinez, H.M. (1974). Automata and Biology, *Annual Review of Biophysics and Bioengineering*, Vol. 3, p255–291 (Volume publication date June 1974). DOI: 10.1146/annurev.bb.03.060174.001351.
- Balaguer, M. (1998). *Platonism and Anti-Platonism in Mathematics*, Oxford University Press, New York, New York.
- Baldwin, J.M. (1896). A new factor in evolution. *American Naturalist*, 30, p441–451.

Bibliography

- Behr, M. J., Wachsmuth, I., Post, T. R. and Lesh, R. (1984). Order and equivalence of rational numbers: A clinical teaching experiment, *Journal for research in mathematics education*, 15:5, p323–341.
- Belew, R.K. (1993). Interposing an Ontogenic Model Between Genetic Algorithms and Neural Networks. In: *Advances in neural information processing (NIPS5)*. J. Cowan (Ed.). Morgan Kaufmann.
- Benson, H. (1990). Meno, the Slave Boy and the Elenchos. *Phronesis*, 35, p 128–58.
- Bereiter, C. (1985). Towards a solution of the learning paradox. *Review of Educational Research*, 55:2, p201–226.
- Bezuk, N. and Cramer, K. (1989). Teaching about fractions: what, when and how? In: *New Directions for Elementary School Mathematics*. Trafton, P. R. and Schulte, A. P. (editors) Reston, VA: National Council of Teachers of Mathematics.
- Bibel, W. (1981) *Automated Theorem Proving*, Vieweg, Braunschweig Wiesbaden, (revised 1987).
- Bickhard, M. H. (1991a). A Pre-Logical Model of Rationality. In L. Steffe (Ed.) *Epistemological Foundations of Mathematical Experience*. New York: Springer-Verlag, p 68–77.
- Bickhard, M. H. (1991b). The Import of Fodor's Anti-Constructivist Argument. In Les Steffe (Ed.) *Epistemological Foundations of Mathematical Experience*. New York: Springer-Verlag, p14–25.
- Bickhard, M. H. (1996). The emergence of representation in autonomous embodied agents. In *Proceedings of the Fall 1996 AAA symposium on embodied cognition and action*.
- Blank, H., Anwander, A., and von Kriegstein, K. (2012). Neural correlates of the encoding of multimodal contextual features, *Learning & Memory*, 19 November 2012, 19, (12), p605–614.
- Boccaro, N. (2007). *Essentials of Mathematica – with applications to Mathematics and Physics*, Springer, New York.
- Bonasso, P. Kortenkamp., D and Thronesbery., C. (2003). Intelligent Control of a water recovery system: Three Years, *AI Magazine*, 2003, vol 24.
- Bonassoo, P. R., Firby, J. R., Gat, E., Kortenkamp, Miller, D., P. and Slack, M. G (1997). Experiences with an architecture for intelligent, reactive agents. *Journal of Experimental and Theoretical Artificial Intelligence*, 9, 237–256.
- Booch, G., Jacobson, I. and Rumbaugh, J. (2000). *OMG Unified Modeling Language Specification*, Version 1.3 First Edition: March 2000. Retrieved 12 August 2008.
- Booth, J.L. and Siegler, R.S. (2006). Developmental and Individual Differences in Pure Numerical Estimation, *Developmental Psychology*, 41:6, p189–201.
- Bratman, M.E., Israel, D. J. and Pollack, M. E. (1988). Plans and resource-bounded practical reasoning. *Computational Intelligence*, 4:4, p349–355.
- Bratman. M. E. (1987). *Intention, Plans and Practical Reason*. Harvard University Press, Cambridge, Massachusetts, 1987.
- Bresser, R. and Holtzman, C. (1999). *Developing number-sense: Grades 3–6*. Sausalito, Calif.: Math Solutions Publications.

Bibliography

- Bright, G., Behr, M., Post, T. and Wachsmuth, I. (1988). Identifying fractions on number lines. *Journal for Research in Mathematics Education*, 19:3, p215–232.
- Brooks, R.A. (1991). Intelligence without representation, *Artificial Intelligence*, 47, p139–159.
- Bugden, S., Price, G.R., McLean, A., and Ansari, D. (2012). The role of the left intraparietal sulcus in the relationship between symbolic number processing and children's arithmetic competence. *Developmental Cognitive Neuroscience*, 2,4, October 2012, p448–457.
- Byers, W. (2007). *How Mathematicians Think: Using Ambiguity, Contradiction, and Paradox to Create Mathematics*, Princeton University Press.
- Cannon, C.M. and Bseikri, M.R. (2004). Is Dopamine Required for Natural Reward? *Physiology and Behavior*, 81:55,p741–748.
- Carbonell, J. G., Blythe, J., Etzioni, O., Gil, Y., Joseph, R., Kahn, D., Knoblock, C., Minton, S., P´erez,A., Reilly, S., Veloso, M. and Wang, X. (1992). PRODIGY4.0: The manual and tutorial. *Technical Report CMU-CS-92-150*, Carnegie Mellon University.
- Carbonell, J. G., Knoblock, C. A. and Minton, S. (1990). PRODIGY: An integrated architecture for planning and learning. In K. Van Lehn (Ed.), *Architectures for intelligence*. Hillsdale, NJ: Lawrence Erlbaum.
- Carpenter, G., A and Grossberg S. (1987). A massively parallel architecture for a self organizing neural pattern recognition machine, *Computer Vision, Graphics and Image Processing*, 37, p54–115.
- Carpenter, G., A and Grossberg S. (2002). Adaptive Resonance Theory, *The Handbook of Brain Theory and Neural Networks*, Second Edition, Michael A. Arbib, Editor, Cambridge, Massachusetts: MIT Press.
- Carpenter, G.A., Grossberg, S., Markuzon, N., Reynolds, J.H. and Rosen, D.B. (1992). Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps, *IEEE Transactions on Neural Networks*, 3, 698–713.
- Carpenter, G.A., Grossberg, S., Rosen, D.B. (1991)., Fuzzy ART:Fast stable learning and categorization of analog patterns by an adaptive resonance system, *Neural Networks*, 4, p759–771.
- Carroll, L (2004). *Through the Looking Glass*, Dover Thrift Editions.
- Case, R. (1984). The process of stage transition: A neo-Piagetian view. In R. J. Sternberg (Ed.), *Mechanisms of cognitive development*. New York: W. H. Freeman. p19–44.
- Case, R. (1996). Introduction: Reconceptualizing the nature of children's conceptual structures and their development in middle childhood. In R. Case and Y. Okamoto, *The role of central conceptual structures in the development of children's thought*. Monographs of the Society for Research in Child Development, Serial No. 246, Vol. 6. p1–26.
- Cassimatis, N. L., Trafton, J., Bugajska, M. and Schultz, A. (2004). Integrating cognition, perception and action through mental simulation in robots. *Journal of Robotics and Autonomous Systems*, 49, p13–23.
- Chang, C. and Lee, R C. (1973) *Symbolic Logic and Mechanical Theorem Proving*, Academic Press, New York.
- Chaput, H. H. (2004). *The Constructivist Learning Architecture: A Model of Cognitive Development for Robust Autonomous Robots*. Unpublished doctoral thesis, The University of Texas, Austin.

Bibliography

- Chilampikunnel, M, A. (2010) *A Manual for Parents, Teachers, and Principals on Early Childhood Education*, Xlibris, Corp.
- Chomsky, N. (1980). *Rules and Representations*, Blackwell, London.
- Chomsky, N. (1986). *Knowledge of Language, Its Nature, Origin and Use*. New York, Praeger.
- Chomsky, N. (1995). *The Minimalist Program*. Cambridge, Mass.: The MIT Press.
- Chomsky, N. (2000). *New Horizons in the Study of Language and Mind*, Cambridge University Press, p55.
- Churchland, Paul. and Churchland, Patricia., (1990). Could a machine think?, *Scientific American*, 262:1, p32–37.
- Clancey, W. J. (2001). Is abstraction a kind of idea or how conceptualization works? *Cognitive Science Quarterly*, 1, (3–4), p389–421. Special issue on abstraction.
- Colton, S. (2000). An Application-based Comparison of Automated Theory Formation and Inductive Logic Programming, *Electronic Transactions in Artificial Intelligence*, Special issue: Selected papers from Machine Intelligence 17, 4, Section B, p97–118.
- Colton, S. (2002). *Automated Theory Formation in Pure Mathematics*. Springer-Verlag.
- Colton, S. and Muggleton, S. (2006) Mathematical applications of Inductive Logic Programming *Machine Learning*, 64:25 p25–64, DOI 10.1007/s10994-006-8259-x.
- Colton, S. and Wagner, D. (2007). Using Formal Concept Analysis in Mathematical Discovery, Towards Mechanized Mathematical Assistants, *Lecture Notes in Computer Science Volume 4573*, 2007, p 205–220.
- Commons, M. L. and White, M. S. (2003). A complete theory of tests for a theory of mind must consider hierarchical complexity and stage: A commentary on Anderson and Lebiere. *Behavioral and Brain Sciences* 26:5, p606–607.
- Copeland, R. (1974). *How Children Learn Mathematics: Teaching Implications of Piaget's Research*, Macmillan Publishing Co, New York, New York, 2nd Edition.
- Cramer, K., Behr, M., Post, T. and Lesh, R. (1997a). *Rational Number Project: Fraction Lessons for the Middle Grades: Level 1*. Dubuque, IA: Kendall/Hunt Publishing.
- Cramer, K., Behr, M., Post, T. and Lesh, R. (1997b). *Rational Number Project: Fraction Lessons for the Middle Grades: Level 2*. Dubuque, IA: Kendall/Hunt Publishing.
- Crutchfield, J. P (1994a). *Is Anything Ever New? Considering Emergence in Complexity: Metaphors, Models and Reality*, G. Cowan, D. Pines and D. Melzner, editors, Santa Fe Institute Studies in the Sciences of Complexity, Addison-Wesley, Redwood City p479–497.
- Crutchfield, J. P (1994b). The calculi of emergence: Computation, dynamics and induction, *Physica D* (1994) special issue on the *Proceedings of the Oji International Seminar on Complex Systems — from Complex Dynamics to Artificial Reality* held 5 – 9 April 1993, Numazu, Japan.
- Cummings, A. (1998). *Painless Fractions*, Barron's Educational Series, Hauppauge, New York.
- Cuoco, A. A. and Curcio, F. R. (2001). The Roles of Representation in School Mathematics, Yearbook, National Council of Teachers of Mathematics.

Bibliography

- Day, J. M. (1994). *Plato's Meno in Focus*, Day, J. M. ed. London: Routledge.
- de Hevia, M. D., Girelli, L., Macchi-Cassia, V. (2012). Minds without language represent number through space: origins of the mental number line, *Frontiers in Psychology*, 2012, 3, 466, Published online 2012 October 31. DOI: 10.3389/fpsyg.2012.00466PMCID: PMC3484654.
- De Jong, K.A. (2006). *Evolutionary Computation: A Unified Approach*. MIT Press.
- Deacon, T. (1996). *The Symbolic Species: The Co-evolution of Language and the Brain*, W.W. Norton and Company, New York.
- Dehaene, S. (1997). *The Number-sense: How The Mind Creates Mathematics*, Oxford Univ. Press, New York, USA.
- Dehaene, S., Izard, V., Spelke, E.S., and Pica, P. (2008). Log or linear? Distinct intuitions of the number scale in Western and Amazonian cultures. *Science*, 320, 5880, p1217–1220.
- Dellaert, F. and R.D. Beer (1994). Toward an evolvable model of development for autonomous agent synthesis. In: *Artificial Life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems*. R. Brooks and P. Maes (Eds.) MIT Press.
- Dennett, D. (1995). *Darwin's Dangerous Idea*, London, Penguin.
- Desimone, R., Albright, T., Gross, C., and Bruce, C. (1984). Stimulus-selective properties of inferior temporal neurons in the macaque. *The Journal of Neuroscience*, 1 August 1984, 4(8): p2051–2062
- Deutsch, A. and Dormann, S. (2005). *Cellular Automaton Modeling of Biological Pattern Formation*, Birkhauser, Springer Science+Business Media.
- Dorigo, M., Birattari, M. and Stutzle, T. (2006) Ant Colony Optimization – Artificial Ants as a Computational Intelligence Technique, *IEEE Computational Intelligence Magazine*, Nov 2006, p28–39.
- Drescher, G (2002). *Made-Up Minds: A Constructivist Approach to Artificial Intelligence*, MIT Press, Artificial Intelligence Series, Cambridge Mass.
- Drummond, M., Bresina, J. and Kedar, S. (1991). The Entropy Reduction Engine: Integrating planning, scheduling and control. *SIGART Bulletin*, 2, 61–65.
- Duch, W., Oentaryo, R, J.,Pasquier, M. (2008). Cognitive Architectures: Where do we go from here? *Frontiers in Artificial Intelligence and Applications*, Vol 171 Ed. Pei Wang, Ben Goertzel and Stan Franklin, IOS Press, p122–136.
- Duffy, T., M. and Cunningham, D. J. (1996). Constructivism: Implications for the design and delivery of instruction. In D.H. Jonassen (ed.) *Handbook of Research for Educational Communications and Technology*, NY: Macmillan Library Reference USA.
- Dugdale, S. (2008). Research in a Pioneer Constructivist Network-Based Curriculum Project For Children's Learning of Fractions, in *Research on Technology in the Teaching and Learning of Mathematics: Cases and Perspectives* Eds, Glendon W. Blume, Kathleen M. Heid, Vol 2, Information Age Publishing, p3–30.
- Durfee, E., Lesser, V. and Corkill, D. (1989). Trends in Cooperative Distributed Problem Solving. *IEEE Transactions on Knowledge and Data Engineering*, Volume KDE-1, Number 1, p63–83.

Bibliography

- Epstein, S. L. (1992). For the right reasons: The FORR architecture for learning in a skill Domain. *Cognitive Science*, 18, p479–511.
- Epstein, S. L. (2004). Metaknowledge for Autonomous Systems. In *Proceedings of AAAI Spring Symposium on Knowledge Representation and Ontology for Autonomous Systems*. AAAI.
- Epstein, S., L. and Pacini, R. (1999). Some basic issues regarding dual-process theories from the perspective of cognitive-experiential self-theory. In: *Dual-process theories in social psychology*, ed. S. Chaiken and Y. Trope. Guilford Press.
- Fauconnier G and Turner, M. (2002). *The Way We Think: Conceptual Blending And The Mind's Hidden Complexities*, Basic Books Inc., US.A.
- Fauconnier, G. (1985) *Mental Spaces: Aspects of Meaning Construction in Natural Language*, Cambridge University Press, Cambridge, UK.
- Fauconnier, G. (1997) *Mappings in Thought and Language*, Cambridge University Press, Cambridge
- Fauconnier, G. and Turner, M. (1996) Blending as a central process of grammar. *Conceptual structure, Discourse, and Language*, ed. A. Goldberg. Stanford: CSLI Publications, p113–129.
- Fauconnier, G., and Turner, M. (2008). Rethinking metaphor. In R. Gibbs (Ed.), *Cambridge Handbook of Metaphor and Thought*, New York: Cambridge University Press, p53–66.
- Fennell, F. and Landis, T. E. (1994). Number-sense and operation sense. In C. A. Thornton and N. S. Bley (Eds.) *Windows of opportunity: Mathematics for students with special needs*. Reston, VA: National Council of Teachers of Mathematics. p187–203.
- Field, T M., Woodson, R., Greenberg, R., and Cohen D. (1989). Discrimination and imitation of facial expressions by neonate, *Science*, 218 (4568): p179–181.
- Filep, L. (2001). The Development and Developing of, the Concept of a Fraction, *International Journal for Mathematics Teaching and Learning*, <http://www.cimt.plymouth.ac.uk/journal/default.htm>, Last Accessed 2008-02-18.
- Fiorillo, C.D., Tobler, P.N. and Schultz, W. (2003). Discrete Coding of Reward Probability and Uncertainty by Dopamine Neurons, *Science*, 299 (5614), p1898–1902.
- Firby, J. R. (1989). *Adaptive Execution in Dynamic Domains*, Ph.D. thesis, Yale University, 1989.
- Fisher, D. and Langley, P. (1985). Approaches to conceptual Clustering, *Proceedings of IJCAI-85*, Morgan Kaufmann, Los Altos, CA, 1985, p691–697.
- Flanagan, J.R., Vetter, P., Johansson, R.S. and Wolpert, D.M. (2003). Prediction Precedes Control in Motor Learning. *Current Biology*, 13, p146–150.
- Fodor, J. A. (1975). *The Language of Thought*, Cambridge, MA: Harvard.
- Fodor, J. A. (1980). On the Impossibility of Acquiring more Powerful Structures. In *Language and Learning: The debate between Jean Piaget and Noam Chomsky*, Piattelli-Palmarini, M., J. Piaget and N. Chomsky eds. Cambridge, MA: Harvard University Press, 1980, p142–162.
- Fodor, J. A. (1991). Yin and Yang in the Chinese Room, in D. Rosenthal (ed.), *The Nature of Mind*, New York: Oxford University Press.

Bibliography

- Fodor, J. A. (2008). *The Language of Thought Revisited*. Oxford: Oxford University Press. p132 .
- Forgy, C. L. (1982). RETE: A fast algorithm for many pattern/many object pattern matching problem. *Artificial Intelligence*, 19, p17–37.
- Freed, M. (1998). Managing multiple tasks in complex, dynamic environments. In *Proceedings of the fifteenth national conference on artificial intelligence*, Madison, WI: AAAI Press, p921–927.
- Friedman–Hill, E. (2003). *Jess in Action: Rule–Based Systems in Java*, Manning Publications Company.
- Fuhrman, O. and Boroditsky, L (2010). Cross–Cultural Differences in Mental Representations of Time: Evidence From an Implicit Nonlinguistic Task, *Cognitive Science*, 34, p1430–1451.
- Furth, H.G. (1969). *Piaget and Knowledge,: Theoretical Foundations*, Prentice Hall, Englewood Cliffs, New Jersey.
- Gat, E. (1998). On Three–Layer Architectures. In *Artificial Intelligence and Mobile Robots*, Kortenkamp, D., Bonasso, R. P. and Murphy, R., Eds.: AAAI Press. p195–210.
- Gay, A. S. and Aichele, D. B. (1997). Middle School Students understanding of number–sense related to percent, *School Science and Mathematics*, 97:1.
- Geertz, C. (1973). *The Interpretation of Cultures: Selected essays*, Basic Books, Inc.
- Georgeon, O. L., Ritter F. E. and Haynes S. R. (2009). Modeling Bottom–Up Learning from Activity in SOAR. *18th Annual Conference on Behavior. Representation in Modeling and Simulation (BRIMS)*. Sundance, Utah. March 30 – April 2, 2009.
- Giarratano, J. C. (1998). *Expert Systems: Principles and Programming* (Third ed.): Brooks Cole.
- Ginsburg, H. (1977). Learning to Count. Computing with Written Numbers. Mistakes. In Ginsburg, H., *Children's Arithmetic: How They Learn It and How You Teach It*, p1–29, 79–129.
- Gödel, K. (1992). *On Formally Undecidable Propositions Of Principia Mathematica And Related Systems*, New York, Dover.
- Gold, E. M. (1967). Language identification in the limit. *Information and Control*, 10:5,p447—474.
- Goldstone, R. L., Steyvers, M. and Larimer, K. (1996). Categorical perception of novel dimensions. *Proceedings of the Eighteenth Annual Conference of the Cognitive Science Society*. Hillsdale, New Jersey: Lawrence Erlbaum Associates. p 243–248.
- Gordon. D. M. (2010). *Ant Encounters: Interaction Networks and Colony Behavior*. Princeton University Press.
- Granger R. (2006a). Engines of the Brain: The computational instruction set of human cognition, *AI Magazine*, 27, p15–32.
- Granger R. (2006b). The evolution of computation in brain circuitry, *Behavioral and Brain Science*, 29, p17–18.
- Gratch, J. (2000). Emile: Marshaling passions in training and education. In *Proceedings of the fourth international conference on autonomous agents*. Barcelona, Spain. p325–332.
- Greenwood, G. W., and Tyrrell, A, M. (2006) *Introduction to Evolvable Hardware: A Practical Guide for Designing Self–Adaptive Systems*, Wiley–IEEE Press.

Bibliography

- Griffin, S. (2004). Contributions of central conceptual structure theory to education. In A. Demetriou and A. Raftopoulos (Eds.), *Cognitive developmental change: Theories, models and measurement*. Cambridge, UK: Cambridge University Press. p264–295.
- Grossberg, S. and Paine, R. W. (2000). A neural model of corticocerebellar interactions during attentive imitation and predictive learning of sequential handwriting movements. *Neural Networks*, 13, p999–1046.
- Guhe, M., Pease, A., Smaill, A., Martinez, M., Schmidt, M., Gust, H., Kühnberger, K., Krumnack, U., (2011). A computational account of conceptual blending in basic mathematics, *Cognitive Systems Research*, 12, 3–4, September 2011, Elsevier Science Publishers B. V. Amsterdam, The Netherlands, p249–265
- Harel, D. and Pnueli A. (1985). On the development of reactive systems. In *Logics and Models of Concurrent Systems*, volume 13 of NATO ASI Series, p477–498, New York.
- Harnad, S. (1990). The Symbol Grounding Problem. *Physica*, D 42, p335–346.
- Harnad, S. (1993). Grounding Symbols in the Analog World with Neural Nets. *Think 2: 12 – 78* (Special Issue on “Connectionism versus Symbolism” D.M.W. Powers and P.A. Flach, eds.), p68–73.
- Hayes–Roth, B., Pflieger, K., Lalanda, P., Morignot, P. and Balabanovic, M. (1995). A domain–specific software architecture for adaptive intelligent systems. *IEEE Transactions on Software Engineering*, 21, p288–301.
- Haynes, T.; Sen, S.; Schoenefeld, D. and Wainwright, R (1995). Evolving Multiagent Coordination Strategies with Genetic Programming, *Technical Report, UTULSA–MCS–95–04*, The University of Tulsa, May 31, 1995.
- Hecht, S. A. (1998). Toward an Information–Processing Account of Individual Differences in Fraction Skills. *Journal of Educational Psychology*. 90:3,p545–559.
- Hiebert, J. (1984). Children’s mathematics learning: The struggle to link form and understanding. *The Elementary School Journal*, 84:5, p497–510.
- Hjelmfelt, A, Weinberger, E, D., Ross, J (1992). *Proceedings of the National Association of Sciences of USA*, 89, p383–387, January 1992, Chemistry.
- Hoffmann, M. H. G. (2003). Pierce’s “Diagrammatic Reasoning” as a Solution to the Learning paradox, *Process Pragmatism: Essays on a Quite Philosophical Revolution*, Ed. Guy Debrock, Amsterdam,: Rodopi, 2003, p121–143 Available at http://works.bepress.com/michael_hoffmann/6.
- Holland, John H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press.
- Holyoak, K. J. and Spellman, B. A. (1993). Thinking. *Annual Review of Psychology*, 44, p265–315.
- Howden, H. (1989). Teaching Number–sense. *Arithmetic Teacher*, 36:6, p6–11.
- Huang, C., Kaur, J., Maguitman, A. and Rocha, L.M. (2007). Agent–Based Model of Genotype Editing. *Evolutionary Computation*, 15:3.
- Hubbard, E. M., Piazza, M., Pinel, P. and Dehaene, S. (2005). Interactions between number and space in parietal cortex. *Nature Reviews Neuroscience*, 6, p435–448.
- Hutchins, E. (2005) Material anchors for conceptual blends, *Journal of Pragmatics*, 37, p1555–1577.

Bibliography

- Jaaregui, J. A., Jauregui, E. and Jauregui, P. (1995). *The Emotional Computer*, Blackwell Publishers, London.
- Jacob, C. (2001). *Illustrating Evolutionary Computation with Mathematica*, (The Morgan Kaufmann Series in Artificial Intelligence), San Francisco, CA: Morgan Kaufmann Publishers.
- Jacobson, I, Booch, G. and Rumbaugh, J. (1999). *The Unified Software Development Process*, Addison Wesley.
- Jennings, N. R., Sycara, K. (1998). A Roadmap of Agent Research and Development, *Autonomous Agents and Multi-Agent Systems*, 1:1, p275–306.
- Johnson, S. (2001). *Emergence: The Connected Lives of Ants, Brains, Cities, and Software*, Penguin Books, London, England.
- Jonyer, I., Holder L.B. and Cook D.J. (2002). Concept Formation Using Graph Grammars, in *Proceedings of the KDD Workshop on Multi-Relational Data Mining*, Edmonton, Alberta, Canada, p71–79.
- Kant, E. (1999). *Critique of Pure Reason*, The Cambridge Edition of the Works of Immanuel Kant, translated by Paul Guyer and Allen Wood, Cambridge University Press.
- Karmiloff-Smith, A. (1995). *Beyond Modularity: A Developmental Perspective on Cognitive Science*, Cambridge, MA: MIT Press.
- Kauffman. S. A. (1969). Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, 22, p437–467.
- Kaylani, A., Georgiopoulos, M., Mollaghasemi, M., Anagnostopoulos, G. C., Sentelle, C. and Zhong, M. (2010). An Adaptive Multiobjective Approach to Evolving ART Architectures, *IEEE Transactions on Neural Networks*, VOL. 21, NO. 4, APRIL 2010.
- Khamassi, M., Lachèze, L., Girard, B., Berthoz, A. and Guillot, A. (2005). Actor-Critic Models of Reinforcement Learning in the Basal Ganglia: From Natural to Artificial Rats, *Adaptive Behavior*, Vol. 13, No. 2, 131–148.
- Kieras, D. E. and Meyer, D. E. (1997). An overview of the EPIC architecture for cognition and performance with application to human-computer interaction. *Human-Computer Interaction*, 12, 391–438.
- Kieren, T. (1988). Personal knowledge of rational numbers: Its intuitive and formal development. In J. Hiebert and M. Behr (Eds.), *Number concepts and operations in the middle grades*, Hillsdale, NJ: Erlbaum and Reston, VA: National Council of Teachers of Mathematics. p162–181.
- Kitano, H. (1994). Evolution of Metabolism for Morphogenesis. In: *Artificial Life IV: proceedings of the fourth international workshop on the synthesis and simulation of living systems*. R. Brooks and P. Maes (Eds.). MIT Press.
- Klahr, D. and Wallace, J. G. (1976). *Cognitive Development: An Information Processing View*. Hillsdale, NJ: Lawrence Erlbaum.
- Kohonen, T. (1997). *Self-Organizing Maps*. Springer.
- Kolodner, J. (1983). Reconstructive Memory: A Computer Model, *Cognitive Science*, 7:4.

Bibliography

- Koza, J. R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, Massachusetts.
- Koza, J. R. (1994). *Genetic Programming II, Automated Discovery of Reusable Programs*, MIT Press, Cambridge, Massachusetts.
- Kumar S. and Bentley, P. (2003). *On growth, form and computers*. Elsevier Academic Press.
- Kurth, W. (2007). Specification of morphological models with L-systems and relational growth grammars, *Journal of Interdisciplinary Image Science*, 5, Themenheft, p4–10.
- Laird, J. E. (2008). Extending the SOAR Cognitive Architecture, *Frontiers in Artificial Intelligence and Applications*; Vol. 171, *Proceeding of the 2008 conference on Artificial General Intelligence 2008*, p224–235, IOS Press Amsterdam, The Netherlands.
- Laird, J. E., Newell, A. and Rosenbloom, P. S. (1987). SOAR: An architecture for general intelligence. *Artificial Intelligence*, 33, p1–64.
- Lakatos, I. (1975). *Proofs and Refutations: The Logic of Mathematical Discovery*.
- Lakoff, G. (1992). *The Contemporary Theory of Metaphor, Metaphor and Thought*, Ortony, A (ed.) (2nd edition), Cambridge University Press, p203–250.
- Lakoff, G. and Johnson, M. (1980). *Metaphors We Live By*, University of Chicago Press, Chicago
- Lakoff, G. and Núñez, R (2001). *Where Mathematics Comes From: How the embodied mind brings mathematics into being*, Basic Books.
- Lakoff, G., and Johnson, M. (1999). *Philosophy in the Flesh: The Embodied Mind and Its Challenge to Western Thought*, Basic Books.
- Lamon, S. J. (1999). *Teaching Fractions and Ratios Understanding*. Mahwah, NJ: Lawrence Erlbaum Associates.
- Lampert, M. (1986). Knowing, doing and teaching multiplication. *Cognition and Instruction*, 3:4, p304–342.
- Langacker, R. (1987). *Foundations in Cognitive Grammar*, Vol 1, Stanford University Press, Stanford, CA.
- Langley, P. and Choi, D. (2006). A unified cognitive architecture for physical agents. In *Proceedings of the twenty-first AAAI conference on artificial intelligence*. Boston: AAAI Press.
- Langley, P. and Messina, E. (2004). Experimental studies of integrated cognitive systems. In *Proceedings of the performance metrics for intelligent systems workshop*. Gaithersburg, MD.
- Langley, P., Laird, J., E. and Rogers, S. (2008). Cognitive architectures: Research issues and challenges, *Cognitive Systems Research*, doi:10.1016/j.cogsys.2006.07.004, p141–160.
- Lappan, G and Bouck, M., K. (1998). Developing algorithms for adding and subtracting fractions. In: *The Teaching and Learning of Algorithms in School Mathematics*. Morow, L. J. and Kenny, M. J. (editors) Reston, VA: National Council of Teachers of Mathematics.
- Le, Q., V, Ranzato, M A., Monga, R., Devin, M., Chen, K., Corrado, G S., Dean, J., and Ng, A Y. (2012). Building High-level Features Using Large Scale Unsupervised Learning, *Proceedings of the 29th International Conference on Machine Learning*, Edinburgh, Scotland, UK, 2012.

Bibliography

- Lebowitz, M. (1983). Memory-Based Parsing, *Artificial Intelligence*, 21 (1983), 363–404.
- Lee, H., Battle, A., Raina, R. and Ng, A. Y. (2007), Efficient sparse coding algorithms, *Advances in Neural Information Processing Systems*, 19, p801–809.
- Lenat, D. B. (1982). AM: Discovery in Mathematics as Heuristic Search. In Davis, R. and Lenat, D.B. (1982). *Knowledge-Based Systems in Artificial Intelligence*. New York: McGraw-Hill International Book Company, p1–225.
- Lenat, D. B., Prakash, M. and Shepard, M (1986). CYC: Using common sense knowledge to overcome brittleness and knowledge acquisition bottlenecks, *AI Magazine*, American Association for Artificial Intelligence, Menlo Park, CA, USA Volume 6, Issue 4 (Winter 1986), p65–85.
- Levy, A. (2002). FiniteAutomata.m Finite Automata Package for Mathematica , <http://library.wolfram.com/infocenter/Demos/75/> , Listed Feb, 2002.
- Liebeck. P. (1984). *How Children Learn Mathematics*. Penguin Books, Middlesex.
- Lieberman, M. D. (2000). Intuition: A social cognitive neuroscience approach. *Psychological Bulletin* 126:1, p109–137.
- Lieberman, M. D., Gaunt, R., Gilbert, D. T. and Trope, Y. (2002). Reflexion and reflection: A social cognitive neuroscience approach to attributional inference. *Advances in Experimental Social Psychology*, 34, p200–250.
- Luke, S., Hamahashi, S. and Kitano, H (1999). Genetic Programming. In Wolfgang Banzhaf and Jason Daida and Agoston E. Eiben and Max H. Garzon and Vasant Honavar and Mark Jakiela and Robert E. Smith editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 2, p1098–1105, Orlando, Florida, USA, 1999.
- Mack, N. (1990). Learning fractions with understanding: Building on informal knowledge. *Journal for research in mathematics education*, 21:1, p16–32.
- Madden, J. (2001) Where Mathematics comes from: How the Embodied Mind Brings Mathematics into Being, *Notices of the AMS*, vol 48, 10, Nov 2001, p1182–1188.
- Maes, P. (1995). Agents that Reduce Work and Information Overload. In: *Readings in Human-Computer Interaction, Toward the Year 2000*, Baecker, Grudin and Buxton, editors, Morgan Kaufman, 1995.
- Maraninchi, F. (1991). The Argos Language: Graphical Representation of Automata and Description of Reactive Systems, *IEEE Workshop on Visual Languages*, October.
- Maraninchi, F. (1992). Operational and Compositional Semantics of Synchronous Automaton Compositions, (1992), *International Conference on Concurrency Theory – CONCUR*, p550–564, 1992. DOI: 10.1007/BFb0084815.
- Maraninchi, F. and Remond, Y. (2001). Argos: an automaton-based synchronous language, *Computer Languages*, 27, (1/3), (2001), Elsevier, p61–92.
- Markovits, Z. and Sowder, J. (1994). Developing number-sense: An intervention study in grade 7. *Journal for research in mathematics education*, 25:1, p4–29.

Bibliography

- Mason, J., Drury, H. and Bills, L. (2007). Studies in the Zone of Proximal Awareness, Mathematics: Essential Research, Essential Practice, Vol 1, *Proceedings of the 30th annual conference of the Mathematics Education Research group of Australasia*, J. Watson and K. Beswick (Eds.), Merga Inc, 2007, p 49.
- Matthews, G. B. (1999). *Socratic Perplexity and the Nature of Philosophy*. New York: Oxford UP.
- Maturana, H. (1988a). *Ontology of Observing. The Biological Foundations of Self Consciousness and the Physical Domain of Existence*, <http://www.inteco.cl/biology/>.
- Maturana, H. (1988b). Reality: The Search for Objectivity or the Quest for a Compelling Argument, *Irish Journal of Psychology*, 9, p25–82.
- Maturana, H. and Varela, F. (1980). *Autopoiesis and Cognition: The realization of the living*. D. Reidel Pub. Co. Boston.
- Mayo, M. J. (2003). Symbol Grounding and its Implications for Artificial Intelligence, Twenty-Sixth Australasian Computer Science Conference (ACSC2003) in Adelaide, Australia. *Conferences in Research and Practice In Information Technology*, Vol. 16. Michael Oudshoorn, Ed., p55–60.
- McCarthy, J. (1968). Programs with common sense, *Semantic Information Processing*, Volume: 1, Publisher: MIT Press, p403–418.
- McCarthy, J. (1997). *The Well Designed Child*, cogprints, <http://cogprints.org/428/>.
- McClelland, J.L. (1995). A connectionist perspective on knowledge and development. In T.J. Simon and G. S. Hallford (Eds.), *Developing cognitive competence: New approaches to process modeling*, p 157–204, Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.
- McClelland, J.L. and Rumelhart, D. E. (1988). *Explorations in Parallel Distributed Processing*. Cambridge, Mass: MIT Press.
- McCulloch, W. S. and Pitts, W. H. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5, p115–133.
- Mealy, G. H. (1955). A Method for Synthesizing Sequential Circuits, *Bell Systems Technical Journal*, 34, p 1045–1079. http://essay.utwente.nl/59482/1/scriptie_C_Baaij.pdf.
- Meyer, M. and Kieras, D. (1997). A computational theory of executive control processes and human multiple-task performance. Part 1: Basic mechanisms. *Psychological Review*, 104, 3–65.
- Miesenböck, G. (2008). Lighting up the brain. *Scientific American*, 299: October 2008, p52–59.
- Miller, K.F. (1984). Child as measurer of all things: Measurement procedures and the development of quantitative concepts. In C. Sophian (Ed.) *The Origin of Cognitive Skills*. Hillsdale, NJ: Erlbaum.
- Minsky, M. (1967). *Computation: Finite and Infinite Machines*, Prentice-Hall, Inc, Englewood Cliffs, NJ.
- Mitchell, M. (1992). Genetic algorithms. In *Lectures in Complex Systems*. L. Nadel and D. Stein (Eds.). SFI Studies in the Science of Complexity Vol. V, Addison-Wesley. p3–87.
- Mithen, S. (1996) *The PreHistory of the Mind: The Cognitive Origins of Art, Religion and Science*, London, Thames and Hudson.

Bibliography

- Modha, D.S. & Singh, R. (2010). Network architecture of the long-distance pathways in the macaque brain, *PNAS*, 107, p13485–13490
- Moline, J. (1969). Meno's Paradox? *Phronesis*, 14, p153–61.
- Montague, P.R., Hyman, S.E. and Cohen, J.D. (2004). Computational Roles for Dopamine in Behavioral Control. *Nature*, 431, p760–767.
- Moss, J. and Case, R. (1999). Developing Children's Understanding of the Rational Numbers: A New Model and an Experimental Curriculum. *Journal for Research in Mathematics Education*, 30,2 122–47.
- Musliner, D.J., Goldman, R.P. and Pelican, M.J. (2001). Planning with increasingly complex executive models. Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ *International Conference on Intelligent Robots and Systems*. Issue Date: 2001, p2124–2130.
- Nason, R. and Martin, C. (1990). A computer based system for developing expertise in the diagnosis and remediation of common error patterns in the domain of fractions. In J. G. Hedberg, J. Steele and M. Mooney (Eds), *Converging Technologies: Selected papers from EdTech '90*, 205–211. Canberra: AJET Publications. <http://www.aset.org.au/confs/edtech90/nason.html>.
- Nehamas, A. (1985). Meno's Paradox and Socrates as a Teacher. *Oxford Studies In Ancient Philosophy*, 3 (1985), p1–30.
- Newell, A. (1980a). Physical Symbol Systems. *Cognitive Science*, 4, p135–183.
- Newell, A. (1980b). Reasoning, problem solving and decision processes: The problem space as a fundamental category. In R. Nickerson (Ed.), *Attention and Performance VIII*. Hillsdale, NJ: Erlbaum.
- Newell, A. (1990). *Unified theories of cognition*. Cambridge, MA: Harvard University Press.
- Newell, A. and Simon, H. A. (1976). Computer Science as Empirical Inquiry: Symbols and Search, *Communications of the ACM*, 19, <http://portal.acm.org/citation.cfm?id=360022>, p116.
- Niemi, D. (1996). A fraction is not a piece of pie: assessing exceptional performance and deep understanding in elementary school mathematics. *Gifted Child Quarterly*, 40:2, p70–80.
- Nuerk, H., Helmreich, I., Zuber, J., Moeller, K., Pixner S., and Kaufmann, L. (2011). Language Effects on Children's Nonverbal Number Line Estimations, *Journal of Cross-Cultural Psychology*, May 2011, 42, p598–613, first published on May 5, 2011.
- Núñez, R. E. (2011), No Innate Number Line in the Human Brain, *Journal of Cross-Cultural Psychology*, 42,4, p651–658, DOI: 10.1177/0022022111406097.
- O'Reilly, R.C. and Munakata Y. (2000). Computational Explorations in *Cognitive Neuroscience: Understanding of the Mind by Simulating the Brain*. Cambridge, MA: MIT Press, 2000.
- Opfer, J. E., Thompson, C.A., Furlong E. E. (2010). Early development of spatial-numeric associations: evidence from spatial and quantitative performance of preschoolers, *Developmental Science*, Sep 2010, 13,5, p761–771.
- Paivio, A. (1986). *Mental representations: a dual coding approach*. Oxford. England: Oxford University Press.

Bibliography

- Paivio, A. (2006). *Mind and its evolution; A dual coding theoretical interpretation*, Mahwah, NJ: Lawrence Erlbaum Associates, Inc.
- Papert, S. (1980). *Mindstorms*. New York: Basic Books.
- Parisi, D. and Schlesinger, M. (2002). *Artificial Life and Piaget*, *Cognitive Development*, 17:3, p1301–1321
- Pascual–Leone J. and Goodman D. (1979). Instructional Science Series 8, *Intelligence and Experience: A Neo Piagetian Approach*, Elsevier Scientific Publishing Company, Amsterdam.
- Pascual–Leone, J. (1970). A mathematical model for the transition rule in Piaget's developmental stages, *Acta Psychologica*, 32, North–Holland Publishing Company, p301–345.
- Pascual–Leone, J. (1980). Constructive Problems for Constructive Theories: The Current Relevance of Piaget's Work and a Critique of Information–Processing Simulation Psychology, In R. Kluwe and H. Spada (Eds.), *Developmental models of thinking*. New York: Academic Press, p263–296.
- Pascual–Leone, J. (1996). Vygotsky, Piaget and the problems of Plato, *Swiss Journal of Psychology*, 55, 2/3, p84–92.
- Pascual–Leone, J. and Johnson, J. (1999). A dialectical constructivist view of representation: Role of mental attention, executives and symbols. In I.E. Sigel, (Ed.), *Development of mental representation: Theories and applications*. Mahwah, NJ: Erlbaum, p.169–200.
- Pascual–Leone, J., Goodman, D., Ammon, P. and Subelman, I. (1978). Piagetian Theory and Neo–Piagetian Analysis as Psychological Guides in Education, in J.M. Gallagher and J.A.Easley (eds.) *Knowledge and development*, Vol 2, New York: Plenum Publishing Corporation, p243–289.
- Pease, A. (2007). *A computational Model of Lakatos–style Reasoning*, Ph.D. Thesis, School of Informatics, University of Edinburgh.
- Pease, A., Colton, S., and Charnley. J. (2012) Automated theory formation: The next generation. IFCoLog Lectures in Computational Logic, 2012 (Forthcoming).
- Pell, B., Bernard, D. E., Chien, S. A., Gat, E., Muscettola, N., Nayak, P. P., Wagner, M., D. and Williams, B. C. (1997). An autonomous spacecraft agent prototype. In *Proceedings of the first international conference on autonomous agents*. Marina del Rey, CA: ACM Press. p253–261.
- Piaget, J. (1952). *The Child's Conception of Number*. New York: W. W. Norton.
- Piaget, J. (1953). How children develop mathematical concepts, *Scientific American*, Nov 1953, Scientific American, Inc, p74.
- Piaget, J. (1954). *The Construction of Reality in the Child*, Basic Books Inc, New York, p5.
- Piaget, J. (1962a). *Play, dreams, and imitation in childhood*. New York: Norton.
- Piaget, J. (1962b). The stages of the intellectual development of the child, *Bulletin of the Menninger Clinic*. Reprinted in *Personality, Development and Learning: A Reader*, London: Hodder and Stoughton in association with the Open University, p31–38.
- Piaget, J. (1963). *Origins of Intelligence in Children*, New York, W.W. Norton, p160–162.

Bibliography

- Piaget, J. (1964). Development and Learning, in R.E. Ripple and V.N. Rockcastle (Eds.), *Piaget Rediscovered*, p7–20.
- Piaget, J. (1968). *Six Psychological Studies*, (D. Elkind, Ed), Vintage, N. Y.
- Piaget, J. (1970a). *Genetic Epistemology* (Woodbridge lectures delivered at Columbia University in October of 1968, no. 8), translated by Eleanor Duckworth, New York: Columbia University Press.
- Piaget, J. (1970b). *Structuralism*. New York: Basic Books.
- Piaget, J. (1976). *Le comportement, moteur de l'évolution*. Paris: Gallimard. Re: *The Motor behavior of evolution*, p18.
- Piaget, J. (1978). *Psychology and Epistemology: Towards a Theory of Knowledge*, Penguin Books, Middlesex, England.
- Piaget, J. (1980a). The psychogenesis of knowledge and its epistemological significance, in M. Piattelli-Palmarini, ed., *Language and Learning: The Debate between Jean Piaget and Noam Chomsky*, Harvard University Press, Cambridge, Mass.
- Piaget, J. (1980b). Schemes of action and language learning, in M. Piattelli-Palmarini, ed., *Language and Learning: The Debate between Jean Piaget and Noam Chomsky*, Harvard University Press, Cambridge, Mass.
- Piaget, J. (1983). Piaget's theory (translated by Guy Cellérier and Jonas Langer), in W. Kessen (Editor), *Handbook of child psychology*, 4th edition, Volume 1: History, theory and methods, New York: Wiley, p125.
- Piaget, J. (2000a). Commentary on Vygotsky. *New Ideas in Psychology*, 18, p241–59.
- Piaget, J. (2000b). *Studies in reflecting abstraction*, edited and translated by Robert L. Campbell; Hove: Psychology Press, 2000, Chapter 2, p57.
- Piaget, J., Inhelder, B. and Szeminska, A. (1960). *The Child's Conception of Geometry*. New York: Basic Books.
- Piattelli-Palmarini, M. (1980). *Language and learning: The debate between Jean Piaget and Noam Chomsky*, Massimo Piattelli-Palmarini (Editor), Cambridge, MA: Harvard University Press.
- Pinker, S. (1994). *The Language Instinct: The New Science of Language and Mind*, New York, Penguin Books.
- Plato. (1985). *Meno*. Trans. and ed. R.W. Sharples. Chicago: Bolchazy-Carducci; Warminster: Aris and Phillips.
- Poldrack, R.A. (2010). Interpreting developmental changes in neuroimaging signals. *Human Brain Mapping*.
- Poli, R. (2001). General Schema Theory for Genetic Programming with Subtree-Swapping Crossover, In *Genetic Programming*, Proceedings of EuroGP 2001, LNCS, Springer-Verlag, p18–20.
- Poli, R. and McPhee, N. (2003a). General schema theory for genetic programming with subtree-swapping crossover: Part i. *Evolutionary Computation*, 11:2, p53–66.
- Poli, R. and McPhee, N. (2003b). General schema theory for genetic programming with subtree-swapping crossover: Part ii. *Evolutionary Computation*, 11:2, p169–206.

Bibliography

- Polya, G. (1988). *How to solve it: A New Aspect of Mathematical Method*, 2nd Ed, Princeton University Press, Princeton, New Jersey.
- Post, T. and Cramer, K. (1987). Research into practice: Children's strategies in ordering rational numbers. *Arithmetic Teacher*, 35:2, p22–25.
- Post, T., Wachsmuth, I., Lesh, R. and Behr, M. J. (1985). Order and equivalence of rational numbers: A cognitive analysis. *Journal for research in mathematics education*, 16:1, p18–36.
- Prigogine, I. (1980). *From Being to Becoming: Time and Complexity in the Physical Sciences*, New York, W.H. Freeman and Company.
- Putnam, H. (1975). The meaning of "meaning". In *Philosophical Papers*, Vol. 2: Mind, Language and Reality. Cambridge University Press, p227.
- Pyysiäinen, I. (2003). Dual-process theories and hybrid systems: A commentary on Anderson and Lebiere. *Behavioral and Brain Sciences*, 26:5, p617–18.
- Raina, R., Battle, A., Lee, H., Packer, B., and Ng, A.Y. (2007). Self-taught learning: Transfer learning from unlabeled data. In *ICML, 2007*, p759–766.
- Rao, A. S. and Georgeff, M. P. (1991). Modeling Rational Agents within a BDI-Architecture. In: *Proc. Of Knowledge Representation and Reasoning (KR&R 1991)*.
- Rao, A. S. and Georgeff, M. P. (1995a). BDI Agent: From theory to Practice" in *Proc. of the First International Conference on Multi-Agent Systems (ICMAS-95)*, San Francisco, CA, June 12–14, 1995, p.312–319.
- Rao, A.S. and Georgeff, M. P. (1995b). BDI Agents: From Theory to Practice April 1995, *Technical Note 56*.
- Rao, R.P.N. and Ballard, D.H. (1999). Predictive Coding in the Visual Cortex: A Functional Interpretation of Some Extra-Classical Receptive-Field Effects. *Nature Neuroscience*, 2:1, p79–87.
- RCS (2009). Real-Time Control Systems Library — Software and Documentation at [nist.gov](http://www.isd.mel.nist.gov/projects/rcslib/index.html). <http://www.isd.mel.nist.gov/projects/rcslib/index.html>, Accessed Aug 4, 2009.
- Reber, A.S. (1989). Implicit Learning and Tacit Knowledge, *Journal of Experimental Psychology*, vol118 no3, p219–235.
- Reeve, R., Lloyd, D., Reynolds, F. and Butterworth, B. (2011). Using mental representations of space when words are unavailable. Studies of enumeration and arithmetic in indigenous Australia, *Journal of Cross-Cultural Psychology*, Volume 42, issue 4 (May 2011), p.630–638, ISSN: 0022-0221 DOI: 10.1177/0022022111406020.
- Ren, J., Orwell, J., Jones G., and Xu, M (2004) A general framework for 3D soccer ball estimation and tracking, *Proc. IEEE Int. Conf. on Image Processing*, p1935–1938.
- Rendell, G. (2008). Learning Fraction-Sense: Applying Piaget's Operative Theory of Meaning in a Multi Agent System to Overcome the Learning paradox, *The International Journal of Learning*, Vol 15, Issue 2, Common Ground Publishing Pty Ltd, Melbourne, Australia, Mary Kalantzis, Bill Cope (Ed.), p233–244.

Bibliography

- Rescher, N. (2001). *Paradoxes: Their Roots, Range and Resolution*, Open Court Publishing Company, Chicago, IL.
- Resnick, L.B., Nesher, P., Leonard, F., Magone, M., Omanson, S. and Peled, I. (1989). Conceptual bases of arithmetic errors: The case of decimal fractions. *Journal for Research in Mathematics Education*, 20, 8–27.
- Rey, G. (1986), What's Really Going on in Searle's "Chinese Room", *Philosophical Studies*, 50: p169–185.
- Rocha, L. M. (1998). Selected Self–Organization and the Semiotics of Evolutionary Systems. In: *Evolutionary Systems: The Biological and Epistemological Perspectives on Selection and Self– Organization*. S. Salthe, et al (eds.) Kluwer, p341–358.
- Roswell, J. A. (1983) Equilibration: Developing the Hard Core of the Piagetian Research Program, *Human Development*, 26, p61–71.
- Rymel, J.D., Renno, J.R., Greenhill, D., Orwell, J., Jones, G.A. (2004). Adaptive Eigen- Backgrounds for Object Detection, *IEEE International Conference on Image Processing*, October, Suntec City, Singapore.
- Sackur–Grisvard, C. and Leonard, F. (1985). Intermediate cognitive organizations in the process of learning a mathematical concept: The order of positive decimal numbers. *Cognition and Instruction*, 2:2, p157–174.
- Schank, R. (1982). *Dynamic Memory: A Theory of Learning in Computers and People*, New York: Cambridge University.
- Schultz, W. (1997). *Reward responses of dopamine neurons: A biological reinforcement signal*, Lecture Notes in Computer Science, Heidelberg: Springer Berlin, 1327, p1–12.
- Schultz, W. (2000). Multiple Reward Signals in the Brain. *Nature Reviews Neuroscience*, 1:p199–207.
- Schultz, W. and Dickinson, A. (2000). Neuronal Coding of Prediction Errors. *Annual Review of Neuroscience*, 23, p473–500.
- Schwartz, M. and Fischer, K. W. (2004). Building general knowledge and skill: Cognition and micro development in science learning. In A. Demetriou and A. Raftopoulos (Eds.), *Cognitive developmental change: Theories, models and measurement*. Cambridge, UK: Cambridge University Press, p157–185.
- Scowen, R, S. (1993). Extended BNF — A generic base standard. *Software Engineering Standards Symposium 1993*.
- Searle, J. (1980). Minds, Brains and Programs, *Behavioral and Brain Sciences*, Volume 3:3, p 417–457, doi:10.1017/S0140525X00005756, <http://web.archive.org/web/20071210043312/http://members.aol.com/NeoNoetics/MindsBrainsPrograms.html>, retrieved May 13, 2009.
- Searle, J. (1984). *Minds, Brains and Science*, Cambridge: Harvard University Press.
- Searle, J. (1990). Is the Brain's Mind a Computer Program?, *Scientific American*, 262: p26–31.
- Searle, J. (1999). The Chinese Room, in R.A. Wilson and F. Keil (eds.), *The MIT Encyclopedia of the Cognitive Sciences*, Cambridge, MA: MIT Press.

Bibliography

- Shaki, S., Fischer, M. H., Göbel, S. M. (2012) Direction counts: A comparative study of spatially directional counting biases in cultures with different reading directions. *Journal of Experimental Child Psychology*, vol. 112 issue 2 June, 2012. ISSN: 0022–0965 DOI: 10.1016/j.jecp.2011.12.005, p275–281.
- Shang, Y., Claridge–Chang, A., Sjulson, L., Pypaert, M. and Miesenböck, G. (2007). Excitatory local circuits and their implications for olfactory processing in the fly antennal lobe. *Cell*,128: p601–612.
- Shapiro, S. C. and Ismail, H. O. (2003). Anchoring in a grounded layered architecture with integrated reasoning. *Robotics and Autonomous Systems*, 43, 97–108.
- Sherman, A. M. and Guillery, R.W. (2006) *Exploring the Thalamus and its Role in Cortical Function*, 2nd edition, The MIT Press, Cambridge, Massachusetts.
- Sherman, S. M. and Guillery, R. W. (2001). *Exploring the Thalamus*, Academic Press, Elsevier, New York.
- Shultz, T. R. (1994). The challenge of representational redescription, *Behavioral and Brain Sciences*,17:4, p728–729.
- Shultz, T. R., Schmidt, W.C., Buckingham, D. and Mareschal, D. (1995). Modeling cognitive development with a generative connectionist algorithm. In T.J. Simon and G.S. Halford (Eds.) *Developing Cognitive Competence: New approaches to process modeling*, p205–261, Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.
- Siekmann, J. and Wrightson, G. (1983). *Automation of Reasoning*, Springer–Verlag, Berlin, 1983 (published in 2 volumes).
- Simon, T. and Klahr, D. (1995). A Computational Theory of Children’s Learning About Number Conservation, In T. Simon and G.S. Halford (Eds.), *Developing Cognitive Competence: New Approaches to Process Modeling*, Hillsdale, NJ: Erlbaum.
- Singh, S., Barto, A. G. and Chentanez, N. (2005). Intrinsically Motivated Reinforcement Learning. In *Advances in neural information processing systems*, 17, Proceedings of the 2004 conference. Cambridge MA: MIT Press.
- Sloman, A. (1985). What enables a machine to understand? In *Proceedings 9th International Joint Conference on AI*, Los Angeles, August 1985, p995 –1001.
- Sloman, A. (1987). Reference without causal links. In J. B. H. du Boulay, D. Hogg and L. Steels, editors, *Advances in Artificial Intelligence – II*, p369–381. North Holland, Dordrecht, 1987.
- Sloman, A. (2001). Varieties of affect and the CogAff architecture schema. In *Proceedings of the AISB’01 symposium on emotion, cognition and affective computing*. York, UK, p39–48.
- Sloman, Steven. A. (1996). The empirical case for two systems of reasoning. *Psychological Bulletin*, 119:1, p3–22.
- Sloman, Steven. A. (1999). Rational versus rational models of thought. In: *The nature of cognition*, ed. R. J. Sternberg. MIT Press.
- Smith L. B. and Thelen, E. (2003). Development as a dynamic system, *Trends in Cognitive Sciences*, Vol.7, No.8, August 2003, p343–348.
- Smuts, J. (1927). *Holism and Evolution*. London: McMillan and Co Limited, p88.

Bibliography

Sørmo, F., Cassens, J. and Aamodt, A. (2005). Explanation in Case-Based Reasoning—Perspectives and Goals. *Artificial Intelligence Review*, Vol 24, 2, October 2005, p109–143.

Sowder, J. (1995). Instructing for rational number-sense. In J.T. Sowder, B.P. Schappelle (Eds), *Providing a foundation for teaching mathematics in the middle grades*, Albany, NY: State University of New York Press, p15–30.

Sowder, J. (1997). Place value as the key to teaching decimal operations. *Teaching children mathematics*, 3:8, p448–453.

Star, J. R. (2000). On the Relationship Between Knowing and Doing in Procedural Learning. In B. Fishman and S. O'Connor-Divelbiss (Eds.), *Fourth International Conference of the Learning Sciences*, Mahwah, NJ: Erlbaum. p80–86.

Stevens, A. (1988). A rational reconstruction of Boyer and Moore's techniques for constructing induction formulas, *Proceedings of ECAL-88*, 1988, p565–570.

Streeter, T., Oliver, J. and Sannier, A. (2006). Verve: A General Purpose Open Source Reinforcement Learning Toolkit. In *Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference 2006*, September 10–13, 2006, Philadelphia, Pennsylvania, USA.

Sun, R. (2004). Desiderata for cognitive architectures. *Philosophical Psychology*, Vol.17, No.3, p341–373. 2004.

Sun, R., Merrill, E. and Peterson, T. (2001). From implicit skills to explicit knowledge: A bottom-up model of skill learning. *Cognitive Science*, 25, p203–244.

Suri, R.E. and Schultz, W. (1998). Learning of Sequential Movements by Neural Network Model with Dopamine-Like Reinforcement Signal. *Experimental Brain Research*, 121:3, p350–354.

Sutton, R.S. and Barto, A.G. (1998). *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA.

Taatgen, N. A., Juvina, I., Herd, S., Jilk, D. and Martens, S. (2007). Attentional blink: An internal traffic jam? *Eighth International Conference on Cognitive Modeling*.

Taddeoa, M. and Floridi, L. (2005). Solving the symbol grounding problem: a critical review of fifteen years of research, *Journal of Experimental and Theoretical Artificial Intelligence*, Volume 17, Issue 4 December 2005, p419 – 445.

Tadepalli, P. (2003). Cognitive architectures have limited explanatory power. *Behavioral and Brain Sciences*, 26:5, p 622–623. Commentary on Anderson and Lebiere, 2003.

Thompson, P. W. and Lambdin, D. (1994). Research into practice: Concrete materials and teaching for mathematical understanding. *Arithmetic Teacher*, 41:9, p556–558.

Tirosh, D. (2000). Enhancing Prospective Teachers' Knowledge of Children's Conceptions: The Case of Division of Fractions. *Journal for Research in Mathematics Education*, 31:1, p5–25.

Bibliography

- Tobler, P. N., O'Doherty, J. P., Dolan, R. J. and Schultz, W. (2006). Human Neural Learning Depends on Reward Prediction Errors in the Blocking Paradigm, *Journal of Neurophysiology*, 95, First Published September 28, 2005, p301–310.
- Turing, A. (1950). Computing Machinery and Intelligence, *Mind*, 59, p433–460.
- Turner, M. (2011) The Way We Imagine, in Theory of Mind and Literature, Eds (Leverage, P., Mancing, H., Schweickert, R. and William, J. M.), Purdue University Press, p41–62.
- Van Noord, G. and Gerdemann, D. (2001). Finite State Transducers with Predicates and Identities, *The Journal of Grammars*, 2001, v4, p263–286.
- Veloso, M.M., Carbonell, J. G., Perez, A., Borrajo, D., Fink, E. and Blythe, J. (1995). Integrating Planning and Learning: The PRODIGY Architecture, *Journal of Experimental and Theoretical Artificial Intelligence*, 7:1, 1995.
- Von Glasersfeld, E (1998). Scheme Theory as a Key to the Learning paradox, *15th Advanced Course, Archives Jean Piaget Geneva*, September 20–24, p9.
- Waddington, C. H. (1942). *The evolution of an evolutionist*. Ithica, NY: Cornell University Press.
- Wang, H., Johnson, T. R. and Zhang, J. (2003). A multilevel approach to cognitive modeling (Commentary on Anderson and Lebiere – The Newell Test for a Theory of Cognition) *Behavioral and Brain Sciences*, 26:5, p626–627.
- Wearne, D. and Hiebert, J. (1988). Constructing and using meaning for mathematical symbols: The Case of decimal fractions. In J. Hiebert and M. Behr (Eds.) *Number concepts and operations in the middle grades*, Hillsdale, NJ: Erlbaum and Reston, VA: National Council of Teachers of Mathematics.
- Wei, H. and Fu, X. (2009). 2009 *International Conference on Machine Learning and Computing*, IPCSIT vol.3 (2011), IACSIT Press, Singapore, p117–p122.
- Wellin, P., Gaylord, R., Kamin, S. (2006). *An introduction to programming with Mathematica*, 3rd edition, Cambridge University Press, Cambridge, UK.
- White, N. (1976). *Plato on Knowledge and Reality*. Indianapolis: Hackett.
- Williamson, J.R. (1995). Gaussian ARTMAP: A Neural Network for Fast Incremental Learning of Noisy Multidimensional Maps, *Technical Report CAS/CNS-95-003*, Boston University, Center of Adaptive Systems and Department of Cognitive and Neural Systems.
- Wirth, N. (1995). *Digital Circuit Design for Computer Science Students: An Introductory Textbook*, Springer-Verlag, Berlin Heidelberg.
- Wolfram, S. (2003). *The Mathematica Book*, Wolfram Media, Fifth Edition, 2003.
- Wooldridge, M. J. (2000). *Reasoning about Rational Agents*. Cambridge, MA: MIT Press.
- Wooldridge, M. J. and Jennings, N. R. (1995). Intelligent Agents: Theory and Practice, *The Knowledge Engineering Review*, 10:2, p115–152.
- Wray, R. E. and Laird, J. E. (2003). An architectural approach to consistency in hierarchical execution. *Journal of Artificial Intelligence Research*, 19, p355–398.

Bibliography

Wray, R.E. and Jones, R. (2005). An introduction to SOAR as an agent architecture. In *Cognition and Multi Agent Interaction From Cognitive Modeling to Social Simulation*. R. Sun (Ed.), Cambridge University Press. Publisher: Cambridge University Press, p53–78.

Yuret, D. (1996). *The Binding Roots of Symbolic AI: A Brief Review of the Cyc Project*, <http://www.ai.mit.edu/people/deniz/publications/cyc96/>

Glossary

Glossary

#	Term	Description
1)	Act	The Piagetian accommodation process of "recognition" (§2.3.6.1 and § 4.2.1.2) is implemented by the system using the act algorithm (§ 4.2.3.16 for algorithm 16). This enables the system to act on the environment.
2)	Artificial Neural Network Implementation	This research created an <i>artificial neural network implementation</i> to test for emergent of permanent object invariants.
3)	ATF	ATF is a synonym for automated theory formation (Colton, 2000).
4)	ATP	ATP is a synonym for automated theory proving (Colton, 2000).
5)	Automaton Response	Each Prediction scheme, as a finite state automata, is designed to take a given input and produce a given output, using its transitions. The output of an automaton is its automaton response (§ 4.2.3.2 for algorithm 2).
6)	Baldwin Effect	The genetic specification alone does not determine what can be learned neurally, but what is learned affects learning. Otherwise referred to as the Baldwin effect (Baldwin, 1896; Waddington, 1942), it can be understood, as what is passed through the genome is not so much neural learned behavior, but the disposition to learn that behavior.
7)	Cognizant failure	Cognizant failure is based on the idea of designing a system to detect failures and recover from them as opposed to never failing (Gat, 1998, p198). Originating on Firby's research on adaptive control (Firby, 1989). Like nature, it sidesteps the issue of designing perfect algorithms, but requires a contingency recovery procedure for each failure. In this implementation, the executive is used to handle error situations.
8)	Complex vs. Complicated	Whereas complicated systems have many parts, each part can be reduced and studied separately through a process of deduction is contrasted with complex systems such as human cognition, which are not fully predictable, so cannot be decomposed. More importantly, the models used to describe these complex systems induce principles based on empirical observations.
9)	Concept	A concept is a mental structure, which generalizes over at least two features of the environment, in order to mediate behavior (Goldstone, Steyvers and Larimer, 1996, p243).

#	Term	Description
		In this sense, generalization is the capability to correctly respond to situations that have previously been available; mediation is the capability to modularize situations, as collections of sub structures where adaptation is the capability for the subject to learn in the environment that it is placed in. Within the system, a concept is a finite state automata that has been evolved through mutation to be able to predict the next state of the environment. The intention of the system is to test the development of concepts using a Piagetian model of learning and development (§2.3).
10)	Constructivism	The interaction between a genome and the environment allows concepts to emerge through the phenotype.
11)	Creating Concepts	The system initially creates a concept through the addition of a class of chromosomes (§ 4.2.3.15 for algorithm 15), and then the evolution through mutation of a population of chromosomes that attempts to meet the definition supplied by the class (§ 4.2.3.8 for algorithm 8).
12)	Curiosity Reward	The curiosity reward is the value of the reward produced by the environment at time t , that is in response to some curiosity behavior on the part of the Agent. The curiosity reward is a special case of reward.
13)	Current Reward	The value of the current reward as produced by the environment at time t .
14)	Design Constraints	Design Constraints limit the design such as requiring an agent design.
15)	Development	Development is assumed to define LP6: Children develop their learning process. In this, the construction of what is already known and the process by which that knowledge is acquired is altered by the maturing child (Furth, 1969, p105).
16)	Dialectic System	The Dialectic System is generated by this research to test a Piagetian model of cognition using hierarchical finite state automata.
17)	Dual Process Theories	The dual mind hypothesis, often referred to as dual coding theory, asserts that visual information processed by the brain are analogue codes with verbal information being symbolic (Paivio 1986; Paivio,2006). These already formed symbols “allow” cognitive modeling researchers to incorporate symbolic information into their systems in the form of propositional and predicate logics (Anderson, 2005). Two supporting biological systems of dual process theories (dual mind hypothesis) are the spontaneous system (using the lateral temporal cortex, amygdala and basal ganglia) and the rational system (using the anterior cingulate, prefrontal cortex and hippocampus). The rational system either generates solutions to problems encountered by the spontaneous system, or it biases its processing in a variety of ways. To quote Pyysiäinen, “A pre-existing doubt concerning the veracity of one’s own inferences seems to be necessary for the activation of the rational system” Pyysiäinen, 2003, p617. The rational system thus identifies problems arising in the spontaneous system, takes control away from it and remembers situations in which such control was previously required. These operations are plausibly considered to consist of generating and maintaining symbols in working memory, combining these symbols with rule based logical schemes and biasing the spontaneous system and motor systems to

#	Term	Description
		behave accordingly (Lieberman, Gaunt, Gilbert, Trope, 2002).
18)	Embedded Figures Task	The Embedded Figures Task (EFT) is designed to measure disembedding, a restructuring skill, which results from the use of style. According to Bonham (Bonham, 1987), the EFT was adapted from Gottschaldt's figures by adding colored patterns to increase complexity. Each complex figure included an embedded simple figure, which the subject is to identify as quickly as possible; there are 24 figures in the EFT.
19)	Embryogenesis and Learning and Development	The development of knowledge is a spontaneous process, tied to the whole process of embryogenesis. Embryogenesis concerns the development of the body and it includes as well the development of the nervous system and the development of mental functions. In the case of the development of knowledge in children, embryogenesis ends only in adulthood. It is a total development process in which one must resituate in its general biological and psychological context. In other words, development is a process, which concerns the totality of the structures of knowledge. Learning presents the opposite case. In general, learning is provoked by situations – provoked by a psychological experimenter; or by a teacher, with respect to some didactic point; or by an external situation. It is provoked, in general, as opposed to spontaneous. In addition, it is a limited process – limited to a single problem, or to a single structure (Piaget, 1964, p8).
20)	Empiricism	All concepts are learned through experience. In empiricism, the brain mediates the interaction between the world and the body, by the formation of concepts about the body it is contained on and the environment it is embedded in.
21)	Executive	An executive is the process by which the governing processes of Act and Sense, Reinforce and Predictive Model, determine if disequilibrium has occurred and following the developmental trend, make corrections to put the system into a state of equilibrium. Evidence suggests that there is a biological basis for this switching (Miesenböck, 2008, p52; Shang, Claridge-Chang, Sjulson, Pypaert and Miesenböck, 2007, p601).
22)	Fitness	Fitness is the measure of a Prediction scheme to adapt to the environment. It augments prediction quality with attributes that assess usefulness (§ 4.2.3.3 for algorithm 3).
23)	FSA	FSA is a synonym of finite state automata.
24)	Functionality	Functionality includes the architecturally significant system-wide functional requirements.
25)	Genetic Epistemology	Genetic Epistemology deals with both the formation and the meaning of knowledge. The essential question is: "By what means does the human mind go from a lower to a higher level of knowledge?" The fundamental hypothesis of genetic epistemology is that there is a parallelism between the progress made in logic (or the rational organization of knowledge) and the corresponding formative psychological processes" (Piaget, 1970a, p13 cited in Copeland, 1974, p5). Piaget, as a constructivist, held some basic assumptions about his genetic epistemology that can be fitted to the research concerns. First, Continuity Vs discontinuity. As a stage theorist, Piaget contends that there is a discontinuity that occurs at various stages of development (As will be seen by this research, the development of levels of discontinuity is not doubted, but the attribution to developmental stages for an individual is suspect). One

#	Term	Description
		<p>interesting question is what is the impact to conflict on development? Would an increase in conflict, show an increase in development?</p> <p>Second, Nature vs. Nurture.</p> <p>1) An individual is born without "mind" i.e., they have some innateness -instincts (Furth, 1969, p261) and primary reactions (Furth, 1969, p146) - but that development of intelligence is operational (Furth, 1969, p244). Using this definition, Piaget supports both nature and nurture. 2) An individual has hereditary organic reactions i.e., there is support for the notion of nature; 3) An individual is naturally active and engages with the world i.e., Piaget thus supports the notion of nurture;</p> <p>Third, Knowledge Generation and Meaning.</p> <p>An individual needs to adapt to their environment. In this sense, Piaget supports innate reason and ideas since these are used and developed over time as schemes. Piaget also supports some of the notions of empiricisms, in that the external world is a construction by the individual, who cannot initially be separated from it. Also, to adapt, an individual organizes his/her thinking into structures (schemes) and for Piaget, the essential determinant of genetic epistemology is the construction of these structures using assimilation and accommodation and the control of these through equilibration that allows for the emergence of intelligence.</p>
26)	Genotype	Genotype describes the genetic constitution of an individual that is the specific allelic makeup (An allele is an alternative form of a gene – which may have a different characteristic, e.g., color of line etc.) of an individual, usually with reference to a specific character under consideration.
27)	HFSA	HFSA is a synonym of hierarchical finite state automata.
28)	Imitation	The Piagetian accommodation process of "imitation" (§2.3.6.2 and § 4.2.1.2) is implemented by the system using the imitation algorithm (§ 4.2.3.18 for algorithm 18). This enables the system to imitate observations in the environment by making an internal accommodation. This internal accommodation, as imitation, leads the reconstruction.
29)	Learning Concepts	The system uses the evolution of classes of chromosomes, as the conceptual learning mechanism. This process of evolution through random mutation, is encapsulated within an algorithm (§ 4.2.3.5 for algorithm 5).
30)	Learning Paradox	"To put it most simply, the paradox is that if one tries to account for learning by means of mental actions carried out by the learner, then it is necessary to attribute to the learner a prior structure that is as advanced or complex as the one to be acquired" (Bereiter, 1985, p202).
31)	Learning Steps	The system may take many attempts to develop a concept (as a constructed automaton within a population of chromosomes with a specific class); this process is encapsulated in the mutation of copies of chromosomes (§ 4.2.3.6 for algorithm 6). The quality of learning is defined by the chromosome's prediction quality.
32)	Learning Strategy	The system uses an evolutionary strategy to implement the Piagetian concept of learning through assimilation and accommodation (§ 4.2.3.7 for algorithm 7).

#	Term	Description
33)	LP	A set of constraints for a system that overcomes the learning paradox, as defined in § 2.1.7: LP1: Works Within The Constraints Imposed By Fodor's Arguments. LP2: Exhibits Emergence of Hierarchical Concepts using evolutionary process. LP3: Operates Autonomously. LP4: Mirrors Real World Behavior of Children Learning Number Sense using a Number line. LP5: Learns Incrementally with Minimal Innate Knowledge and Reflexes. LP6: Develops its Learning Process. LP7: Acts in Novel, Opportunistic and Noisy Situations.
34)	MNL	MNL is a synonym for the mental number line (Dehaene, Izard, Spelke and Pica, 2008).
35)	Molecules of Cognition	In this research, an argument is made that cognition is the emergence of structure from networks of FSA and HFSA (schemes as discrete machines). Since there is reuse of binary components depending on need, that these machines are molecular. From this, one could argue that cognition (in this simulation at least) is molecular and that the general approach described in this research is the appearance of molecules of cognition.
36)	Nativism	All concepts have an innate basis: the available atomic symbols and the rules for combination are specified beforehand and remain unchanged. Thus all concepts are provided in the genome, at a genetic level. This contrasts with the experimental evidence of stage transition.
37)	Neurons	Sensory neurons transmit information from sensory receptors throughout the body. Motor neurons transmit information from brains to muscles. Neurons contain nucleus to hold genetic information as well as a membrane, mitochondria, golgi body and cytoplasm. Neurotransmitters released by axon (nerve fibers) and dendrites (branches) pass through to synapses to other neurons. There are 100's of types of neurotransmitters with each using a distinctive chemical signal. The human brain has 10^9 neurons and 10^{12} synaptic connections. FSA can represent any neuron (Minsky, 1967).
38)	Ontogenesis	Sequence of events involved in the development of an individual organism from the translation of the genome onto the actual phenotype.
39)	Operators	There are a set of mutation operators that are designed to enable the construction of finite state automata using random evolution (§ 4.2.3.4 for algorithm 4).
40)	Package	A package is a self-contained unit that can be developed and deployed in a controlled fashion to achieve the aims of the system. A package has a series of interfaces that enable it to be used and tested independently and by design reduce the risk of the whole system not functioning correctly. By design, each package aims to have high cohesion and low coupling with other packages. They also need to be self-contained and can be consumed by other packages. A package is generally related to a feature or set of features of a system, or they may meet a specific need or set of needs as defined in the vision of the system. The package becomes a mechanism to trace the developed solution back to the initial requirements and scope of the solution.
41)	Performance	Performance involves throughput of data, system response time and recovery time.

#	Term	Description
42)	Permanent Object Invariant	The permanent object invariants are emergent from the sensory motor processing, and in this system represent the external environment features of point, line, direction, penState (penUp, penDown, and stop) and movement. (Furth, 1969, p248) (§ 2.3.5).
43)	Phenotype	Phenotype is the observable constitution of an organism; Phenotype is the appearance of an organism resulting from the interaction of the genotype and the environment; A phenotype describes any observed quality of an organism, such as its morphology, development, or behavior, as opposed to its genotype – the inherited instructions it carries, which may or may not be expressed.
44)	Phylogenesis	Evolutionary development of a species or feature of an organism.
45)	Physical Constraints	Physical constraints include physical hardware and networking environment.
46)	Prediction Quality	Prediction Quality is the measure of a Prediction Scheme to correctly predict a sequence of input values from the environment (§ 4.2.3.1 for algorithm 1).
47)	Reconstruction	The Piagetian assimilation process of “reconstruction” (§4.6.2 and § 2.3.6.3) is implemented by the system using the reconstruction algorithm (§ 4.2.3.19 for algorithm 19). This enables the system to reconstruct observations in the environment through the generation of chromosomes.
48)	Reliability	Reliability includes aspects such as availability, scalability, accuracy, recoverability.
49)	Scheme and Schema	Schema does not refer to the work of Poli (Poli, 2001, p19; Poli and McPhee, 2003a, 2003b) who presents a general schema theory for genetic programming. A scheme for Piaget is “the coordination and organization of adaptive action, considered as a behavioral structure within the organism, such that the organism can transfer and generalize the action to similar and analogous circumstances” (Furth, 1969, p44).
50)	Selection	The system uses a number of selection mechanisms in its evolutionary strategy, these include best selection (§ 4.2.3.9 for algorithm 9), elite selection (§ 4.2.3.10 for algorithm 10), fitness proportionate selection (§ 4.2.3.11 for algorithm 11), rand based selection (§ 4.2.3.12 for algorithm 12), random selection (§ 4.2.3.13 for algorithm 13) and tournament selection (§ 4.2.3.14 for algorithm 14).
51)	Sense	The Piagetian assimilation process of “perception” (§2.3.1) is implemented by the system using the sense algorithm (§ 4.2.3.17 for algorithm 17). This enables the system to observe (sense) the environment.
52)	TCO	TCO is a synonym of Theory of Constructive Operators (Pascual-Leone, 1970).
53)	UML	Software architecture and design is commonly organized into a set of views, with each view representing a set of system elements and the relationships among them. Since no consensus exists on which symbol-set or language should be used to describe each view, the Unified Modeling Language version 2.2 will be used (Booch, Jacobson and Rumbaugh, 2000). The UML delineates into two categories, with structural diagrams defining the structure of the solution and behavior diagrams, defining the dynamic behavior of the functional requirements. The RUP method of classification adopted by this thesis is known as FURPS+. Where “FURPS” indicates F unctionality, U sability, R eliability, P erformance and S upportability and “+” indicates design constraints, implementation constraints,

#	Term	Description
		<p>interface constraints and physical constraints that the system must perform number.</p> <p>Further, these non-functional requirements can be considered those features of the system that provide the structure to enable the Agents, in the simulation, to have the necessary observed behavior. It is this epigenetic framework, which the agent cannot manipulate. Over generations, student performance is enhanced, through selective breeding based with students with better performance characteristics.</p>
54)	Usability	Usability includes user interface issues such as accessibility, interface aesthetics, consistency.
55)	WE	<p>The worked example, as defined in § 2.3.14</p> <p>WE1 – Base Level.</p> <p>WE2 – Constrained Level – Conservation of Length, Measurement, Point.</p> <p>WE3 – Differentiated Level – Object – Conservation of Object.</p> <p>WE4 – Hierarchical Level – Segmentation into Unit – Conservation of Number.</p> <p>WE5 – General Level – Relationships of Unit Values.</p> <p>WE6 – Rational Level – Relationships of Unit Values to Continuous Values.</p>

Index

Index

- Accommodation, viii, xxii, xxiii, xxiv, xxviii, 18, 19, 20, 27, 32, 33, 34, 35, 36, 37, 38, 39, 41, 42, 46, 47, 51, 52, 62, 74, 84, 91, 95, 96, 112, 114, 116, 117, 118, 119, 122, 123, 124, 125, 126, 133, 137, 138, 139, 141, 142, 143, 144, 161, 162, 163, 206, 208, 209, 211, 212, 213, 218, 219, 220, 222, 224, 228, 229, 233, 264, 265, 267, 268, 273, 298, 300, 323, 325, 326
- Act
- Algorithm 16, iii, xxii, xxiii, xxiv, xxvi, xxix, 7, 18, 29, 32, 34, 37, 38, 46, 47, 53, 63, 66, 92, 105, 106, 107, 110, 122, 123, 125, 138, 141, 142, 143, 159, 161, 163, 167, 174, 175, 178, 179, 182, 186, 187, 190, 191, 206, 219, 222, 248, 256, 262, 265, 270, 323
- Act And Sense, xxii, xxiii, xxiv, xxix, 37, 47, 66, 105, 106, 107, 122, 123, 141, 142, 143, 161, 167, 178, 186, 190, 265, 270
- Adaptation, xx, xxii, xxiv, xxviii, 19, 20, 26, 27, 28, 30, 33, 35, 47, 61, 77, 109, 117, 122, 123, 133, 137, 142, 143, 148, 149, 205, 208, 210, 214, 218, 225, 235, 240, 246, 264, 323, 325
- Add class
- Algorithm 15, xiii, 158
- Addition, 3, 8, 13, 19, 28, 41, 42, 46, 51, 53, 54, 55, 66, 79, 81, 83, 151, 189, 199, 212, 236, 241, 249, 254, 257, 258, 266, 272, 273, 277, 285, 324, 325
- Full-Adder, xiv, xv, xix, xxi, xxiv, xxviii, xxix, xxx, 130, 134, 202, 203, 204, 205, 209, 210, 211, 212, 219, 221, 282, 285, 292, 293, 294, 295, 296, 297, 300
- Agent, xx, xxiv, xxv, 28, 29, 58, 68, 77, 90, 92, 110, 111, 112, 127, 162, 165, 166, 167, 168, 169, 170, 171, 247, 251, 255, 256, 257, 258, 259, 260, 261, 262, 266, 306, 315, 322, 324, 328
- Algorithm
- Algorithm 1

- Prediction Quality, xxv, xxvi, xxvii, 127, 131, 132, 133, 145, 146, 147, 148, 149, 150, 152, 158, 159, 160, 172, 173, 174, 176, 177, 178, 191, 192, 193, 195, 325, 326
- Algorithm 10
 - Elite, 153, 154, 155, 158, 328
 - Elite Selection, 155, 328
- Algorithm 11
 - Fitness Proportionate Selection, 153, 155, 156, 183, 328
- Algorithm 12
 - Rank Based Selection, 156
- Algorithm 16
 - Act, iii, xxii, xxiii, xxiv, xxvi, xxix, 7, 18, 29, 32, 34, 37, 38, 46, 47, 53, 63, 66, 92, 105, 106, 107, 110, 122, 123, 125, 138, 141, 142, 143, 159, 161, 163, 167, 174, 175, 178, 179, 182, 186, 187, 190, 191, 206, 219, 222, 248, 256, 262, 265, 270, 323
- Algorithm 17
 - Sense, iii, xxiv, xxvi, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 18, 25, 26, 29, 32, 37, 42, 43, 45, 48, 51, 52, 53, 54, 55, 56, 57, 61, 64, 67, 70, 74, 76, 79, 81, 83, 84, 85, 98, 99, 101, 102, 103, 105, 107, 108, 122, 123, 137, 141, 143, 159, 161, 163, 165, 170, 174, 175, 180, 181, 182, 184, 188, 190, 191, 194, 196, 205, 206, 216, 218, 219, 220, 222, 224, 226, 227, 229, 230, 231, 232, 233, 234, 244, 247, 248, 253, 256, 258, 261, 264, 268, 271, 280, 303, 306, 307, 308, 309, 312, 313, 320, 323, 325, 328
- Algorithm 18
 - Imitation, xxiii, 33, 36, 38, 40, 41, 45, 46, 120, 123, 133, 144, 159, 160, 162, 267, 307, 309, 315, 326, 344
- Algorithm 19
 - Reconstruction, xxiii, 36, 38, 46, 114, 119, 123, 124, 144, 159, 160, 161, 162, 249, 267, 320, 326, 328
- Algorithm 3
 - Fitness, xxv, xxvi, xxvii, 127, 131, 132, 145, 147, 148, 149, 150, 152, 153, 155, 156, 157, 158, 172, 173, 174, 176, 177, 183, 184, 192, 195, 246, 328
- Algorithm 5
 - Evolve, xxv, 31, 72, 80, 86, 113, 131, 149, 150, 151, 152, 153, 161, 162, 173, 174, 202, 225, 245, 246, 264, 265, 278
 - Evolve Classes, 150, 161
- Algorithm 6
 - Evolve Chromosome, 150, 151, 152
- Algorithm 9

- Best Selection, 154, 155, 328
- Algorithm 9 thru 15
- Mutation, xxv, xxvi, xxvii, xxviii, 22, 127, 130, 144, 145, 148, 149, 150, 151, 152, 154, 155, 156, 157, 161, 178, 186, 189, 190, 191, 192, 193, 197, 202, 203, 204, 206, 208, 215, 217, 218, 219, 225, 230, 243, 289, 295, 323, 324, 326, 327
- Mutation Selection, 289
- Selection, xxiv, 12, 20, 21, 22, 81, 82, 87, 105, 110, 112, 131, 135, 149, 152, 153, 154, 155, 156, 157, 158, 173, 174, 183, 243, 245, 246, 252, 256, 257, 262, 328, 344
- Algorithm 16, 159, 161, 219, 323
- Algorithm 18, xxiv, 144, 145, 159, 161, 326
- Architecture, xx, xxiii, xxviii, 7, 8, 22, 35, 75, 76, 80, 82, 85, 90, 93, 98, 102, 109, 110, 119, 125, 163, 164, 172, 196, 202, 203, 206, 215, 217, 221, 222, 225, 232, 251, 254, 255, 256, 260, 261, 262, 295, 302, 303, 304, 307, 309, 310, 311, 314, 319, 322, 328
- Argos, iii, xiv, xv, xix, xx, xxi, xxiv, xxvii, xxviii, xxix, xxx, 31, 101, 105, 118, 125, 126, 129, 130, 133, 134, 135, 136, 137, 138, 139, 141, 152, 158, 162, 163, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 225, 226, 229, 235, 237, 245, 246, 247, 265, 267, 268, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 300, 309, 312
- Argos Scheme, xxi, xxvii, xxviii, xxix, xxx, 125, 126, 130, 133, 137, 139, 162, 163, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 211, 213, 214, 215, 216, 218, 220, 267, 268, 282, 283, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297
- Artificial Neural Network Implementation, xxiii, 7, 8, 9, 31, 57, 97, 98, 99, 101, 104, 105, 106, 107, 108, 109, 113, 163, 164, 165, 172, 218, 219, 222, 225, 228, 231, 232, 233, 234, 236, 239, 323
- Assimilation, viii, xxii, xxiii, xxiv, xxviii, 18, 19, 20, 27, 32, 33, 34, 35, 36, 37, 39, 41, 42, 46, 47, 51, 52, 62, 74, 84, 91, 95, 96, 112, 114, 116, 117, 118, 119, 122, 123, 124, 125, 126, 133, 138, 139, 141, 142, 143, 144, 159, 161, 162, 163, 206, 208, 209, 211, 212, 213, 218, 219, 220, 222, 224, 228, 229, 233, 264, 265, 267, 273, 298, 300, 325, 326, 328
- Algorithm 16, 159, 161, 219, 323

Index

- Algorithm 18, xxiv, 144, 145, 159, 161, 326
- ATF, 5, 58, 59, 60, 61, 62, 98, 323
- ATP, 5, 58, 59, 61, 62, 98, 323
- Attention, 4, 22, 23, 27, 70, 74, 85, 240, 261, 279, 344
- Attentional Energy, 23, 24, 25
- Autonomy, 75, 89, 177, 239
- Basic Gate, 139, 146, 147, 148, 162, 296
- Bead Problem, xxii, xxvii, 4, 7, 26, 27, 28, 29, 44, 49, 50, 51, 102, 103, 105, 107, 108, 137, 170, 182, 187, 188, 189, 190, 218, 219, 222, 268, 271, 276
- Bereiter, vii, 1, 2, 3, 5, 6, 11, 13, 14, 15, 18, 22, 25, 29, 76, 91, 99, 220, 225, 226, 231, 303, 326, 344
- Defined Factors, 14, 15, 29, 344
- Best Selection
- Algorithm 9, 154, 155, 328
- Bootstrapping, 13, 14, 15, 29, 60, 71, 73, 91, 225
- Buffer, xxiv, xxv, 131, 132, 135, 140, 142, 146, 147, 148, 173, 174, 197, 198, 288
- Buffer Machine, xxiv, 131, 135, 140, 142, 198
- Chaput, 10, 90, 93, 94, 95, 171, 218, 231, 304
- Chomsky, vii, 6, 7, 10, 11, 12, 13, 15, 18, 19, 21, 25, 29, 78, 99, 248, 305, 307, 316, 344
- Chromosome, xiii, xxiv, xxvi, xxvii, xxviii, 127, 128, 129, 131, 133, 134, 145, 146, 147, 149, 150, 151, 152, 153, 154, 157, 158, 159, 160, 161, 172, 174, 176, 177, 183, 184, 185, 190, 191, 192, 193, 195, 196, 202, 203, 208, 209, 283, 293, 295, 326
- CLA, xi, 93, 94, 95
- Class, xx, xxiv, xxv, xxvi, xxvii, xxviii, 7, 95, 114, 125, 127, 131, 132, 133, 137, 141, 142, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 158, 160, 161, 172, 173, 174, 175, 177, 183, 184, 185, 190, 195, 203, 248, 249, 265, 266, 267, 273, 277, 278, 279, 293, 295, 324, 326
- Cognitive Development, iii, xx, 5, 6, 7, 8, 10, 13, 14, 32, 61, 74, 89, 96, 98, 100, 124, 224, 229, 232, 251, 304, 319
- Cognizant Failure, 84, 92, 210, 218
- Concept, 2, 4, 5, 6, 7, 9, 11, 15, 16, 17, 18, 20, 22, 26, 27, 33, 39, 44, 45, 46, 49, 50, 53, 54, 58, 59, 60, 61, 62, 66, 67, 73, 77, 80, 82, 85, 86, 88, 91, 96, 98, 99, 114, 119, 164, 187, 224, 228, 229, 230, 231, 232, 233, 234, 247, 255, 268, 279, 280, 318, 323, 324, 326, 344
- Conceptual Knowledge, 280
- Concrete Operations, 34, 36, 39, 44, 62, 205, 226, 267, 271, 279
- Conservation, xxii, xxvii, 4, 7, 23, 28, 34, 36, 38, 39, 44, 46, 48, 50, 53, 113, 171,

Index

- 188, 190, 205, 219, 222, 264, 265, 266,
267, 271, 272, 276, 278, 279
- Constraint, xx, 2, 4, 27, 47, 51, 59, 61, 68,
79, 135, 238
- Continuous Sensor, xx, xxii, xxv, xxvi,
18, 25, 47, 48, 81, 106, 109, 111, 165,
166, 167, 168, 169, 171, 181, 185, 258,
264, 266, 273
- Counting, xxi, xxiv, xxviii, xxx, 26, 29,
35, 44, 46, 50, 51, 55, 69, 70, 71, 103,
114, 118, 130, 139, 140, 203, 206, 208,
210, 211, 212, 215, 216, 217, 218, 219,
222, 264, 265, 268, 271, 273, 276, 277,
289, 294, 298, 299, 300, 319
- Curiosity, xxv, 110, 111, 167, 168, 169,
170, 225, 324
- Declarative, 28, 77, 88, 252
- Decoder, xxi, xxix, 202, 209, 287, 288,
289, 290
- Deduction, 19, 58, 72, 92, 100, 101, 114,
323
- Design, xx, xxiii, xxviii, 8, 13, 34, 35, 52,
58, 75, 89, 90, 93, 99, 100, 104, 105,
108, 109, 112, 119, 120, 125, 142, 161,
163, 164, 172, 196, 202, 206, 212, 213,
215, 225, 227, 232, 241, 244, 246, 255,
256, 261, 306, 324, 327, 328
- Development, iii, iv, xx, xxiii, xxviii, 2, 3,
4, 5, 6, 7, 8, 9, 10, 12, 13, 14, 16, 17,
18, 19, 20, 21, 22, 25, 26, 27, 28, 29,
30, 32, 35, 36, 38, 40, 41, 42, 43, 44,
45, 46, 48, 49, 51, 53, 54, 55, 57, 58,
59, 61, 62, 63, 64, 66, 67, 68, 72, 74,
75, 76, 78, 82, 83, 85, 86, 87, 89, 91,
93, 94, 95, 97, 98, 99, 102, 103, 104,
105, 106, 111, 113, 114, 133, 137, 148,
153, 163, 166, 167, 171, 189, 199, 205,
208, 211, 217, 218, 219, 220, 222, 224,
225, 226, 227, 228, 229, 230, 232, 233,
234, 236, 241, 249, 255, 259, 264, 265,
266, 267, 268, 272, 276, 277, 279, 280,
281, 290, 300, 304, 306, 309, 310, 313,
314, 315, 318, 323, 325, 327, 328
- Development Of Logic, 59, 205, 304, 305,
311, 315
- Developmental Trend, xxii, 19, 32, 41, 42,
43, 46, 89, 137, 219, 222, 259, 260,
268, 281, 325
- Dialectic System, iii, xxiii, 8, 9, 31, 57,
97, 98, 99, 101, 105, 106, 108, 109,
113, 161, 163, 164, 171, 172, 182, 190,
219, 225, 228, 233, 234, 282
- Discrete Sensor, xx, xxv, 47, 48, 82, 106,
107, 109, 111, 112, 162, 165, 166, 167,
168, 169, 170, 171, 172, 182, 186, 248,
266, 273, 327
- Disequilibrium, xxviii, xxx, 32, 96, 126,
203, 210, 211, 222, 265, 294, 299, 300
- Dual Process Theory, 87, 90, 324
- Elite
Algorithm 10, 153, 154, 155, 158, 328
- Elite Selection

Index

- Algorithm 10, 155, 328
- Embryogenesis, 19, 36, 325
- Emergence, iii, viii, x, xv, xvi, xvii, xx, xxii, xxiii, xxxi, 2, 3, 6, 7, 8, 9, 10, 12, 14, 25, 26, 27, 28, 29, 30, 31, 47, 48, 49, 50, 51, 52, 57, 62, 66, 67, 69, 70, 72, 73, 74, 83, 85, 90, 91, 94, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 163, 164, 165, 170, 172, 182, 190, 194, 206, 217, 219, 222, 224, 227, 228, 229, 231, 232, 233, 234, 235, 236, 244, 247, 248, 249, 250, 259, 276, 278, 303, 305, 310, 325, 327, 344
- Empiricism, 2, 19, 44, 82, 325
- Encoder, xxi, xxviii, 196, 197, 198, 199, 200, 201, 202, 290
- Entity Relationship Diagram, xxiii, 114, 115, 116, 117, 121
- Equilibration, xxviii, 19, 20, 21, 22, 23, 24, 25, 32, 35, 37, 42, 46, 47, 61, 96, 112, 126, 133, 203, 210, 217, 219, 222, 264, 265, 294, 325
- Equivalence, xviii, xx, xxii, xxiii, xxiv, xxix, 28, 44, 46, 49, 50, 51, 107, 117, 118, 126, 133, 135, 136, 137, 138, 139, 140, 141, 162, 189, 190, 222, 247, 268, 271, 273, 274, 276, 277, 278, 303, 317
- Evocation, xxiii, 36, 38, 39, 124, 162, 226
- Evolution, xx, xxiv, xxv, xxvi, 5, 7, 8, 10, 12, 13, 18, 20, 28, 29, 30, 31, 39, 52, 56, 65, 66, 76, 78, 87, 92, 99, 112, 119, 125, 127, 131, 132, 133, 136, 141, 142, 150, 153, 160, 163, 172, 173, 174, 175, 182, 183, 190, 194, 203, 219, 222, 224, 227, 229, 230, 232, 233, 235, 236, 243, 244, 245, 246, 249, 250, 262, 276, 295, 302, 306, 308, 315, 316, 321, 324, 326, 327
- Evolutionary Trend, xxii, 19, 27, 32, 42, 43, 51, 89, 259, 260, 281
- Evolve
- Algorithm 5, xxv, 31, 72, 80, 86, 113, 131, 149, 150, 151, 152, 153, 161, 162, 173, 174, 202, 225, 245, 246, 264, 265, 278
- Evolve Chromosome
- Algorithm 6, 150, 151, 152
- Evolve Classes
- Algorithm 5, 150, 161
- Figurative Scheme, viii, xxii, 32, 33, 34, 35, 37, 41, 44, 45, 46, 69, 130, 189, 209, 211, 222, 238
- Finite State Automata, iii, xvi, xvii, xxiii, xxiv, xxx, xxxi, 6, 7, 8, 30, 31, 98, 105, 106, 107, 112, 118, 119, 120, 123, 125, 126, 127, 128, 129, 130, 133, 134, 135, 137, 142, 145, 146, 147, 148, 149, 154, 160, 161, 162, 174, 187, 211, 217, 219, 222, 224, 225, 226, 228, 233, 235, 236, 237, 238, 243, 245, 246, 247, 249, 250, 265, 268, 279, 295, 296, 300, 323, 324, 325, 326, 327

Index

- Transition, xxv, xxvi, xxvii, xxix, 17, 20, 23, 24, 70, 130, 131, 133, 147, 148, 173, 184, 197, 198, 199, 237, 244, 289, 304, 315, 327
- Fitness**
- Algorithm 3, xxv, xxvi, xxvii, 127, 131, 132, 145, 147, 148, 149, 150, 152, 153, 155, 156, 157, 158, 172, 173, 174, 176, 177, 183, 184, 192, 195, 246, 328
- Fitness Proportionate Selection**
- Algorithm 11, 153, 155, 156, 183, 328
- Fodor, vii, viii, xv, 1, 2, 5, 6, 7, 10, 11, 15, 16, 17, 18, 19, 21, 22, 25, 27, 29, 46, 47, 48, 49, 50, 51, 62, 73, 79, 80, 81, 83, 84, 89, 93, 99, 137, 163, 189, 217, 225, 229, 231, 245, 247, 256, 258, 259, 260, 261, 262, 281, 303, 307, 308, 327, 344
- Formal Operations Stage, 62**
- Furth, xxii, 3, 6, 7, 8, 9, 10, 13, 19, 21, 28, 29, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 49, 52, 59, 62, 73, 74, 76, 78, 83, 84, 86, 89, 91, 98, 99, 103, 104, 113, 114, 119, 120, 125, 126, 133, 162, 164, 171, 190, 205, 217, 218, 222, 224, 225, 226, 227, 232, 259, 260, 267, 279, 308, 324, 325, 328, 344
- Genetic Epistemology, viii, 6, 7, 8, 9, 11, 12, 17, 21, 22, 23, 26, 32, 47, 51, 52, 164, 218, 224, 225, 228, 229, 230, 232, 233, 264, 316, 325, 344**
- Genotype, 328**
- Greater Than, xxiii, 46, 51, 169, 269, 272, 273, 280**
- HFSA, xxx, 219, 295**
- Hierarchical Finite State Automata, xvii, xxiii, xxiv, xxviii, xxxi, 6, 7, 31, 32, 40, 54, 98, 105, 106, 107, 112, 117, 118, 119, 120, 122, 123, 124, 126, 127, 128, 129, 133, 134, 135, 136, 137, 139, 162, 196, 204, 211, 214, 217, 219, 225, 226, 227, 228, 229, 230, 237, 238, 239, 243, 245, 247, 248, 249, 250, 264, 265, 267, 268, 271, 272, 273, 276, 277, 278, 279, 326, 327**
- Synchronization, xxviii, xxx, 195, 198, 199, 203, 207, 214, 215, 297**
- Imitation, xxiii, 33, 36, 38, 40, 41, 45, 46, 120, 123, 133, 144, 159, 160, 162, 267, 307, 309, 315, 326, 344**
- Algorithm 18, xxiii, 33, 36, 38, 40, 41, 45, 46, 120, 123, 133, 144, 159, 160, 162, 267, 307, 309, 315, 326, 344**
- Induction, 2, 15, 18, 58, 72, 92, 101, 113, 114, 265, 279, 305, 320**
- Innate Scheme, viii, ix, xv, 3, 24, 27, 28, 35, 56, 218, 247, 314, 327**
- Interiorization, xxii, 19, 32, 36, 40, 41, 42, 43, 95, 96, 118, 130, 189, 190, 209, 216, 218, 219, 222, 229, 233, 276, 279**

Index

- Invariant, 78, 95, 171, 259, 268
- Knowing Circle, viii, xxii, xxiv, xxxi, 32, 34, 35, 36, 39, 40, 45, 66, 84, 89, 91, 119, 138, 206, 209, 219, 229, 259, 260
- Language, 19, 68, 71, 303, 305, 306, 307, 308, 312, 314, 316, 317, 328
- Learning Paradox, iii, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 17, 18, 22, 23, 24, 25, 26, 29, 30, 31, 44, 46, 47, 48, 51, 52, 58, 62, 65, 66, 71, 74, 75, 77, 78, 82, 86, 87, 88, 89, 90, 94, 97, 98, 99, 100, 101, 102, 103, 109, 113, 163, 164, 217, 223, 224, 225, 228, 230, 231, 232, 233, 234, 235, 239, 243, 247, 249, 250, 254, 257, 258, 261, 303, 327
- Less Than, xxiii, xxix, 25, 46, 51, 118, 160, 269, 270, 271, 273, 274, 275
- LM Learning, 344
- LP1–Fodor, viii, xv, xx, 7, 8, 11, 17, 25, 26, 27, 29, 30, 31, 32, 47, 51, 52, 62, 73, 74, 75, 77, 78, 81, 83, 84, 87, 89, 90, 93, 94, 98, 101, 102, 103, 104, 105, 112, 137, 163, 164, 189, 206, 217, 223, 229, 230, 232, 233, 234, 235, 239, 245, 247, 250, 256, 258, 259, 260, 261, 263, 327
- LP2–Emergence, viii, xv, 2, 28, 31, 32, 50, 52, 77, 78, 79, 83, 84, 85, 86, 89, 90, 93, 94, 98, 163, 217, 236, 237, 238, 247, 249, 250, 259, 327
- LP3–Autonomy, viii, xv, 24, 28, 31, 32, 48, 52, 74, 86, 89, 90, 95, 163, 217, 239, 243, 245, 247, 250, 327
- LP4–Real World Behavior, viii, xv, 4, 28, 31, 50, 52, 79, 83, 84, 86, 89, 90, 94, 163, 218, 239, 250, 259, 260, 263, 327
- LP5–Learns Incrementally, viii, xv, 3, 27, 28, 30, 32, 49, 52, 74, 79, 84, 89, 95, 97, 98, 163, 203, 218, 235, 239, 245, 247, 249, 258, 295, 327
- LP6–Develops Learning Process, viii, xv, xxii, 13, 29, 30, 31, 32, 42, 43, 48, 52, 79, 85, 86, 89, 90, 93, 94, 95, 98, 163, 218, 219, 235, 236, 247, 249, 259, 260, 261, 324, 327
- LP7–Acts In Novel Noisy Opportunistic Situations, viii, xv, xx, 7, 8, 11, 18, 26, 29, 30, 31, 32, 47, 48, 51, 52, 74, 75, 77, 78, 83, 87, 89, 90, 94, 98, 101, 102, 103, 104, 105, 112, 163, 164, 217, 219, 223, 229, 230, 232, 233, 234, 235, 239, 245, 247, 249, 250, 259, 260, 327
- M-Capacity, 204
- M-Demand, 204
- Mealy Machines, xxiv, xxvii, 130, 131, 163, 172, 190, 192, 194, 217, 220, 236, 237, 245, 246, 247, 313
- Mental Attentional Energy, 23, 24, 29, 226, 315
- Multiplication, 42, 51, 220, 222, 271, 273, 277, 311

Index

Mutation

Algorithm 9 thru 15, xxv, xxvi, xxvii, xxviii, 22, 127, 130, 144, 145, 148, 149, 150, 151, 152, 154, 155, 156, 157, 161, 178, 186, 189, 190, 191, 192, 193, 197, 202, 203, 204, 206, 208, 215, 217, 218, 219, 225, 230, 243, 289, 295, 323, 324, 326, 327

mutation selection

Algorithm 9 thru 15, 289

Nativism, 16, 22

Neural Network, iii, xx, xxiii, xxv, 6, 7, 8, 9, 30, 31, 52, 54, 55, 63, 81, 83, 85, 86, 88, 90, 91, 92, 93, 95, 96, 98, 106, 109, 164, 165, 166, 170, 171, 223, 232, 235, 236, 237, 243, 248, 249, 255, 259, 262, 266, 303, 304, 309, 319, 323

Not Machine, 132, 146, 147, 148, 162

Number Sense, 3, 48, 49, 58

Number Sense, 58

Number-Line, iii, xx, xxii, xxiii, xxv, xxvi, xxvii, xxviii, xxix, 4, 6, 7, 8, 10, 26, 31, 33, 44, 46, 47, 48, 49, 50, 51, 53, 54, 55, 56, 57, 62, 65, 69, 70, 71, 73, 78, 97, 100, 101, 102, 103, 104, 105, 106, 107, 110, 111, 112, 113, 115, 116, 117, 118, 121, 122, 124, 126, 133, 139, 141, 162, 163, 165, 166, 167, 170, 171, 180, 181, 182, 183, 184, 186, 188, 189, 190, 195, 205, 208, 209, 212, 214, 216, 219, 222, 224, 227, 228, 232, 235,

239, 246, 248, 250, 264, 266, 267, 271, 272, 276, 278, 280, 281, 282, 283, 300, 304, 306, 327

Operational Function, 36

Operative Scheme, viii, xxii, 27, 32, 33, 34, 35, 36, 37, 38, 41, 45, 47, 49, 50, 69, 97, 139, 162, 189, 209, 222, 238

Over-Fitting, 86

Pascual-Leone, 24, 204, 344

Perception, xxiii, 2, 8, 10, 31, 33, 35, 36, 37, 38, 41, 44, 49, 66, 75, 82, 94, 95, 97, 123, 141, 159, 162, 180, 239, 253, 254, 257, 259, 261, 262, 266, 272, 273, 277, 304, 308, 328

Permanent Object Invariant, iii, xxiii, xxvi, 9, 36, 57, 87, 91, 101, 102, 104, 106, 107, 113, 114, 130, 132, 133, 165, 171, 172, 177, 180, 182, 183, 186, 218, 219, 233, 323, 328

Phenotype, 324, 327, 328

Piaget, iii, iv, vii, ix, xxii, xxiv, xxvii, xxviii, xxxi, 1, 4, 5, 6, 7, 9, 10, 11, 12, 17, 18, 19, 20, 21, 22, 23, 26, 27, 29, 32, 33, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 56, 59, 61, 62, 63, 66, 69, 71, 73, 74, 76, 78, 82, 83, 84, 85, 87, 88, 91, 92, 93, 94, 95, 96, 97, 98, 99, 101, 102, 105, 107, 108, 113, 120, 122, 124, 126, 133, 138, 140, 141, 163, 164, 171, 188, 189, 190, 203, 204, 205, 206, 209,

Index

- 210, 216, 218, 219, 220, 222, 224, 225,
226, 228, 229, 230, 231, 232, 233, 234,
247, 264, 265, 268, 271, 272, 277, 279,
280, 294, 297, 305, 307, 308, 315, 316,
317, 321, 325, 328, 344
- Piagetian Scheme, xxiv, xxvii, xxviii,
xxix, 29, 89, 107, 112, 117, 125, 126,
129, 137, 138, 139, 141, 196, 206, 208,
210, 211, 212, 213, 214, 215, 216, 217,
218, 220, 221, 222, 226, 233
- Play, xxiii, 33, 41, 46, 71, 117, 124, 225,
226, 229, 230, 237
- Policy, 93, 109, 111, 258
- Predicate, 6, 17, 27, 58, 59, 60, 61, 62, 72,
73, 74, 77, 78, 84, 93, 163, 206, 233,
246, 254, 257, 258, 259, 324
- Prediction Machine, 118, 125, 133, 162,
163, 267
- Prediction Quality
- Algorithm 1, xxv, xxvi, xxvii, 127, 131,
132, 133, 145, 146, 147, 148, 149,
150, 152, 158, 159, 160, 172, 173,
174, 176, 177, 178, 191, 192, 193,
195, 325, 326
- Prediction Scheme, xiv, xxi, xxv, xxvi,
xxvii, xxviii, xxix, xxx, 125, 130, 131,
133, 134, 141, 142, 146, 147, 158, 160,
161, 163, 172, 173, 174, 175, 176, 177,
178, 179, 180, 181, 182, 183, 184, 185,
186, 188, 189, 190, 191, 192, 193, 194,
195, 196, 197, 198, 201, 202, 203, 204,
206, 209, 213, 214, 215, 216, 217, 218,
219, 221, 222, 282, 283, 287, 289, 297,
300, 323, 325, 328
- Predictive Model, xxii, xxv, xxx, 37, 40,
41, 110, 111, 119, 124, 125, 126, 137,
141, 168, 169, 212, 217, 218, 225, 226,
227, 229, 230, 239, 240, 249, 265, 268,
272, 273, 276, 277, 278, 279, 298, 299,
300
- Pre-Operational Stage, 49
- Primary Reaction, xx, xxii, 19, 34, 35, 36,
37, 47, 48, 51, 63, 78, 104, 105, 162,
171, 217, 248, 325
- Problem Solving, x, xi, xii, xiv, xvi, xvii,
xviii, 61, 73, 75, 77, 83, 84, 85, 86, 95,
96, 100, 187, 236, 237, 245, 259, 260,
261, 262, 306, 309
- Procedural Knowledge, 88
- Propositional, xxvii, 6, 16, 17, 31, 107,
112, 163, 191, 193, 194, 206, 219, 222,
233, 245, 250, 257, 259, 262, 324
- Random selection
- Algorithm 13, 153, 157, 328
- Rank Based selection
- Algorithm 12, 155, 156
- Rational Numbers, 14, 73, 76, 230, 265,
271, 278, 279, 303, 310, 317
- Recognition, xxiii, 36, 37, 38, 72, 75, 86,
95, 97, 123, 159, 162, 236, 262, 304,
323

Index

- Reconstruction, viii, xiii, xxiii, 36, 38, 46, 114, 119, 123, 124, 144, 159, 160, 161, 162, 249, 267, 320, 326, 328
Algorithm 19, xxiii, 36, 38, 46, 114, 119, 123, 124, 144, 159, 160, 161, 162, 249, 267, 320, 326, 328
- Reflective Abstraction, ix, xxii, 14, 24, 25, 26, 27, 32, 36, 41, 42, 44, 51, 63, 88, 92, 95, 96, 113, 126, 133, 218, 233, 236, 279, 281, 305, 316
- Reinforce, xx, xxii, xxiii, xxiv, xxvi, 37, 118, 123, 125, 141, 142, 143, 144, 159, 161, 177, 211, 212, 265, 267, 325
- Reinforcement Learning
Actor Critic, 242
TD Learning, xxi, 110, 217, 225, 228, 241, 242
- Representation, xx, xxv, xxvi, 18, 19, 23, 25, 26, 38, 45, 46, 51, 54, 56, 59, 68, 70, 75, 77, 84, 85, 91, 109, 111, 130, 139, 141, 165, 173, 183, 184, 211, 227, 248, 255, 256, 258, 261, 273, 280, 303, 304, 315
- Requirements, i, xxvi, 21, 100, 146, 147, 164, 165, 172, 174, 251, 260, 261, 325, 327, 328
- Reward, xxi, xxii, xxv, 28, 31, 34, 93, 95, 109, 110, 111, 112, 113, 117, 126, 130, 133, 137, 139, 150, 166, 167, 168, 169, 170, 171, 199, 209, 211, 212, 215, 225, 239, 240, 241, 242, 243, 246, 250, 252, 258, 276, 278, 300, 324
- Scheme, viii, xiv, xv, xix, xx, xxi, xxii, xxiv, xxv, xxvi, xxvii, xxviii, xxix, 14, 20, 21, 25, 27, 33, 34, 35, 36, 37, 40, 41, 44, 46, 48, 49, 69, 80, 88, 110, 112, 113, 114, 117, 119, 120, 122, 124, 125, 126, 127, 130, 132, 133, 135, 136, 137, 138, 139, 142, 145, 152, 155, 158, 159, 160, 161, 172, 173, 174, 176, 177, 183, 184, 189, 190, 192, 195, 196, 197, 198, 199, 200, 202, 203, 204, 205, 206, 207, 208, 210, 211, 212, 213, 214, 215, 216, 217, 219, 221, 226, 243, 247, 258, 262, 274, 275, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 298, 300, 321, 328, 344
- Selection
Algorithm 9 thru 15, xxiv, 12, 20, 21, 22, 81, 82, 87, 105, 110, 112, 131, 135, 149, 152, 153, 154, 155, 156, 157, 158, 173, 174, 183, 243, 245, 246, 252, 256, 257, 262, 328, 344
- Semantics, 78, 80, 82, 83, 246, 247, 256
- Semiotic Function, 36, 38, 45, 73, 91
- Sense
Algorithm 17, iii, xxiv, xxvi, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 18, 25, 26, 29, 32, 37, 42, 43, 45, 48, 51, 52, 53, 54, 55, 56, 57, 61, 64, 67, 70, 74, 76, 79, 81, 83, 84, 85, 98, 99, 101, 102, 103,

Index

- 105, 107, 108, 122, 123, 137, 141, 143, 159, 161, 163, 165, 170, 174, 175, 180, 181, 182, 184, 188, 190, 191, 194, 196, 205, 206, 216, 218, 219, 220, 222, 224, 226, 227, 229, 230, 231, 232, 233, 234, 244, 247, 248, 253, 256, 258, 261, 264, 268, 271, 280, 303, 306, 307, 308, 309, 312, 313, 320, 323, 325, 328
- Sensory-Motor Stage, 205
- Simulation, iii, xx, xxii, xxvi, xxvii, xxviii, xxx, 2, 5, 6, 8, 9, 10, 11, 23, 25, 26, 28, 29, 30, 31, 46, 47, 48, 49, 50, 51, 52, 54, 57, 75, 78, 85, 91, 99, 100, 101, 102, 103, 109, 112, 113, 114, 117, 119, 120, 126, 127, 133, 162, 163, 172, 174, 176, 177, 178, 179, 180, 181, 182, 183, 185, 187, 188, 189, 190, 193, 194, 196, 197, 203, 204, 205, 206, 208, 209, 210, 215, 216, 217, 218, 219, 222, 224, 225, 226, 227, 229, 231, 232, 233, 235, 236, 238, 241, 243, 245, 246, 247, 249, 250, 255, 265, 281, 283, 296, 304, 310, 327, 328
- SOAR, x, xi, xvii, xxix, 24, 76, 77, 80, 81, 84, 86, 89, 90, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260, 308, 311, 322
- State Value Function, xxv, 93, 109, 111, 166, 167
- Structural Function, 36
- Student, xx, xxiii, xxix, 4, 14, 29, 34, 47, 48, 104, 105, 107, 112, 116, 117, 119, 120, 121, 122, 125, 126, 139, 141, 162, 177, 180, 194, 211, 212, 219, 222, 227, 264, 265, 266, 267, 272, 276, 278, 279, 280, 281, 300, 328
- Subtraction, 3, 46, 53, 54, 205, 273, 277, 279, 285
- Full-Subtractor, xix, xxi, xxix, 205, 282, 283, 284, 285, 286, 287
- Symbol, 3, 5, 6, 12, 18, 19, 28, 32, 33, 34, 36, 38, 45, 46, 52, 54, 56, 57, 63, 71, 73, 74, 76, 77, 78, 79, 80, 81, 82, 83, 84, 86, 87, 88, 89, 90, 91, 92, 93, 95, 113, 114, 148, 171, 172, 205, 217, 224, 237, 244, 245, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260, 262, 280, 304, 320, 324, 328
- Symbol Grounding Problem, 6, 28, 77, 79, 81, 82, 84, 86, 87, 88, 92, 113, 217, 224, 255, 257, 258, 259, 320
- Symbolic Function, 33, 36, 38, 45, 46, 84, 91, 172, 205, 244
- Teacher, xiii, 14, 19, 95, 105, 107, 117, 120, 121, 122, 126, 139, 141, 162, 177, 180, 187, 194, 211, 212, 222, 271, 276, 300, 309, 314, 317, 320, 325
- Theory Of Constructive Operators, 22, 23, 24, 25, 26, 90, 125, 265, 328
- Tournament Selection Algorithm 14, 157, 158, 328

Index

- WE1, ix, xxii, xxv, xxvi, 7, 8, 32, 46, 47, 52, 74, 75, 98, 102, 103, 105, 107, 113, 162, 164, 166, 170, 171, 172, 180, 181, 182, 187, 190, 194, 205, 206, 216, 222, 223, 227, 230, 232, 233, 234, 329
- WE2, ix, xxii, xxvi, 48, 52, 113, 171, 181, 182, 190, 222, 329
- WE3, ix, xxii, xxv, xxvii, 47, 49, 52, 113, 166, 171, 187, 188, 190, 222, 329
- WE4, ix, xxii, 49, 50, 52, 171, 187, 206, 216, 217, 329
- WE5, ix, xxii, 7, 8, 32, 46, 47, 50, 52, 74, 75, 98, 102, 103, 105, 107, 162, 164, 170, 172, 180, 182, 187, 190, 194, 205, 206, 216, 222, 227, 230, 231, 232, 233, 234, 264, 274, 276, 280, 299, 329
- WE6, xix, xxix, 227, 264, 280, 329
- Worked Example, iii, xxv, xxviii, xxix, xxx, 7, 8, 10, 32, 35, 46, 47, 51, 52, 62, 73, 74, 97, 98, 99, 102, 105, 107, 108, 113, 161, 164, 166, 170, 172, 180, 181, 182, 187, 190, 194, 205, 206, 208, 212, 216, 217, 222, 227, 230, 231, 232, 233, 234, 264, 274, 276, 280, 299, 329
- Worksheet, xxiii, xxix, 105, 115, 117, 121, 162, 170, 283

End Notes

End Notes

ⁱ Various theories are discussed in the literature review, these include: Generative Linguistics (Chomsky, 1980); Fixation of Belief (Fodor, 1980); Genetic Epistemology (Furth, 1969; Copeland, 1974; Piaget, 1954; Piaget, 1964); Theory of Constructive Operators (Pascual-Leone, Goodman, Ammon and Subelman, 1978); Constructivist Theories (Liebeck, 1984; Furth, 1969; Dehaene, Izard, Spelke and Pica, 2008); Thalamic loops (Albus, 2008; Albus, 2010b, Granger, 2006a; Granger 2006b); Event Hierarchies (Albus, 2008; Albus, 2010a and 2010b); Receptive Field Hierarchies (Albus, 2008; Albus, 2010b); Herbrand's theorem as used in automated theory formation and proving systems (Chang and Lee, 1973; Pease, Colton. and Chamley, 2012; in Colton and Muggleton, 2006; Colton, 2002; Colton, 2000); Theory of Metaphors (Lakoff and Núñez, 2001); Conceptual Blending Theory (Fauconnier, 1985; Fauconnier and Turner, 2002; Hutchings, 2005;Guhe, Pease, Smaill, Martinez, Schmidt, Gust, Kühnberger and Krumnack, 2011); Theory Tethering (Sloman, 1985); Adaptive Resonance Theory (ART) (Carpenter and Grossberg, 2002) ; ACT-R theory (Anderson, 2007; Anderson, Bothell, Byrne, Douglass, Lebiere, Qin, 2004); Scheme Theory (Von Glasersfeld, 1998); Intelligent Agent Theory (Wooldridge and Jennings, 1995); Dual-Process Theory (Pyysiäinen, 2003; Epstein and Pacini, 1999) and Theory of Emergence (Crutchfield, 1994a, 1984b) amongst others.

ⁱⁱ For Bereiter, the defined factors are: chance plus selection; piggy-backing; affective boosting of schemas (attention is regulated by effect; attention is regulated by field effects; attention is regulated by bias); emotion as a separate information processing system; field facilitation; imitation; learning support systems (scaffolding, field facilitation in LM Learning, LC learning, See § 2.6), use of spare mental capacity and the need for coherent self-concept (Bereiter, 1985, p208).

ⁱⁱⁱ Mental spaces are described as an aspect of cognitive linguistics that develop constructs during discourse. Distinct from linguistic constructs, they are established by linguistic expressions (Fauconnier, 1985), and provide explanation of problems in philosophy and cognitive science concerning thought and language. Mental spaces consist of elements and the relations between them (Fauconnier & Turner 2002,p102). These elements, potentially from multiple domains located in both short and long term memory are related in working memory and when "organized as a package that we already know about, we say that the mental space is framed" (Fauconnier and Turner 2002, p103).

^{iv} In simplexes, one input consists of a frame and the other consists of specific elements. A frame is a conventional and schematic organization of knowledge such as “buying gasoline” (Fauconnier & Turner 2002, p120). In mirrors, a common organizing frame is shared by all spaces in the network (Fauconnier & Turner 2002, p122). In single-scope, the organizing frames of the inputs are different, and the blend inherits only one of those frames (Fauconnier & Turner 2002, p126). In double-scopes, essential frame and identity properties are brought in from both inputs (Fauconnier & Turner 2002, p131). It is double-scope blending which can resolve clashes between inputs that differ fundamentally in content and topology and is the powerful source of human creativity (Fauconnier & Turner 2002, p139).

^v In composition, the projections from the inputs make new relations available that did not exist in the separate inputs (Fauconnier & Turner 2002, p146; Fauconnier, 1997, p150; Hutchins, 2005, p1556). The mapping schemes, by contrast, are “predictable from the language forms used to evoke them” (Fauconnier & Turner 2002, p147). In completion, knowledge of background frames, cognitive and cultural models, allows the composite structure projected into the blend from the inputs to be viewed as part of the larger self-constrained structure in the blend. The patterns in the blend triggered by the inherited structures is “completed” into the larger, emergent structure (Fauconnier & Turner 2002, p44; Fauconnier, 1997, p151; Hutchins, 2005, p1556). In elaboration (also called running the blend), cognitive work is performed within the blend according to its own emergent logic (Fauconnier & Turner 2002, p44; Fauconnier, 1997, p151; Hutchins, 2005, p1556).

^{vi} Pascual-Leone, Goodman, Ammon and Subelman’s abstract and technical theory of Constructive operators seeks to “represent explicitly the underlying mechanism of dialectical equilibration and structural growth” (Pascual-Leone, Goodman, Ammon and Subelman, 1978, p252). The theory of constructive operators integrates three sorts of organismic constructs: schemes, basic factors and basic principles within a process model of the psychological organism, the meta-subject. It is called the meta-subject, to indicate that it is a “highly active hidden organization which is causally responsible for the subject’s performance.” Refined over several years, the theory of constructive operators (TCO) (Pascual-Leone and Goodman, 1979, p303) is based on a series of principles: Principle of assimilatory praxis as well as the associated principle of schematic over-determination of performance (SOP), Principle of equilibration and the Principle of bi-level psychological organization. These principles need to be combined with several other elements of Pascual-Leone’s work, including: Executive schemes, Interrupt function, M Capacity, Principle of scheme inhibition and decay, Schemes, Silent operators. Some aspects of Pascual-Leone’s work had to be set aside; this was due to restrictions on development time, though in any future research, their utility and warranty may be proven. These deprecated features include: Affective schemes, personality biases and belief systems, Sensorial field factor, Content Learning, LM Learning, LC Learning.

^{vii} The Cartesian product operator is implemented as parallel composition of independent systems, where each system is a FSA. When the FSA communicate with each other, it is through encapsulation with dedicated signals. The encapsulation allows for refinement and the parallel composition is always symmetric (Maraninchi and Remond, 2001).

^{viii} The refinement operator otherwise called the encapsulation operator, is a unary operator that restricts the scope of signals i.e., it makes them local to the encapsulation for both parallel and hierarchical consumption. The refinement operator is asymmetric, since one of the FSA is the controller and the other(s) FSA are controlled. The refinement operator allows for interruption, exceptions and terminations of programs (Maraninchi and Remond, 2001).

^{ix} The inhibiting operator can be preemptive and prevent a parameter to a FSA from being present. The inhibiting operator also allows the controller to “kill” the controlled FSA (Maraninchi and Remond, 2001)

^x Temporized states is the capability to attach “delays” to states within the macro-notation of “multiform time.” This “multiform time” allows an external input event to be used a clock for the system, “which may count meters as well as seconds” (Maraninchi and Remond, 2001, p77).

^{xi} Incorrect compositions or causally incorrect compositions of components occur when “an encapsulation operator is applied to a reactive or deterministic component and yields a system which is either non-deterministic or non-reactive” (Maraninchi and Remond, 2001, p79). This breaks the basis of reactive systems.