COMPUTER AIDED RELIABILITY PREDICTION

This thesis is presented to the Council for National Academic Awards for the degree of Doctor of Philosphy

by

Christopher David Partridge B.Sc.

School of Electrical and Electronic Engineering
Kingston Polytechnic

January 1976

ACKNOWLEDGEMENTS

The work described in this thesis was carried out under the supervision of Dr. A. A. Kaposi of Kingston Polytechnic and Mr. N. A. Walter and Mr. E. Robinson of the Admiralty Surface Weapons Establishment.

The author wishes to acknowledge the contributions of the personnel of the Reliability Group of A.S.W.E. and their sub contractors, particularly Rex Thompson and Partners, and Plesscy Assessment Laboratory.

Grateful thanks are due to the staff of the Computer Unit at Kingston Polytechnic.

Most of all the author wishes to thank Dr. A. A. Kaposi for her considerable effort during the whole period of the project.

Contents

Acknowledgements

Abstract

1. Reliability Analysis

1.1. Background to Reliability Prediction 1.1

1.2. Mathematical Methods for obtaining Reliability Performance 1.1

2. Reliability Modelling

2.1. The Reliability Block Diagram 2.1

2.2. Dependency 2.2

2.2.1. The Concept of Dependency 2.2

2.2.2. Implementation of Dependency 2.2

2.2.3. The Switch 2.4

2.2.4. Standby 2.5

2.2.5. Duplicate Elements 2.6

2.2.6. Time Operated Switch 2.7

2.3. N out of M Redundancy 2.8

2.4. The Concept of Nesting 2.9

2.5. A Complex Modelling Example 2.10

2.6. Modelling Experience 2.11

2.7. Modelling for Spares Estimation 2.12

3. Systems Aspects of the Prediction Program RELY

3.1. The Program and the 1900 Computer System 3.1

3.2. Use of Program 3.6

4. The Prediction Program RELY

4.1. Objectives of Program 4.1

4.2. The Structural Components of the Program 4.3

4.2.1. Data Input Modules 4.4

4.2.2. Route Searching and Production of Status Table 4.10

4.2.3. Generation of Probability Tables 4.15

4.2.4. Time Simulation 4.20

4.2.5. Statistics of Simulation 4.28

4.2.6. Nesting 4.32

4.2.7. Spares Estimation 4.35

4.2.8. Output 4.44

4.2.9. Housekeeping 4.47

4.2.10 Control 4.52

5. Reliability of Mechanical Engineering Systems

5.1. Introduction 5.1

5.2. Modelling Mechanical Engineering Systems 5.1

5.3. Conclusions 5.3

6. Documentation of Program

6.1. The User Level 6.1

6.2. The Systems Level 6.4

6.3. The Program Level 6.4

7. Design Examples

7.1. A Shipborne Communications System 7.1

7.2. A Private Hospital 7.7

7.3. A Control System for a Disabled Driver's Car 7.17

8. Interaction with Users

8.1. The Role of the User 8.1

8.2. Comments of Users 8.2

8.2.1. Facilities Requested by Users 8.2

8.2.2. Errors Reported by Users 8.5

9. Furthur Developments

10. Other Related Projects

10.1. Internal Projects 10.1

10.2. External Projects 10.1

11. Bibliography

Appendix i: Conferences and Publications

Appendix ii: List of symbols and abbreviations

Appendix iii: Definitions

ABSTRACT

This thesis describes a project, sponsored by the Admiralty Surface Weapons Establishment (A.S.W.E.), whose objective is to investigate the use of Computer-Aided Design (C.A.D.) methods in reliability engineering and, in particular, in reliability prediction.

The project evolved as a result of continuous interaction with users whose requirements and comments have assisted in the definition of the project specification which, in turn, implied the method of computation (Monte Carlo analysis) and the form of implementation (a modularly structured program).

The project produced a CAD method which aimed to provide:

i) a means of predicting the reliability of complex heterogeneous systems and an aid to estimate their spares requirements in an efficient way.

ii) software which is easily extendable, modifiable and, while oriented towards the ICL 1900 range of computers, optimally portable.

iii) a mode of documentation which permits the use of the program by reliability engineers who have no previous computing experience.

In order to fulfil these requirements it was necessary to incorporate a number of novel features which include:

i) the use of hierarchical structures as a means of modelling the reliability of large and complex systems.

ii) the introduction of a modelling device in the form of a controlled switch which allows the modelling of a wide range of dependent failure and repair mechanisms.

iii) the transformation of any type of failure and repair distribution into a uniform data structure which permits the easy and efficient handling of any type of distribution function.

iv) the use of modular programming and program documentation as a means of providing the necessary efficiency, flexibility and user-accessability.

This thesis includes the description of the CAD method and illustrates it by means of a number of examples. Further, it discusses some of the problems of using this method to predict the reliability of mechanical engineering systems.

The use of the program by A.S.W.E. contractors and Polytechnic students is described by reference to diverse design examples. Further areas of research and development in relation to the project are given.

To assist the reader who may not be equally familiar with the standard terminologies of reliability engineering, statistics and computing used in this thesis, a set of selected definitions is included in one of the appendices.

1. RELIABILITY ANALYSIS

1.1 BACKGROUND TO RELIABILITY PREDICTION

Reliability studies as a separate discipline rose to prominence during the second world war. Until then the application of mathematical techniques to improve the reliability of systems was restricted to simple cases of one or two elements.

With the ever increasing complexity of technological systems since the war, a means of estimating the reliability of systems became important. As the number of elements in a system increased, overdesign for reliability became an unacceptable solution due to the prohibitive cost. Performing life cycle tests with items of equipment in order to compile data proves impossible with the size of systems being designed at the present time. For these reasons reliability prediction has become important as an area of study. As part of the process of design, it is necessary to estimate the reliability performance of physical systems by means of calculation, thus ensuring that a new product conforms to such requirements as minimum availability or mean time between failures.

1.2 MATHEMATICAL METHODS FOR OBTAINING RELIABILITY PERFORMANCE

There are at present two main classes of methods for obtaining the reliability performance of a system.

i) Analytical methods.
ii) Simulation methods.

i) Analytical methods aim to provide an explicit expression for the reliability behaviour of a system. These methods are based on the assumption that the system may be partitioned into a number of units (elements) each of which may exist in one of two reliability states, namely, functioning or failed. The reliability model of a system of n elements may be described by a Boolean state transition diagram with 2^n states, each of which represents a unique combination of element reliability states. These 2^n system states may then be partitioned into two mutually exclusive subsets representing the functioning and failed state of the system, respectively. If now the probability of transition from one state to another is known for each element for all values of $t(0 \leqslant t \leqslant t_M)$, where t is time and t_M is the 'mission time', one may attempt to obtain the system reliability behaviour as a solution of a set of simultaneous differential equations whose parameters are the transition probabilities of elements.

The most usual analytical method regards failure and repair mechanisms as a Markov process whose full description is given, among others, in reference 5.

In many cases a Markov model may only be devised after radical simplifications of the real-life behaviour of a system under analysis. In any case, at the start of this project and at the time of writing, the method was restricted to systems of modest size and complexity which are devoid of sophisticated reliability problems such as dependency. References 6 and 24 describe early methods which aimed to overcome the problem of size. The most promising method concerned truncation of the transition probability matrix with the resultant drawback of loss of accuracy. Even at the present state of the art (references 18, 33) modelling of dependent elements is usually not possible unless a very simple mechanism operates. Quite sophisticated list processing and pattern recognition techniques (reference 18) will still only allow systems of moderate size (20 elements) to be analysed.

ii) Simulation methods provide an estimate of the system reliability based on statistical experiments with the reliability model of the system. For each experiment, random samples are taken from the probability distributions characterising the failure and repair mechanisms of the elements. One technique that employs this process is known as Monte Carlo analysis which is a method involving repeated statistical sampling in order to obtain an estimate of the solution of a stochastic problem. Mihram (Ref. 36) regards this method as a branch of experimental mathematics and verifies its validity by means of the Law of Large Numbers. The statistical foundations of the method are widely discussed in the literature (see for instance, references 37 and 38), and one of its uses in the context of reliability assessment is described in reference 10. It is this technique which forms the basis for the method described in this thesis.

A comparison of these two main methods at the start of the project (1972) indicated that the Monte Carlo method would provide a more promising approach for the class of system being considered, namely, large complex interdependent structures. In the course of this work it was found that the Monte Carlo method was able to cater for the practical problems encountered by reliability engineers while, even at the present state of the art, Markov methods would not be able to handle the complexities of the systems shown as design examples in this thesis, chapter 7.

2. RELIABILITY MODELLING

2.1 THE RELIABILITY BLOCK DIAGRAM

The type of structure in reliability analysis most easily understood by designers is the reliability block diagram which is a contact network of branches connected between nodes.

Each element of the system is represented by a branch which contains a contact. The contact can be in one of two states; made or broken, representing functioning or failure of the element, respectively.

The prime purpose of the reliability block diagram is to model the physical system as a number of routes or paths between two specified nodes. The system is said to be functioning if sufficient routes are available (usually one) between these two nodes. Failure of an element in a route breaks the contact and disables that route. Figure 2.1 shows the reliability block diagram of a four-element system containing three parallel redundant elements. The system will be said to function if at least one route is maintained between nodes 1 and 3.

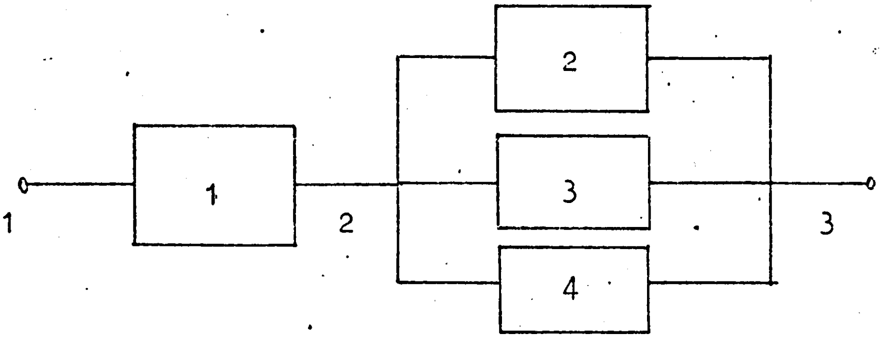


figure 2.1

Should one of the parallel elements fail, two of the three routes through the system would still be available. However, if the series element '1' failed then all three routes would be disabled and the system would cease to function.

Modelling systems reliability in terms of contact networks is a well known procedure. An alternative is the use of logic gate models (reference 13) While both methods, in their normal form, are limited, the contact network model offered more scope for extension of modelling facilities to allow for additional complexities such as shelf life standby, etc. A third alternative method of modelling is the failure or fault tree. An example of this type of model is shown in section 2.5. Reference to this example shows that this is a dual of the contact network model.

2.2 DEPENDENCY

2.2.1 THE CONCEPT OF DEPENDENCY

The failure and repair of an element is governed by one of two basic mechanisms.

i) An element can change state independently of any other element or event. This means that the element fails and repairs according to some random process characterised by the distributions apertaining to that element.

ii) The operation of an element may be governed by events occuring to another element elsewhere in the reliability structure. Alternatively, an element may change state at certain pre-determined times as demanded by the environment in which the system operates. In both these cases the failure and/or repair of the element is not an independent random event. To cope with such situations the concept of dependency has been introduced. By using this concept it is possible to model a variety of practical situations such as shelf life, standby redundancy, scheduled maintenance, etc.

2.2.2 IMPLEMENTATION OF DEPENDENCY

Let us assume that a system contains, among others, two elements, A and B. A is an independent element whose failure and repair are random events whose occurrence is characterised by specific distributions. The repair of B is similarly independent but B only starts to operate after A has failed. Thus the failure distribution of B does not give a full account of the failure mechanism of the element since failure also depends upon events occuring in element A.

Figure 2.2 shows a sequence of random events occuring in A and B. The manner in which the program handles the situation is as follows:

At the start of simulation the time of the first event is generated for each branch. Let these be t1A and t'1B for element A and B, respectively. t1A is an independent event which is recorded by the program. t'1B is examined with respect to the state of A. Since at t'1B A was working, this failure time is not taken into account until later. Element B fails at t1B = t.1A + t'1B. After this, both elements are under repair. B becomes available first and starts service; hence the subsequent failure time t3B is recorded, although A becomes available in the meantime. Events at t3A and t4A are independent. t'5B is once again suspended because A is functioning. The next failure of B is computed as t5B = t5A + t'5B.

In general, failure and/or repair may be a dependent mechanism. At the time of sampling the appropriate distribution of the dependent branch a check is made to see if the independent branch is operating in the same mode. If so, the time of the next dependent event is adjusted according to the next event in the independent branch; otherwise it is treated in the same way as for an independent event.

Inverse dependency is an identical mechanism except that:

i) The initial state of the dependent element is inverted i.e. fail to repair and vice versa.

ii) The next event time of the dependent branch is modified if its mode (failed or working) is opposite to that of the branch on which it is dependent.

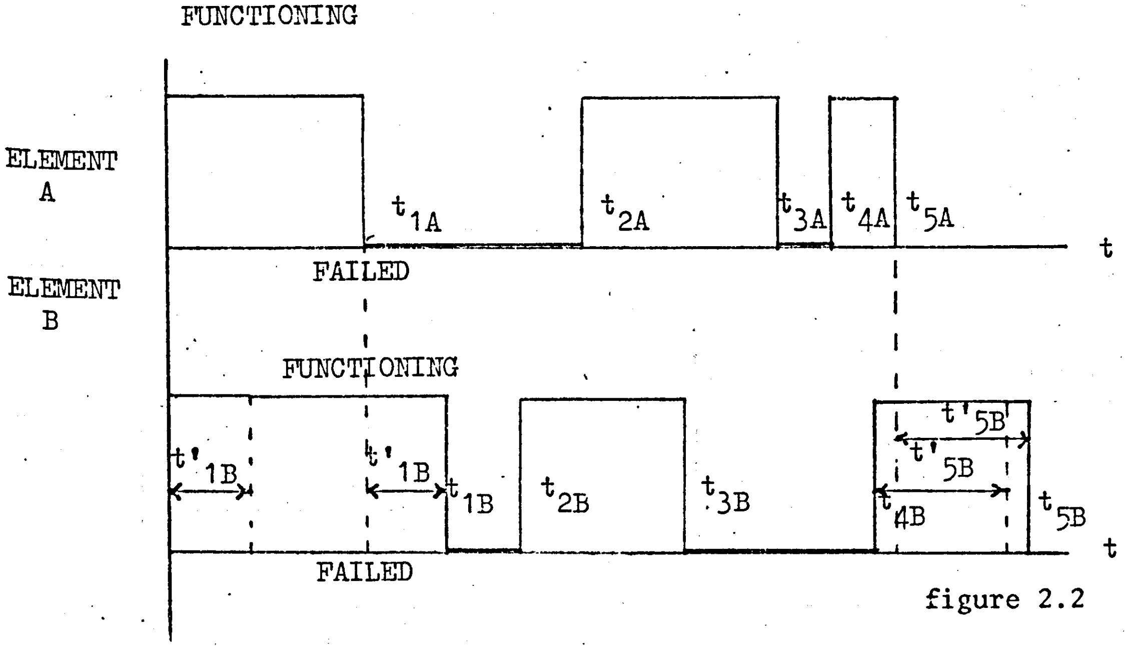


figure 2.2

2.2.3 THE SWITCH

In order that more complex situations may be modelled an additional element has been invented. This element, called the switch, has a discontinuous cumulative distribution function characterised by figure 2.2A. The time t' at which the probability of the event changes from 0 to 1 is known as the switch time. By making the operation of the switch dependent on some independent event or events it is possible to use this auxiliary modelling device in a variety of apparently different practical situations. In the light of experience it can be supposed that the switch may be a universal modelling tool. This means that new modelling problems may be solved with the judicious use of the switch.

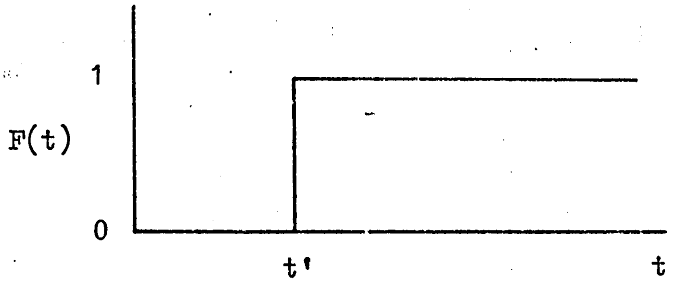


figure 2.2A

The operation of a dependent switch is shown in figure 2.3. The 'switch time' of the switch can be set so short that, compared with the timing of any other event in the system, the switch appears to act instantaneously. The user sets this time as desired.

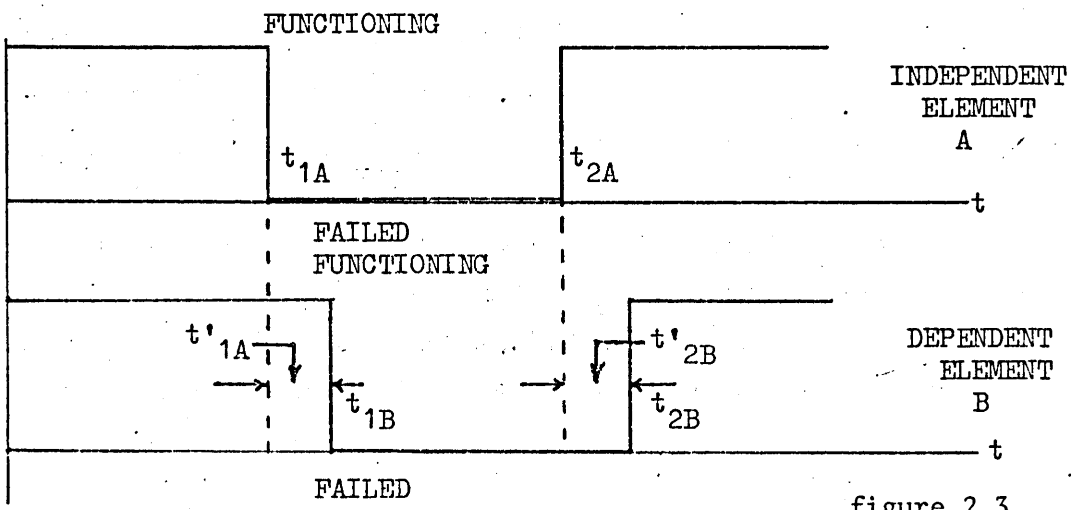


figure 2.3

2.2.4 STANDBY

In the classic standby situation an element is in service only when the corresponding independent element fails. Assuming the standby element has infinite shelf life then it can be modelled in two ways. In the example of figure 2.4A, X is an element whose operation is dependent on M. X will not come into service initially until M fails for the first time.

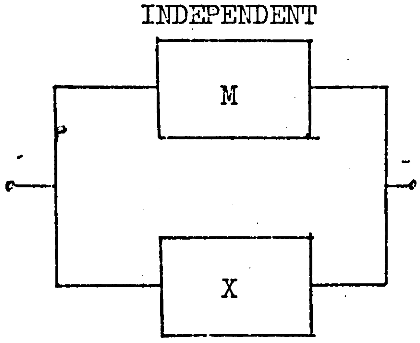


figure 2.4A

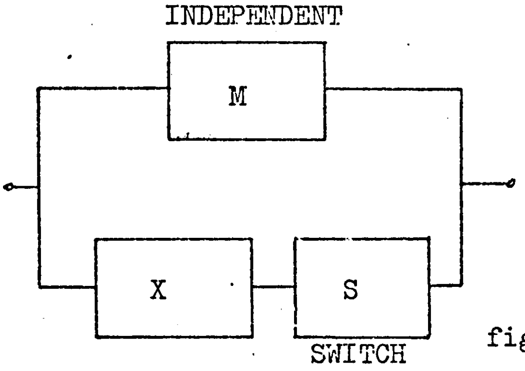


figure 2 .4B

Assuming normal repair functions on both X and M, X will always be on standby unless M has actually failed. The disadvantage of this model is that, following a failure of M, X remains operational even after repair of M as can be seen from the explanation of the dependency concept. This will not affect the performance of some systems but in other cases it may cause misleading or ambiguous results, i.e. when more than one route is required for the system to operate. Hence, in general, the alternative model shown in figure 2.4B is recommended. An additional series element is inserted in the standby path. This is an inverse dependent switch which fails when M repairs and repairs when M fails. Thus the standby path is not available while M is functioning. An additional constraint can be imposed by making the failure or repair of M dependent on the failure or repair of X. This has the effect of not bringing M back into service until X has failed.

The standby models so far described all have the property that, for the duration of the switch time, there are no working routes available in this part of the system. The switch time may be chosen by the user to represent the period of failure while the standby is brought into service. If the standby is in a series branch, for this period the system will be in a failed state.

In some systems the changeover to standby is accomplished practically instanteously and the functioning of the system is not disrupted for any appreciable time. Since the models of figure 2.4A and figure 2.4B record a system failure for the duration of the switch time, all the output parameters dependent on the number of systems failures (M.T.B.F. M.T.F.F. etc) will be distorted, as in many practical systems these failures are ignored. An alternative model is shown in figure 2.4C which correctly represents such a situation.

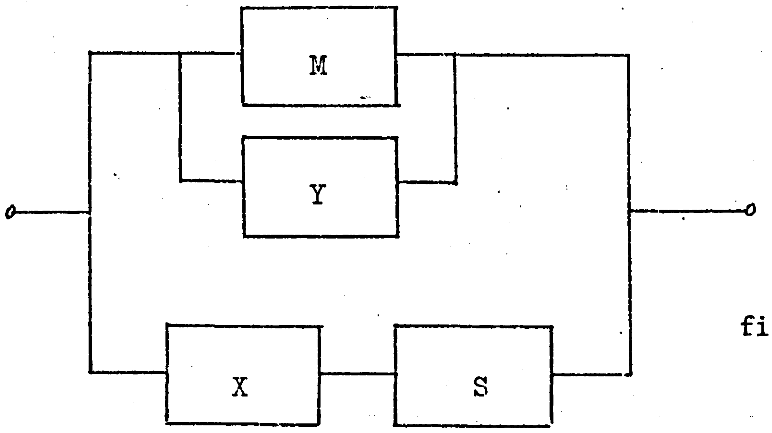


figure 2.4C

Y is an additional switch element which fails when M fails and repairs when M repairs. The switch time of Y though is made longer than that of S to cover the transition that causes the system failure. Figure 2.5 illustrates the model.

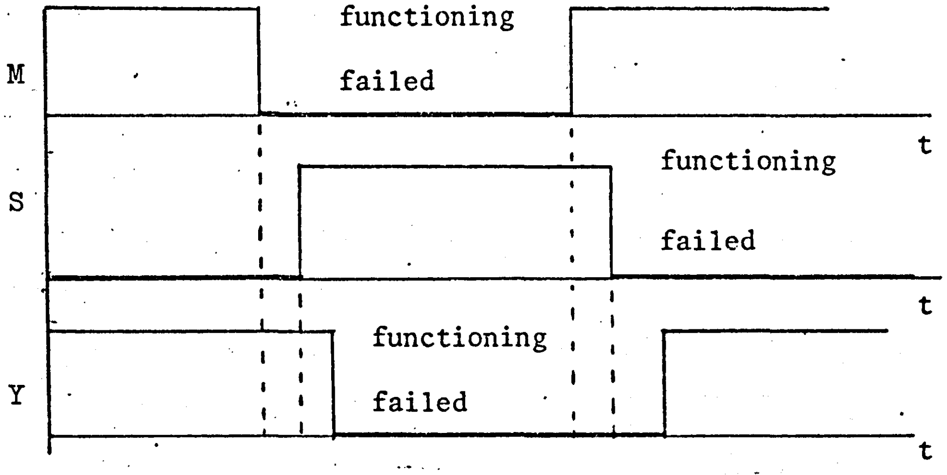


figure 2.5

2.2.5 DUPLICATE ELEMENTS

It is sometime necessary to include the same physical element a number of times in the same reliability structure. Such a situation may arise in the course of modelling a complex system where it may be quite difficult to ensure that some element forms part of all the correct routes. By using dependent switches in the structure it is possible to achieve the required model if the switches (with a small switch time) are made dependent on an element which has the distributions of the required single element.

2.2.6 TIME OPERATED SWITCH

Some systems require a model that is able to cater for predetermined or periodic switching of equipment. For example, two receiver channels might be available in a system, one being the in-service channel, the other the out-of-service channel. Every n hours the roles of these channels would be switched round. By using the switch as a free-running clock it is possible to model this. Figure2.5A shows a possible block diagram for modelling the two receiver channel system.

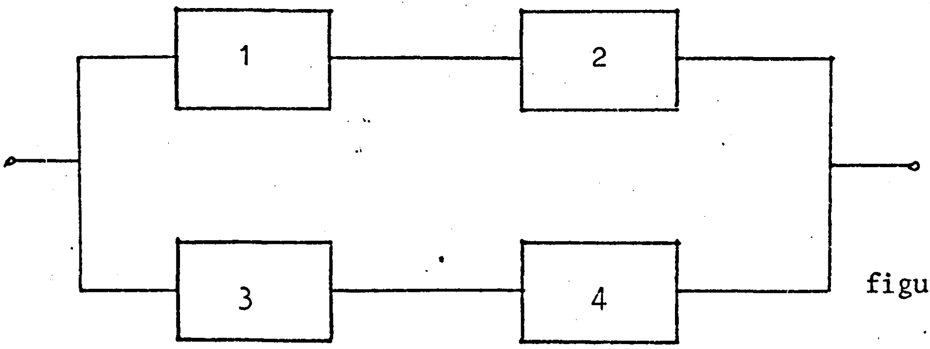


figure 2.5A

Element 3 is a switch that opens at a certain time t
Element 1 is a switch that closes at a certain time t

Hence the effect is to switch in element 2 at t, switching out element 4 at the same time.

A multiple switch shown in figure 2.6A could be modelled as in figure 2.6B

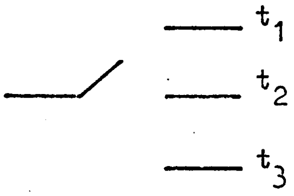


figure 2.6A

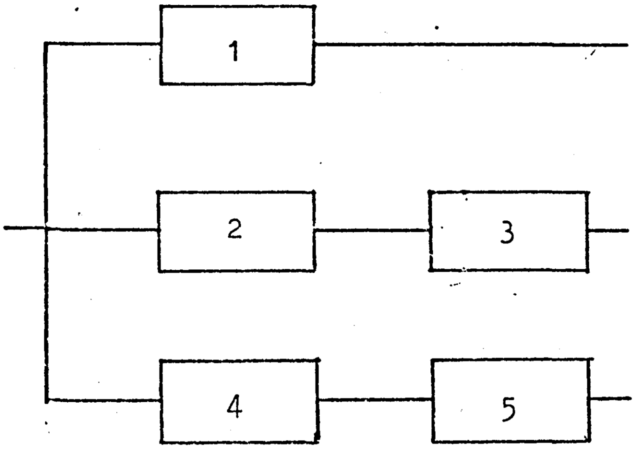


figure 2.6B

A time sequence governing the operation of the model could be represented as in figure 2.7.

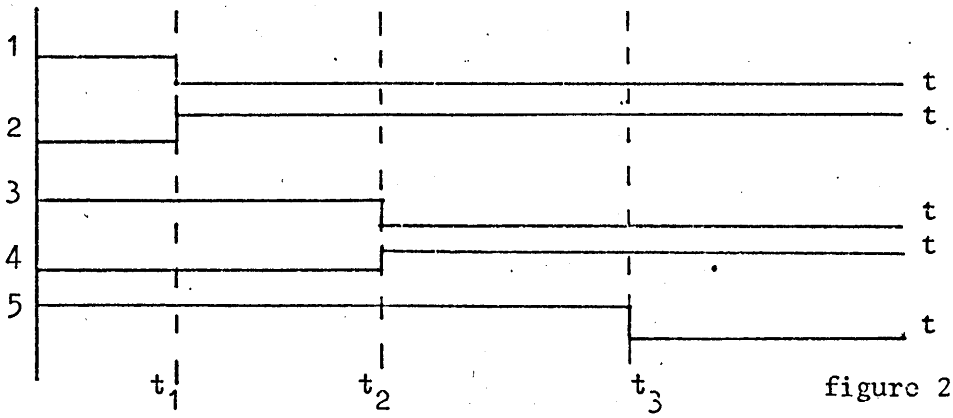


figure 2.7

t_1, t_2, t_3 represent the times at which the switch changes. Element 1 fails at t_1, element 2 being repaired at the same time. Element 3 fails at t_2, element 4 being repaired at the same time. Element 5 fails at t_3. A cyclic switch could be made by having dependent failure and repair with an initial independent switch.

2.3 N OUT OF M REDUNDANCY

This is a special case of parallel redundancy where, by definition, a specified minimum number (n) of all m independent parallel elements must be functioning for the system to function.

The situation may be modelled in two ways:

1) The n out of m section may be regarded as a separate nest of the system, (see section 2.4 for nesting). 'm' is an input parameter. RELY records a nest failure when the number of functioning branches falls below m.

2) A parallel structure containing all combinations of n out of m elements may be set up using the facility of duplicate elements. The network shown, figure 2.8, illustrates the model for the 2 out of 4 case.

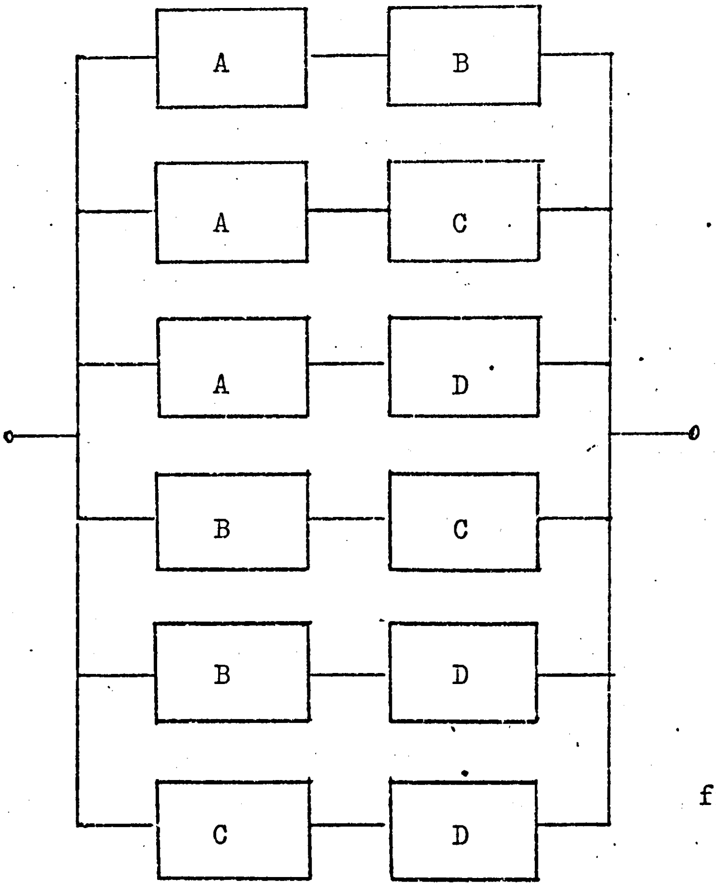


figure 2.8

This method is clumsy for any but the simplest cases.

2.4 THE CONCEPT OF NESTING

The model of a complex system may contain many elements. In order to simulate this model in an efficient way it is necessary to partition it into smaller, two-terminal units called nests. Thus, nesting is a facility provided so that a complex system can be subdivided into nests of readily manageable size. In the case of a complex system such sub-divisions may need to be organised hierarchically in order to keep the number of elements contained in each nest and the number of routes through the structure within reasonable bounds. The ideal nest size, governed by machine space and time considerations, is discussed in section 4.2.6 .

Figure 2.9 shows a simple system which has been nested to illustrate the concept. Normally systems as simple as this would not be nested but this example demonstrates the technique. Elements 10 and 20 are the lowest level of resolution. No distributions are available describing the failure and repair of these elements. Nests 2 and 3 are the actual representation of elements 10 and 20 and distributions are available for elements 1, 2 and 7. Element 8 is further broken down to Nest 4, elements 11 and 12.

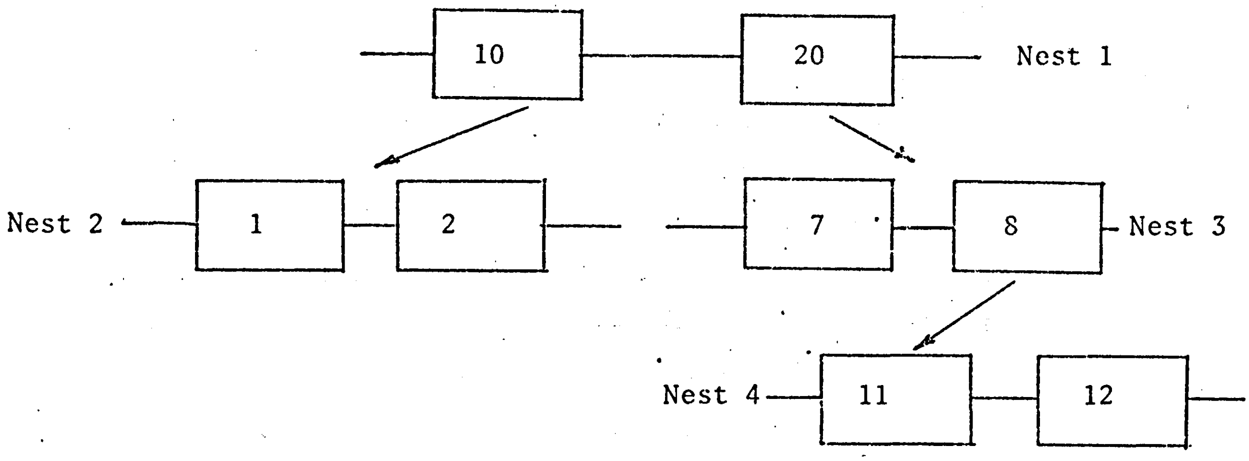


figure 2.9

Simulation of this system would start with nest 4. The outcome of that simulation would be used to obtain distributions describing the performance of element 8, nest 3. Then nests 2 and 3 would be simulated, the results used for elements 10 and 20 of nest 1. Finally nest 1 would be simulated and the overall system performance obtained.

2.5 A COMPLEX MODELLING EXAMPLE

Figure 2.10 represents the fault tree of a gas production unit as described by the reliability engineers of a chemical plant. This is the boolean dual of the success model of the contact network used by RELY. The corresponding contact network model of figure 2.10 is shown in figure 2.11 incorporating a number of auxiliary elements for the purpose of describing specific features of the system as discussed below. The data for this model is contained in figure 2.12. The unusual modelling features are as follows:

Units 31,33. The condition B excludes A is catered for by the series combination of 31 and 33. When B fails the path 31,33 is broken.

Units 3, 4. These are clocks with periods of 1 month and 6 months respectively. Any unit that has a T.P. repair is repaired after the T.P. period if it had failed.

The repair on unit 1233 is provided by the switch, 1222. Unit 122 represents the alarm failure. If failure occurs then 1222 fails and takes 10 hours to repair, 122 itself being repaired after 1 month. The repair of 1233 is dependent on 1222 but delayed by 0.1 hours. Thus if 1233 (the alarm) had failed then repair is delayed by 10 hours.

Figure 2.13 shows the branch and system failure and repair information for 200 runs of 10 years each (87600 hours). The system is only available to fail if there is a demand on the gas to absorption unit (31 and 33). This demand only occurs either 4 times a year for 8 hours each or, exponentially, with μ of 0.8/year and μ repair of 0.1 hours. It can be seen from figure 2.13 that of the 9425 times there was a demand (failures of 31 and 33), on 41 of these occasions the rest of the system was unavailable, thus making 31 or 33 cause a system failure. On 6 other occasions when there was a demand, another unit failed during the demand. The time to first failure for the 47 system failures is 58393 hours (6.67 years). The system repair time is 6.92 hours.

The model could be further refined by taking some practical considerations such as ensuring certain vital units are available before allowing a demand. For example if both the running fan and the stand-by fan were unavailable then clearly the system would fail at once.

If this example were a design study then the next stage would be to remove 31 and 33 so as to see which units in the system caused the system not to be available for 41 of the times of demand.

Furthur examples of modelling may be found in chapter 7 where 3 design studies are described.

2.6 MODELLING EXPERIENCE

The majority of experience gained in the use of these modelling techniques (and subsequent simulation of the models) concerns complex equipment systems such as control and communication systems. Although the diversity of reliability problems of such systems might be considered sufficient proof of the flexibility of the modelling methods, efforts were directed to gain experience in modelling and simulating totally different classes of systems. This experience includes the prediction of the reliability of the service provided to patients in a newly built hospital.

Another example, the investigation of the safety of a gas production unit, is detailed in the previous section of this chapter.

2.7 MODELLING FOR SPARES ESTIMATION

Spares estimation is a facility that was incorporated late on into the project at the request of the sponsoring establishment, A.S.W.E. The concepts and implementation involved in spares estimation are described in section 4.2.7 but some aspects involve modelling.

In a spares situation, a system normally has a number of identical pieces of equipment appearing in different parts of the associated reliability block diagram. Consequently, it is necessary when devising the overall reliability block diagram to take this into account. Should the identical units be spread over a number of nests, it is a requirement of the method used for spares estimation that these nests be the parts of a series chain. All the normal modelling techniques will work for spares estimation.

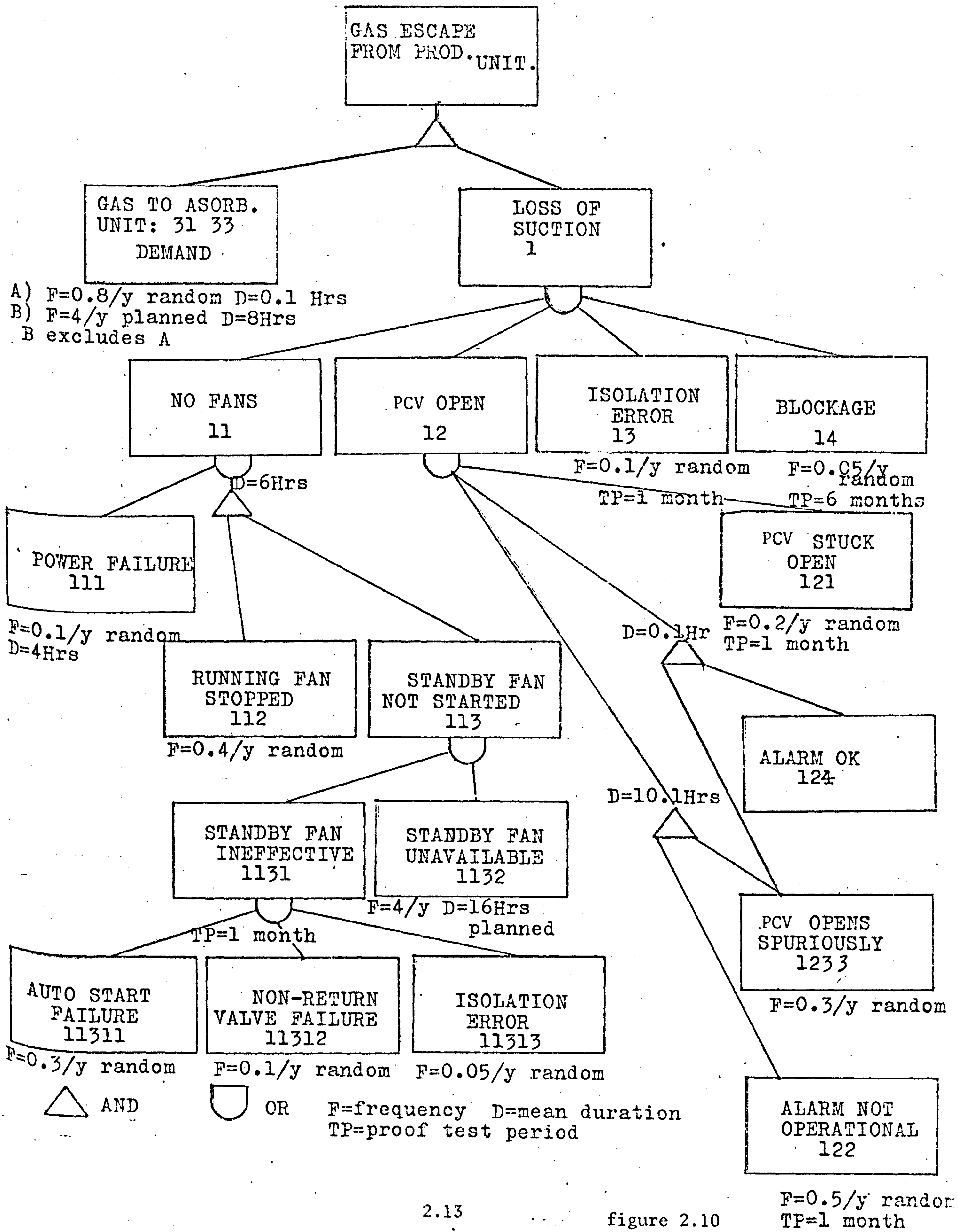

GAS ESCAPE FROM PROD. UNIT.
GAS TO ASORB. UNIT: 31 33 DEMAND
A) F=0.8/y random D=0.1 Hrs
B) F=4/y planned D=8Hrs
B excludes A
LOSS OF SUCTION 1
NO FANS 11
PCV OPEN 12
ISOLATION ERROR 13
F=0.1/y random
TP=1 month
BLOCKAGE 14
F=0.05/y random
TP=6 months
D=6Hrs
POWER FAILURE 111
F=0.1/y random
D=4Hrs
RUNNING FAN STOPPED 112
F=0.4/y random
STANDBY FAN NOT STARTED 113
PCV STUCK OPEN 121
F=0.2/y random
TP=1 month
D=0.1Hr
ALARM OK 124
D=10.1Hrs
STANDBY FAN INEFFECTIVE 1131
STANDBY FAN UNAVAILABLE 1132
F=4/y D=16Hrs planned
TP=1 month
PCV OPENS SPURIOUSLY 1233
F=0.3/y random
AUTO START FAILURE 11311
F=0.3/y random
NON-RETURN VALVE FAILURE 11312
F=0.1/y random
ISOLATION ERROR 11313
F=0.05/y random
AND
OR
F=frequency D=mean duration
TP=proof test period
ALARM NOT OPERATIONAL 122
F=0.5/y random
TP=1 month

figure 2.10

MODEL FOR ASSESSING THE PROBABILITY OF GAS ESCAPE FROM PRODUCTION UNIT

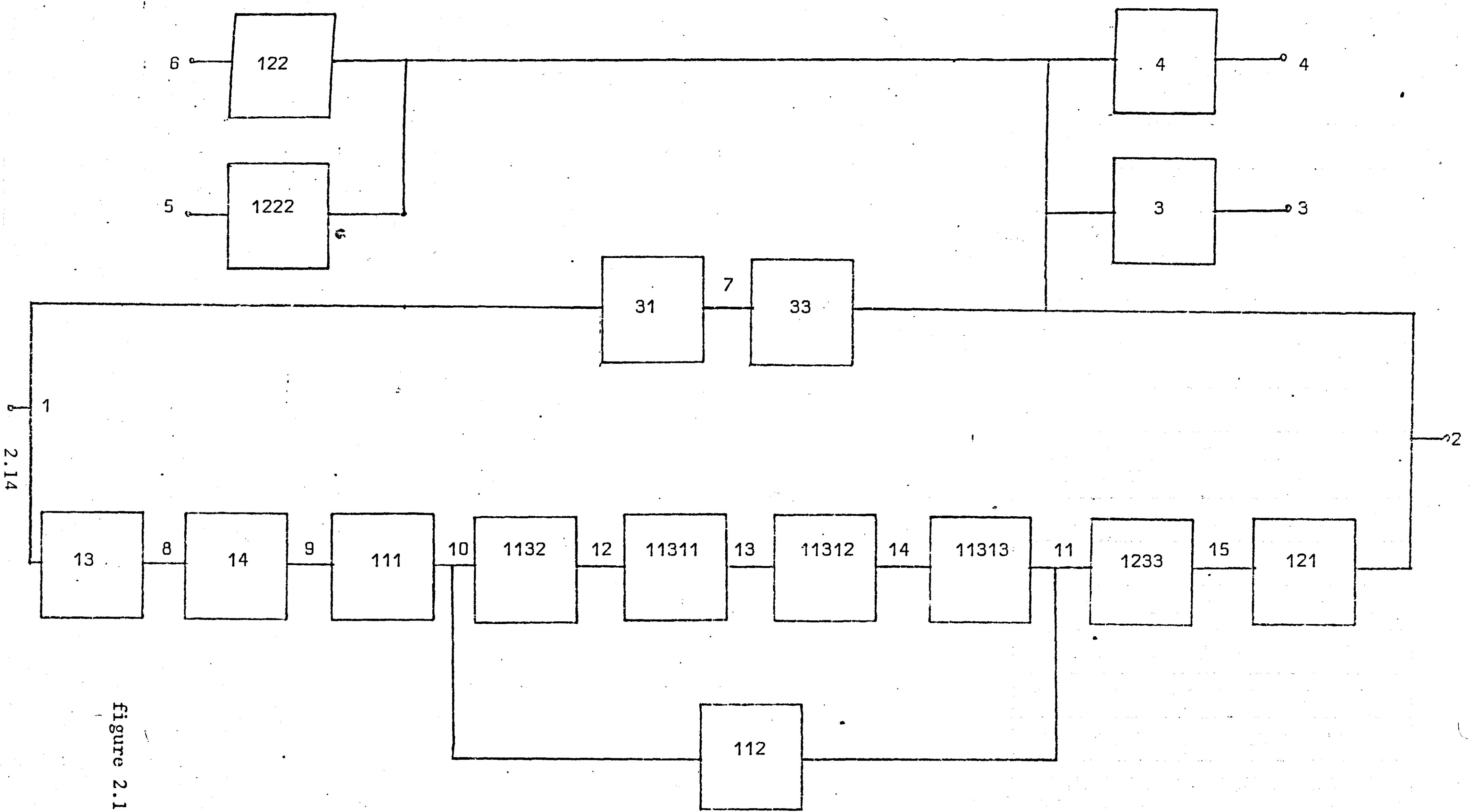


figure 2.11

ALL TIME PARAMETERS GIVEN IN HOURS

FAILURE AND REPAIR CODES: '1' – EXPONENTIAL DISTRIBUTION
'3' – SWITCH (ABRUPT DISTRIBUTION)

MISSION TIME 17520 (2 YEARS)

ELEMENT	NODE NUMBERS		FAIL CODE PARAMETERS AND DEPENDENCIES	REPAIR CODE PARAMETERS AND DEPENDENCIES	ELEMENT NAME	REMARKS
31	1	7	1,0.913x10^{-4}	1, 01	GAS TO ABSORPTION UNIT 'A'	
33	1	2	3, 2190	3, 8	GAS TO ABSORPTION UNIT 'B'	
13	1	8	1, 1.14x10^{-5}	3, 0.1x10^{-5},(3)	ISOLATION ERROR	REGULAR MONTHLY REPAIR AVAILABLE
14	8	9	1, 0.57x10^{-5}	3, 0.1x10^{-5},(4)	BLOCKAGE	REGULAR 6-MONTHLY REPAIR AVAILABLE
111	9	10	1, 1.14x10^{-5}	1, 4	POWER FAILURE	
112	10	11	1, 4.56x10^{-5}	3, 6	RUNNING FAN STOPPED	
1233	11	15	1, 3.42x10^{65}	3, 0.1 (1222)	PCV OPENS SPURIOUSLY	REPAIR DELAYED IF ALARM NOT OPERATIONAL
121	15	2	1, 2.28x10^{-5}	3, 0.1x10^{-5},(3)	PCV STUCK OPEN	REGULAR MONTHLY REPAIR AVAILABLE
1132	10	12	3, 2190	3, 22	STANDBY FAN UNAVAILABLE	
11311	12	13	1,3.42x10^{-5}	3, 6 (3)	AUTO-START FAILURE	REGULAR MONTHLY REPAIR AVAILABLE
11312	13	14	1, 1.14x10^{-5}	3, 6 (3)	NON-RETURN VALVE FAILURE	REGULAR MONTHLY REPAIR AVAILABLE
11313	14	11	1, 0.7x10^{-5}	3, 6 (3)	ISOLATION ERROR	REGULAR MONTHLY REPAIR AVAILABLE
122	2	6	1, 5.7x10^{-5}	3, 0.1x10^{-5},(3)	ALARM NOT OPERATIONAL	REGULAR MONTHLY REPAIR AVAILABLE
1222	2	5	3, 0.1x10^{-5},(122)	3, 10	DUMMY ELEMENT	REPAIR DELAY 10 HOURS IF ALARM NOT OPERATIONAL
4	2	4	3, 2160	3, 2160	DUMMY ELEMENT	REGULAR 6-MONTHLY REPAIR
3	2	3	3, 360	3, 360	DUMMY ELEMENT	REGULAR MONTHLY REPAIR

figure 2.12

ANALYSIS OF BRANCH FAILURES AND REPAIRS

NEST LEVEL 1

BRANCH NO	SYSTEM FAILURES CAUSED BY BRANCH	BRANCH FAILURES	SYSTEM REPAIRS CAUSED BY BRANCH	BRANCH REPAIRS
3	0	24400	0	24200
4	0	4000	0	4000
13	1	186	1	186
14	0	112	0	112
31	8	1625	8	1625
33	33	7800	31	7800
111	2	201	2	201
112	0	842	0	842
121	2	383	4	383
122	0	1014	0	1012
1132	0	7800	0	7800
1222	0	10455	0	10453
1233	1	609	1	609
11311	0	600	0	599
11312	0	178	0	176
11313	0	127	0	127
	47	60332	47	60125

figure 2.13

3. SYSTEMS' ASPECTS OF THE PREDICTION PROGRAM RELY

3.1 THE PROGRAM AND THE 1900 COMPUTER SYSTEM

The choice of computer system influences the structure and format of the program. The 1900 system was chosen for this program primarily because of the existance of a 1905 machine at the Polytechnic. Access to 1900 machines is fairly easy and so most potential users would be able to have a copy of the program on a convenient machine. The other major area of choice in program development is the programming language used. ALGOL 60 was chosen for a number of reasons:

i) It is the preferred language of the sponsoring establishment A.S.W.E.

ii) Implementation of algorithms is fairly straightforward as ALGOL is an 'algorithmic language'.

iii) The sophistication of ALGOL 60 over all other high level languages available, (ALGOL 68 was not generally available at the time), makes the design of modules and control software fairly simple.

iv) The program makes heavy use of dynamic array structures which are not available in other languages.

v) It is the programming language most familiar to the author.

vi) The objective of portability is maintained as ALGOL 60 is widely available on other major computer systems.

vii) The 'procedure' concept in ALGOL 60 enables a modular structure to be realised fairly easily. The modular implementation chosen is described in chapter 4.2.

The choice of the 1900 system gives access to certain software and hardware facilities available with this range. The ones that were considered are:

i) Segmentation: A segment is the smallest unit that can be compiled. By the use of segmentation one may devise a program which can consist of a number of segments in different languages. Each segment is compiled by its own compiler and the whole program is constructed by consolidating the appropriate segments together by means of a universal consolidator.

ii) Overlaying: This is a facility available on machines with a backing store, ideally disc, which allows part of a program to reside in store during run time rather than needing to hold all the program in store. This is useful as it increases the data area and may allow a program to run on a small machine.

iii) Interrupt facility: This exists in some form on all modern machines and enables the processor to timeshare peripherals. It is implemented by use of certain extracodes which perform distinct tasks. These extracodes are available at assembly code level programming. They enable a peripheral transfer to be initiated and processing to be carried on until the stage is reached where the buffer of that peripheral would be overwritten before it had been emptied. The other possibility is subprogramming where a program is divided into members and each member is allowed to be active. However, it is a difficult technique to implement practically.

iv) Backing Store: Large volumes of data not actually being analysed can be kept on backing store. In ALGOL, arrays or part arrays can be written and read from backing store which can either be magnetic tape or disc.

v) Multiprogramming: This enables more than one program to run at any time. It is timesharing at program level.

vi) Trusted program status: Programs which can run other programs or have a program under control are called trusted programs. They are allowed to do certain tasks, by the machine's executive, which ordinary programs are not allowed to do.

Some of these facilities have been incorporated. Figure 3.1 shows the phases of a compilation process. A module or subroutine library is constructed by passing the source program statements through the compiler and producing semi-compiled output which is stored in a library file on disc. To produce a working program the master segment or module (supervisor or control section). is passed through the ALGOL compiler. The semi-compiled output is combined with the semi-compiled library modules by a universal consolidator to produce an overlaid binary object program. The overlay units are held on disc, the master segment being permanently held in store.

Overlaying is achieved by dividing the store into overlay areas and a permanent area. In each overlay area are assigned a number of overlay units of which only one can be present at any one time. The permanent area contains the supervisor which is as small as possible. The structure envisaged allows a number of overlay areas to be used for applications modules so that more than one application module could be operating. Likewise with housekeeping modules, a special type of module described in section 4.2.9.This would allow a form of subprogramming by timesharing. This sophisticated structure was not implemented fully but a subset was used where one overlay area was assigned.

An important consideration is the peripheral requirements of the program. The 1900 system has peripherals of three main types accessible by program.

i) Basic Input devices:
Paper tape reader
Card reader
Magnetic or optical encoding reader
Timesharing terminal

ii) Basic Output devices:
Paper tape punch
Card punch
Lineprinter
Graph Plotter
Timesharing terminal

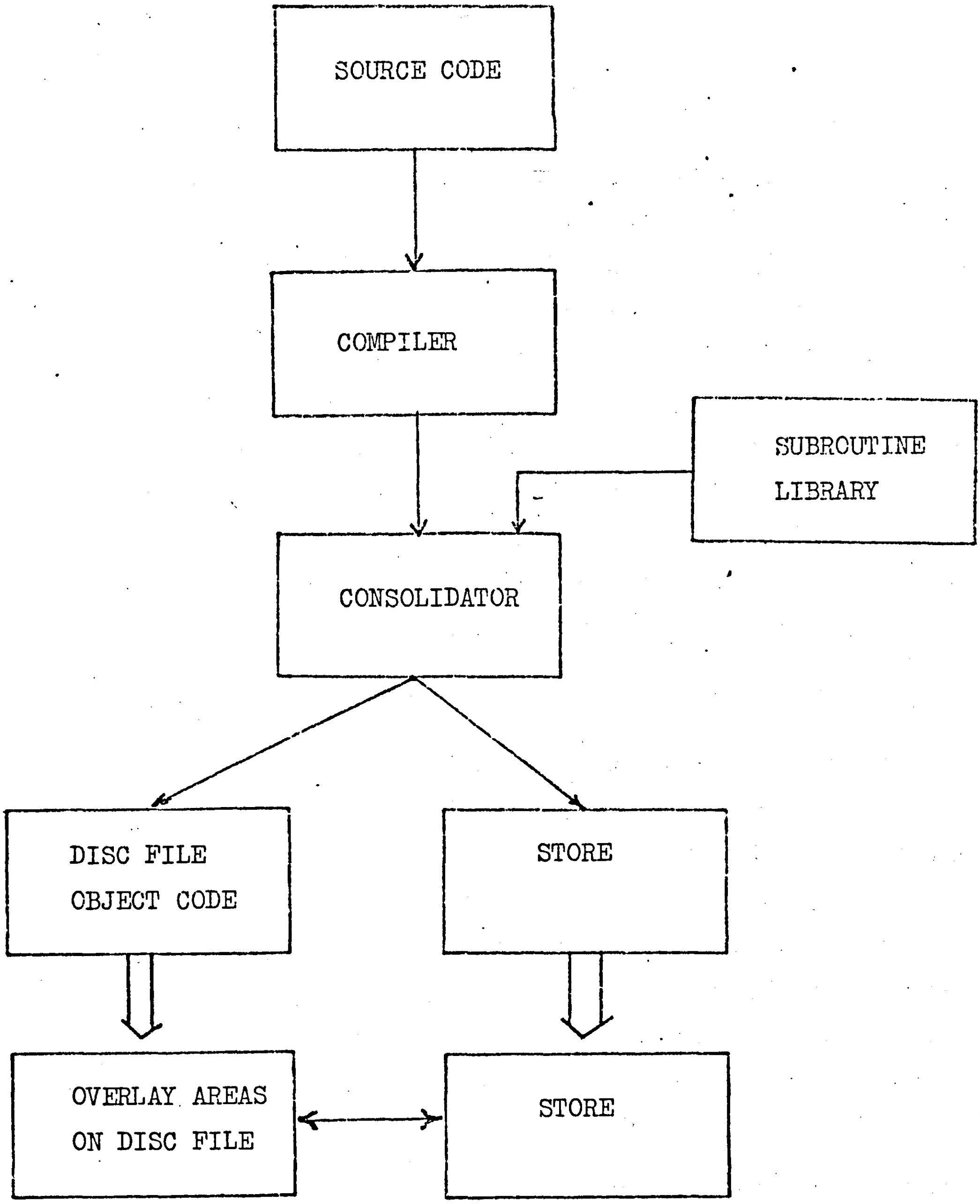

SOURCE CODE
COMPILER
SUBROUTINE LIBRARY
CONSOLIDATOR
DISC FILE OBJECT CODE
STORE
OVERLAY AREAS ON DISC FILE
STORE

figure 3.1

iii) Storage devices :

Magnetic tape
Disk (fixed or exchangeable)
Drum

In deciding which peripherals to make use of, the availability of the different types on 1900 installations must be borne in mind. All normal installations have either a paper tape reader or card reader. RELY allows use of either. All normal installations have a lineprinter so RELY only requires a lineprinter as an output device. All normal installations have at least two tape decks and disc storage. However, disc storage is a limited utility. It is available as named files of a fixed size or workspace available for the duration of a run. Tapes are available as complete units capable of mass data storage. As a result, the backing store medium chosen was tape, and disc storage used as workspace for a small amount of data handling. Tapes have the disadvantage of being slow, although this is not a drawback from a user point of view. The operating system on large 1900 machines allows tapes to be simulated by disc files and this removes the disadvantage of slowness. The main drawback of a tape backing store, however, is that tape is a serial storage device as opposed to disc which is a random access device. This drawback has been overcome by the backing store housekeeping module described in section 4.2.9 which treats tape as though it were a random access device. Combined with the operating system facility, it provides a flexible backing store system.

This choice of peripherals has enabled the installation of the program on other 1900's without alterations. In addition, the overlay technique has allowed machines of 32K store to be used, this being chosen as the smallest practical installation.

3.2 USE OF PROGRAM

The description of the use of the program can be divided into two sections.

i) Operating considerations.
ii) User considerations.

i) Operating a computer program can be performed by an operator (manual method) or by an operating system (automatic method). The latter mode is normally used and it is desirable that the operating instructions be kept as simple as possible in order to simplify the job description. RELY requires the following sequence of operations.

1) Load two magnetic tapes.
2) Make available an input device (card or paper tape reader).
3) Make available a lineprinter.
4) Load the program into the computer.
5) Set switches to control mode of program operation.
6) Activate the program.
7) Delete the program on completion.

ii) A complex computer program requires that the user should have a certain level of knowledge so that efficient use may be made of the program. To this end a user guide has been issued, see chapter 6.1.

The user has control over:

i) The input data.
ii) The outputs available.
iii) The mode of operation of the program.

i) The choice of input data determines the efficiency of the simulation. By tailoring the reliability block diagram according to the recommendations of the user guide (nesting if necessary), the user is able to extract maximum information from a simulation. The choice of mission time and number of runs is particularly important.

ii) A choice of outputs exist. Chapter 8.2.1 outlines the full and summarised output facilities. The user can control the number and range of output by five output control parameters.

iii) The mode of operations is controlled by the setting of eight system switches (a software representation of console switches). The switches perform the following operations if set:

1) Input all data on paper tape. Default is card.
2) Terminate run after ROUTEANALYSIS output. This is used where simulation is not required.
3) Start program at simulation stage with a previously assembled data file. This is used if 2) had been used on the previous run.
4) Provide a summary output. This is used for remote terminal users.
5) Extend data file from 400 records to 1000 records.
6) Start program at output stage. This will output all the results of a previous simulation. It would be used if a previous run had 4) set.
7) Provide Spares Estimation. This will cause the spares data to be input.
8) Provide a repeatable random sequence. This allows an identical random number stream to be used.

A start number can be placed in the program by the user if desired.

4. THE PREDICTION PROGRAM RELY

The major part of the research into computer aided reliability prediction was devoted to the design and implementation of a computer program for use by reliability engineers. The program, which is described in detail in this chapter, utilises the modelling techniques discussed in Chapter 2.

4.1 OBJECTIVES OF PROGRAM

It was envisaged from the conceptual stage that certain objectives would control the way in which the program was to be developed.

i) The program must be user orientated.

ii) The program must provide some useful facilities for reliability engineers as soon after the start of the project as possible, gradually extending these throughout the duration of the project.

iii) The program must have inputs and outputs that are recognised as standard methods of system reliability description by designers in the reliability field.

iv) The program must be efficient.

v) The program must be portable.

vi) The program must not be constrained by an unrealistic demand of central processor and peripheral resources.

i) This was the fundamental objective because it was intended that the sponsors, A.S.W.E., would use the program from an early stage. The most important requirement of A.S.W.E. was the ease of use, even, if necessary, at the expense of efficiency and accuracy.

ii) The design philosophy of many programs demands the delay of the release of software until, sometimes, years of development have been carried out. In order that the sponsors, A.S.W.E., could get an indication of the state of development and could comment upon the facilities provided, the first issue of the program was made available about one year after the start of the project. An additional advantage of this design philosophy was that, since sophisticated new facilities were gradually introduced, users were educated in the use of the program by easy stages. Thus, the project encountered none of the usual user resistance to CAD.

iii) The input and output formats affect user acceptance but go further. It is possible to design an input language which offers ease of coding but requires a transformation from a standard system description to an unusual one. The chosen method of modelling (section 2.1) uses a standard formulation of the problem and, accordingly, this type of description was chosen.

Standard recognised output formats, such as tables, histograms, graphs etc. have been provided to suit user preference.

iv) The efficiency of a program relates to user acceptance in terms of cost of a program run. If the cost is prohibitive compared with existing programs then users will be reluctant to make extensive use of the program in spite of its sophisticated facilities. Chapter 4.2.4 describes the methods that were employed to improve the efficiency of the program.

v) Program portability can be considered in two stages. Firstly, portability amongst computers of the same range (the ICL 1900 series in this case). Secondly, portability amongst computers of comparable size to the prime computer for which the program was designed. The first consideration is fairly straightforward apart from machines at the top end of the 1900 range. There are features present on the 1906A, such as paging, that require fundamental changes in the structure of a program developed on a more modest 1900 computer. The second consideration conflicts with efficiency because making a program more efficient usually involves employing special features of a particular computer. A compromise involves modular design, (section 4.2), where sections of the program are written in assembler code as a replacement for high level language equivalents. This improves efficiency at the expense of machine dependence. Portability is achieved by replacing the assembler code with the high level language equivalents and efficiency improved on the new machine by the conversion of these modules to that machine's assembler. Identifying those parts of the program which critically affected efficiency and incorporating these into a single, assembler-language module of relatively modest size, it has proven possible to find a compromise between the conflicting demands of efficiency and portability.

vi) It is easy to make wide use of all the peripherals of a computer available to the programmer. This has two major drawbacks. Firstly, should one of these peripherals be unavailable due to breakdown or use by another program then this would/might curtail a run of the program. Secondly, it would affect the feasibility of using the program on other, more modest, installations. Chapter 3.1 details how this object has been met.

4.2 THE STRUCTURAL COMPONENTS OF THE PROGRAM

The fundamental feature of the program is its modular construction. A modular program comprises a number of units known as modules, organised by a control or supervisor. This type of structure is very flexible as a working program can be produced after a much shorter time than compared with a monolithic approach. Extra facilities and sophistication can be achieved by adding modules or replacing modules with better equivalents. Modular programming is an area of study in itself and consequently the structure used by RELY is fairly simple. The main features of the RELY implementation are:

i) Set processing tasks like input, histogram output, etc. are entities called applications modules and exist as procedure segments normally held on the module library file.

ii) Data file communication is achieved by a housekeeping module. Requests for data are handled by this module.

iii) With the exception of the housekeeping module, applications modules do not communicate with other applications modules. All interaction is achieved by the control section.

iv) Applications modules are written in ALGOL except for a few modules where assembler code has been used to improve efficiency. ALGOL equivalents do exist for portability and development purposes.

With this structure it is possible to incorporate new facilitites with relatively minor modifications to the control structure. Examples of this are nesting and spares estimation, both of which are fairly complex facilities but were implemented at late stages of development of the project without much trouble and without disturbing facilities developed earlier.

The linkpin of the structure, however, is the housekeeping module. As mentioned in the previous section, this module allows the data file on tape to be treated as if it were held on a random access device. It gives complete flexibility in the order in which applications modules are called within the limits of the logical data flow, i.e. input must come before output. It allows the data file to be treated as a black box into which data is entered and then extracted as necessary. Thus, devising a new module to perform a task is straightforward from a data file consideration; if this new module requires a record of information placed on the file somewhere a request to the housekeeping module for this record will be all that is required. It is analogous to asking a librarian for a book and waiting while he fetches it.

The file is held on magnetic tape which is a cheap mass storage medium. It allows very long simulations to be carried out without the danger of the data file overflowing. The partial results of one run can be retained and used for the next run e.g. data input and verification. In addition to the data file, a second tape is used as a temporary storage device. This second tape is particularly useful during simulation, allowing the output to be written in assembler format (not standard ALGOL records as the data file) for speed of processing.

The description of the component parts of the program follows. The modules are described using the format:

a) Purpose of modules.
b) Theoretical considerations and practical context.
c) Structure of the modules in terms of information flow.
d) Algorithms used.

Modules are grouped, when appropriate, according to the function they perform.

4.2.1 DATA INPUT MODULES.

Three modules are responsible for data input. INPUTROUTINE is the main input module, CONPARIN is used for output control parameters and SPARESINPUT inputs sparing data. The first two of these are dealt with here while SPARESINPUT is described in section 4.2.7.

a) Purpose of modules

INPUTROUTINE is designed to input a reliability block diagram, see section 2.1. The topology is described by branch numbers and the node numbers between the branches.

The probablistic information about the failure and repair characteristics of the elements is described by distribution types in the form of code numbers together with the parameters appropriate to the distributions. Interdependencies, see section 2.2, are described by branch numbers. In addition to the reliability block diagram, information is required about the characteristics of the simulation. Parameters specifying the length of simulated time, the number of simulations, the number of routes required for a working system and nesting structure, see section 2.4, are also input by INPUTROUTINE. Control of output is achieved by CONPARIN. Five outputs are available:

i) System Failure Rate (SFR)
ii) System Repair Time (SRT)
iii) Availability
iv) Time Between Failure (MTBF)
v) Time to First Failure (MTTF)

A parameter is associated with each of these outputs which will:

i) Suppress the output.
ii) Allow choice of the histogram Y axis maximum.
iii) Give automatic histogram Y axis maximum.

Error checks are performed on the input data. The errors that are detected are:

i) Number of branches < 1 or number of nodes < 2.
ii) Incorrect number of parameters for current branch.
iii) Missing branch delimiter (/) See part b) of this section.
iv) Two identical branch numbers.

v) Two identical node numbers.

vi) Specified input/output node doesn't match an encountered node.

vii) Nest termination parameter not encountered.

At the end of a line of branch input a / (solidus) is searched for. This is used as a check on the correct number of branch parameters being input. The parameter specification takes the following form.

One set of nest parameters per nest:

A,X,B,C,D,E,F,G

A =	Block Replacement Number	(Positive Integer)
X =	Replacement Nest Level	(Positive Integer)
B =	Nest Number	(Positive Integer)
C =	Number of Branches	(Positive Integer)
D =	Number of Nodes	(Positive Integer)
E =	Number of Routes for System to Operate	(Positive Integer)
F =	Mission Time	(Positive Integer)
G =	Number of Simulations	(Positive Integer)

One set of branch parameters per branch:

A,B,C,D,E,F,G,H,J,K,L

A =	Branch Number	(Positive Integer)
B,C =	Node Numbers	(Positive Integer)
D =	Fail Code	(Positive Integer)
E,F =	Fail Parameters	(Real)
G =	Repair Code	(Positive Integer)
H,J =	Repair Parameters	(Real)
K,L =	Dependency Parameters	(Integer)

One set of terminal nodes per nest:

A,B

A =	Input Node	(Positive Integer)
B =	Output Node	(Positive Integer)

One nest terminal parameter:

* =	More Nests to follow
↑ =	Current Nest is last Nest.

b) Theoretical considerations and practical context

A practical example will illustrate the use of the input structure.

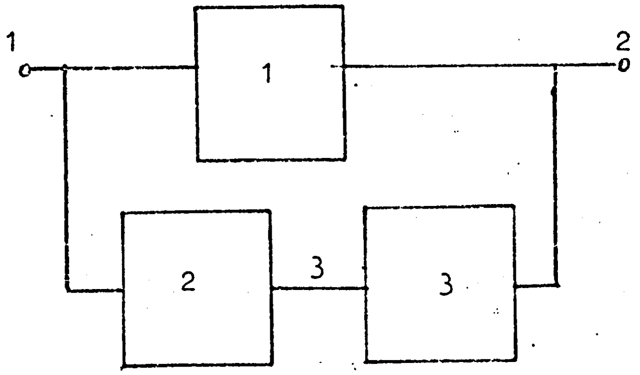


figure 4.1

Consider the example shown in figure 4.1. Branch 1 is independent of 2 and 3, and is characterised by a Weibull distribution. Branch 2 is a standby of branch 1 and its failure is dependent on it. It too is charactèrised by a Weibull distribution. Branch 3 is a switch that switches in 2 on a failure of 1. It operates in 1 second. This model is a standby dependency described in section 2.2.4. The code letters for the distributions are described in section 4.2.3, but for this example the correct numbers are used as described in that section. The nesting parameters A,X,B are set as for a single nest system. The complete input is:

```
0,1,1,3,3,1,100,10
1,1,2,1,2,3,1,2,3,0,0,/
2,1,3,1,2,3,1,2,3,1,0,/
3,3,2,3,.017,0,3,.017,0,-1,-1,/
1,2,↑
0,0,0,0,0
```

The mission time has been set at 100 hours with 10 runs.

c) Structure of the modules in terms of information flow.

The Input device is selected (cards or paper tape) and the nest control parameters read in. The number of branches and number of nodes are used to set up arrays to hold the connection table describing the reliability block diagram. A loop is then entered to perform the following steps:

i) A complete line of branch data (11 parameters) is read into a temporary vector.

ii) Error checks 2 and 3 are performed.

On completion of this loop the next phase of data checking is carried out, error checks 4 and 5.

For internal manipulations the user branch and node numbers are converted into a sequential set of numbers starting at 1. This process is carried out by first sorting the branches into ascending order. Then a sequential set is constructed and a node directory compiled of user node numbers versus internal node numbers.

To illustrate, consider figure 4.2.

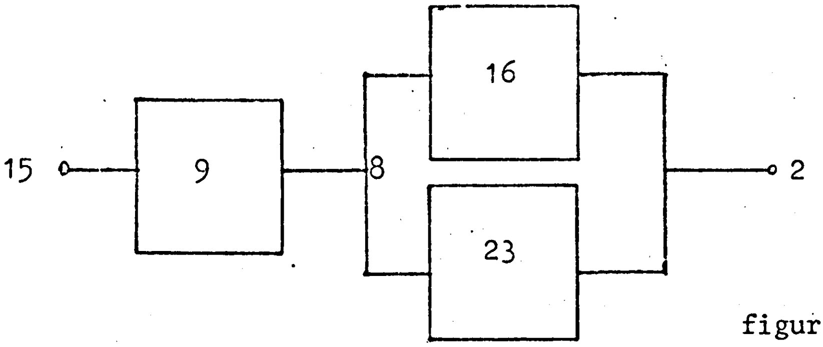


figure 4.2

The connection table after input:

branches	Nodes	
16	8	2
23	8	2
9	15	8

figure 4.3a

After numeric sorting on the branch numbers:

branches	Nodes	
9	15	8
16	8	2
23	8	2

figure 4.3b

The node directory would appear:

Internal node	User node
1	15
2	8
3	2

figure 4.3c

The connection table would then be altered to:

branches	Nodes	
9	1	2
16	2	3
23	2	3

figure 4.3d

Branch 9 would be internal branch number 1 (1st row)
Branch 16 would be internal branch number 2 (2nd row)
Branch 23 would be internal branch number 3 (3rd row)

The dependency branch numbers are also converted to internal branch numbers. The terminal nodes are now input and error check 6 carried out. Then the nest terminator is searched for and if it is not encountered within 5 papertape characters or 80 card characters (the remaining blanks on an 80 column card are treated as spaces by 1900 hardware) then error 7 is displayed. This makes the search finite. The arrays are then written to the data file by the housekeeping module and if more nests follow (nest terminator * encountered) then the whole process is repeated.

d) Algorithms used

No special algorithms are used. Because the number of branches per nest is small, typically 20, no special sorting algorithms are used for the ascending branch order sort.

4.2.2 ROUTE SEARCHING AND PRODUCTION OF STATUS TABLE

These functions are carried out by one module ROUTESEARCH.

a) Purpose of modules

The purpose of this module is to take the connection table produced by INPUTROUTINE and construct a status table.

b) Theoretical considerations and practical context

In order to be able to perform simulation it is necessary to convert the reliability block diagram into a more suitable format. A reliability block diagram is coded by assigning unique positive integer numbers to each node and to each branch and then describing the nodes linked by each branch. This method is user orientated. The process of obtaining a more machine orientated format involves constructing the connection matrix of the reliability block diagram. A connection matrix has branches by rows and nodes by columns. The terminations of a branch are indicated by two 1's in the appropriate columns. The connection matrix for figure 4.4 would be:

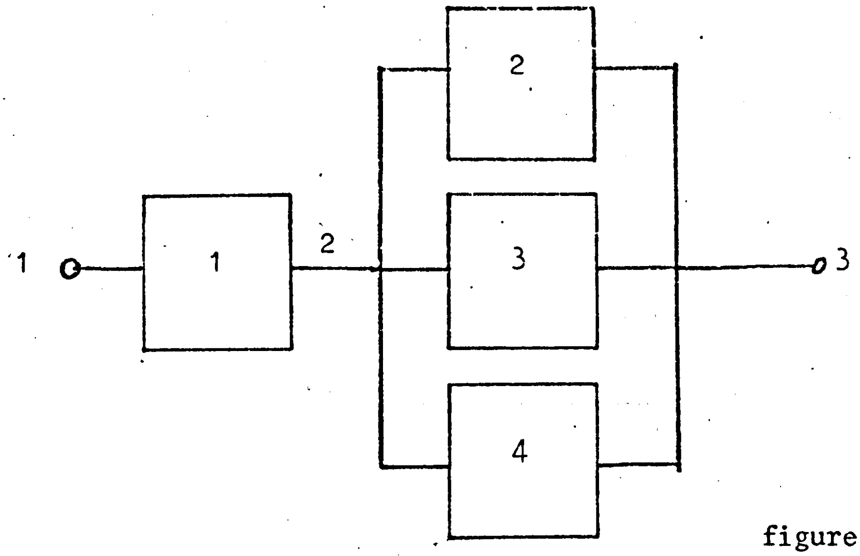


figure 4.4

	1	2	3
1	1	1	0
2	0	1	1
3	0	1	1
4	0	1	1

This connection matrix is used as a basis for obtaining the status table. This table provides information about the state function which, when combined with a time sequence, provides an immediate assessment of the state of the system. The state function of figure 4.4 would be:

(1 'AND' 2)'OR'(1 'AND' 3)'OR'(1 'AND' 4)

alternatively written (1.2)+(1.3)+(1.4)

Another way of considering the state function is to imagine it describing all the routes from the input to the output (the number of 'AND's). The status table displays all these routes and the current state of the routes during the time sequence is given by altering the values of the elements from 1 to -1. The status table for figure 4.4 would be:

		Routes		
branches	1	1	1	1
	2	1	0	0
	3	0	1	0
	4	0	0	1

This would indicate a working system (all routes functioning). Failure of branch 1 would disable all the routes and the system would fail. A full description of time simulation can be found in section 4.2.4.

c) Structure of the modules in terms of information flow

The module is called by control on a nest by nest basis. The connection table for the current nest is obtained by the housekeeping module. The algorithm, see d), is used to construct each column of the status table on disc. This solves the problem of not knowing the number of routes until completion. At the end of the module the complete status table is read back and placed on the data file by the housekeeping module.

d) Algorithms used

The only algorithm used is that for obtaining the routes.

Starting at the input node and proceeding always along the first available branch of lowest serial number, a route is traced out. A record of the branches comprising the route is held in vector PR i.e. an entry PR[N] = 1 indicates the Nth branch lies on that route. A node list is kept in vector NL, i.e. an entry NL[M] = 1 indicates the Mth node has been traversed. As each branch in the route is traced out, the nodes of that branch are marked as unavailable in the connection matrix but the destination or second node of that branch is not marked until a check has been made in NL to ensure that this node has not already been traversed. If the node has already been encountered then the branch leading to this node is retraced, the other node of that branch then made available again and the path continued if possible taking the next branch of lowest serial number out of that node. Should there not be a branch available then the branch that initially lead into that node is retraced, that node then being removed from the node list, that entry branch from the path record and that node associated with that branch in the connection matrix made available. When the output node is encountered the path record is copied away. The branch that leads to the output node is retraced, the output node being deleted from the node list. The branch and the node associated with it in the connection matrix are made available again from the path record. On arriving at the other node of the final branch, the search is then carried on as before, attempting to find the next branch of lowest serial number. When the retracing has been sent all the way back to the input node and no more branches are available then the algorithm stops as all the routes have been found.

The algorithm:

I = Rows, J = Columns. CONMAT is the connection matrix of dimension, number of branches (by row) by number of nodes (by column). IMAX = number of rows + 1. NL, the node list, is a vector of dimension number of nodes. PR, the path record, is a vector of dimension number of branches. CONMAT [I,J] has three states 0,1 or 2. 0 indicates that a branch is not connected to that node, 1 indicates that it is connected and 2 indicates that the node has been encountered.

```
 1.  Set I = 3, J = Input node, clear all lists, Set NL[J] to 1.
 2.  If CONMAT[I,J] = 1 go to 5.
 3.  Set I = I + 1.
 4.  If I = IMAX go to 11 otherwise go to 2.
 5.  Put CONMAT[I,J] = 2.
 6.  Find J for which CONMAT[I,J] = 1, I retains previous value J varied.
 7.  If NL[J] = 1 go to 19.
 8.  Set NL[J] = 1.  PR[I] = 1, CONMAT[I,J] = 2.
 9.  If J = output node go to 22.
10.  Set I = 1 and go to 2.
11.  If J = Input node then Stop.
12.  Set I = I - 1.
13.  If CONMAT[I,J] = 2 go to 14 otherwise go to 12.
14.  Set CONMAT[I,J] = 1.
15.  Reset NL[J] = 0, PR[I] = 0.
16.  Find J for which CONMAT[I,J] = 2 and set CONMAT[I,J] = 1.
17.  Set I = I + 1.
18.  If I = IMAX go to 11 otherwise go to 2
19.  Find J for which CONMAT[I,J] = 2 and set CONMAT[I,J] = 2.
20.  Set I = I + 1.
21.  If I = IMAX then go to 11 otherwise go to 2.
22.  Write away PR vector, set CONMAT[I,J] = 1, set NL[J] = PR[I] = 0.
23.  Find J for which CONMAT[I,J] = 2 and set CONMAT[I,J] = 1.
24.  Set I = I + 1.
25.  If I = IMAX then go to 11 otherwise go to 2.
```

4.2.3 GENERATION OF PROBABILITY TABLES

Information about the failure and repair characteristics in the form of standard distributions is supplied at the input stage. Although this is in a usable form for simulation it is very time consuming to sample equations with random numbers in order to obtain event times, (see section 4.2.4 for a theoretical description of time simulation). To speed up the process the cumulative distribution function is converted into a tabular format.

a) Purpose of modules

The purpose of the module, PROBPOINTS, is to convert a standard distribution function into a table of values of t for 50 equally spaced values of F(t) going from 0 → 1.

b) Theoretical considerations and practical context

The input routines, section 4.2.1, have provision for code numbers describing distribution types for each branch together with parameters pertaining to the appropriate distribution. The distributions available are:

0	=	No distribution
1	=	Weibull
2	=	Lognormal
3	=	Switch

Weibull: 1st parameter is the scale parameter (K)
2nd parameter is the shape parameter (M)

An exponential distribution is obtained by putting the 2nd parameter to zero. The first parameter then becomes λ the failure rate.

Lognormal: 1st parameter is the lower 3σ point
2nd parameter is the upper 3σ point

Switch: 1st parameter is the actual switch time
2nd parameter not used

Both Weibull and Lognormal are distributions that have the shape as shown in figure 4.5.

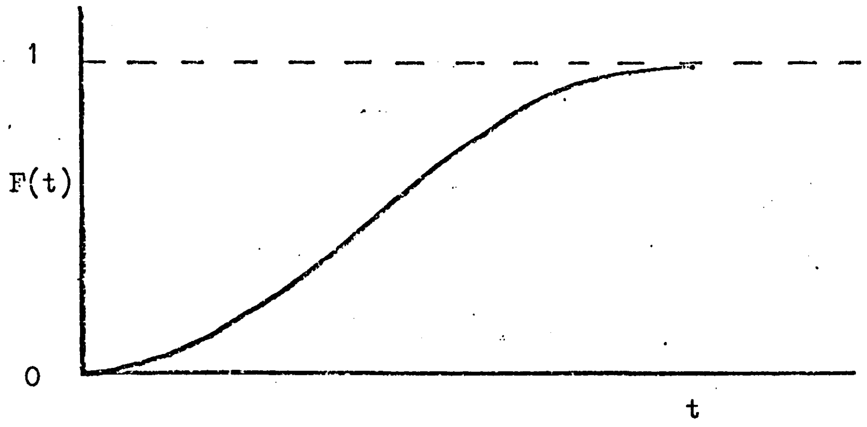


figure 4.5

They are characterised by the properties:

$t = 0$ at $F(t)=0$
$t = \infty$ at $F(t)=1$

As the table of values of t is designed to accept F(t)0-1 a difficulty is created for F(t)=1. It is impossible to store ∞ as the final value in the table. Instead 10^{20} was used as a representation of ∞. However this gave rise to very large errors in the final section of the table. Should a random sample be generated in the last interval, see section 4.2.4 then the corresponding value of t obtained by linear interpolation will be large, probably greater than the mission time. Consider the case of an exponential distribution with $\lambda = 1$. The 49th value in the table is 3.8632. The largest number that can be generated below 1 is 0.999999 (generator definition is 10^{-6} in range 0 - 1). The value of t corresponding to this F(t) on an exponential distribution is 13.8155. Thus all values of t on a true exponential, apart from the 1 in 10^6 value of 1, lie below 13.8156. Consider what actually happens with limits 3.8632 and 10^{20}. Taking the next possible value of F(t), after the 49th interval, which is 0.979001 results in a value of t of 5.10^{15}

which will be outside any possible mission time as the mission time is an integer parameter which cannot lie above 8388607 ($2^{23} - 1$). This method was replaced by a better approximation suggested by the users (reference 30) where $t50 = t49 + 2/\lambda$. λ can either be an input parameter as with exponential or calculated from the values in the table already found.

c) Structure of the modules in terms of information flow

The module is called on a nest by nest basis by control. The record containing the distribution code letters and parameters is obtained from the data file by the housekeeping module. The code letter is used to switch to the correct section of the module depending on the distribution type. The actual mechanism used to fill the table is described in d). When the table is full, the housekeeping module places the table on the data file.

d) Algorithms used

The Weibull distribution is described by the equation:

$$F(t) = 1\text{-EXP}\left[\frac{-K_t^{m+1}}{m+1}\right]$$

Writing as an inverse function:

$$t = \text{EXP}\left[\frac{\text{LOG}_E - \left(\frac{m+1}{K}\,\text{LOG}_E\,(1 - F(t))\right)}{m + 1}\right]$$

Putting m = 0 in the distribution function:

$F(t) = 1 - \text{EXP}(-Kt)$ which is an exponential distribution.

The process of obtaining the table is fairly straightforward. $F(t)$ is incremented from 0 to 0.979 in 49 equal steps and the equation solved.

The Lognormal distribution is obtained by generating a normal distribution and taking EXP t for given values of F(t). The process used is lengthy and involves the following steps:

i) Generate values of F(t) for given values of t on a normal distribution, by numerical integration. The algorithm used to do this is number 226, Generation of Normal Distribution Function from the collected algorithms published by the A.C.M., (reference 7). The mu and sigma must first be calculated from the lognormal upper and lower 3 σ's by the following formulas:

$$\mu = \mathrm{LOG}_E \left[\frac{(3\sigma u + 3\sigma L)/2}{\sqrt{1 + \left(\frac{3\sigma u - 3\sigma L}{3(3\sigma u + 3\sigma L)} \right)^2}} \right]$$

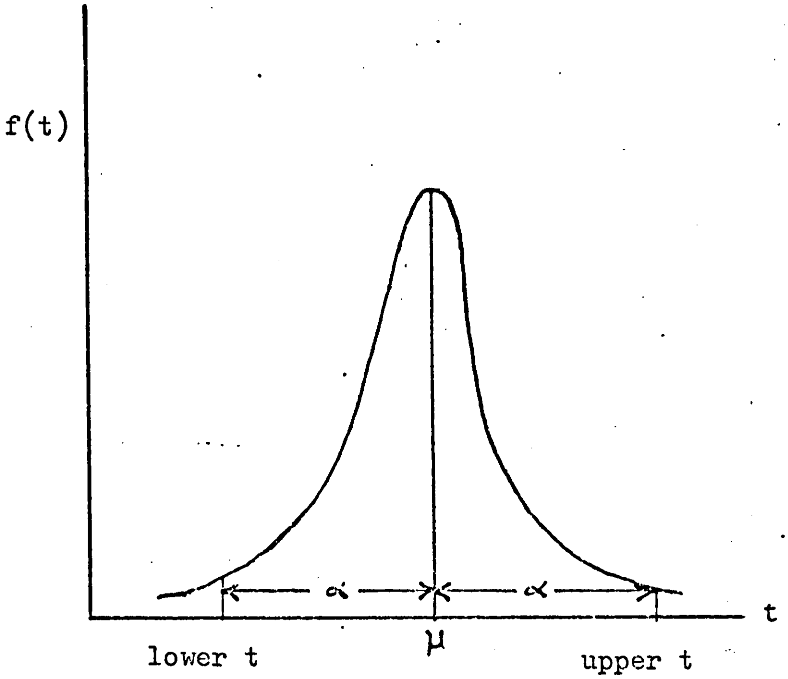
$$\sigma = \sqrt{\mathrm{LOG}_E \left[1 + \left(\frac{3\sigma u - 3\sigma L}{3(3\sigma u + 3\sigma L)} \right)^2 \right]}$$

The range of t is found from:

UPPER $t = \sigma \times 2.0055 + \mu$

LOWER $t = \mu - \sigma \times 2.0055$

figure 4.5a illustrates the validity of this.



Normal density function figure 4.5a

$\alpha = \sigma \times 2.0055$. For probability of 0.979 i.e. area under curve normalised $t = 2.0055$.

100 values of F(t) are generated using this method.

ii) Interpolate values of F(t) first obtained, to obtain values of t for required values of F(t) (at equal intervals).

iii) Transform the values of t so obtained by calculating EXP(t).

The switch distribution is obtained by placing the switch time (the 1st parameter of the distribution) into the table for all fifty values.

4.2.4 TIME SIMULATION

Three modules are responsible for time simulation; MTPOSITION, TIMESEQ and TIMESEQMT. TIMESEQ is the main module, the other two being backing store manipulation modules.

a) Purpose of modules

The purpose of the modules is to generate nest failure and repair times for a given number of simulations each of a specified length using the status table generated by ROUTESEARCH, section 4.2.2, and the probability table generated by PROBPOINTS, section 4.2.3.

b) Theoretical considerations and practical context

Time simulation is achieved by the following process.

i) Random samples are generated by the random number generator.

ii) These samples are applied to the failure distribution tables and linear interpolation carried out to obtain a first failure time for each element in the nest.

iii) The times so obtained are chained together so that rapid accessing of the sequence can be achieved.

iv) The times are applied to the status table and the state of the system checked after each time used.

v) The time previously used is replenished and chained into the sequence.

vi) This process is carried on until the required number of simulations each of the required length of time have been performed.

A simple example will illustrate the basic method. Consider the status table, figure 4.6, obtained from the reliability block diagram, figure 4.7.

		Routes 1	2	3
Branches	1	1	1	1
	2	1	0	0
	3	0	1	0
	4	0	0	1

figure 4.6

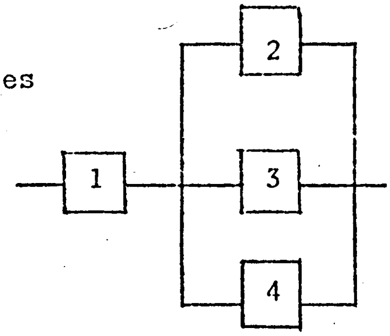


figure 4.7

Figure 4.8 shows a possible time sequence for first failure together with its chaining.

1	10	4	F
→ 2	8	1	F
3	16		F
4	14	3	F

figure 4.8

After the first time, 8 hours, the status table will be:

1	1	1
-1	0	0
0	1	0
0	0	1

figure 4.9

and an additional vector, the routes available vector will appear:

0	1	1

figure 4.10

indicating 2 routes still available. Replenishment of the event time for branch 2 could produce a new time sequence thus:

→	10	4	F
	15	3	R
	16		F
	14	2	F

figure 4.11

Applying the next event, failure of branch 1 at 10 hours would produce a status table and routes available vector:

-1	-1	-1
-1	0	0
0	1	0
0	0	1

0	0	0

figure 4.12

Thus the system would fail at 10 hours and this would be recorded.

The simulation requires a number of input records from the data file. The modules are all written in Assembler code for reasons explained later on in this section and consequently the request to the housekeeping module for the records has to originate from the control section. The events produced by the simulation are stored on the 2nd magnetic tape, the work file, because the housekeeping module cannot be used to write simulated output on the data file as the data format is assembler and not ALGOL produced. MTPOSITION is responsible for positioning the tape prior to simulation. The simulation itself is performed by an assembler code module, written in PLAN, for efficiency considerations which are:

i) Array accessing can be simplified once the address of the first element has been found.

ii) Extracodes can be eliminated.

iii) ALGOL 'FOR' loops can be accomplished very simply.

iv) Subroutine calls can be made with one instruction.

v) The writing of vectors to magnetic tape backing store can be timeshared.

vi) Floating point operations can be carried out without the need to store intermediate products.

Examining the reasons for these advantages:

i) Arrays are stored column by column. Once the address of the 1st element has been obtained, any other element in a repetitive loop can be obtained by a simple add to a modifier. Modification is the process where an element is accessed by taking a base address and adding on to it an incremental value. The advantage comes because the base addresses can be obtained outside all the repetitive loops whereas in ALGOL the base address has to be obtained on every array access. A 2-dimensional array element takes 250μS in ALGOL compared with 20μS in PLAN.

ii) Certain instructions in the basic machine instruction set exist not as hardware functions but as a series of hardware instructions performed by the executive of the machine. The cheaper machines in a range usually have more of these instructions, known as extracodes. But the major disadvantage of them is the time required to perform them. Not only must the actual instructions comprising the extracode be obeyed but the executive must be entered and left. Two instructions in particular are used in the simulation module. 'FIX' converts from floating point to fixed point and 'FLOAT' does the opposite. They both take about 650μS to perform compared with a standard instruction time of 7μS. There were 9 calls of these instructions before attempts were made to remove them. 6 of them were relocated outside repetitive loops, 1 was replaced but the 2 remaining (FLOATs) were found impossible to replace. It was decided to replace these by the 7 executive obeyed instructions which consequently cut the time down inside the loop from 5.85mS to 98μS.

iii) An ALGOL for loop when obeyed has to incorporate certain checks to ensure that the loop is not corrupted by the statements inside it. These precautions can be dispensed with, making a loop a simple increment and compare.

iv) In ALGOL the calling of procedure takes 250µS if the procedure has no parameters and an extra 250µS for every parameter. An equivalent call in PLAN takes 7µS, the parameters being available anyway.

v) When a vector is to be written away to magnetic tape, the ALGOL routine stops the program until the operation is completed This is because the next ALGOL statement might write to that vector while it is still being written to the tape thus corrupting it. In PLAN the peripheral transfer can be activated and the program allowed to continue until the vector is written to. By having two vectors it is possible for the transfer to have very little effect on the main program. This process is similar to timesharing and is called double buffering.

vi) When floating point arithmetic operations are carried out in ALGOL all intermediate products must be stored even though it may not be necessary for the final result. In PLAN the equation can be rearranged so as to minimise this.

The result of implementing these changes was to reduce the simulation time for a simple problem from 20 minutes with the ALGOL version to 9 seconds with the PLAN version, a factor of 130:1.

c) Structure of the modules in terms of information flow

The subroutines of TIMESEQ are each responsible for a specific task. The subroutines are:

TSE: This routine contains the random number generator. It is responsible for producing an event time when requested. The random number generator is a multiplicative congruent type. It can be primed by a combination of the date and time or else continue a sequence already started. Initially the length of number used was a 24 bit integer but this was found to cause problems with the degree of randomness in a small interval, see chapter 8.2.2. The length was increased to 48 bits and the randomness found to be sufficient within the limits of the simulation technique. The actual process involved in obtaining a random number requires the following steps:

i) Obtain the previous number of the sequence.
ii) Multiply it by 25681
iii) Collate off the middle 20 bits of the least significant word.

10 random numbers are generated at one time and then used. Having obtained a random number, the two values of t required for linear interpolation are obtained and linear interpolation performed to produce an event time.

CFT: This routine is responsible for chaining the first event times of all the branches in the nest. On exit, the 1st event time is held and the branch number for that event time is also held.

SST: This routine is designed to alter the status table for the current event. The entries for the correct row (the current branch) are altered to -1 if the event is a failure or 1 if the event is a repair. Associated with each route (column of the status table) is a counter which counts the number of failed branches in the route. Zero indicates the route is working. Thus if a branch is repaired then the counter is decremented. If a change to or from zero occurs then a routes counter is incremented or decremented accordingly. This mechanism saves a time consuming scan of the status table after every event. If the current event has caused a system event then SST calls another routine FAS to transfer the event time to a buffer.

FAS: This routine accepts event times and places them in the correct output buffer (the one that is being filled). When the buffer is full, a transfer is initiated, the other buffer now being used. Normally, simulation is not held up by transfers unless system events happen at every branch event.

REP: This routine is designed to chain in to the time sequence the replacement event generated for the one just used.

While the control sequence in TIMESEQ calls the subroutines, it in itself is fairly complex. It comprises:

i) An initial phase to set up constants and data areas.
ii) A loop on number of missions.
iii) An inner loop on mission time.
iv) A final 'clean up' phase.

The constituents of the control are:

i) The output tape is prepared for reception of events. The arrays containing the status table, connection table and probability table are located and their base addresses calculated. The random number generator is primed. Various floating point constants are formed.
ii) The status table is set to a virgin state at the start of the mission. The route counter is cleared. The time sequence of first events is generated by successive calls of TSE. Dependencies are examined, the times in the time sequence adjusted and the repair and failure pointer column set. CFT is called.
iii) In this loop, SST, TSE and REP are called until the mission time is reached. At the end of the mission FAS is called with the mission time as the event time.

iv) The buffers are written to the tape so that partial information in them is preserved.

After simulation, the output tape is transferred to the data file by TIMESEQMT. Blocks of information are read from the simulated output and placed on the data file by the housekeeping module.

d) Algorithm Used

The adjustment to the event times caused by dependency depends on:

a) The state of the independent element (working or failed).

b) Whether dependency or inverse dependency is in operation.

The algorithm used:

1) Is there dependency. If so go to 3
2) Go to Out
3) Is the dependency inverse. If so go to 9
4) Has the independent failed. If yes go to 7
5) Is the dependent working. If yes go to 14
6) Go to Out
7) Has the dependent failed. If yes go to 14
8) Go to Out
9) Has the independent failed. If yes go to 12
10) Is the dependent working. If no go to 14
11) Go to Out
12) Is the dependent working. If yes go to 14
13) Go to Out
14) Adjust time
15) Out

4.2.5 STATISTICS OF SIMULATION.

The output statistics are produced by two modules, STATISTICAL and CELLMAP.

a) Purpose of modules

STATISTICAL is responsible for taking time simulation output and generating five statistics.

i) Times Between failure
ii) Availability
iii) System Repair Times
iv) Times to First Failure
v) System Failure Rates

The means, standard deviations, maxima and minima of these quantities are calculated. The output from STATISTICAL is passed onto CELLMAP.

CELLMAP takes the values making up each of the five statistics and performs a sort in order to be able to divide up the range of values into ten equal intervals called cells. The number in each interval is obtained and stored so that it can be used by the histogram output module, section 4.2.7.

b) Theoretical considerations and practical context

The events output by the time simulation, section 4.2.4 consist of a list of system failure and repair times in ascending order up to the mission time, which is included on the output. A pictorial representation of this is shown in figure 4.13.

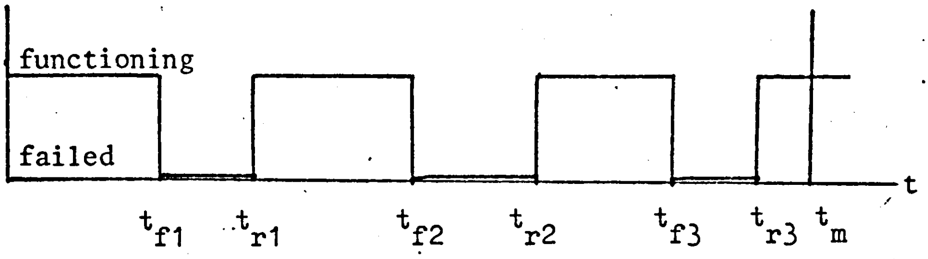


figure 4.13

In addition to the actual values a marker is placed next to each value on the data file to indicate the type of event.

1	=	Repair
-1	=	Failure
0	=	Mission Time

c) Structure of the modules in terms of information flow

STATISTICAL is entered and the first action is to request a record of simulated output which is serviced by the housekeeping module. A loop is then set up on the events contained in the record which automatically replenishes the record when all the events have been processed. The events are extracted from the record and the module control passes to one of three sections depending on the value of the marker. The next section, d, gives the actual formulae used. If the event is the first of its type in the current mission then a reference is stored with this value. If this first event is a failure then the time is stored in the output buffer as a time to first failure. As the events are processed, the times between failure are obtained by subtracting the reference from the current failure, the reference then being updated by the current failure time. The repair time is obtained by subtracting the current repair time from the last failure time. The availability is found by accumulating all the repair times in a given mission, subtracting them from the mission time and dividing by the mission time. The system failure rate is found by counting the number of failures in a mission and dividing it by the mission time. While these quantities are being assembled, the mean and standard deviations are found. In addition, the maximum and minimum of each quantity is found by comparing the current value of the quantity with a reference maximum and minimum and updating if necessary. When the output buffer is full the housekeeping module is requested to write it to the data file.

The markers used next to the output quantities are:

0	=	Time Between Failure
1	=	Time to Repair

2 = Time to first Failure
3 = Availability
4 = Failure Rate

Finally a 30 word vector is filled with all the mean's, standard deviations, maxima and minima.

CELLMAP is then entered and the first phase is to obtain the abscissa maximum. This is achieved by estimating the next number, above the maximum value of the quantity, from a base of 2, 5 or 10, e.g. if the maximum value was 0.042 then the abscissa maximum would be set to 0.05. This gives uniform scaling of the 10 cells of the histogram. A loop is then entered which extracts the quantities from the output records of STATISTICAL, sorts them into one of the five types and increments the appropriate cell of the correct type. The output of CELLMAP is five 10 element vectors.

d) Algorithms used

The formulae used for computation of the statistics are:

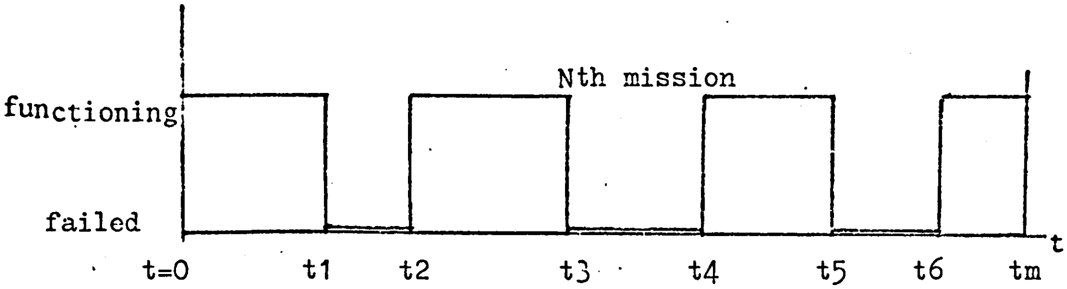


figure 4.13 a

Time to First Failure :

for Nth mission = t_1

hence mean for L missions = $\dfrac{t_{11} + t_{12} + t_{13} \dots t_{1L}}{L}$

If a mission has no failures then the quotient is decremented by 1 and a counter incremented by 1. This counter is stored and output.

Time Between Failure :

for Nth mission = $t_3 - t_1,\ t_5 - t_3,$

hence mean for L missions $= \left[\sum_{j=1}^{L} \frac{t_{Kj} - t_{1j}}{\text{number of failures} - 1} \right] \times \frac{1}{L}$

where K is the last failure

If a mission has less than 2 failures then it is not included for the calculation of MTBF. L is decremented by 1 and a counter is incremented, stored and output.

Repair Time :

for Nth mission $= t_2 - t_1, \; t_4 - t_3, \; t_6 - t_5$ (3 failures)

Hence mean for L missions $= \left[\sum_{j=1}^{L} t_{2j} - t_{1j} + t_{4j} - t_{3j} \ldots\ldots \right] \times \frac{1}{L}$

If a mission has no repairs then it is not included for the calculation of MRT. L is decremented by 1 and a counter is incremented, stored and output.

Availability :

for Nth mission $= \frac{(t_1 + t_3 - t_2 + t_5 - t_4 + t_m - t_6)}{tm}$

hence for L mission $= \left[\sum_{j=1}^{L} \text{availability for jth mission} \right] \times \frac{100}{L}$

Failure Rate:

For Nth mission $= \frac{3}{tm}$

hence mean for L missions $= \left[\sum_{j=1}^{L} \frac{\text{number of failures in jth mission}}{tm} \right] \times \frac{1}{L}$

4.2.6 NESTING

Nesting is carried out by one module, TABULAR, and by control. The control aspects are described in section 4.2.10.

a) Purpose of modules

TABULAR is designed to take the output of a simulation, in the form of nest events, and use this output to obtain a tabular distribution describing the performance of the nest in the same format as PROBPOINTS, namely a table containing 50 values of t for equal intervalled values of f(t) going from 0 to 1.

b) Theoretical considerations and practical context

The object of nesting is to obtain distributions for elements by a process of simulation. The simulated output, which can be represented as in figure 4.13., can be converted to histograms of times to failure and times to repair. The histogram represents the density function for that nest. The cumulative function, which is the tabulated function, can be obtained by taking the area under the histogram. The advantage of this technique is that no curve fitting needs to be carried out on the function.

Although the distribution obtained is a record of a simulation it does not define the boundaries i.e. at $F(t) = 0$ and at $F(t) = 1$. The case for $F(t) = 0$ is easily catered for by putting t equal to 0. At the top end $F(t) = 1$ it is not so simple. It is possible to estimate with standard distributions what the top value should be, but when the form of the distribution is not known, there is no way of calculating the top value. An estimate based on the mean can be found from:

$$tmax = tmax - 1 - 98/ \sum_{j=1}^{max-1} tj$$

A justification for this method can be found in reference 30.

The process of obtaining the distribution is as follows:

i) The first 1000 times to failure (or repair for the repair distribution are taken. The technique will work with as few as 50 values but a more accurate result will be obtained with 1000 values.

ii) The values are sorted into ascending order. Section d has details of the algorithm used.

iii) The table is constructed by finding the interval n, $^1/49$ of the number of values, and placing in it every Nth value.

This method,although fairly easy to implement,lacks sophistication. The users commented on a number of points and their suggested improvements to the method are given in reference 30. The ideas have been incorporated into an improved TABULAR which is being written.

The drawbacks in the existing module are:

i) The maximum number of events that are utilised is 1000.

ii) Storage space requirements are quite high. 4000 words are needed to hold 2000 events in floating point format.

iii) A numeric sort is required on the numbers.

The improved method has the following features:

i) The space required is only 400 words.

ii) A sort only needs to be carried out on 200 event times.

iii) The remaining event times are sorted as they are generated.

c) Structure of the modules in terms of information flow

The routine TABULAR is entered and a loop set up to obtain the simulation output. The events are extracted and up to 1000 times to failure and times to repair are obtained and placed in two vectors. Both vectors are sorted into ascending order by use of the disc backing store. The vectors are stored on the backing store and the space that was used to hold the values is now used for the sort. On completion of the two sorts, the table of 50 values is filled by taking every Nth value where n is $^1/49$ of the number of values. The output of the module is one row of the probability table (50 failure values and 50 repair values) which can be slotted into the probability table by control.

The improved method uses the same theoretical technique for filling the row of the probability table but it differs in a number of details. An initial 200 values of times to failure and times to repair are used to estimate the tmax's of the generated distributions. These tmax's define the time interval of each cell of a 400 word vector which has tmax as the top value. The events, used to estimate tmax, together with the other events of the simulation are then examined to see which cell they lie in, and the appropriate cell incremented by 1. Should the estimate of tmax be too small then another estimate of tmax is made and the scan restarted.

d) Algorithms used

The algorithm used for the numeric sort is called QUICKERSORT, (reference 29). In comparison with three other sorts, Shellsort, Stringsort and Treesort it is much faster over the whole range of numbers used in the module.

The method used is to continually split the vector into parts such that all elements of one part are less than all elements of the other, with a third part in the middle consisting of a single element. An element with value t is chosen arbitrarily, i and j give the lower and upper limits of the segment being split. After the split has taken place a value q will have been found such that a $[q] = t$ and a $[I] \leqslant t \leqslant a[J]$ for all I,J such that $i \leqslant I<q<J \leqslant j$. The algorithm then performs operations on the two segments $a[i:q-1]$ and $a[q+1:j]$ as follows: The smaller segment is split and the position of the larger segment is stored in a lower temporary and upper temporary area. If the segment to be split has two or fewer elements it is sorted and another segment obtained from the lower temporary and upper temporary areas.- When no more segments remain, the vector is completely sorted.

4.2.7 SPARES ESTIMATION

Spares estimation is the process of ascertaining the spares requirements of a system so that it will survive a stipulated mission with a given probability. At A.S.W.E. there exists a software routine that performs spares estimation on the basis of:

i) Stock out risk

ii) Length of Mission

iii) Failure rate of each unit in the system

iv) Unit cost of each unit

v) Number of each unit type in system

This routine, called OPTCOST, optimises on the basis of cost, i.e. given two units whose effect on the system is the same, then it would be better to stock the cheaper of the two. It is virtually impossible to guarantee that a system will not run out of spares; (this would entail holding many times the number of spares for each item, clearly impractical on a ship). The measure of this guarantee is the stock out risk, set by the designers of the system.

The existing routine has three disadvantages:

i) The reliability structure of the system is assumed to be a series chain. This could lead to oversparing in systems with redundancy.

ii) The routine assumes that all elements of the system have exponential failure distributions. As this assumption is used to obtain the expected number of failures, errors will result when other distributions are specified.

iii) Very fast repair time (compared with failure time) is assumed. Repair time is ignored when the expected number of failures is estimated. This will result in oversparing in systems with significant repair times.

Information available during a RELY run could be used to improve on the assumption made by OPTCOST.

i) Information is available about the topology of the system in the form of the STATUS TABLE, section 4.2.2. Section b describes a modification to the expected number of failures based on the STATUS TABLE.

ii) Information is available about the number of failures occurring for each branch during simulation. This takes account of all distribution types as the mechanisms invoked to reach this stage e.g. PROBPOINTS, section 4.2.3, are specialised to each distribution.

iii) Repair time is taken account of automatically in ii).

a) Purpose of modules

Spares estimation is performed by three modules; SPARESINPUT, SPARESROUTES, OPTCOST.

SPARESINPUT is designed to input additional information, (additional to the reliability block diagram) about the sparing. The format of the input data is:

CODE, NT, SO, NB   for each nest to be spared
MN, UC, MS, (BN_1, NN_1, BN_2, NN_2,...) for each type

CODE determines the method used to obtain the number of failures
NT is the number of different item types (require different spares)
SO is the system stock out risk
NB is the number of branches to be spared
MN is the type number
UC is the unit cost
MS is the minimum number of spares for this type
BN is the user branch number
NN is the nest level for the branch

All the parameters are integer with the exception of SO which is real.

Certain rules exist about the data format:

i) The Branch number, BN, must correspond to a branch number of the same nest in the connection table.
ii) All the nest numbers NN must have the same nest replacement i.e. the nests to be spared must be at the same level in the system; see example in section b.
iii) NB must equal the sum of all the branches in each nest NN.
iv) There must be NT types.

In addition system switch 7 must be set to invoke the whole sparing section. Two other switches, 2 and 4, can be used together with 7 and CODE, to provide a flexible use of the sparing facility, Switch 2 terminates the run after the ROUTEANALYSIS output i.e. before simulation is started. Switch 4

		SWITCH			
	CCDE	2	4	7	
1	0	OFF	OFF	ON	SPARING. ANALYTICAL DISTRIBUTIONS
2	0	OFF	ON	ON	SPARING. ANALYTICAL DISTRIBUTIONS. SUMMARY OUTPUT
3	0	ON	ON	ON	SPARING. ANALYTICAL DISTRIBUTIONS. TERMINATE RUN AFTER ROUTEANALYSIS
4	1	OFF	OFF	ON	SPARING. SIMULATED DISTRIBUTIONS
5	1	OFF	ON	ON	SPARING. SIMULATED DISTRIBUTIONS. SUMMARY OUTPUT
6	1	ON	ON	ON	NOT APPLICABLE. RESULTS IN ERROR NO. 22
7	2	OFF	OFF	ON	SPARING. SIMULATED DISTRIBUTIONS. NO NESTING PERFORMED
8	2	OFF	ON	ON	SPARING. SIMULATED DISTRIBUTIONS. NO NESTING PERFORMED. SUMMARY OUTPUT
9	2	ON	ON	ON	NOT APPLICABLE. RESULTS IN ERROR NO. 22

figure 4.14

provides a summarised output i.e. PRINTSHORT is used instead of PRINTHISTOGRAMS. See section 4.2.8. Figure 4.14 shows the options available with the four variables. SPARESINPUT places the parameters into a vector ready for use by the next module.

SPARESROUTE adjusts the expected number of failures depending on the conditions existing in figure 4.14. The method used is described in section b.

OPTCOST performs the spares estimation as carried out by the original routine. It is slightly more efficient than the original.

b) Theoretical considerations and practical context

The method used to make allowance for topology can be illustrated using the following example.

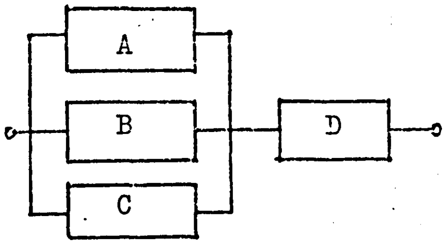


figure 4.15

The nest shown has 3 routes. A,B and C appear in 1 route each, D appears in all 3. Taking an n/m ratio (number of appearances/number of routes) gives 1/3 for A,B and C and 1 for D. A complication arises when a nest contains more than 1 element of a given type, e.g. A and D could be identical. The total n/m ratio can be obtained by adding ratios, 4/3, giving the combined significance of elements A and D in this nest. This, adjusted for one element gives 2/3. The advantage of this method is that it can be used for nested systems provided:

i) Sparing is performed on the same level of replacement
ii) The mother nest is a series chain.

Consider figure 4.16

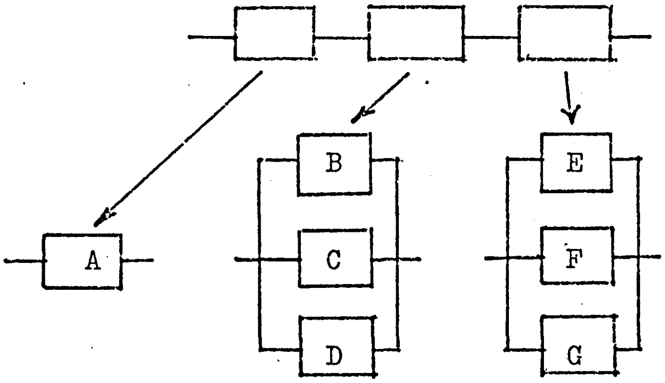


figure 4.16

The number of routes in each nest is clearly 1,3,3 respectively. Suppose A,C and G are of an identical type. If all 3 nests were combined then there would be a total of 9 routes. Hence the ratio for these 3 elements normalised to 1 element is:

$(1 \times 3 \times 3 + 1 \times 1 \times 3 + 1 \times 1 \times 3)/(9 \times 3) = 5/9 = 0.55$

Since element A appears in one route in its nest but would appear in all the routes in the other 2 nests; element C appears in 1 route out of three in its nest but would appear in the single route in the first nest and in all 3 routes in the last nest; likewise for G. The effect of increasing the number of identical elements depends upon the topology. In this case if a nest contains more than one branch of the same type, the ratio would be reduced. It was shown with A,C and G identical, the ratio is 0.55. With A,B,C and G identical the ratio drops to 0.5. Considering the example further, clearly A on its own has a ratio of 1 (the series case). But if A is removed from the group of identical types, then the ratio remains constant at 0.33. This is because B-G all have equal significance as does every element in a series chain. The simplest modification to the existing method of obtaining the expected number of failures is to multiply it by the ratio so obtained (ratio $\leqslant$ 1 thus reducing expected number of failures). Field trials are being carried out to see if this relationship is valid.

Should exponential failure and repair be used then the expected number of failures is estimated from:

$$\text{mission time} \; / \; ({}^{1}/_{\lambda_f} + {}^{1}/_{\lambda_R}).$$

This is an approximation of the expected number from two exponential distributions which can be found by recourse to queuing theory. This approximation gives about 8% error when $\lambda_f = \lambda_n$ the worst case. Should more accuracy be required then the expected number of failures can be obtained from simulation.

When simulation is used to provide the expected number of failures the mean is found for all the branches comprising each type to be spared.

To demonstrate the improvement in sparing estimation brought about by use of the RELY facilities the following system was used.

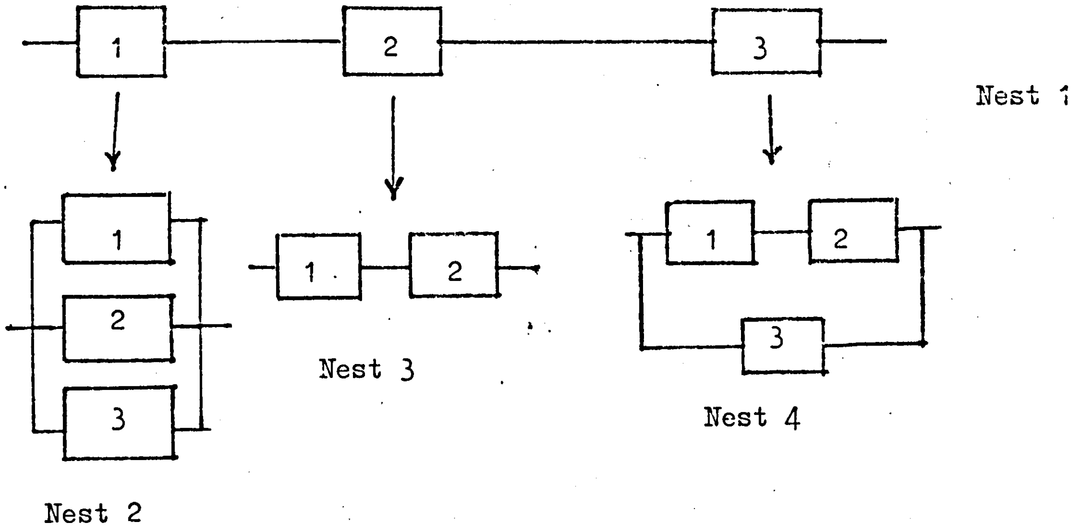


figure 4.17

Type No.	Unit Cost	BN's NN's	Failure Rate	Repair Rate
1	10	2,2,2,3,3,4,	0.015	0.007
2	12	1,2,1,3,1,4,	0.01	0.005
3	15	3,2,2,4,	0.012	0.01

The example was run in three different ways with a 100 missions of 1000 hours each.

i) Using the original OPTCOST

ii) Using analytical distributions

iii) Using simulated output

The results obtained were:

Type No.	Fails/hour i)	ii)	iii)	Spares allocated i)	ii)	iii)
1	0.015	0.0029	0.0031	54	13	15
2	0.01	0.002	0.0024	38	10	11
3	0.012	0 0023	0.0024	30	7	7
		Total cost		1446	355	387

The system has a large amount of redundancy hence the dramatic reduction between i) and ii), iii). Because repair is significant ii) differs from iii). iii) is the most accurate result. A re-run of iii) with 500 missions (instead of 100) gave estimates for types 1,2 and 3 of 15,10 and 8 spares respectively.

c) Structure of the modules in terms of information flow

SPARESINPUT is entered after INPUTROUTINE, section 4.2.1. The four nest parameters are input and used to calculate the bounds of the input vector. A loop is set up on each row of type data to input and check. Error checks are performed to ensure that the parentheses match, i.e. no parameters have been omitted. When the vector is complete, the housekeeping module is requested to transfer the vector on to the data file.

SPARESROUTES is entered after simulation. A special jump is instigated by control if no simulation is performed i.e. switch 2 on. Housekeeping is requested to transfer the vector from the data file into store. In addition the status table and connection table are obtained. Because a number of nests may be involved (the same type may comprise branches of different nests) it is necessary to pre-process the vector so that the complete status table can be used before the next status table is obtained. This is achieved by scanning the vector for branches in the current status table (nest) and building the ratio for each type. The number of routes in each nest is also compiled. When this process has been completed the next phase is to gather all ratios of a given type and obtain the complete ratio by dividing by the number of branches in this type. This ratio is stored and the module left.

OPTCOST is entered after SPARESROUTES and one of two routines are entered depending on whether analytical or simulated distributions are used. If the former is the case then the expected number of failures is computed by multiplying the ratio by the failure rates. If the latter is the case then the simulated output is obtained by the housekeeping module and the mean failure rate for each type obtained. This is multiplied by the ratio to give the expected number of failures. The original OPTCOST routine is then entered. On completion, the sparing estimates are output in a tabular format.

d) Algorithms used

The algorithm used in OPTCOST for the optimisation process is security classified by A.S.W.E. and hence no details can be given here.

4.2.8 OUTPUT

The output from a RELY computer run falls into one of two categories:

i) Results of an analysis or simulation

ii) Error messages

Specific modules are responsible for type i): ROUTEANALYSIS is responsible for analytical information about the structure of the system being simulated.

EVENTSDATA is responsible for outputting the number of branch and system failures and repairs experienced by each branch.

PRINTSHORT is responsible for the summarised output required by remote terminal users.

PRINTHISTOGRAMS is responsible for the generation of histograms from the CELLMAP output.

The figures in chapter 7 (Design examples) illustrate the output formats.

Type ii) output can be generated from any point in the program. The user guide gives details of the reasons for each message see figure 6.4 for format.

The selection of output modules is governed by control which itself is subject to user choice. The section on control, 4.2.10 describes in more detail the mechanisms used.

a) Purpose of modules

The module specification is taken in the above order. ROUTEANALYSIS is designed to accept:

i) The input connection table

ii) The probability values table

iii) The status table

From these tables:

i) The input data is displayed in an ordered format.

ii) The nesting structure is provided.

iii) The structure of each route is displayed.

iv) Summarised information is provided about the routing system for the whole system.

EVENTSDATA is designed to accept a table containing, for each branch:

i) The number of failures experienced by the branch.

ii) The number of repairs experienced by the branch.

iii) The number of systems failures caused by the branch.

iv) The number of system repairs caused by the branch.

This information is output in tabular form for each nest immediately after simulation.

PRINTSHORT is designed to accept the 30 word vector output by STATISTICAL, section 4.2.5 and output, in a reduced format suitable for a terminal, all the parameters generated by STATISTICAL. PRINTHISTOGRAMS is designed to take the 5, 10 box cell vectors output by CELLMAP, section 4.2.5 and convert them into lineprinter histograms. Additionally the module provides all the summarised parameters as output by PRINTSHORT.

b) Theoretical considerations and practical context

All the output is normally steered to the lineprinter. However, when running under an advanced operating system such as GEORGE 3, found on large 1900 systems, the output is placed in specified output files. The destination of the output is then controlled by the job description. It is the use of this technique that enables PRINTSHORT output to be displayed on a terminal.

As mentioned in section 3.1, the only output peripheral that is used is the lineprinter. Accordingly the output is formatted to a standard lineprinter page of 60 rows of 120 print positions. This is important for tables in order to ensure alignment; (additional characters over the last print position appear on the next line.) It is however, vital for histograms because any line overflow would make nonsense of a pictorial representation.

c) Structure of the modules in terms of information flow

ROUTEANALYSIS is entered and the housekeeping module is requested to obtain the Connection table, Node directory, probability table and Status table for the current nest. The printing then begins. First the nesting structure is displayed. Then each set of branch parameters is displayed, the branches having been sorted into numerical order. The number of routes required is output followed by the nest number and then the branches comprising each route are given. Finally a summarised table showing:

i) the number of routes each branch appears in

ii) the number of routes containing n branches where $1 \leqslant n \leqslant$ no. of branches, is displayed.

EVENTSDATA is entered and the contents of the simulation events table is output branch by branch.

PRINTSHORT is entered and the following parameters are output.

Number of missions, mission time, the five output parameters (MTBF, MTFF, SFR, MRT, Availability) and the minimum, maximum and standard deviation of each of the five.

PRINTHISTOGRAMS is entered and the y axis maximum calculated from either:

the largest cell

or:

the user selected maximum

A new page is thrown and a loop is then entered which outputs the boundaries of the axis and the cells. The cell content (number of items) is displayed above the top of the cell. The y axis scaling is printed on every 5th line and the x-axis scaling on every 2nd cell boundary. When printing histograms on a line-printer, the entire line has to be built up in the printer buffer. This involves fairly complex control. After the x-axis labelling the PRINTSHORT parameters are displayed.

d) Algorithmsused

No special algorithmsare used because the majority of the output is rearrangement of internal tables.

4.2.9 HOUSEKEEPING

Housekeeping describes the general manipulations necessary on data areas in order to free CONTROL from the restriction of requiring to keep track on the current location of all the data held on backing store.

In RELY these manipulations are carried out by a single module, MTFILE. The full implications of MTFILE in relation to the structure of the program are described in section 4.2.

a) Purpose of modules

MTFILE is designed to:

i) Open a scratch tape (unused tape) for the data file.

ii) Open an existing data file.

iii) Transfer a record to the data file (write operation).

iv) Transfer a record from the data file (read operation).

v) Close the data file.

b) Theoretical considerations and practical context

The module contains a map of the layout of the tape. This map is a permanent vector in store and therefore has to be of a fixed size. The size first chosen allowed a maximum of 400 records. This was found to be too small for long simulations where the volume of data generated was large. A user option was provided where the size of the data file could be increased to 1000 records. The drawback of having a large map area is the consequent reduction in available data space.

Because the module is in the form of an ALGOL procedure, steps have to be taken to ensure that the map vector is not destroyed on exit from the procedure. There are only two ways that this can be achieved:

i) Making the map area an 'OWN' array.
ii) Making the map area global to the whole program.

The first method is the best one from programming and structural considerations. The 'OWN' variable concept is a device used in ALGOL to retain the contents of variables outside the block in which they were declared (a procedure counts as a block). Variables are declared within blocks and are normally only valid within the blocks or inner blocks in which they were declared. Unfortunately the use of 'OWN' variables in 1900 ALGOL creates an additional problem. All data space is contained in an area of store at the end of the program called the stack. A request for more space, (e.g. a dynamic array declaration), than the stack has would cause the size of the stack to be increased. As soon as an 'OWN' variable is requested the stack is used in a different manner. Figure 4.18 shows the situation. The stack is fixed in size from the outset and all is well until the two areas meet, at which point the program fails. It is therefore necessary to make the stack as

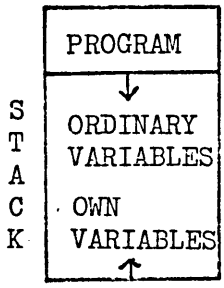


figure 4.18

large as possible thus removing the dynamic data space facility of ALGOL.

The second method has the drawback of requiring the global area to be declared in the MTFILE procedure and listed in the parameters at call time. If MTFILE is used inside another module then that module must declare the area. But as this method is the only viable alternative it was decided to structure the map in this manner.

c) Structure of the modules in terms of information flow

In order to understand the mode of operation of MTFILE it is necessary to describe the data structure used.

Arrays:

RECNAME. This array contains the names of the records. When a new record is written a check is made to see if the record name already exists. If not it is added into RECNAME. Names are stored 4 characters to a word, new names starting at the next vacant word.

	1				2				3				4				5
RECNAME	J	O	E		B	I	L	L	Y		`		S	I	D		

RECNAMEP: This array points to the start address of the record names in RECNAME. The corresponding RECNAMEP for the RECNAME shown would be

	1	2	3	4	5
RECNAMEP	1	2	4	5	

Thus the 1st name (JOE) starts in word 1. The 2nd name starts in word 2, the 3rd in word 4 and the 4th in word 5.

MAP: This array contains a map of the data file. Each element represents a record and the contents of the array represent the subscript of the RECNAMEP array which relates to the record.

A typical map could be

	1	2	3	4	5	6	7	8	
MAP	1	1	2	1	3	1	2	1	

Thus the 1st record is of name 1 (which is JOE). The 5th record has the 3rd name in the RECNAME list, which starts at word 4 of RECNAME (SID).

Variables:

NUMBER: This contains the number of records on the tape.

TAPEPOS: This shows the current position of the tape. 0 = beginning.

RECNAMESUB: This shows the next free name in the RECNAMEP array.

Assuming the tape is positioned to write away a new record the values of each of the three variables in the example shown would be:

NUMBER = 8
TAPEPOS = 8
RECNAMESUB = 4

The entry parameters in the MTFILE call are:

MTFILE (A,B,C,D,E,F,G)

where A = An integer parameter to control the mode

- 0 = Write
- -1 = Read
- -2 = Open new file
- -3 = Close file
- -4 = Open existing file

B = An integer parameter to determine which occurrence of a given record name is required when reading.

C = An integer array to hold the information being transferred to or from the record.

D = A real array to hold the information being transferred to or from the record.

E = An integer marker to determine which of C or D is to be used.

E = 0 C is the array to be used
E ≠ 0 D is the array to be used

F = The name of the record being written or read.

G = The directory of the tape.

The action taken depends on the mode (A) chosen.

Writing to the file: The record name (F) is examined. If it already exists then it is not added to the list in RECNAME. The next vacant element in Map is filled with the number of the record name. The tape is moved to record number (NUMBER-TAPEPOS). The contents of C or D depending on E are written. NUMBER is increased by 1 and TAPEPOS is set equal to it.

Reading from the file: The record name parameter F is examined. The corresponding subscript of RECNAMEP is found. Parameter B is examined and search is made of MAP to locate the Bth occurrence of the number of the record name, e.g. suppose it were required to obtain the 5th occurrence of record JOE in the example. JOE is the 1st record name and therefore has a number 1. A scan of MAP shows that the 5th occurrence of number 1 is the 8th record. The tape is moved to (8- TAPEPOS -1) and the record read and placed in either C or D depending on E.

Opening a new file: A scratch tape is relabelled as the data file, NUMBER & TAPEPOS set to zero, RECNAMESUB set to 1 and MAP cleared.

Closing a file: The file is closed by writing (using mode 0) the actual map as record. The final call of MTFILE then closes the tape and rewinds it.

Opening an existing file: The file tape is opened and a search made for the Map. The map was written with a unique record name so the map is not required for this operation consequently the normal software read routine can be used. When the map has been read into store it is adjusted to remove all trace of the previous writing of the map (as this was the last record written to the tape before it was closed). This involves reducing NUMBER by 1, setting TAPEPOS=0, setting RECNAMESUB to point to the start of the unique map record name (so that it can be overwritten) and clearing the record number from MAP. When this 'bootstrapping' process has been completed the data file can be used normally.

d) Algorithms used

No special algorithms are used.

4.2.10 CONTROL

The control section of the program is responsible for data flow organisation. It calls the applications modules in a required order to suit the user need for the current run. Ideally control should consist of a list of module calls with decision statements interspersed. However, this is not a practical possibility as machine software does not permit this elegance of structure.

The structure of control has the following component parts:

i) A list of program description statements.
ii) A declaration of all the procedures (modules and standard routines)
iii) The body comprising the module calls and necessary control software.

Examining these in detail:

i) Certain statements are necessary in order to inform the compiler of the actions that need to be taken. These include such things as:

   the name of the module library
   the overlay structure
   the disc file to be used to store the program.

ii) A declaration of all procedures is needed so that the named libraries can be searched for the body of each procedure.
iii) The body can be considered as a list of actions and decisions.

Check date; The program is issued every three months and a purge date ensures that obsolete copies do not proliferate.

Open data file; A test is made to see if an existing file is to be extended or a new file to be created.

Open work file;

Call INPUTROUTINE; (4.2.1) Input data

Call SPARESINPUT; (4.2.7) If spares estimation has been invoked then input spares data.

Free input device;

Open work disc area; .

Call ROUTESEARCH; (4.2.2) Find routes in nest

Call PROBPOINTS; (4.2.3) Fill probability tables

Transfer completed tables to data file; The previous two steps are carried out on each nest. On completion, the finished tables are written to the data file.

Call SPARESROUTES; (4.2.7) Perform spare routing if required

Call ROUTEANALYSIS; (4.2.8) Output all route tables

Transfer all tables for simulation from data file; Set up status table, probability table and various work arrays.

Call TIMESEQ; (4.2.4) Performs time simulation

Call EVENTSDATA; (4.2.8) Output branch histories

Invoke nesting control; This control is responsible for simulating a nested structure in the correct order, see chapter 2.4.

Call TABULAR; (4.2.6) This is called by the nesting control and is responsible for producing probability tables for nested structures

Call OPTCOST; (4.2.7) This performs the actual spares estimation if necessary.

Call STATISTICAL; (4.2.5) This obtains the simulation statistics for the top nest.

Call CELLMAP; (4.2.5) This obtains the cell sizes for the ten box histogram

Call PRINTSHORT; (4.2.8) This is responsible for remote terminal output and is called if necessary.

Call PRINTHISTOGRAMS; (4.2.8) This outputs all the histograms.

Close all files;

5. RELIABILITY OF MECHANICAL ENGINEERING SYSTEMS

5.1 INTRODUCTION

In the early stages of this project, the systems used for field trials were all electronic equipment systems such as communications networks. It was suggested by the sponsors, A.S.W.E., that it would be profitable to try to apply the prediction method to predominantly mechanical systems. As the prediction technique is a general one, in principle there seemed to be no limitations on the extension of the method to mechanical systems. However, a number of practical problems were revealed when a feasibility study was carried out. It became clear that, in most practical cases, the general methods of reliability prediction can only be applied to mechanical engineering systems after the solution of certain modelling problems particular to such systems. This chapter aims to outline the nature of these problems and indicate the need for further work in this area. The nature and volume of this work is such that it was considered unreasonable to include it within the bounds of this project.

5.2 MODELLING MECHANICAL ENGINEERING SYSTEMS

As shown in earlier chapters of this thesis, when carrying out a reliability analysis of any system it is necessary to supply:

i) A reliability block diagram modelling all relevant aspects of the physical system.

ii) Failure and repair distributions that characterise the performance of each element in the reliability block diagram.

Both of these may be problematic in a mechanical engineering system:

i) In a mechanical system there are three main failure modes, free, locked and runaway. Analysis of failure modes and their effects on this type of system results in a very complex pattern of interactions between the system components which, in turn, results in a very involved block diagram with a large number of interdependences. Figure 5.1 shows a simple example relating to a rotating waveguide joint. This assembly has been used by A.S.W.E. in order to see how complex the modelling problem becomes with a fairly simple physical structure. The resultant model became practically unmanageable and could not be used for obtaining quantitative reliability figures for the system.

Experience indicates that resolving a mechanical system into a reliability block diagram at the present time requires detailed mechanical engineering knowledge of the system under consideration in order that a full failure modes and effects analysis may be carried out. A skilled mechanical-engineer is therefore needed to perform the modelling, unlike the case with an electronic system where the interdependences are not so great, and the modelling problem is relatively straightforward.

ii) Mechanical systems tend to be purpose built as individual items and therefore there is little standardisation of fabrication and assembly processes. Even if the individual items comprising an equipment are standardised, and hence provide a sufficient population for statistical analysis on the basis of large samples, the methods of assembly can be critical in determining the overall performance of the system. It is difficult to isolate for test purposes the component parts from their assemblies. Limited time and availability of specialised prototype assemblies prevents the accumulation of satisfactory volumes of test data. All these factors hinder obtaining accurate failure rates and distribution types.

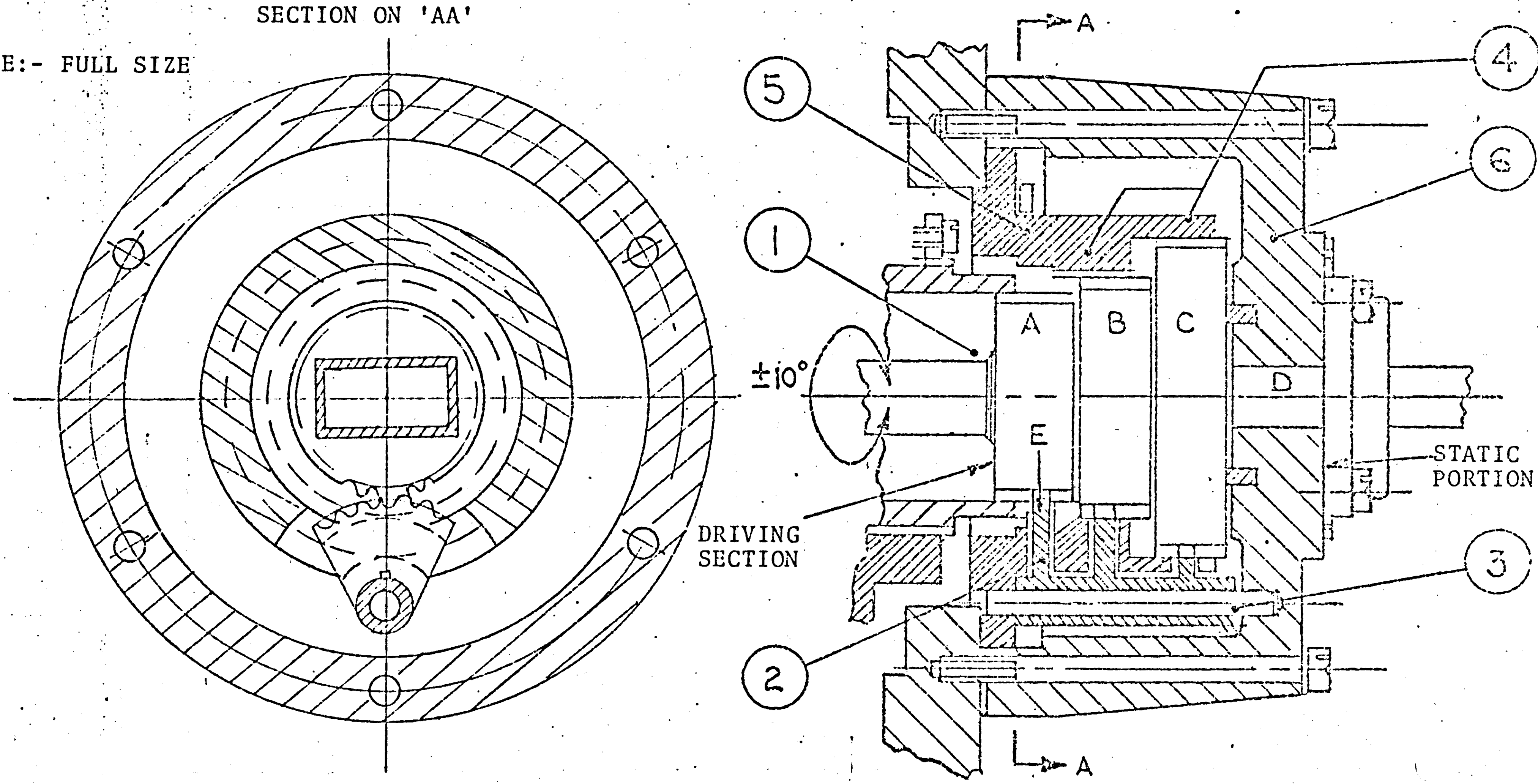
In the rare cases when data is available from standard sources, very little confidence can be placed on it due to the wide differences between the data quoted for the same item by alternative sources. Figure 5.2 shows some data for the assembly of figure 5.1 (reference A.S.W.E. failure rate tables.) The corresponding items detailed in MIL 217

(U.S. Defence publication) differ by factors of 30 from the A.S.W.E. list.

5.3 CONCLUSIONS

As a result of these problems it was decided that no further work could be carried out in this area of mechanical reliability prediction. Although the prediction method is applicable to any system, the state-of-the-art of mechanical engineering systems does not usually allow the direct application of the suggested modelling procedures. Thus, further research will need to be carried out by mechanical engineers into modelling: this work will necessitate development of:

- methods of partitioning mechanical engineering systems into minimally interdependent components.
- methods of data collection and analysis under conditions of scarce and unreliable data.



ROTATING W/G JOINT

figure 5.1

1 SOLDERED JOINT
2 COMPOUND GEAR PINION
3 GEAR SPINDLE BEARING
4 P.T.E.E. BEARING
5 INNER BEARING HOUSING
6 MAIN HOUSING

OPERATION

THE JOINT CONSISTS OF 4 SECTIONS. SECTION A IS ATTACHED TO THE MAIN PITCH SPINDLE AND MOVES WITH IT. B AND C ARE DRIVEN FROM A BY MEANS OF GEAR E. THE GEAR RATIOS OF B AND C AND THE GEAR PINION VARY SO THAT THEY MOVE AT DIFFERENT SPEEDS. D IS FIXED, AND THE TOTAL MOVEMENT OF A, RELATIVE TO D IS 20°, EACH SECTION BEING DISPLACED 5° RELATIVE TO THE PREVIOUS ONE.

SEMI-ROTATING WAVEGUIDE JOINT. INFORMATION TABLE

G.A. Item No.	Item Description	Remarks	Number Off N.	Failure Mode Score			Proportioning Factor	Failure Rate ppmh.	f.
				Locked	Free	Runaway			
1	Waveguide Soldered Joint. (Input)	-	2	1	2	1	4/2	0.007	0.0035
1a	Waveguide Soldered Joint. (Output)	-							
2	Geared Sections (Rack & Pinion)	Partial Rotation	3	1	-	1	2/3	0.20	0.30
3	Multiple Pinion Bearing Spindle (Plain Bearing)	-	1	1	-	1	2/1	0.50	0.25
4	Plain Bearings	-	2	1	-	-	1/2	0.50	1.00
5	Machined Inner Housing	Static, Extra Reliable	1	-	1	-	1/1	0.41/20	0.02
6	Machined Main Casting	"	1	-	1	-	1/1	0.41/20	0.02

Note. Items 5 and 6 are considered to be extra reliable because of their static nature, and therefore to have a lower than normal failure rate. i.e. 1/20th. the rate quoted in the list.

figure 5.2

6. DOCUMENTATION OF PROGRAM

The documentation of the reliability program RELY is designed as a hierarchical structure starting with elementary user information and gradually resolving into detailed information about the program statements. The documentation may be considered on three levels of resolution:

i) The user level. This level contains sufficient information to enable a user to use the program without needing to know structural details. Specifications of the modules are given in a user orientated format.

ii) The system level. Information is provided about the overall program structure and its use of 1900 system software. The documentation at this level enables a version of the program to be constructed from a library of modules.

iii) The program level. The actual operation of each module is described. This level is the most detailed and would enable actual changes to be made in a module.

All documentation examples are taken from the actual manuals.

6.1 THE USER LEVEL

Two manuals exist at this level.

i) The User Guide.

ii) The Introductory Guide.

User Guide:
This guide is a formal statement of the facilities of the program and operations required on the part of the user. It is a modular document which permits modifications, additions and deletions with each new issue of the program ensuring that users have up to date documentation. The user guide consists of:

i) The specification of the current issue of the program.

ii) The description of the users guide.

iii) Program instructions.

iv) The specification of the control module.

v) List and specifications of modules associated with input.

vi) Specification of each module.

vii) General error reports.

The information contained in each part of the guide is as follows:

i) Each issue has its own publication and expiry date. In addition, the size of the program and the modules in the current issue are also given.

ii) The description of the user guide, includes the index and informs the user how to achieve maximum efficiency in the use of the guide.

iii) The format is shown in figure 6.1

iv) The particular control module for the current issue is specified as a list of module calls and intermodule operations.

v) Because the formal description of input parameters is contained in the module specifications vi),it is necessary to list the modules responsible for input so that no part of the input structure is omitted by mistake. Using this format enables module revision documents to be incorporated without affecting the overall structure. Should a new module be included which requires input then a document of type figure 6.2 would be issued together with a revised input module list showing the amended order, figure 6.3

vi) Each module contained in the current issue has a specification document. An example is shown in figure 6.2 The specification sets out:

i) The mark number.

ii) The module issue date.

iii) The size.

iv) The programming language.

v) A description of the module.

vi) Error reports generated by the module.

vii) Figures 6.4, 6.5 illustrate the format used. The messages are those which appear on the output. These refer to the module where an explanation of the error can be found. Thus occurrence of error No.17 (detected by the message) figure 6.5, would be caused by a missing parenthesis, figure 6.2.

Introductory Guide

This guide is designed to give advice on the best use of the program parameters depending on the problem being simulated. The introductory guide consists of:

i) Specification of the program.

ii) Modelling examples.

iii) Recommendations on input parameter choice.

iv) Recommendations on output parameter choice.

v) Recommendations on use of the nesting facility.

vi) User control diagram.

vii) Recommendations on the use of spares estimation.

The information contained in each section of the guide is as follows:

i) A brief description of the simulation technique is given.

ii) Suggested methods for modelling standard situations are given. Chapter 2 describes the models in detail.

iii) The format is shown in figure 6.6.

iv) The format is similar to figure 6.6.

v) Advice is given on the design of a nesting structure for handling large systems. Chapter 2.4 describes the nesting concept.

vi) The user control diagram is a flow chart showing the effect of the control switches on the mode of operation of the program.

vii) The sparing facilities are described together with a table showing the various options available with the sparing facility.

6.2 THE SYSTEMS LEVEL

One manual exists at this level, the systems organisation manual. The documents comprising this manual are:

i) System organisation.

ii) Source master segment on magnetic tape.

iii) Object code dump on magnetic tape.

iv) Module library on magnetic tape.

v) Module library organisation.

The information contained in each section of the manual is as follows:

i) The compiling system used is described and a basic outline given of the process of compilation of RELY.

ii) The steps necessary in producing an object code copy of RELY on any suitable 1900 computer are described. The parameters required to produce a magnetic tape copy of the master segment are given. Figure 6.7 shows the format.

iii) The parameters required to produce a magnetic tape object code dump of RELY are given.

iv) The parameters required to produce a magnetic tape copy of the module library are given.

v) The organisation of the module library is described together with the precautions that have to be taken when deleting and adding modules.

6.3 THE PROGRAM LEVEL

At this level it is possible to take the source listings and make changes to the modules. The purpose of each program statement is described as well as the interface (procedure parameters) with the control section. This level would normally only be of interest to a programmer. Each document, whose format is shown in figure 6.8, has two main sections.

i) The procedure heading specification. This contains a list of all the parameters together with their function.

ii) The statement by statement description of the module.

Operating Instructions

These operating instructions apply to a manually run version. If an operating system is to be used then these instructions must be incorporated into the job description pertaining to the system in use.

Load the program into store by means of a find	FI≠≠RELY≠≠(ABCD)
Set switches N=1 Input on paper tape (cards default) N=2 Terminate run after route analysis output N=3 Enter simulation with data already assembled N=4 Provide summary output N=5 Increase size of data file N=6 Provide complete output of previous run N=7 Perform sparing N=8 Repeatable random sequence required	ON≠≠RELY N
Activate the program	GO≠≠RELY 20
At message	0≠≠RELY:-HALTED ST
Make more store available and	GO≠≠RELY
Successful termination	0≠≠RELY:-HALTED AH
Errors occurred	0≠≠RELY:-HALTED EE
Terminate program after an error	GO≠≠RELY 28

figure 6.1

Module SPARESINPUT

MK 1 FEB 75 SIZE 315 ALGOL

This module inputs data for spares calculations. The format of the input is:

Spares Parameters CODE, NT, SO, NB

CODE	= Expected number of failures code	(Integer)
NT	= Number of Types	(Positive Integer)
SO	= Stockout Risk	(Positive Real)
NB	= Number of Branches	(Positive Integer)

Type Parameters MN, UC, MS, (BN, NN, BN, NN, ...)

MN	= Type identification number	(Integer)
UC	= Unit Cost	(Positive Integer)
MS	= Minimum Number of spares	(Positive Integer & 0)
BN	= User Branch number	(Positive Integer)
NN	= Nest No of BN	(Positive Integer)

The type parameters are repeated according to NT.

CODE = 0 (Parameters E, H of PROBPOINTS, 6.2.1. used to estimate expected number of failures)
CODE = 1 (Simulated Branch events used for estimating expected number of failures)
CODE = 1 (Simulated Branch events used for estimating expected number of failures but no nesting is carried out)

Error Conditions

General Errors are given in the error report section. Error reports originating from this module are:

17 The (has not been encountered after MS,
105 Results from 17. Abandon run.

figure 6.2

INPUT DATA

Certain modules are responsible for input of parameters. They are:

Inputroutine;
Probpoints;
Conparin;

A detailed description of the order and specification of parameters can be found in the module description. The order can be summarised as follows:

Nest Parameters Branch Parameters . . . Terminal Parameters Delimiter Nest Parameters Branch Parameters . . . Terminal Parameters Delimiter . . .	Inputroutine & Probpoints
Spares Date Type Data	Sparesinput (if switch 7 is on)
Control Parameters	Conparin

figure 6.3

Error reports and Exception conditions

Errors can be caused by:-

i) Incorrect data which is mispunched
ii) Incorrect data which is numerically invalid
iii) Overflow of procedures due to chance combinations
iv) Hardware failures.

The 1st can be detected by the input module and error reports are generated by the module.

The 2nd are more difficult to trap and can cause a system error. These are generated by the error reporting system incorporated in any ALGOL program and allow the user to send a report to the designer of the version of the program being used.

The 3rd type of error can be caused by the designer of the version of the program being used not implementing sufficient safeguards in the purpose built control module.

Lineprinter Error Messages

Error No.	Message	Module
1	ERROR IN SPECIFICATION PARAMETERS (N)(M)	INPUTROUTINE
2	NEST TERMINATOR ENCOUNTERED INSTEAD OF / WITH BLOCK NEST (N)(M)	INPUTROUTINE
3	TERMINATOR HAS NOT BEEN ENCOUNTERED ON A BRANCH FOR BLOCK NEST (N)(M) BRANCH IS PROBABLY (P)	INPUTROUTINE INPUTROUTINE
4	INCORRECT NUMBER OF NUMBERS FOR BRANCH (B) IN BLOCK NEST (N)(M) (P) NUMBERS WERE INPUT FOR THIS BRANCH	INPUTROUTINE
5	TWO IDENTICAL BRANCHES FOR BLOCK NEST (N)(M)	INPUTROUTINE
6	NON-EXISTENT TERMINAL NODES FOR BLOCK NEST (N)(M)	INPUTROUTINE
7	INCORRECT TERMINATOR AT END OF BLOCK NEST (N)(M) NO MORE DATA HAS BEEN ACCEPTED INCLUDING ABOVE MENTIONED BLOCK NEST	INPUTROUTINE
8	NEST WITH ERROR HAS BEEN IGNORED	INPUTROUTINE
9	RECORD NAME DOES NOT EXIST	MTFILE
10	CANNOT FIND SPECIFIED RECORD	MTFILE

Console Error Messages (And Exception Conditions)

Note: If an operating system is being used then these messages will appear on the lineprinter.

100	0 # RELY	HALTED 99	MTFILE
101	0 # RELY	HALTED NS	TIMESEQ
102	0 # RELY	HALTED FINISHED	FINE
103	0 # RELY	HALTED GO RELY AT N	ENTER
104	0 # RELY	HALTED X(N)	INPUTROUTINE

figure 6.4

Errors contd.

Error No.	Message	Module
11	Nest Level LE Nest Level Replacement	VALIDATE
12	Must Be Only One Nest Level Replacement Number	VALIDATE
13	Insufficient Failures In Nest (N)	TABULAR
14	Insufficient Repairs In Nest (N)	TABULAR
15	Insufficient Failures and Repairs In Nest (N)	TABULAR
16	Non Compatible Block Number At Level (N)	TABULAR
Error Type 80	Caused by disc area being too small, i.e. current nest has too many branches and hence routes. Either reduce size of nest or switch on 5 (see operating instructions) to increase size of data area	
105	0~~==~~ Rely Halted 80	SPARESINPUT SPARESROUTES
17	(Not Encountered For Type (N)	SPARESINPUT
18	Non-Existent Nest No In Spares	SPARESROUTES
19	Different Replacement Levels in Spares	SPARESROUTES
20	Branch Not In Nest Level For Spares	SPARESROUTES
21	Non-Existent Branch No For Spares	SPARESROUTES
22	Cannot Spare Without Simulation	OPTCOST
23	More Than 180 Expected Failures	OPTCOST

figure 6.5

Choice of Input Parameters:

The user must specify the number of branches (elements) and the number of nodes contained in the structure. The length of simulation increases with the number of branches and with the complexity of the structure which manifests itself by a large number of routes through it. A rough guide to the complexity of the structure is the ratio of the number of nodes to the number of branches. The advisory figure of maximum size of structure is 30 branches.

The number of branches also influences the storage space required by the program. The space requirements may be judged by the following information:-

(i) Every branch requires 200 words for its probability information. Thus, 30 branches require 6000 words.

(ii) Every route requires b words where b is the number of branches. Thus, 30 routes require 900 words in a 30 branch system.

Control can be exercised over the number of routes required for the system to function. This facility has been designed for n-out-of-m but it can be used in a broader sense for a complete system. Care must be exercised over this because in a complex system with the number required approaching the number of routes actually in the system, element failures and repairs become system failures and repairs. Thus, the amount of stored simulated output rises sharply. A stage is reached where much more time is required to compute the statistics than to actually perform the simulation. If operating systems permit the use of disc files in place of actual magnetic tapes then it is possible to fill the file rapidly. A typical 2400' tape can hold about 4M words whereas a disc file is set at an upper limit of about 256k words.

The choice of mission time is important in determining the nature of the output. If a system fails and is not repaired then it is clearly pointless having a long mission time:- it is only necessary to make it slightly larger than the longest first failure. Repairless elements are modelled by means of a switch for the repair distribution with an independent switch time greater than the mission time; (events occurring after the mission time are not considered). If the times between failure are important then it is necessary to have a high mission time compared with the average time between failure.

The number of missions is an extension of the mission time because the volume of simulated data will be controlled by the number of missions. However, the number of missions directly controls the times to first failure and availability. Too few missions will not provide valid figures for these.

Both the block replacement number and nest level are dummy parameters at this stage and 1 should be inserted for each one.

figure 6.6

Source Master Segment on Magnetic Tape

It is sometimes necessary to produce an object code copy of the program on the machine on which it is to be used. The need originally arose because of the non compatability of the 1900 overlay package between different environments.

The sequence of operations required are:

1) Produce magnetic tape containing body of master segment using editor program XKYA.

2) Dump SUBGROUPCHRI, the private library of semi compiled segments onto magnetic tape.

3) With the destination machine, load SUBGROUPCHRI onto a suitable disc file.

4) Compile and consolidate the master segment from magnetic tape preceded by program description statements on cards or paper tape.

5) Load the program into a disc file prior to running (necessary because RELY is overlaid).

XKYA parameters:

```
OFW SCRATCH TAPE
RENAME RELYSOURCE (1)
OSF SUBFILE1, B2A4
BLOCK, 128
LINEDIT
*ALTER
↕ MASTER SEGMENT SOURCE CARDS
* END
CSF
FINISH
```

The master segment source cards comprise the statements between the first 'BEGIN' and last 'END' including the 'BEGIN' and 'END'.

XPEK to produce dump of SUBGROUPCHRI:

The library program XPEK is used to copy the subroutines to magnetic tape. Its parameters are:

```
(TAB) IDF (TAB) (TAB) SUBGROUPCHRI, 1
(TAB) OMT (TAB) (TAB) SUBGROUPCHRI, 1,9Ø
(TAB) COM (TAB) (TAB) MT
(TAB) F
```

The program description statements will be those used for a normal compilation including a 'READ FROM' (MT, RELYSOURCE(1).SUBFILE1)

figure 6.7

Module Validate

Procedure Heading

VALIDATE (DIRECTORY, SIZE, MARKER)

DIRECTORY: The Directory of the tape size [1: Nests, 1:3] Integer array

SIZE: The number of nests being simulated : Integer

MARKER: Indicates a non recoverable error to control. Integer

Structure

A comparison between the 2nd and 3rd column of directory is performed to ensure that 2 col > 3 col (Actual level must be greater than replacement level). On detection of invalid number, error message 11 is generated, marker set to 1 and the procedure left.

A check is made to see if there is more than 1 zero in the 3rd column. This is the replacement level at the level of resolution and is used to trigger the statistics. On detection of more or less than one zero error message 12 is generated, marker set to 1 and the procedure left.

The nesting structure is output in tabular format with nest level, nest replacement number and block number.

figure 6.8

7. DESIGN EXAMPLES

The purpose of the project, in the main, has been to investigate methods for reliability prediction that would be suitable for computerisation. The outcome has been a fairly complex, modularly structured, computer program. As one means of assessing the program it was decided to show here design exercises in three different areas:

i) A shipborne communications system.

ii) A private hospital.

iii) A control system for an electronically steered car for disabled drivers.

7.1 A SHIPBORNE COMMUNICATIONS SYSTEM

This design example was supplied by the sponsors A.S.W.E. and was described in a paper presented at C.A.D. 74 (see appendix i). The system is fairly straightforward. An aerial, with a standby, is connected to an amplifier, with a standby, via a multicoupler. A single power supply is used and this is connected to a distribution point which feeds three separate receiving stations or consoles. The three stations have one, two and three receivers, respectively, as indicated on the reliability block diagram, figure 7.1. Each station has its own power supply. This physical configuration leads to a reliability model with standby switches whose operation is quite fast, bringing the standby rapidly into service if needed. The design problem is to improve the reliability of the system without increasing the cost. Certain constraints exist, namely:

i) The system configuration cannot be altered.

ii) Balancing the performance of each element, to ensure that the relative contribution of each element towards system reliability is the same, is not possible in practice.

iii) Repair times cannot be improved.

RELIABILITY BLOCK DIAGRAM OF SIX CHANNEL COMMUNICATION SYSTEM

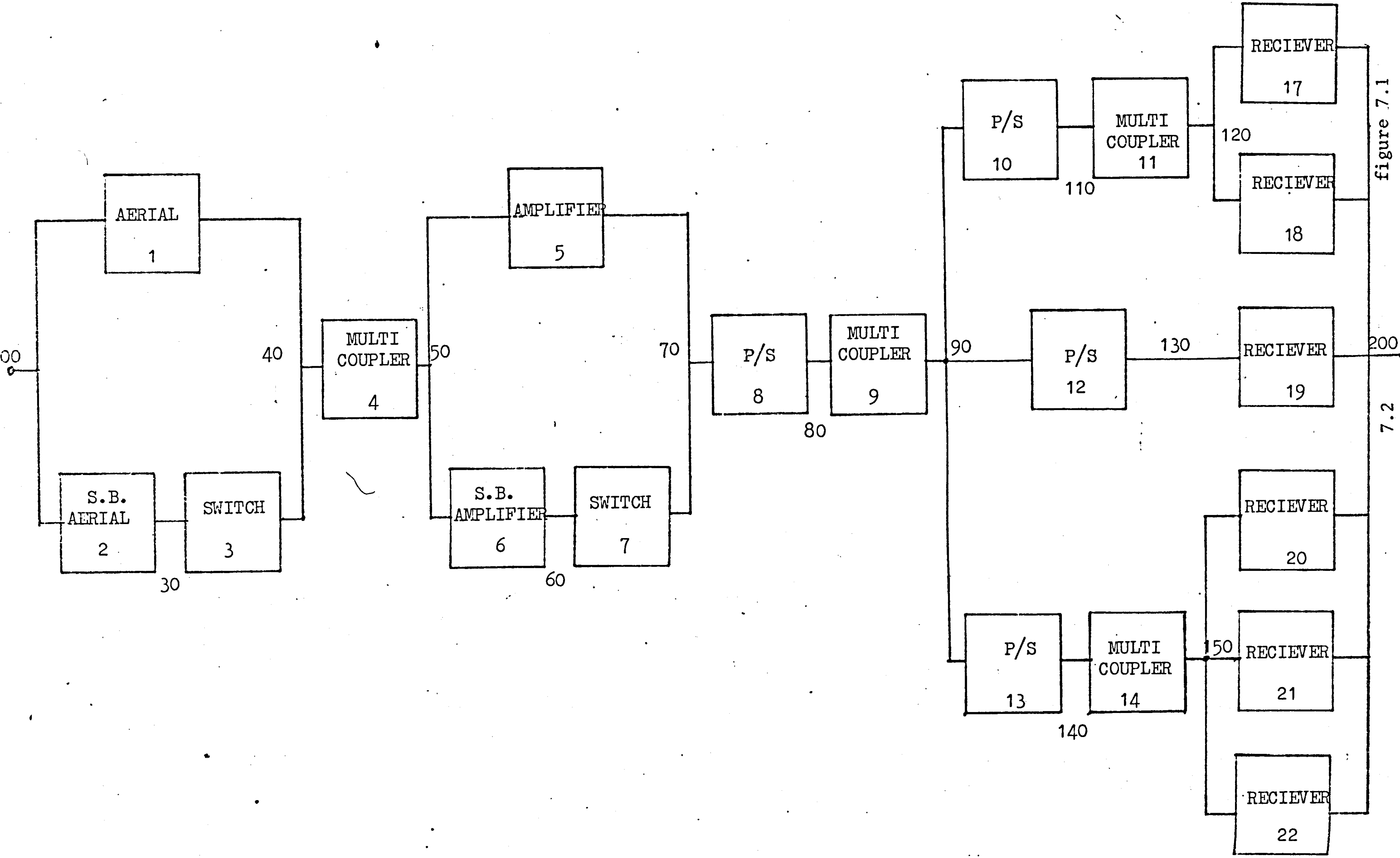
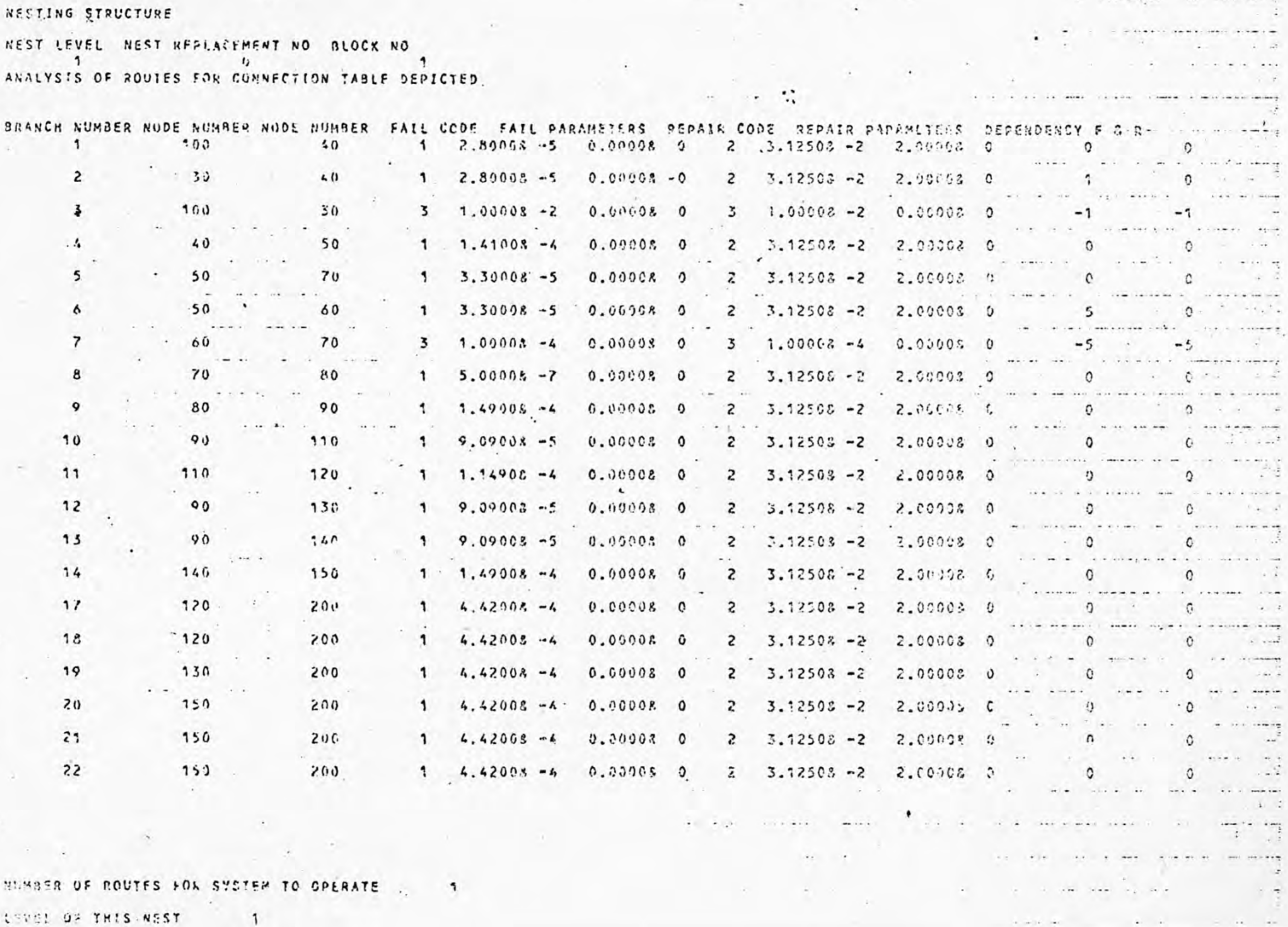


figure 7.1

7.2

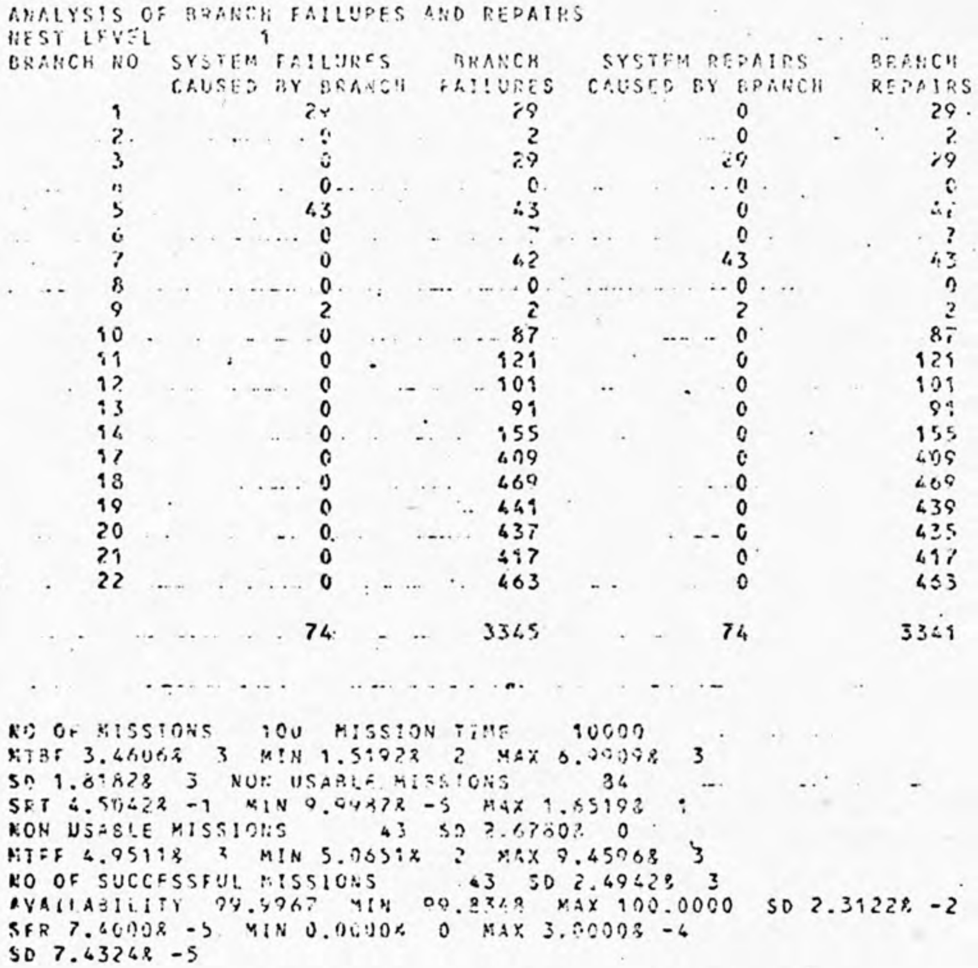


```
NESTING STRUCTURE

NEST LEVEL  NEST REPLACEMENT NO  BLOCK NO
     1               0              1
ANALYSIS OF ROUTES FOR CONNECTION TABLE DEPICTED.


BRANCH NUMBER NODE NUMBER NODE NUMBER  FAIL CODE  FAIL PARAMETERS  REPAIR CODE  REPAIR PARAMETERS  DEPENDENCY F G R
      1         100          40          1   2.80008 -5   0.00008  0    2   3.12508 -2   2.00008  0        0        0

      2          30          40          1   2.80008 -5   0.00008 -0    2   3.12508 -2   2.00008  0        1        0

      3         100          30          3   1.00008 -2   0.00008  0    3   1.00008 -2   0.00008  0       -1       -1

      4          40          50          1   1.41008 -4   0.00008  0    2   3.12508 -2   2.00008  0        0        0

      5          50          70          1   3.30008 -5   0.00008  0    2   3.12508 -2   2.00008  0        0        0

      6          50          60          1   3.30008 -5   0.00008  0    2   3.12508 -2   2.00008  0        5        0

      7          60          70          3   1.00008 -4   0.00008  0    3   1.00008 -4   0.00008  0       -5       -5

      8          70          80          1   5.00008 -7   0.00008  0    2   3.12508 -2   2.00008  0        0        0

      9          80          90          1   1.49008 -4   0.00008  0    2   3.12508 -2   2.00008  0        0        0

     10          90         110          1   9.09008 -5   0.00008  0    2   3.12508 -2   2.00008  0        0        0

     11         110         120          1   1.14908 -4   0.00008  0    2   3.12508 -2   2.00008  0        0        0

     12          90         130          1   9.09008 -5   0.00008  0    2   3.12508 -2   2.00008  0        0        0

     13          90         140          1   9.09008 -5   0.00008  0    2   3.12508 -2   2.00008  0        0        0

     14         140         150          1   1.49008 -4   0.00008  0    2   3.12508 -2   2.00008  0        0        0

     17         120         200          1   4.42008 -4   0.00008  0    2   3.12508 -2   2.00008  0        0        0

     18         120         200          1   4.42008 -4   0.00008  0    2   3.12508 -2   2.00008  0        0        0

     19         130         200          1   4.42008 -4   0.00008  0    2   3.12508 -2   2.00008  0        0        0

     20         150         200          1   4.42008 -4   0.00008  0    2   3.12508 -2   2.00008  0        0        0

     21         150         200          1   4.42008 -4   0.00008  0    2   3.12508 -2   2.00008  0        0        0

     22         150         200          1   4.42008 -4   0.00008  0    2   3.12508 -2   2.00008  0        0        0



NUMBER OF ROUTES FOR SYSTEM TO OPERATE      1

LEVEL OF THIS NEST        1
```

figure 7.2

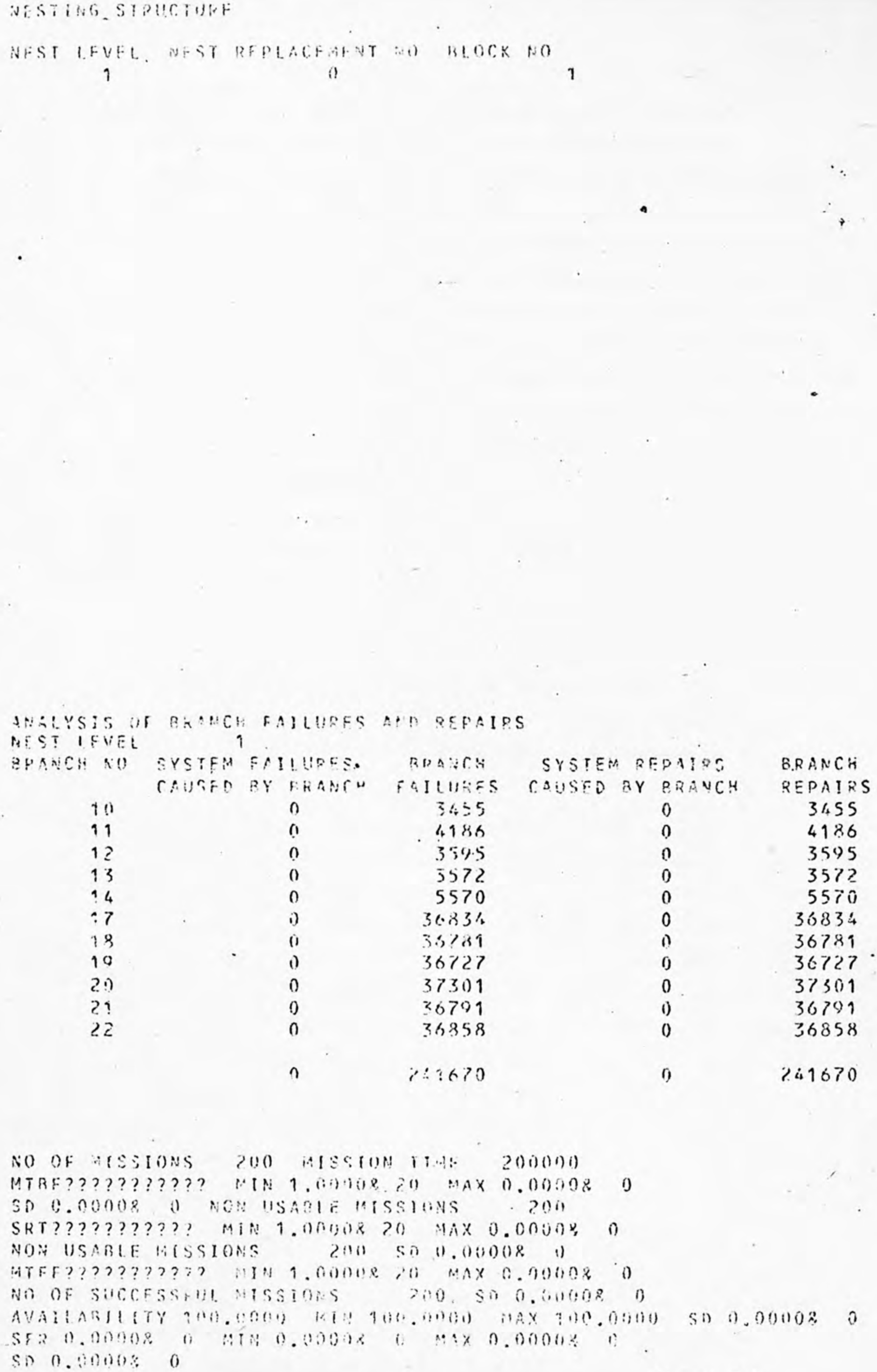
```
       1                              12                                              0                          1
       2                              12                                              0                          2
       3                              12                                              0                          3
       4                              24                                              0                          4
       5                              12                                              0                          5
       6                              12                                              0                          6
       7                              12                                              1                          7
       8                              24                                              7                          8
       9                              24                                             11                          9
      10                               8                                              5                         10
      11                               8                                              0                         11
      12                               4                                              0                         12
      13                              12                                              0                         13
      14                              12                                              0                         14
      17                               4                                              0                         15
      18                               4                                              0                         16
      19                               4                                              0                         17
      20                               4                                              0                         18
      21                               4                                              0                         19
      22                               4                                              0                         20




ANALYSIS OF BRANCH FAILURES AND REPAIRS
NEST LEVEL       1
BRANCH NO  SYSTEM FAILURES    BRANCH   SYSTEM REPAIRS     BRANCH
           CAUSED BY BRANCH  FAILURES  CAUSED BY BRANCH   REPAIRS
       1         29            29            0              29
       2          0             6            0               6
       3          0            29           29              29
       4        127           127          126             126
       5         36            36            0              36
       6          0             4            0               4
       7          0            36           36              36
       8          1             1            1               1
       9        126           126          126             126
      10          0           105            0             105
      11          0           140            0             140
      12          0           100            0             100
      13          0            87            0              87
      14          0           141            0             141
      17          0           423            0             423
      18          0           425            0             424
      19          0           437            0             436
      20          0           449            0             449
      21          0           450            0             448
      22          0           426            0             425

                319          3577          318            3571
```

figure 7.3



```
ANALYSIS OF BRANCH FAILURES AND REPAIRS
NEST LEVEL        1
BRANCH NO   SYSTEM FAILURES      BRANCH     SYSTEM REPAIRS      BRANCH
            CAUSED BY BRANCH   FAILURES   CAUSED BY BRANCH     REPAIRS
        1               29          29                  0          29
        2                0           2                  0           2
        3                0          29                 29          29
        4                0           0                  0           0
        5               43          43                  0          43
        6                0           7                  0           7
        7                0          42                 43          43
        8                0           0                  0           0
        9                2           2                  2           2
       10                0          87                  0          87
       11                0         121                  0         121
       12                0         101                  0         101
       13                0          91                  0          91
       14                0         155                  0         155
       17                0         409                  0         409
       18                0         469                  0         469
       19                0         441                  0         439
       20                0         437                  0         435
       21                0         417                  0         417
       22                0         463                  0         463

                        74        3345                 74        3341


NO OF MISSIONS    100  MISSION TIME     10000
MTBF 3.4606E  3  MIN 1.5192E  2  MAX 6.9909E  3
SD 1.8182E  3  NON USABLE MISSIONS       84
SRT 4.5042E -1  MIN 9.9987E -5  MAX 1.6519E  1
NON USABLE MISSIONS         43  SD 2.6780E  0
MTFF 4.9511E  3  MIN 5.0651E  2  MAX 9.4596E  3
NO OF SUCCESSFUL MISSIONS       43  SD 2.4942E  3
AVAILABILITY  99.9967  MIN  99.8348  MAX 100.0000  SD 2.3122E -2
SFR 7.4000E -5  MIN 0.0000E  0  MAX 3.0000E -4
SD 7.4324E -5
```

figure 7.4



```
NESTING_STRUCTURE

NEST LEVEL, NEST REPLACEMENT NO  BLOCK NO
      1                  0              1
```

```
ANALYSIS OF BRANCH FAILURES AND REPAIRS
NEST LEVEL        1
BRANCH NO  SYSTEM FAILURES.   BRANCH    SYSTEM REPAIRS       BRANCH
           CAUSED BY BRANCH  FAILURES  CAUSED BY BRANCH     REPAIRS
     10              0          3455                0          3455
     11              0          4186                0          4186
     12              0          3595                0          3595
     13              0          3572                0          3572
     14              0          5570                0          5570
     17              0         36834                0         36834
     18              0         36781                0         36781
     19              0         36727                0         36727
     20              0         37301                0         37301
     21              0         36791                0         36791
     22              0         36858                0         36858

                     0        241670                0        241670


NO OF MISSIONS    200  MISSION TIME    200000
MTBF????????????  MIN 1.0000% 20  MAX 0.0000%  0
SD 0.0000%  0  NON USABLE MISSIONS     200
SRT????????????  MIN 1.0000% 20  MAX 0.0000%  0
NON USABLE MISSIONS       200  SD 0.0000%  0
MTFF????????????  MIN 1.0000% 20  MAX 0.0000%  0
NO OF SUCCESSFUL MISSIONS      200  SD 0.0000%  0
AVAILABILITY 100.0000  MIN 100.0000  MAX 100.0000  SD 0.0000%  0
SFR 0.0000%  0  MIN 0.0000%  0  MAX 0.0000%  0
SD 0.0000%  0
```

figure 7.5

The first simulation using the data shown in figure 7.2 results in an availability figure of 99.566%. It is necessary to improve this to 99.99% on a 10,000 hour mission (approximately one year). Examining the branch table shown in figure 7.3 it can be seen that the two serial multicouplers (4, 9) have caused most of the system failures. Those caused by elements 1 and 5 are the momentary switch times of 3 and 7 respectively. For element 3, its switch time is 10^{-2} hours. Thus the total time lost due to this element = 10^{-2} x 29 = 0.29 hours. For a total simulated time of 10^{6} hours, this is relatively insignificant. For element 7 with a switch time of 10^{-4} hours, the effect is even less.

In order to effect improved availability the performance of the crucial elements 4 and 9 must be changed. The first attempt, not shown, consisted of improving λ for 4 and 9 from 1.41 x 10^{-4} and 1.49 x 10^{-4} to 3 x 10^{-5} which did not produce the desired increase in availability. This was only achieved by making λ = 3 x 10^{-6} as demonstrated by figure 7.4. (The usage of the standby facilities is also illustrated by figure 7.4.) The improvements represent increase of costs which has to be compensated by further design changes.

Seeking to keep the number of design changes to a minimum, engineers would suggest degrading the performance of the six receivers 17-22. The result of such a degradation is shown in figure 7.5. Only the receiving consoles are considered (nodes 90 200) and λ for the receivers has been doubled to 1 x 10^{-3}. This simulation, run for 200,000 hours, shows that the poorer quality receivers are still having no effect on the systems overall reliability performance and could be degraded further, if necessary, for additional cost reduction.

7.2 A PRIVATE HOSPITAL

The hospital considered for this design exercise was constructed within the last two years. The hospital has been structured according to the latest hospital design methods. It was decided that it would be interesting to model the hospital using reliability concepts and then attempt a simulation using assumed but realistic data.

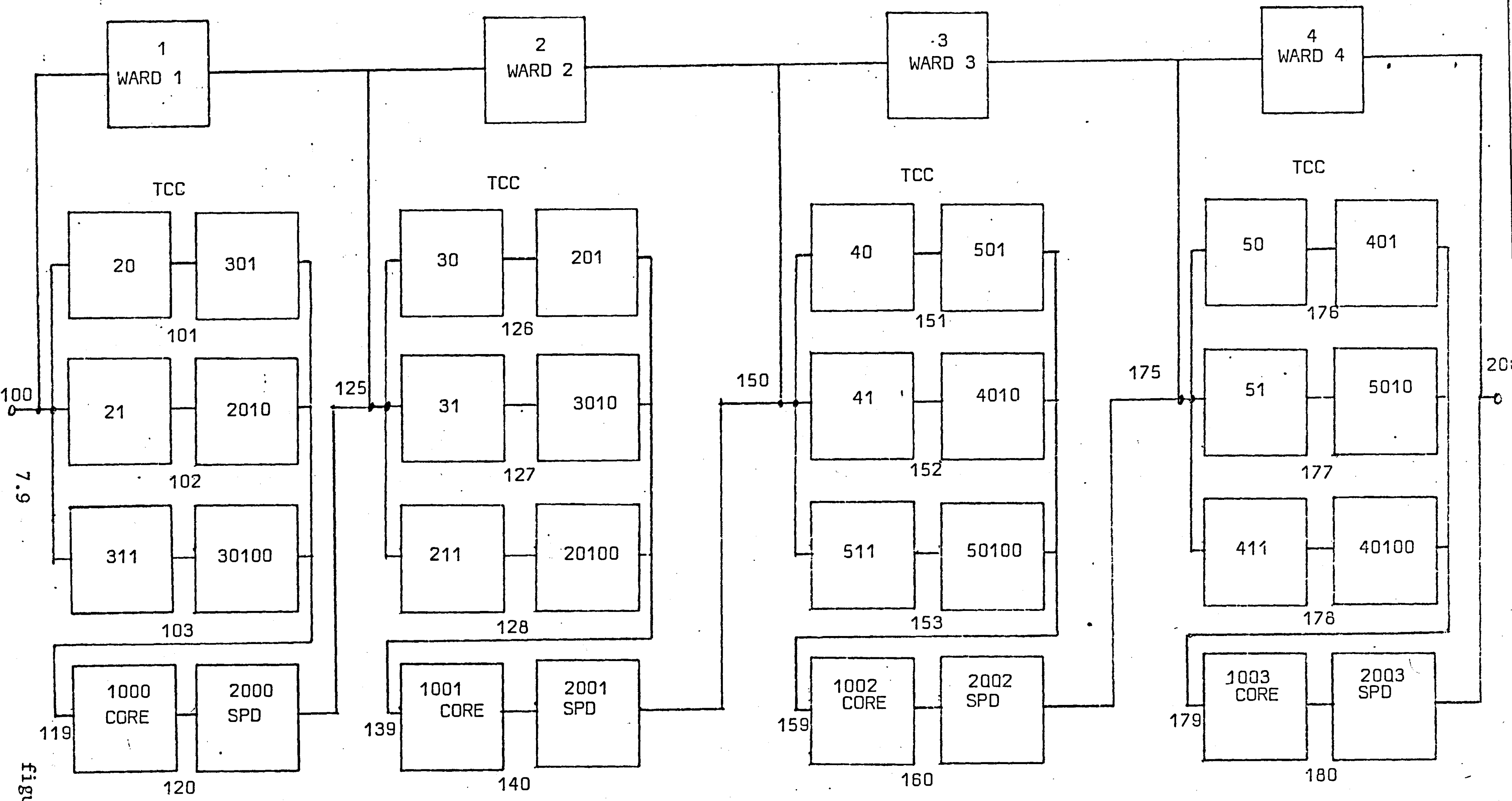
The hospital consists of 80 private rooms divided into wards of 20 rooms each. Two wards exist on the ground floor and two wards on the first floor. Each floor has shared hardware facilities in a core. In each ward there exists a room for conferences, discussions, etc., which should should always be manned. This room is called the Team Control Centre (TCC). Situated on the ground floor is a central Supply and Processing Department (SPD).

The reliability model of this hospital has certain features common to the Gas Production Plant described in chapter 2.5, namely, the object of the model is to assess the ability of the services to meet a demand from the wards. Failure to meet a demand constitutes a system failure. Referring to figure 7.6, the ward demands are in parallel with the services. When a demand is made, the corresponding ward branch fails.

The operation of the system is discussed with reference to figure 7.6. Each floor has a core (1000, 1002) with an additional dependent core (1001, 1003) to cater for demands from wards 2 and 4. The SPD is represented by 2000 and 2002 with dependent SPDs 2001 and 2003. It is necessary to have two SPDs in the model because of the nested structure. The TCC provides the most complex part of the model. It is assumed that, should a TCC be unavailable, the other TCC on the same floor can take over, but with halved reliability of providing a service. This is modelled for one floor as follows: 20 and 30 represent the TCCs. 21 and 31 represent the TCCs working at half reliability. Should 20 fail, then 201 fails. 2010 repairs, bringing in 21. Also 20100 repairs, bringing in a route between nodes 125 and 139. If 21 fails before 20 is repaired then the system fails because 211 depends directly on it. The purpose of 211 is to cope with the circumstance when TCC ward 1 fails and TCC ward 2 has already failed (20 fails but 21 has also failed). The same processes apply on a failure of TCC ward 2 (30).

The data for the model is shown in figures 7.7, 7.9, 7.12. For the purpose of simulation the system is divided into two nests, one for each floor. The element data, which has been assumed, is as follows:

ST. ANTHONYS HOSPITAL RELIABILITY BLOCK DIAGRAM



7.9

figure 7.6

```
NESTING STRUCTURE

 NEST LEVEL  NEST REPLACEMENT NO  BLOCK NO
       2              1               1
       3              1               2
       1              0               1
ANALYSIS OF ROUTES FOR CONNECTION TABLE DEPICTED


BRANCH NUMBER NODE NUMBER NODE NUMBER  FAIL CODE  FAIL PARAMETERS  REPAIR CODE  REPAIR PARAMETERS  DEPENDENCY F & R
         1        100         125       1  2.0000& 0   0.0000&  0    3   1.6000& -1   0.0000&  0         0         0
         2        125         150       1  2.0000& 0   0.0000&  0    3   1.6000& -1   0.0000&  0         0         0
        20        100         101       1  1.0000& 0   2.5000&  0    2   1.6000& -1   3.0000& -1         0         0
        21        100         102       1  5.0000& -1  2.5000&  0    3   1.6000& -5   0.0000&  0        20        30
        30        125         126       1  1.0000& 0   2.5000&  0    2   1.6000& -1   3.0000& -1         0         0
        31        125         127       1  5.0000& -1  2.5000&  0    3   1.6000& -5   0.0000&  0        30        20
       201        126         139       3  1.6000& -5  0.0000&  0    3   1.6000& -5   0.0000&  0        20        20
       211        125         128       3  1.6000& -5  0.0000&  0    3   1.6000& -5   0.0000&  0        21        21
       301        101         119       3  1.6000& -5  0.0000&  0    3   1.6000& -5   0.0000&  0        30        30
       311        100         103       3  1.6000& -5  0.0000&  0    3   1.6000& -5   0.0000&  0        31        31
      1000        119         120       1  1.3900& -3  0.0000&  0    2   5.0000& -1   3.0000&  0         0         0
      1001        139         140       3  1.6000& -5  0.0000&  0    3   1.6000& -5   0.0000&  0      1000      1000
      2000        120         125       1  4.6000& -4  0.0000&  0    2   2.5000& -1   5.0000& -1         0         0
      2001        140         150       3  1.6000& -5  0.0000&  0    3   1.6000& -5   0.0000&  0      2000      2000
      2010        102         119       3  1.6000& -5  0.0000&  0    3   1.6000& -5   0.0000&  0       -20       -20
      3010        127         139       3  1.6000& -5  0.0000&  0    3   1.6000& -5   0.0000&  0       -30       -30
     20100        128         139       3  1.6000& -5  0.0000&  0    3   1.6000& -5   0.0000&  0      2010      2010
     30100        103         119       3  1.6000& -5  0.0000&  0    3   1.6000& -5   0.0000&  0      3010      3010



 NUMBER OF ROUTES FOR SYSTEM TO OPERATE        1

 LEVEL OF THIS NEST       2



 ANALYSIS OF ROUTES ASSOCIATED WITH INPUT NODE/OUTPUT NODE      100         150
```

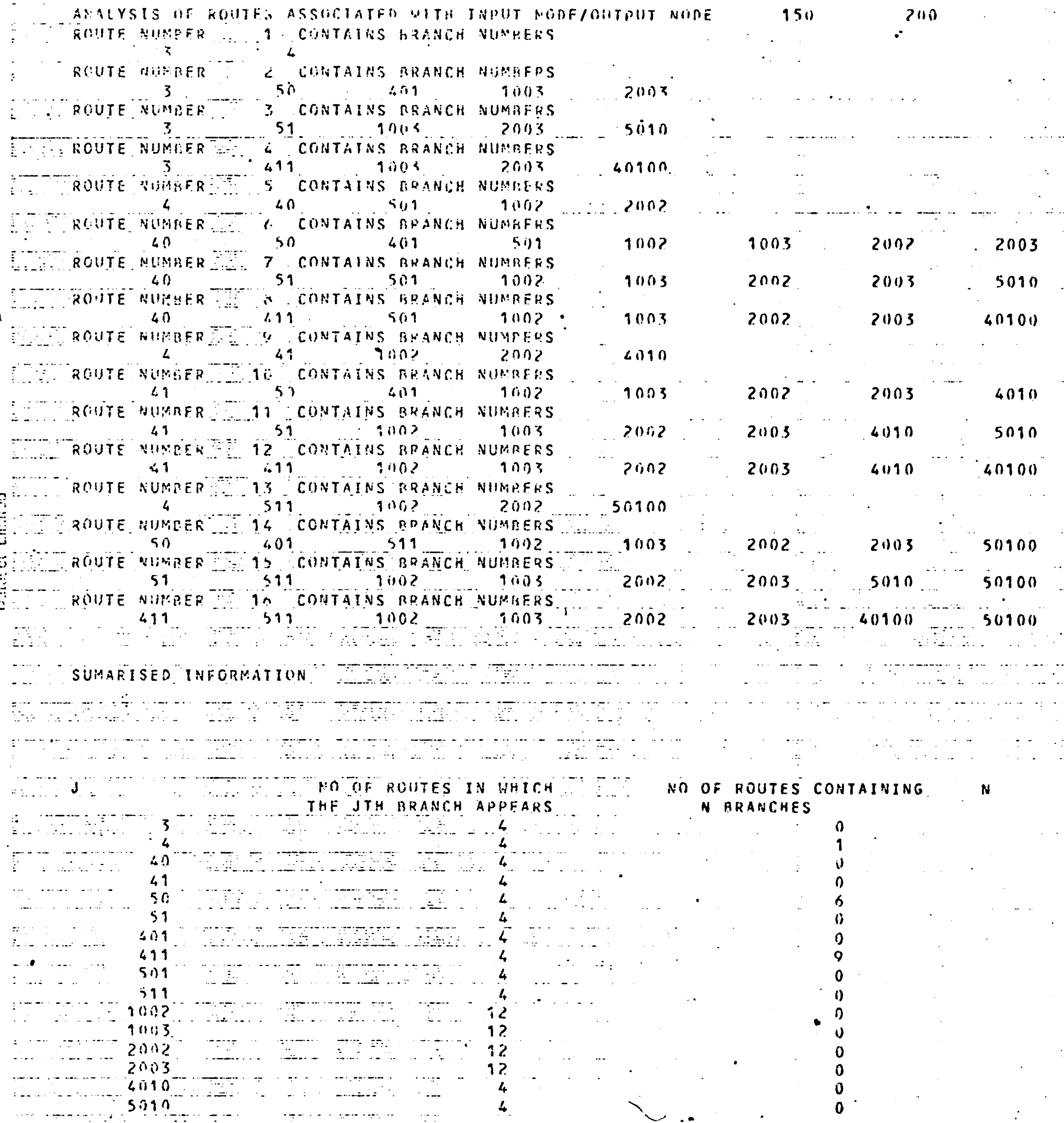
figure 7.7

```
        ROUTE NUMBER    1  CONTAINS BRANCH NUMBERS
              1          2
        ROUTE NUMBER    2  CONTAINS BRANCH NUMBERS
              1         30        201       1001       2001
        ROUTE NUMBER    3  CONTAINS BRANCH NUMBERS
              1         31       1001       2001       3010
        ROUTE NUMBER    4  CONTAINS BRANCH NUMBERS
              1        211       1001       2001      20100
        ROUTE NUMBER    5  CONTAINS BRANCH NUMBERS
              2         20        301       1000       2000
        ROUTE NUMBER    6  CONTAINS BRANCH NUMBERS
             20         30        201        301       1000       1001       2000       2001
        ROUTE NUMBER    7  CONTAINS BRANCH NUMBERS
             20         31        301       1000       1001       2000       2001       3010
        ROUTE NUMBER    8  CONTAINS BRANCH NUMBERS
             20        211        301       1000       1001       2000       2001      20100
        ROUTE NUMBER    9  CONTAINS BRANCH NUMBERS
              2         21       1000       2000       2010
        ROUTE NUMBER   10  CONTAINS BRANCH NUMBERS
             21         30        201       1000       1001       2000       2001       2010
        ROUTE NUMBER   11  CONTAINS BRANCH NUMBERS
             21         31       1000       1001       2000       2001       2010       3010
        ROUTE NUMBER   12  CONTAINS BRANCH NUMBERS
             21        211       1000       1001       2000       2001       2010      20100
        ROUTE NUMBER   13  CONTAINS BRANCH NUMBERS
              2        311       1000       2000      30100
        ROUTE NUMBER   14  CONTAINS BRANCH NUMBERS
             30        201        311       1000       1001       2000       2001      30100
        ROUTE NUMBER   15  CONTAINS BRANCH NUMBERS
             31        311       1000       1001       2000       2001       3010      30100
        ROUTE NUMBER   16  CONTAINS BRANCH NUMBERS
            211        311       1000       1001       2000       2001      20100      30100


        SUMARISED INFORMATION



        J                  NO OF ROUTES IN WHICH          NO OF ROUTES CONTAINING       N
                            THE JTH BRANCH APPEARS             N BRANCHES
              1                          4                            0                 1
              2                          4                            1                 2
             20                          4                            0                 3
             21                          4                            0                 4
             30                          4                            6                 5
             31                          4                            0                 6
            201                          4                            0                 7
            211                          4                            9                 8
            301                          4                            0                 9
            311                          4                            0                10
           1000                         12                            0                11
           1001                         12                            0                12
           2000                         12                            0                13
           2001                         12                            0                14
           2010                          4                            0                15
           3010                          4                            0                16
          20100                          4                            0                17
          30100                          4                            0                18
```

figure 7.8

figure 7.9

```
ANALYSIS OF ROUTES FOR CONNECTION TABLE DEPICTED

      BRANCH NUMBER NODE NUMBER NODE NUMBER  FAIL CODE  FAIL PARAMETERS  REPAIR CODE  REPAIR PARAMETERS  DEPENDENCY F & R
             3      150         175            1     2.00008  0   0.00008  0    3   1.60008 -1   0.00008  0       0       0
             4      175         200            1     2.00008  0   0.00008  0    3   1.60008 -1   0.00008  0       0       0
            40      150         151            1     1.00008  0   2.50008  0    2   1.60008 -1   3.00008 -1       0       0
            41      150         152            1     5.00008 -1   2.50008  0    3   1.60008 -5   0.00008  0      40      50
            50      175         176            1     1.00008  0   2.50008  0    2   1.60008 -1   3.00008 -1       0       0
            51      175         177            1     5.00008 -1   2.50008  0    3   1.60008 -5   0.00008  0      50      40
           401      176         179            3     1.60008 -5   0.00008  0    3   1.60008 -5   0.00008  0      40      40
           411      175         178            3     1.60008 -5   0.00008  0    3   1.60008 -5   0.00008  0      41      41
           501      151         159            3     1.60008 -5   0.00008  0    3   1.60008 -5   0.00008  0      50      50
           511      150         153            3     1.60008 -5   0.00008  0    3   1.60008 -5   0.00008  0      51      51
          1002      159         160            1     1.39008 -3   0.00008  0    2   5.00008 -1   3.00008  0       0       0
          1003      179         180            3     1.60008 -5   0.00008  0    3   1.60008 -5   0.00008  0    1002    1002
          2002      160         175            1     4.60008 -4   0.00008  0    2   2.50008 -1   5.00008 -1       0       0
          2003      180         200            3     1.60008 -5   0.00008  0    3   1.60008 -5   0.00008  0    2002    2002
          4010      152         159            3     1.60008 -5   0.00008  0    3   1.60008 -5   0.00008  0     -40     -40
          5010      177         179            3     1.60008 -5   0.00008  0    3   1.60008 -5   0.00008  0     -50     -50
         40100      178         179            3     1.60008 -5   0.00008  0    3   1.60008 -5   0.00008  0    4010    4010
         50100      153         159            3     1.60008 -5   0.00008  0    3   1.60008 -5   0.00008  0    5010    5010


      NUMBER OF ROUTES FOR SYSTEM TO OPERATE          1

      LEVEL OF THIS NEST      3
```



```
ANALYSIS OF ROUTES ASSOCIATED WITH INPUT NODE/OUTPUT NODE       150         200
ROUTE NUMBER     1  CONTAINS BRANCH NUMBERS
         3           4
ROUTE NUMBER     2  CONTAINS BRANCH NUMBERS
         3          50        401       1003       2003
ROUTE NUMBER     3  CONTAINS BRANCH NUMBERS
         3          51       1003       2003       5010
ROUTE NUMBER     4  CONTAINS BRANCH NUMBERS
         3         411       1003       2003      40100
ROUTE NUMBER     5  CONTAINS BRANCH NUMBERS
         4          40        501       1002       2002
ROUTE NUMBER     6  CONTAINS BRANCH NUMBERS
        40          50        401        501       1002       1003       2002       2003
ROUTE NUMBER     7  CONTAINS BRANCH NUMBERS
        40          51        501       1002       1003       2002       2003       5010
ROUTE NUMBER     8  CONTAINS BRANCH NUMBERS
        40         411        501       1002       1003       2002       2003      40100
ROUTE NUMBER     9  CONTAINS BRANCH NUMBERS
         4          41       1002       2002       4010
ROUTE NUMBER    10  CONTAINS BRANCH NUMBERS
        41          50        401       1002       1003       2002       2003       4010
ROUTE NUMBER    11  CONTAINS BRANCH NUMBERS
        41          51       1002       1003       2002       2003       4010       5010
ROUTE NUMBER    12  CONTAINS BRANCH NUMBERS
        41         411       1002       1003       2002       2003       4010      40100
ROUTE NUMBER    13  CONTAINS BRANCH NUMBERS
         4         511       1002       2002      50100
ROUTE NUMBER    14  CONTAINS BRANCH NUMBERS
        50         401        511       1002       1003       2002       2003      50100
ROUTE NUMBER    15  CONTAINS BRANCH NUMBERS
        51         511       1002       1003       2002       2003       5010      50100
ROUTE NUMBER    16  CONTAINS BRANCH NUMBERS
       411         511       1002       1003       2002       2003      40100      50100


SUMARISED INFORMATION



J                    NO OF ROUTES IN WHICH          NO OF ROUTES CONTAINING      N
                     THE JTH BRANCH APPEARS           N BRANCHES
         3                     4                               0                     1
         4                     4                               1                     2
        40                     4                               0                     3
        41                     4                               0                     4
        50                     4                               6                     5
        51                     4                               0                     6
       401                     4                               0                     7
       411                     4                               9                     8
       501                     4                               0                     9
       511                     4                               0                    10
      1002                    12                               0                    11
      1003                    12                               0                    12
      2002                    12                               0                    13
      2003                    12                               0                    14
      4010                     4                               0                    15
      5010                     4                               0                    16
```

figure 7.10

```
      40100                    4                    0                    17
      50100                    4                    0                    18

   ANALYSIS OF ROUTES FOR CONNECTION TABLE DEPICTED

   BRANCH NUMBER NODE NUMBER NODE NUMBER   FAIL CODE  FAIL PARAMETERS  REPAIR CODE  REPAIR PARAMETERS  DEPENDENCY F & R
         1             1           2          0    0.0000&  0   0.0000&  0     0   0.0000&  0   0.0000&  0        0        0
         2             2           3          0    0.0000&  0   0.0000&  0     0   0.0000&  0   0.0000&  0        0        0

   NUMBER OF ROUTES FOR SYSTEM TO OPERATE        1

   LEVEL OF THIS NEST         1

   ANALYSIS OF ROUTES ASSOCIATED WITH INPUT NODE/OUTPUT NODE          1          3
   ROUTE NUMBER      1  CONTAINS BRANCH NUMBERS
         1.        2

   SUMARISED INFORMATION

   J                     NO OF ROUTES IN WHICH           NO OF ROUTES CONTAINING        N
                        THE JTH BRANCH APPEARS                N BRANCHES
         1                     1                                  0                     1
         2                     1                                  1                     2
```

figure 7.12

7.14

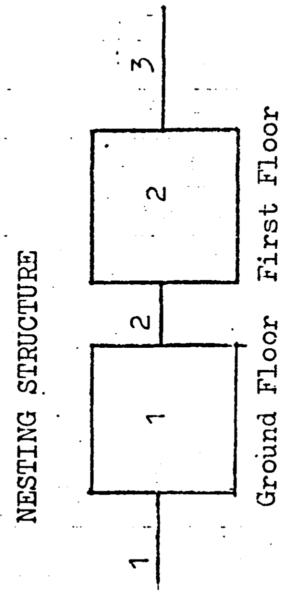
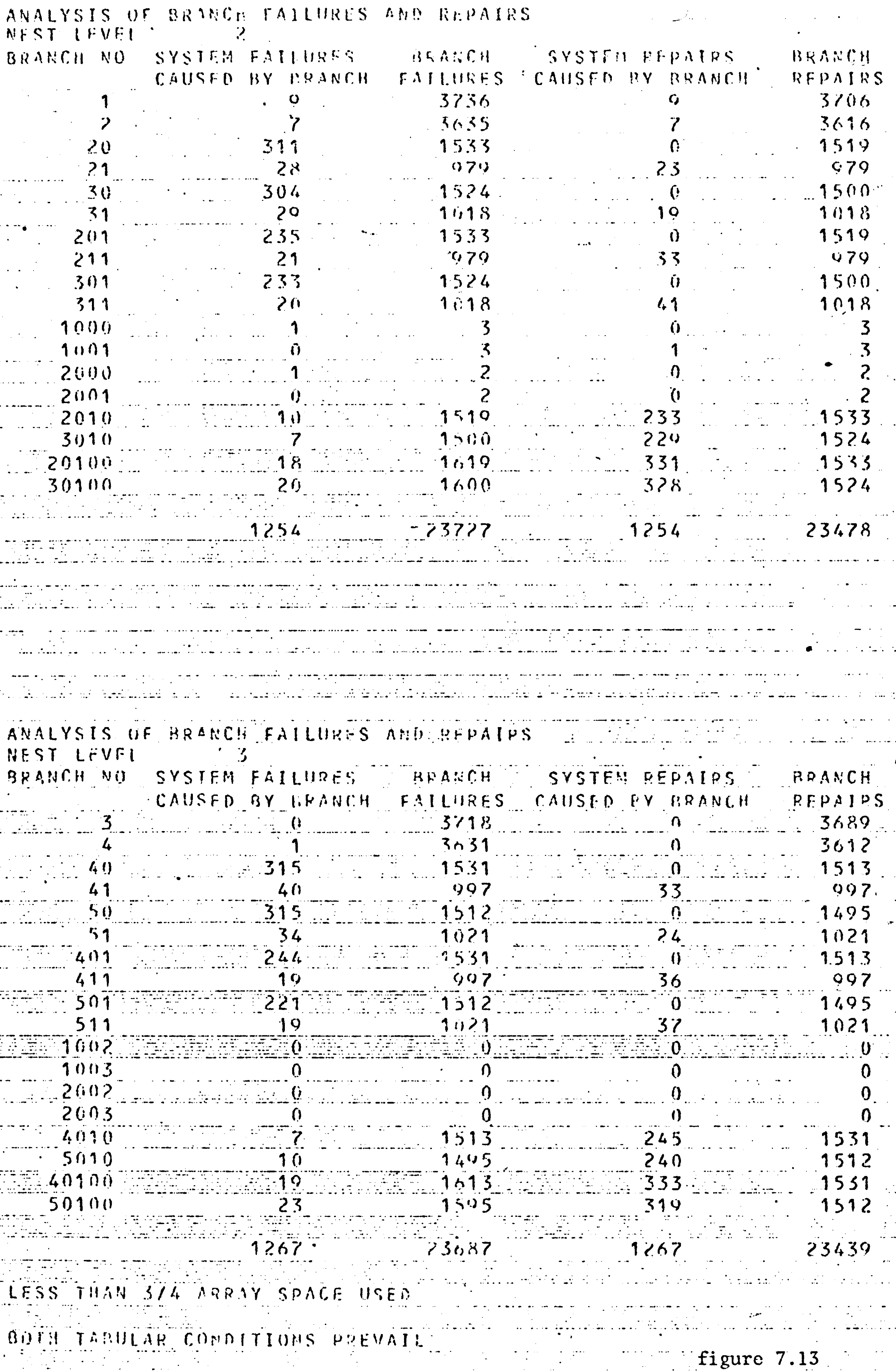


figure 7.11

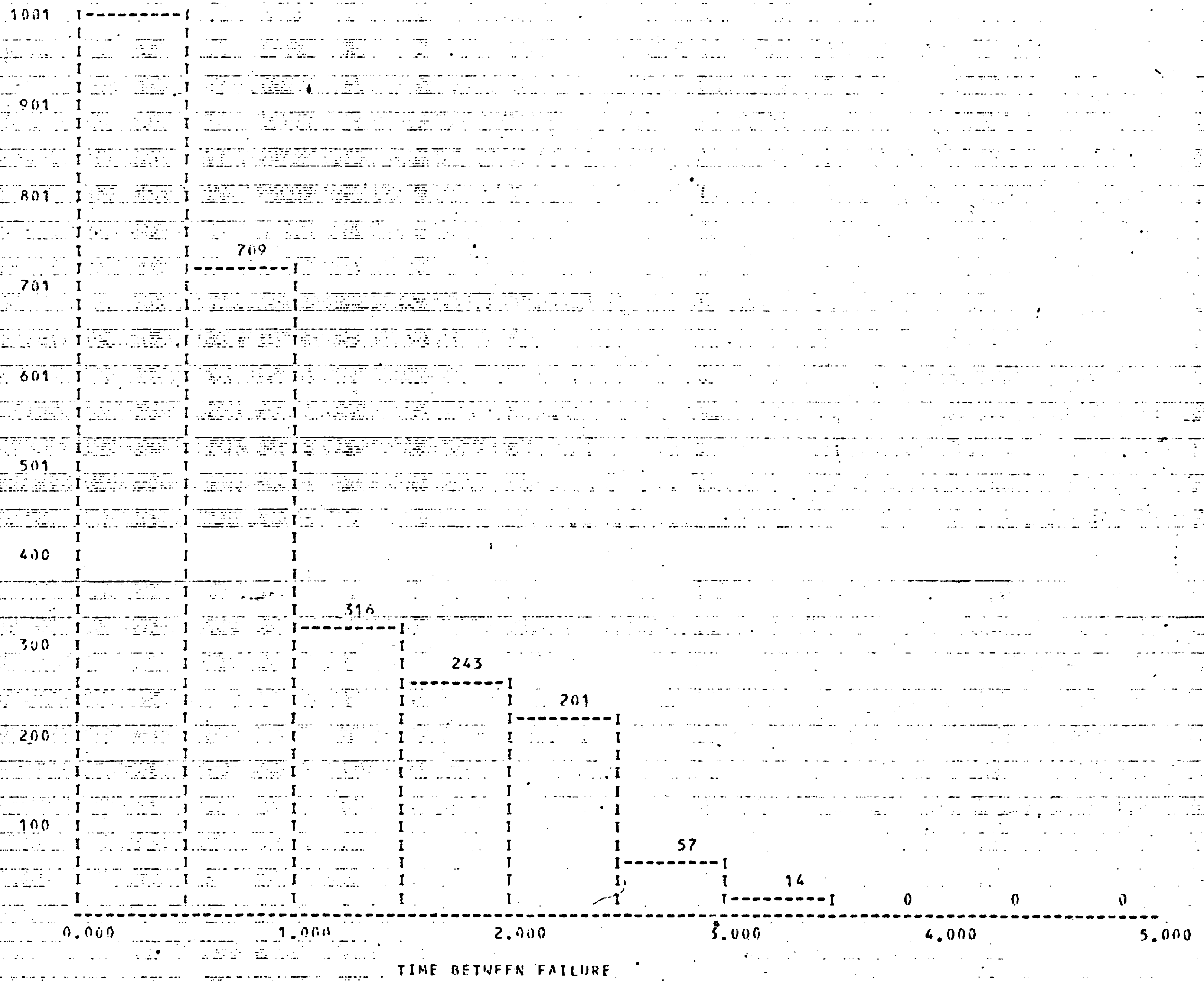


```
ANALYSIS OF BRANCH FAILURES AND REPAIRS
NEST LEVEL       2
BRANCH NO   SYSTEM FAILURES     BRANCH     SYSTEM REPAIRS     BRANCH
            CAUSED BY BRANCH  FAILURES  CAUSED BY BRANCH    REPAIRS
        1                9      3736                 9       3706
        2                7      3635                 7       3616
       20              311      1533                 0       1519
       21               28       979                23        979
       30              304      1524                 0       1500
       31               29      1018                19       1018
      201              235      1533                 0       1519
      211               21       979                33        979
      301              233      1524                 0       1500
      311               20      1018                41       1018
     1000                1         3                 0          3
     1001                0         3                 1          3
     2000                1         2                 0          2
     2001                0         2                 0          2
     2010               10      1519               233       1533
     3010                7      1500               229       1524
    20100               18      1619               331       1533
    30100               20      1600               328       1524

                      1254     23727              1254      23478
```

```
ANALYSIS OF BRANCH FAILURES AND REPAIRS
NEST LEVEL       3
BRANCH NO   SYSTEM FAILURES     BRANCH     SYSTEM REPAIRS     BRANCH
            CAUSED BY BRANCH  FAILURES  CAUSED BY BRANCH    REPAIRS
        3                0      3718                 0       3689
        4                1      3631                 0       3612
       40              315      1531                 0       1513
       41               40       997                33        997
       50              315      1512                 0       1495
       51               34      1021                24       1021
      401              244      1531                 0       1513
      411               19       997                36        997
      501              221      1512                 0       1495
      511               19      1021                37       1021
     1002                0         0                 0          0
     1003                0         0                 0          0
     2002                0         0                 0          0
     2003                0         0                 0          0
     4010                7      1513               245       1531
     5010               10      1495               240       1512
    40100               19      1613               333       1531
    50100               23      1595               319       1512

                      1267     23687              1267      23439

LESS THAN 3/4 ARRAY SPACE USED

BOTH TABULAR CONDITIONS PREVAIL
```

figure 7.13

FREQUENCY DIAGRAM OF TIMES BETWEEN FAILURE



```
NO OF MISSIONS   100     MISSION TIME      24  UNITS    OVERALL MEAN 8.8178E -1    MIN 8.3109E -3    MAX 3.2784E  0
STANDARD DEVIATION 7.1737E -1
NO OF MISSIONS FOR WHICH MTBF CANNOT BE OBTAINED        0
```

figure 7.14

i) Wards: Exponential failure (demand) with a mean of half-hour and switch repair fixed at 10 minutes. (duration of demand).

Core: Exponential failure with a mean of 30 days and lognormal repair with a mean of 1.22 hours.

SPD: Exponential failure with a mean of 90 days and lognormal repair with a mean of 22 minutes.

TCC: Weiball failure with a mean of 1 hour, shaping parameter $m = 2.5$ and lognormal repair with a mean of 13 minutes.

The output of 100 runs of 24 hours each is shown in figure 7.13. There was an average of 3680 demands in this time, i.e., 36 per 24 hours. During these demands, the core and SPD proved to be very reliable, only causing one system failure. On an average of four occasions, the ward demand was not satisfied as soon as the demand occurred. On the other occasion the system broke down during a demand. On an average of 1089 occasions, one TCC had to cope with demands from both wards. On an average of 33 occasions even the single TCC broke down.

The time-between-failure histogram shown in figure 7.14 is exponential in shape and has a mean of 53 minutes. The mean time to first failure is 41 minutes and the mean system repair time is 2 seconds. The availability is very high (99.93%) indicating that the system is unavailable on average for one minute in 24 hours.

7.3 A CONTROL SYSTEM FOR A DISABLED DRIVER'S CAR

The Transport and Road Research Laboratory has been undertaking an investigation into various control systems for a disabled driver's car. Part of this investigation required an assessment of the reliability of the proposed control system. A design exercise was undertaken under the sponsorship of TRRL. The initial part of the reliability assessment exercise is documented here.

The objectives of the exercise were to:

i) locate critical elements.

ii) obtain a mean time to first failure of two channels (no repair being undertaken).

iii) obtain a mean time to first failure of one channel as an indication of the 'safety margin' in which repair of the channel must be carried out before another channel fails.

The system chosen for investigation is a duplex monitored system shown in figure 7.16. The system requires at least two out of three channels to be operational. The monitoring channel is designed to detect a faulty channel and switch it out by means of control relays placed in the main power lines. Three separate batteries are used to prevent common mode supply faults.

In order to provide the two-out-of-three facility, the system may be modelled as in the conceptual block diagram shown on figure 7.15.

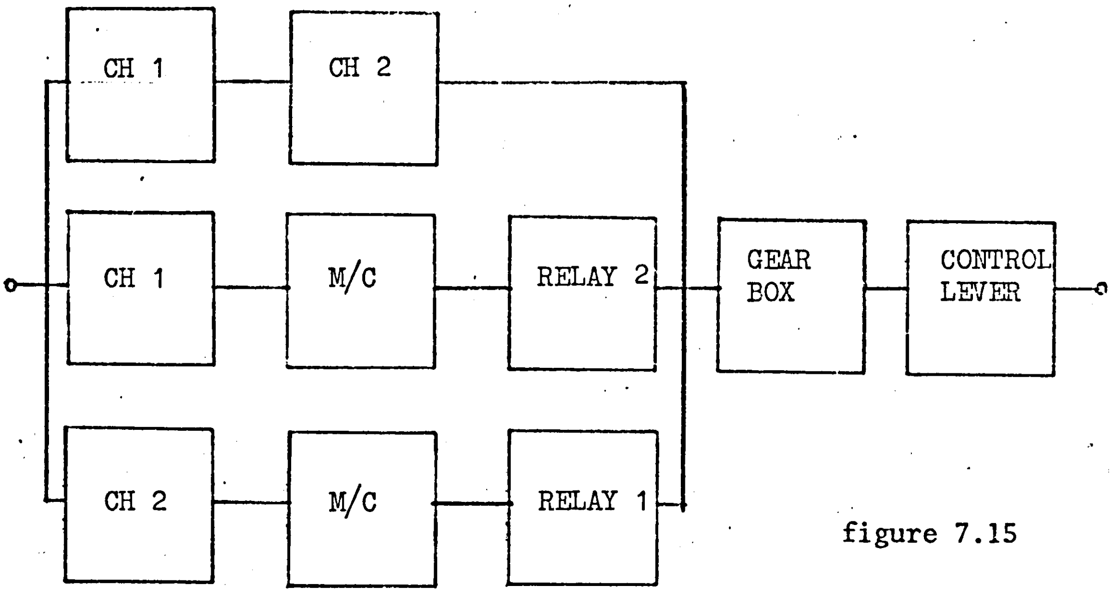


figure 7.15

Should either channel 1 or channel 2 fail then one route will remain open, provided that the corresponding isolating relay works. The gearbox and control lever are the only series (common) elements.

Figure 7.17 shows the full block diagram. The most complex modelling feature is the battery drain. This can be caused by:

i) an alternator failure.

ii) a diode open circuit.

iii) a diode short circuit, together with a battery failure of another channel.

Case i) is catered for by elements 20, 30 & 40, 25, 35 & 45, all of them switch dependent on 15, the switch time being the drain time. Case ii) is catered for by elements 1008, 2010, 2002, 10061, 1006 & 1004 being dependent on their respective diodes. Case iii) is catered for by the elements modelled as for those between nodes 102 and 104. If diode 1 short circuits then branch 22 fails. If this is followed by battery 2 or battery 3 (M/C) failing then 23 or 24 fails shortly after. The lumped parameter elements combine the command pot, comparator, A/D converter, amplifier, motor, clutch, clutch driver power transistor, switch, fuse and wiring. This is valid because all these elements have exponential failure with no repair. The model shown in figure 7.17 has 53 elements. This is more than the optimum number but, since the model has only 12 routes, the extra number of branches do not prove much of a disadvantage. Figure 7.18 shows the input data with all the dependencies and switch times.

The output is shown in figure 7.19. With the exception of four random events, all 100 system failures have been caused by the alternator failing or a battery failing. Because of the interdependencies of figure 7.16 the actual cause of a system failure is sometimes fairly complex. Six branches depend on an alternator failure, 20, 25, 30, 35, 40 and 45. Of these, 30 and 40 have the fastest drain rates. Thus 30 and 40 fail causing two of the three parallel paths to fail. It is 20 failing that fails the system as this element lies in the only remaining path. The six systems failures caused by 30 and 40 are a result of channel 1 or channel 2 having already failed. The batteries provide an even more complex set of interdepndencies. A battery failure will cause a drain and fail the whole system if a diode goes short circuit. But the diode is a very reliable element so it is the second of the three batteries to fail that will cause system failure.

With this system, catastrophic failure will result if two channels fail. Another simulation was performed to assess the system in terms of failure being classified as one channel failure. For this purpose the system was modelled as in figure 7.21. Certain simplifications can be made. Element 10 represents the battery drain caused by an alternator failure. This drain is that undergone by battery 3 because this has the shortest drain time therefore making it unnecessary to include the drain for batteries 1 and 2. The lumped parameter element has in addition to the model of figure 7.17 the battery and the diode S.C. The battery failure rate is so high compared with the other elements, that the lumped parameter element can be considered to be the battery. Figure 7.22 shows the input data and figure 7.23 the branch statistics. The system failures have been caused principally by the battery failing with a contribution of alternator failure. Figure 7.24 the time to first failure histogram, shows an exponential distribution of first failures times with a mean of 286 hours. This compares with the two out of three time of 755 hours and represents the safety margin time in which repair of the faulty channel must be carried out.

DUPLEX MONITORING SYSTEM

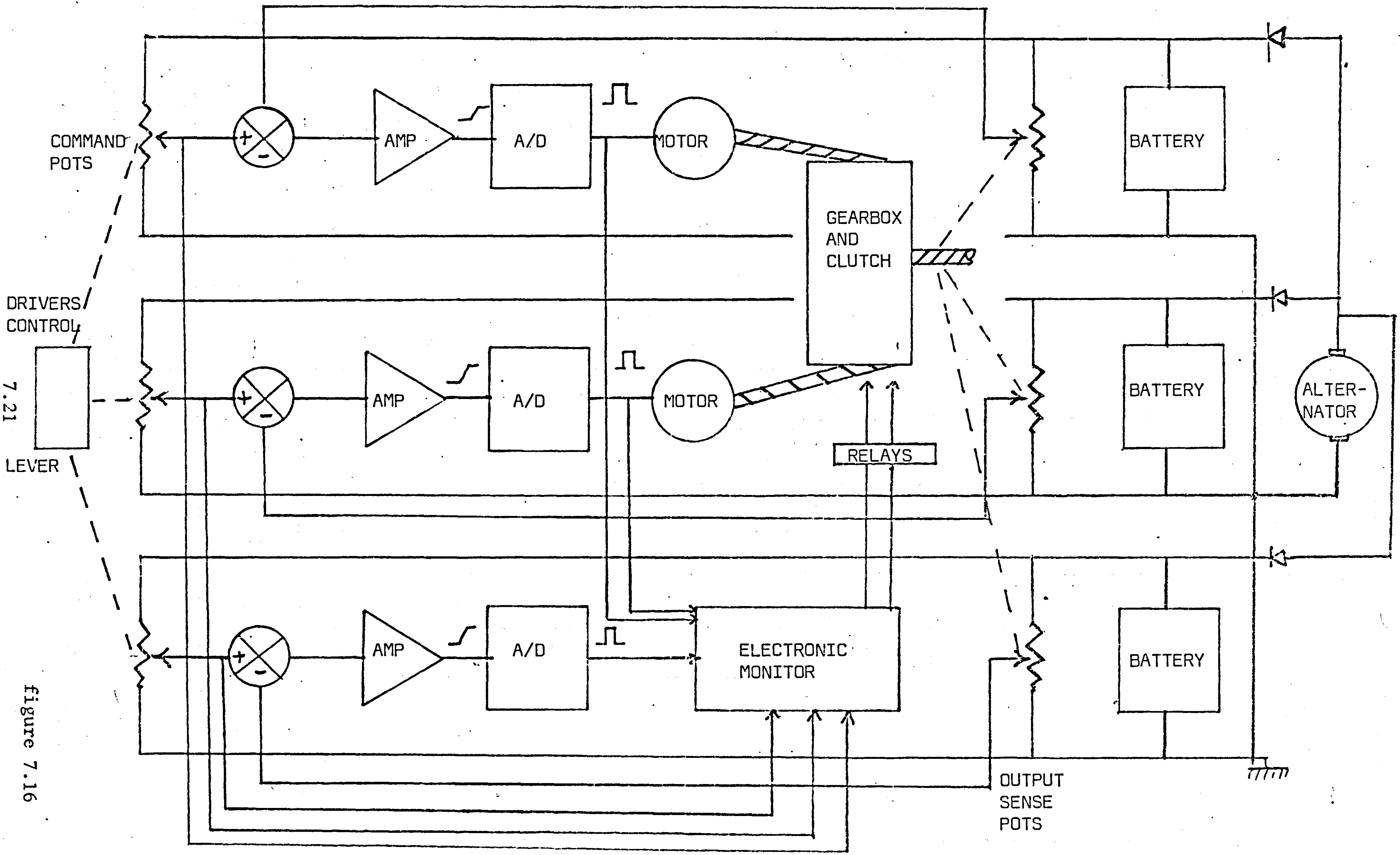


figure 7.16

CONTROL SYSTEM FOR DISABLED DRIVER CAR RELIABILITY BLOCK DIAGRAM

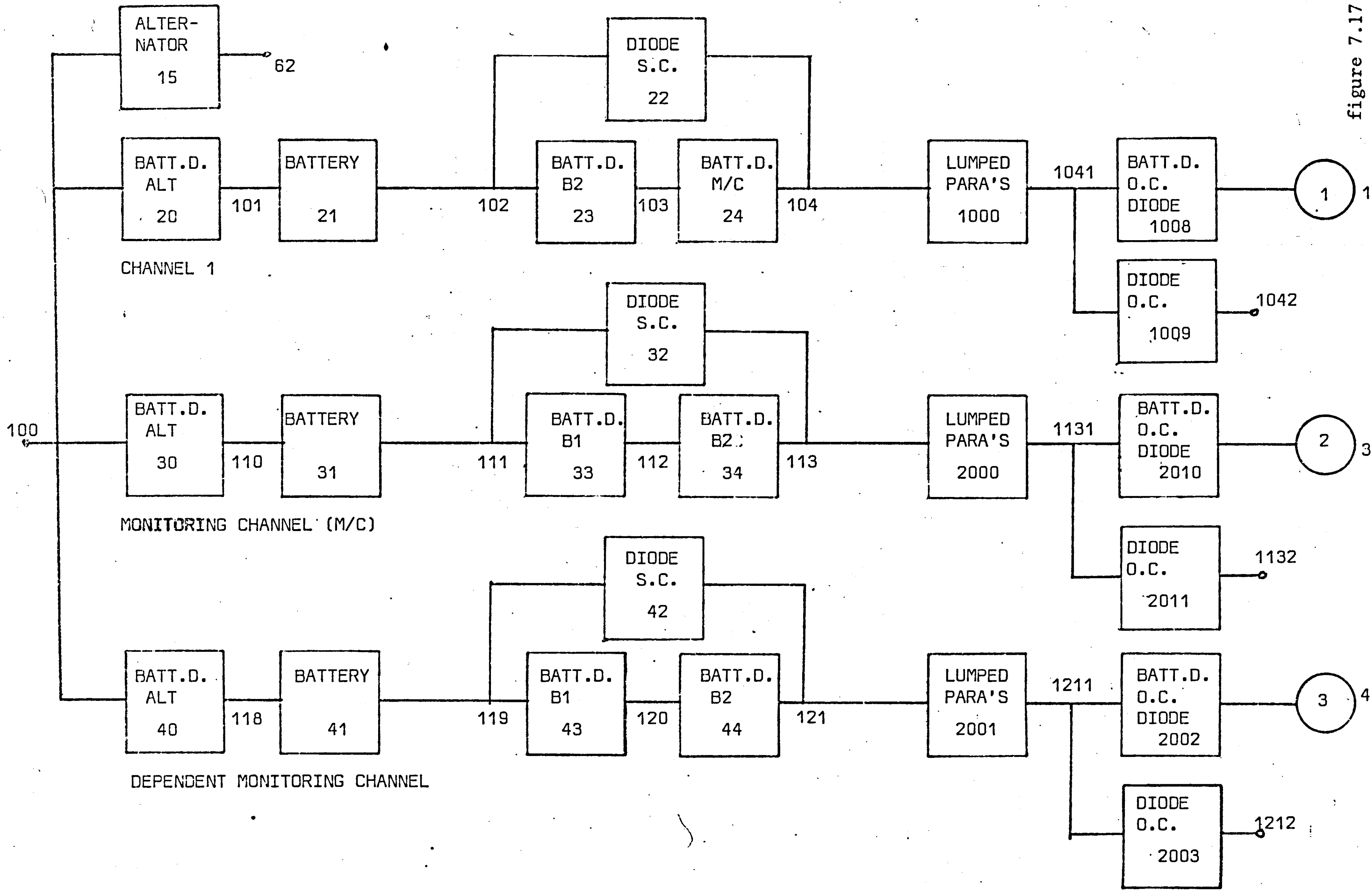


figure 7.17

CONTROL SYSTEM FOR DISABLED DRIVER CAR RELIABILITY BLOCK DIAGRAM (CONT)

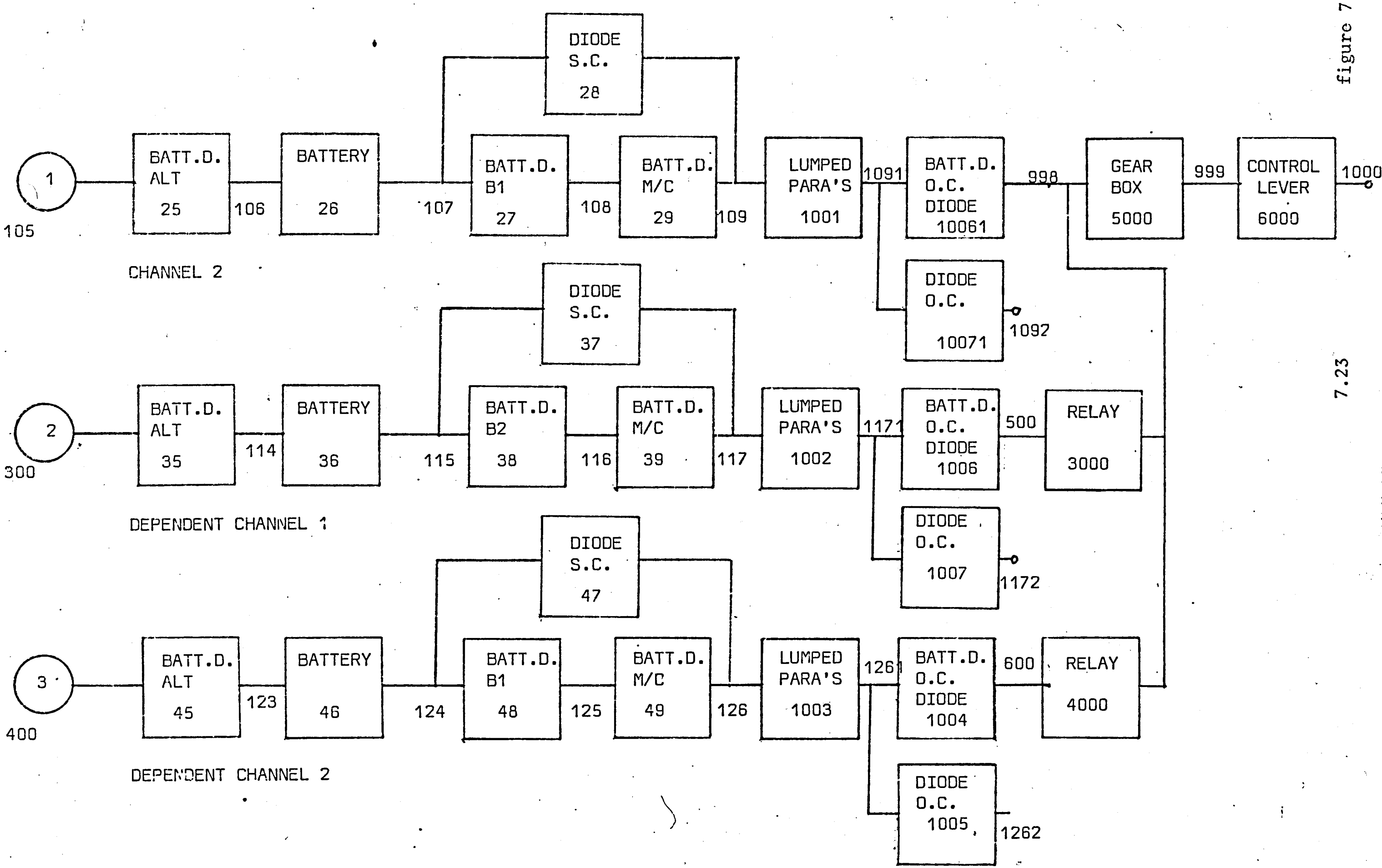


figure 7.17

```
NESTING STRUCTURE

NEST LEVEL  NEST REPLACEMENT NO  BLOCK NO
    1                0               1
ANALYSIS OF ROUTES FOR CONNECTION TABLE DEPICTED

BRANCH NUMBER NODE NUMBER NODE NUMBER  FAIL CODE  FAIL PARAMETERS  REPAIR CODE  REPAIR PARAMETERS  DEPENDENCY F & R
      15          100          62        1    2.0000& -4   0.0000&  0    3   1.0000& 6   0.0000& 0        0         0
      20          100         101        3    3.3000& -1   0.0000&  0    3   1.0000& 6   0.0000& 0       15         0
      21          101         102        1    1.0040& -3   0.0000&  0    3   1.0000& 6   0.0000& 0        0         0
      22          102         104        1    1.5000& -6   0.0000&  0    3   1.0000& 6   0.0000& 0        0         0
      23          102         103        3    3.3000& -1   0.0000&  0    3   1.0000& 6   0.0000& 0       26         0
      24          103         104        3    3.3000& -1   0.0000&  0    3   1.0000& 6   0.0000& 0       31         0
      25          105         106        3    3.3000& -1   0.0000&  0    3   1.0000& 6   0.0000& 0       15         0
      26          106         107        1    1.0040& -3   0.0000&  0    3   1.0000& 6   0.0000& 0        0         0
      27          107         108        3    3.3000& -1   0.0000&  0    3   1.0000& 6   0.0000& 0       21         0
      28          107         109        1    1.5000& -6   0.0000&  0    3   1.0000& 6   0.0000& 0        0         0
      29          108         109        3    3.3000& -1   0.0000&  0    3   1.0000& 6   0.0000& 0       31         0
      30          100         110        3    8.3000& -2   0.0000&  0    3   1.0000& 6   0.0000& 0       15         0
      31          110         111        1    1.0040& -3   0.0000&  0    3   1.0000& 6   0.0000& 0        0         0
      32          111         113        1    1.5000& -6   0.0000&  0    3   1.0000& 6   0.0000& 0        0         0
      33          111         112        3    8.3000& -2   0.0000&  0    3   1.0000& 6   0.0000& 0       21         0
      34          112         113        3    8.3000& -2   0.0000&  0    3   1.0000& 6   0.0000& 0       26         0
      35          300         114        3    3.3111& -1   0.0000&  0    3   1.0000& 6   0.0000& 0       15         0
      36          114         115        3    1.0000& -6   0.0000&  0    3   1.0000& 6   0.0000& 0       21         0
      37          115         117        3    1.0000& -6   0.0000&  0    3   1.0000& 6   0.0000& 0       22         0
      38          115         116        3    3.3111& -1   0.0000&  0    3   1.0000& 6   0.0000& 0       26         0
      39          116         117        3    3.3111& -1   0.0000&  0    3   1.0000& 6   0.0000& 0       31         0
      40          100         118        3    8.3110& -2   0.0000&  0    3   1.0000& 6   0.0000& 0       15         0
      41          118         119        3    1.0000& -6   0.0000&  0    3   1.0000& 6   0.0000& 0       31         0
      42          119         121        3    1.0000& -6   0.0000&  0    3   1.0000& 6   0.0000& 0       32         0
      43          119         120        3    8.3110& -2   0.0000&  0    3   1.0000& 6   0.0000& 0       21         0
      44          120         121        3    8.3110& -2.  0.0000&  0    3   1.0000& 6   0.0000& 0       26         0
```

figure 7.18

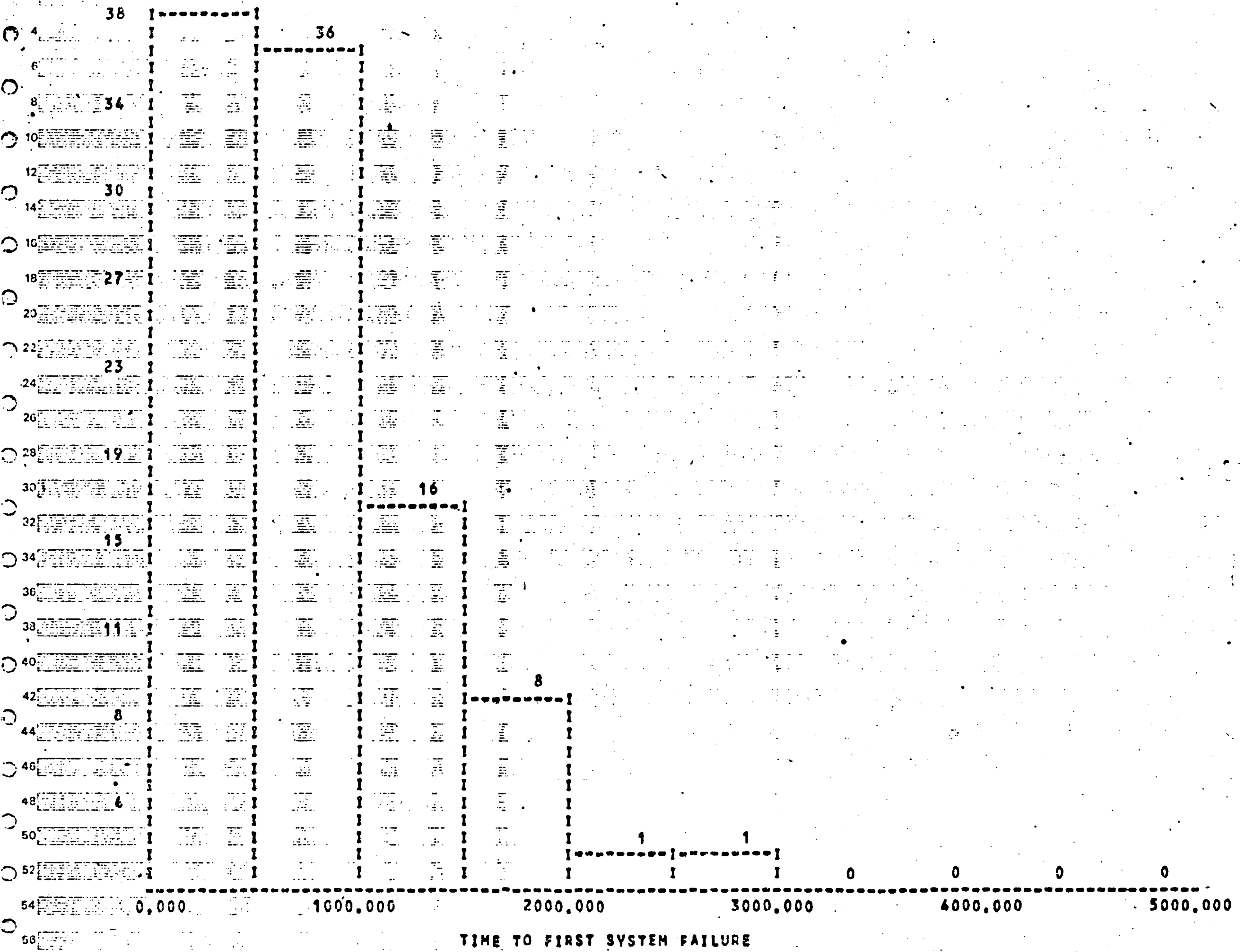
```
   45     400      123     3  3.3120& -1  0.0000&  0   3  1.0000&  6  0.0000&  0      15      0
   46     123      124     3  1.0000& -6  0.0000&  0   3  1.0000&  6  0.0000&  0      26      0
   47     124      126     3  1.0000& -6  0.0000&  0   3  1.0000&  6  0.0000&  0      28      0
   48     124      125     3  3.3110& -1  0.0000&  0   3  1.0000&  6  0.0000&  0      21      0
   49     125      126     3  3.3110& -1  0.0000&  0   3  1.0000&  6  0.0000&  0      31      0
 1000     104     1041     1  6.7980& -5  0.0000&  0   3  1.0000&  6  0.0000&  0       0      0
 1001     109     1091     1  6.7980& -5  0.0000&  0   3  1.0000&  6  0.0000&  0       0      0
 1002     117     1171     3  1.0000& -6  0.0000&  0   3  1.0000&  6  0.0000&  0    1000      0
 1003     126     1261     3  1.0000& -6  0.0000&  0   3  1.0000&  6  0.0000&  0    1001      0
 1004    1261      600     3  3.3100& -1  0.0000&  0   3  1.0000&  6  0.0000&  0   10071      0
 1005    1261     1262     3  1.0000& -6  0.0000&  0   3  1.0000&  6  0.0000&  0   10071      0
 1006    1171      500     3  3.3100& -1  0.0000&  0   3  1.0000&  6  0.0000&  0    1009      0
 1007    1171     1172     3  1.0000& -6  0.0000&  0   3  1.0000&  6  0.0000&  0    1009      0
 1008    1041      105     3  3.3000& -1  0.0000&  0   3  1.0000&  6  0.0000&  0    1009      0
 1009    1041     1042     1  1.5000& -6  0.0000&  0   3  1.0000&  6  0.0000&  0       0      0
 2000     113     1131     1  3.6480& -5  0.0000&  0   3  1.0000&  6  0.0000&  0       0      0
 2001     121     1211     3  1.0000& -6  0.0000&  0   3  1.0000&  6  0.0000&  0    2000      0
 2002    1211      400     3  8.3100& -2  0.0000&  0   3  1.0000&  6  0.0000&  0    2011      0
 2003    1211     1212     3  1.0000& -6  0.0000&  0   3  1.0000&  6  0.0000&  0    2011      0
 2010    1131      300     3  8.3000& -2  0.0000&  0   3  1.0000&  6  0.0000&  0    2011      0
 2011    1131     1132     1  1.5000& -6  0.0000&  0   3  1.0000&  6  0.0000&  0       0      0
 3000     500      998     1  3.9000& -6  0.0000&  0   3  1.0000&  6  0.0000&  0       0      0
 4000     600      998     1  3.9000& -6  0.0000&  0   3  1.0000&  6  0.0000&  0       0      0
 5000     998      999     3  1.0000&  6  0.0000&  0   3  1.0000&  6  0.0000&  0       0      0
 6000     999     1000     3  1.0000&  6  0.0000&  0   3  1.0000&  6  0.0000&  0       0      0
10061    1091      998     3  3.3000& -1  0.0000&  0   3  1.0000&  6  0.0000&  0   10071      0
10071    1091     1092     1  1.5000& -6  0.0000&  0   3  1.0000&  6  0.0000&  0       0      0

NUMBER OF ROUTES FOR SYSTEM TO OPERATE      1
```

figure 7.18

```
ANALYSIS OF BRANCH FAILURES AND REPAIRS
NEST LEVEL     1
BRANCH NO   SYSTEM FAILURES    BRANCH     SYSTEM REPAIRS      BRANCH
            CAUSED BY BRANCH  FAILURES   CAUSED BY BRANCH    REPAIRS
       15               0        100               0               0
       20               6        100               0               0
       21              14        100               0               0
       22               1         17               0               0
       23               0        100               0               0
       24               0        100               0               0
       25               0        100               0               0
       26              14        100               0               0
       27               0        100               0               0
       28               0         12               0               0
       29               0        100               0               0
       30               3        100               0               0
       31               8        100               0               0
       32               0         14               0               0
       33               0        100               0               0
       34               0        100               0               0
       35               0        100               0               0
       36              19        100               0               0
       37               0         17               0               0
       38               0        100               0               0
       39               0        100               0               0
       40               3        100               0               0
       41              16        100               0               0
       42               0         14               0               0
       43               0        100               0               0
       44               0        100               0               0
       45               0        100               0               0
       46              13        100               0               0
       47               0         12               0               0
       48               0        100               0               0
       49               0        100               0               0
     1000               0        100               0               0
     1001               1        100               0               0
     1002               1        100               0               0
     1003               0        100               0               0
     1004               0         10               0               0
     1005               0         10               0               0
     1006               0         11               0               0
     1007               0         11               0               0
     1008               0         11               0               0
     1009               0         11               0               0
     2000               1         97               0               0
     2001               0         97               0               0
     2002               0          7               0               0
     2003               0          7               0               0
     2010               0          7               0               0
     2011               0          7               0               0
     3000               0         31               0               0
     4000               0         31               0               0
     5000               0          0               0               0
     6000               0          0               0               0
    10061               0         10               0               0
    10071               0         10               0               0

                      100       3354               0               0
```

figure 7.19

FREQUENCY DIAGRAM OF TIMES TO FIRST SYSTEM FAILURE



```
NO OF MISSIONS   100     MISSION TIME   100000  UNITS    OVERALL MEAN 7,5516E  2    MIN 1,1426E  2    MAX 2,6445E  3
STANDARD DEVIATION 4,9068E  2
NO OF MISSIONS WITH NO SYSTEM FAILURE     0
```

figure 7.20

CONTROL SYSTEM FOR DISABLED DRIVER CAR RELIABILITY BLOCK DIAGRAM

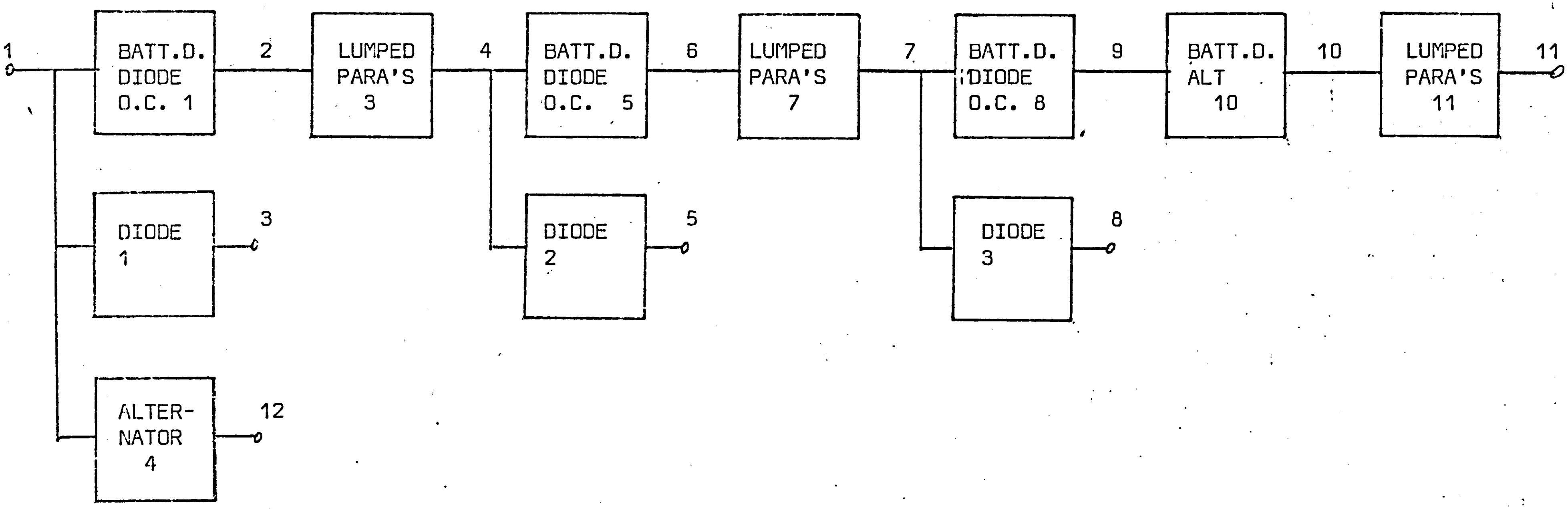


figure 7.21

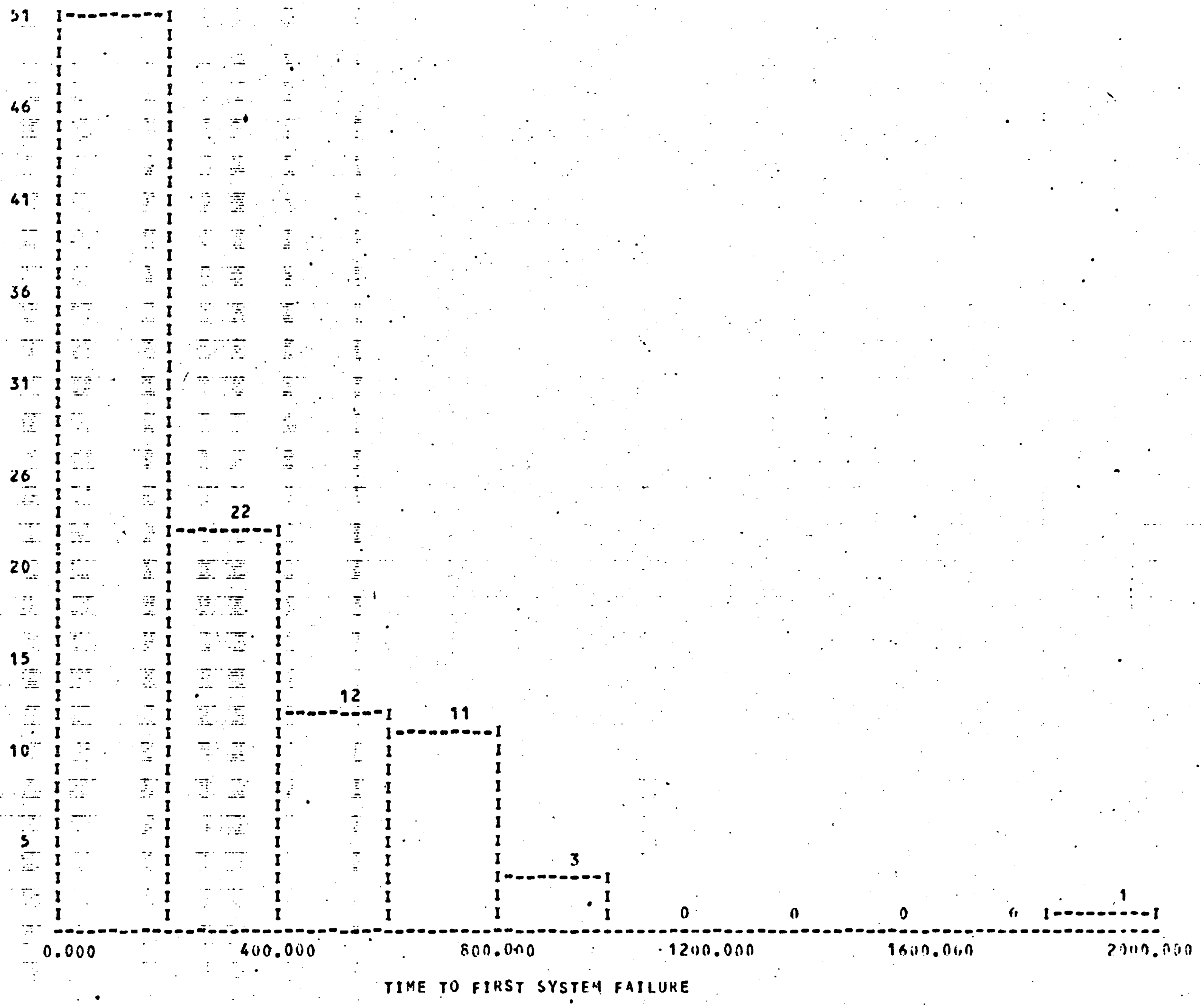
```
NESTING STRUCTURE

NEST LEVEL  NEST REPLACEMENT NO  BLOCK NO
    1              0                1
ANALYSIS OF ROUTES FOR CONNECTION TABLE DEPICTED


BRANCH NUMBER NODE NUMBER NODE NUMBER  FAIL CODE  FAIL PARAMETERS  REPAIR CODE  REPAIR PARAMETERS  DEPENDENCY F & R
     1             1           2          3    3.3000& -1   0.0000&  0     3    1.0000&  6   0.0000&  0      2        0
     2             1           3          1    1.5000& -6   0.0000&  0     3    1.0000&  6   0.0000&  0      0        0
     3             2           4          1    1.0735& -3   0.0000&  0     3    1.0000&  6   0.0000&  0      0        0
     4             1          12          1    2.0000& -4   0.0000&  0     3    1.0000&  6   0.0000&  0      0        0
     5             4           6          3    3.3000& -1   0.0000&  0     3    1.0000&  6   0.0000&  0      6        0
     6             4           5          1    1.5000& -6   0.0000&  0     3    1.0000&  6   0.0000&  0      0        0
     7             6           7          1    1.0735& -3   0.0000&  0     3    1.0000&  6   0.0000&  0      0        0
     8             7           9          3    8.3000& -2   0.0000&  0     3    1.0000&  6   0.0000&  0      9        0
     9             7           8          1    1.5000& -6   0.0000&  0     3    1.0000&  6   0.0000&  0      0        0
    10             9          10          3    8.3000& -2   0.0000&  0     3    1.0000&  6   0.0000&  0      4        0
    11            10          11          1    1.0420& -3   0.0000&  0     3    1.0000&  6   0.0000&  0      0        0



NUMBER OF ROUTES FOR SYSTEM TO OPERATE        1

LEVEL OF THIS NEST        1
```

figure 7.22

```
ANALYSIS OF BRANCH FAILURES AND REPAIRS
NEST LEVEL      1
BRANCH NO  SYSTEM FAILURES     BRANCH    SYSTEM REPAIRS     BRANCH
           CAUSED BY BRANCH   FAILURES  CAUSED BY BRANCH   REPAIRS
        1          0             20            0              0
        2          0             20            0              0
        3         29            100            0              0
        4          0            100            0              0
        5          0             18            0              0
        6          0             18            0              0
        7         30            100            0              0
        8          0             12            0              0
        9          0             12            0              0
       10          5            100            0              0
       11         36            100            0              0

                 100            600            0              0
```

figure 7.23

FREQUENCY DIAGRAM OF TIMES TO FIRST SYSTEM FAILURE



```
NO OF MISSIONS   100      MISSION TIME   100000  UNITS    OVERALL MEAN 2.86428  2    MIN 1.78088  0    MAX 1.94448  3
STANDARD DEVIATION 2.85364 .2
NO OF MISSIONS WITH NO SYSTEM FAILURE        0
```

figure 7.24

8. INTERACTION WITH USERS

This chapter is concerned with the users of the reliability program during the three year development phase. The type of user and the amount of his usage is detailed. The feedback from users is described both in terms of new facilities and critical comments ranging from minor dislikes to error reports.

8.1 THE ROLE OF THE USER

The development of the reliability prediction program RELY was structured so that it became possible to use the program within three months of starting the project. After a year it became feasible to allow distribution of the program to users outside Kingston Polytechnic. It was therefore at this stage that the users became an important explicit consideration where previously they had been implicit. Users were encouraged to make comments and suggestions and to this end frequent meetings were held with A.S.W.E., their subcontractors using the program, and the designers at Kingston. The total usage of the program for the two years of issue is estimated as in figure 8.1. The major users at R.A.E. used a 1904A and a 1906A. The students at Kingston used the program for part of their course work. No development time or use by the designers is included.

	Run Hours	Machine	Cost
A.S.W.E. Internal users	0.25 [NOTIONAL] 0.1 [MILL TIME]	1907 SOUTHAMPTON 1906A R.A.E.	£30 £30
A.S.W.E. Sub-contractors	10 [MILL TIME] 0.5 [NOTIONAL]	1904A & 1906A R.A.E. 1907 SOUTHAMPTON	£800 £50
Kingston Polytechnic students	10 [ELAPSE TIME]	1905 KINGSTON	-
Transport & Road Research Lab.	1 [ELAPSE TIME]	1903A E.R.A.	£30

figure 8.1

8.2 COMMENTS OF USERS

During the two years of usage, the comments of users requiring action from the designer can be divided into facilities requested and errors reported.

8.2.1 FACILITIES REQUESTED BY USERS

The facilities requested ranged from minor output title changes to complex structural modifications. Sometimes the changes were anticipated by the designers and incorporated. On other occasions users requested facilities that they had no immediate use for but felt that the program would not be complete without them. The majority of changes did not, however, fall into this last category.

The first improvements came in the area of data error reporting. Prior to general release, the only users were the program designers. But once inexperienced users started designing and coding input data it became clear that the standard of error reporting would have to be improved. A number of comprehensive error checks were incorporated and these can be seen in detail in figure 6.4 of chapter 6.

Once an input error reporting system had been devised it became clear that some form of user guide was required. A modular format was laid down and this formed the basis of all the documentation which is described fully in chapter 6.

One problem that had not been resolved by the time of the first program issue was the use of magnetic tape data files. While the program was only being used at Kingston, the run times for simulation were not critical as the computer time was free. This was not the case for outside users and it became clear that the file accessing was causing large time delays. As a result, a modular file accessing method was devised and this is described in chapter 4.2.9.

The first outside 1900 computer used was the 1904A at the Royal Aircraft Establishment. Installing the program on this machine presented no problems. However, the second outside 1900 computer, the 1907 at Southampton University, presented problems mainly due to the local operating system. A number of minor changes had to be made to the way the program halted because of errors in the data. The Southampton operating system provided remote job entry (R.J.E.) and in order to take full advantage of this, extra facilities were provided. R.J.E. permits a job (a computer run) to be entered into a batch processing stream from a terminal. Unlike timesharing the job will not be processed immediately but will go into the normal batch queue. When processed, the output can be listed on the terminal if desired. A summarised output format was devised and an option given to the user to allow this form of output in place of the normal full output. Additionally, an option was provided that allowed a full output of a previous run should a greater level of resolution be required on the evidence of the summarised output. This would have to be posted or collected unless a remote lineprinter was available.

When users had become familiar with the program and started getting meaningful results a general request for more comprehensive output was made.

The output was improved in the following ways:

i) The user's input data was output in a branch numerical order. This was required as a document and as a check against input errors.

ii) Route Analysis information provided. The routing information shows the weighting of each element determined by the number of routes it appears in. As well as this, information is available about the complexity of each route. This can be used to determine the topological importance of an element.

iii) A histogram of systems availability.

iv) Standard deviation of the output quantities, MTFF, MTBF, MRT, SFR and availability.

v) Choice of histogram ordinate maximum.

vi) Abscissa intervals scaled to factors of 2, 5 or 10.

vii) Branch simulation histories. This table, examples of which can be found in chapter 7, was requested as a means of identifying the vital branches in a system. The design examples in chapter 7 illustrate the importance of this table.

Chapter 2.3 describes the n out of m facility that is a useful modelling technique. The program facility was incorporated at the request of users who found the conventional method too tedious. Users made comments about the standby mechanism described in chapter 2.2.4. The conventional standby situation gave rise to a system failure, which some users wanted suppressed. By suggesting an alternative model a major program modification was avoided.

Random number generation caused many problems and arising out of these a request for a repeatable random number facility was made. This allowed users to repeat a run with the same random number stream removing this variable from the simulation. The problems are described in the next section.

When users attempted long simulations they found that the data files could not cope with the volume of data. Consequently a facility was provided that allowed users to extend their data files. This is described in chapter 4.2.9.

Towards the end of the project it was suggested by A.S.W.E. that spares estimation could usefully be incorporated into the program. A method for doing this using existing techniques evolved by A.S.W.E. was devised and is described in chapter 4.2.7.

The nesting facility, which is described in chapter 2.4, proved to be too unsophisticated to cater for the diverse outputs of a simulation. The users suggested an improved method and this was duly incorporated as described in chapter 4.2.6.

8.2.2 ERRORS REPORTED BY USERS

The major error reported by users concerned discrepancies in the number of expected events occurring during a simulation. After the provision of facility vii) of section 8.2.1 it became possible to compare the number of failures suffered by a branch with the expected number if:

i) The failure distribution is exponential.

ii) The repair time is small compared with failure time.

Chi squared tests were performed on the number of failures and in about 50% of cases the results failed the test with a confidence level of 90%. In some cases the observed number of failures differed from the expected number by as much as four standard deviations. The causes of the problem were twofold.

i) The technique used to generate the tabular distributions, detailed in chapter 4.2.3, stored values of f(t) of the cumulative function from 0 → 1 with corresponding t's of 0 and ∞. Because of the technique used where the F(t) range (0 → 1) is split into 50 equal intervals and linear interpolation performed, the slope of the last linear segment is very small. As a result almost all values of t generated in this area are very large and lie outside the mission time. (10^{20} was used as a representation of ∞.) Hence every 50th value of t obtained, biased the simulation by either causing very long repair time in a few cases thus reducing the amount of mission time available for failures, or causing very long failure times producing the same result. The users suggest an improved method, reference 30, which was incorporated.

ii) The second cause was due to the random number generator having too short a cycle. The generator used is a multiplicative congruent type and only had a bit length of 24. When this was increased to 48 the Chi squared test was passed with a confidence level of 90%.

When the program was installed at Southampton University it was discovered that an overlaid version would not work. Because of the size of program it became necessary to overlay it, a technique described in chapter 3.1. This problem seriously affected the entire use of the program because a non-overlaid version left a very small area for data. The fault was traced to the machine's overlay package being different from the one used at Kingston. The problem was solved by compiling the program at Southampton thus incorporating the overlay package of that machine.

9. FURTHER DEVELOPMENTS

The project has been devised to allow for continued development and updating of the software. by people who may not have had close contact with its early or detailed design. It was assumed at the outset that the modular software structure, coupled with standardised documentation, would allow for ease of development and maintenance.

In an attempt to prove this in the course of the project and thus illustrate the way in which further development may be carried out in the future, some of the program modules of RELY have been developed by:

i) adaptation of algorithms and software independently developed by others (spares optimisation).

ii) inviting a programmer to devise an algorithm and develop a program module according to set specifications.

Future developments of RELY range from providing minor modifications for greater convenience of users to incorporating powerful new features which would involve considerable research and development effort. Some suggestions, based on user feedback, are given here.

i) Providing information from the result of a simulation about the way an output quantity (time between failure, etc.) varies with time. This could be presented in the form of a histogram.

ii) Providing a facility that enables the user to terminate the mission after first failure for the case when mean time to first failure is the only output quantity of interest. By making repair a switch distribution that is longer than the mission time it is possible to do this at present provided an estimate can be made of M.T.F.F.

However, with complex systems this is not always easy and an automatic facility would be useful.

iii) Allow elements to be 'shut down' during a system failure, i.e., prevent them failing during this time.

This would require adding on to these elements a time equal to the duration of the system shut down time.

iv) Providing information about the number of working routes throughout a simulation, i.e., calculating whether the system spends most of the time with a high percentage or a low percentage of routes available. This would give an overall measure of the redundancy of a system.

v) Providing sensitivity analysis. This would guide the designer towards the most advantageous design modifications without the need for repeated and expensive simulation runs. The provision of this facility would represent a major undertaking to be supported by research and extensive algorithm development.

vi) Providing an assessment of the level of confidence to be associated with a set of results. This would permit the adjustment of the number of simulations according to the measure of accuracy required.

vii) Providing some degree of automation of the number of simulations according to the desired level of confidence and/or to the allowed cost of the simulation.

ix) Allowing dependent elements to be controlled by more than one independent element.

x) Providing a means for describing the failure and repair characteristics of an element as a numerical table.

10. OTHER RELATED PROJECTS

This topic can be divided into two areas of activity:

i) Projects inside the School of Electrical and Electronic Engineering at Kingston Polytechnic.

ii) Projects outside.

10.1 INTERNAL PROJECTS

The major related project is in Markov methods for the assessment of the performance of probabilistic systems. The aim of the project is to investigate the application of Markov models to certain types of stochastic processes. These include the processes governing system reliability. The main problem that has been encountered in the course of this project is the large number of states required by a reliability model of practical systems. This leads to very large matrices and the size problem at present prevents the application of Markov methods to all but the simplest reliability models.

Another project that relates indirectly to the reliability project is an investigation into structured programming and programme proving techniques.

This, however, would only affect the methodological aspects of the reliability program and not the facilities offered to reliability engineers.

10.2 EXTERNAL PROJECTS

This section is divided into three parts:

i) Survey of programs in U.K.

ii) Related projects as a result of liaison with other firms.

iii) Projects outside U.K.

i) A number of reliability computer programs exist in the U.K. Many of them are private to the company that developed them but their existence shows work in this area.

Smiths Industries Ltd.

A number of programs have been developed that perform analysis as opposed to synthesis.

RPRED8, SPRED2, TPRED1: Calculates the failure rate, MTBF and reliability using Smiths own data bank.

DUANE1: Reliability growth assessor.

RTEST2: Truncated sequential reliability test plans for exponential distribution using MIL 781B.

BRDAP2: Reliability block diagram analysis program.

MTBFAV: Calculates MTFF and availability from transition rate diagram.

PSIP4: Probability state interpretation program.

HILO: Strictly a logic simulator but some reliability comparisons may be made.

GEC Marconi

Only one program exists, which is fairly basic.

MIL: Calculates the failure rate and MTBF using MIL 217B.

REDAC Software Ltd.

One program is available as a bureau service.

REDAP25: Reliability prediction using MIL 217A.

Honeywell Information Systems

A simulator is available as a bureau service.

MCREC2: Monte Carlo simulation.

Easams Ltd.

AWAZUV: Analytical reliability calculation.

WAZUST: Simulation of an aircraft mission.

U.K.A.E.A. Risely

NOTED: Analytical reliability calculation.

RASP: Statistical evaluation of reliability data.

ITTE

STCML: Monte Carlo simulation.

COMSYS: Markov reliability prediction.

ii) Reliability prediction is being carried out by the Ordnance department of the Navy at Bath. As a result of this work, a program has been developed for reliability assessment. The sponsors of the Kingston Project, A.S.W.E., have taken this program and assessed it along with RELY. The results of the comparison are given in reference 20.

iii) A large number of projects into reliability are being carried out both in Europe and U.S.A. Details of these can be found in the bibliography section.

11. BIBLIOGRAPHY AND REFERENCES

1) S.K. Banerjee and K. Rajamani
Optimization of System Reliability Using a Parametric Approach.
I.E.E.E. Trans Reliability, Vol. R22, April 1973.
The paper details a new method for optimization using a parametric approach with a classical non-linear programming technique.

2) H.S. Blanks
A Review of New Methods and Attitudes in Reliability Engineering.
Proc. I.R.E.E., April 1972.
This paper examines current reliability prediction techniques with particular reference to BS 9000.

3) R. Brewer
A Personal View of some Reliability Activities in the United Kingdom.
I.E.E.E. Trans Reliability, Vol. R23, August 1974.
This paper described the efforts in Reliability Prediction in Britain in the areas of Industrial, Military and Academic Institutions.

4) G.E. Brown and H.C. Rutemiller
Evaluation of Pr[x $\geqslant$ y] when both x and y are from three parameter Weibull Distributions.
I.E.E.E. Trans Reliability, Vol. R22, June 1973.
This paper develops a computer algorithm for determining the probability that the failure time of an element from one population will exceed that of a second element from another population.

5) J.A. Buzacott
Markov Approach to Finding Failure Times of Repairable Systems.
I.E.E.E. Trans Reliability, Vol. R19, Nov. 1970.
This paper describes the classical Markov method for solving reliability problems.

6) J.A. Buzacott
Network Approach to finding the Reliability of Repairable Systems.
I.E.E.E. Trans Reliability, Vol. R19, Nov. 1970.
This paper describes a network (block diagram) method based on the classical Markov method described in 5).

7) S.J. Cyvin
Generation of Normal Distribution Function.
Algorithm 226, Collected Algorithms from CACM.

8) D.Y. Downham and F.D. Roberts
Multiplicative Congruential Pseudo-Random Number Generators.
Computer Journal, 1967, Vol. 10, p. 74.
This paper describes 6 types of generator and draws conclusions about the suitability of each type.

9) L.R. Doyon and M.W. Berssenbrugge
Solving Complex Reliability Models by Computer.
Proc. Annual Sym. Reliability Boston U.S.A. 1968
This paper discusses the use of Markov methods for use with Reliability problems.

10) D.B. Gilmore
Application of a Monte Carlo Method to the Determination of Mean Time to Failure of Complex Systems.
Proc. Annual Sym. Reliability Boston U.S.A. 1968
This paper describes a computer orientated method for Monte Carlo simulation for determining M.T.T.F.

11) A.E. Green
The Systems Reliability Service and its Generic Techniques.
I.E.E.E. Trans Reliability, Vol. R23, August 1974.
This paper describes the activities of the S.R.S. of the U.K.A.E.A.

12) A.E. Green and A.J. Bourne
Reliability Technology.
Wiley.
Basic reference book for Reliability Analysis.

13) D.R. Holmes
PROLOG: Logic Analysis System
School of Electrical Engineering Report No. 5.

14) L. Tin Htun
Reliability Prediction Techniques for Complex Systems.
I.E.E.E. Trans Reliability, Vol. R15, August 1966.
This paper describes a method for reliability prediction by means of transition diagrams in state space.

15) T.E. Hull and A.R. Dobell
Mixed Congruential Random Number Generators for Binary Machines.
Journal of A.C.M., Vol. 11, 1964, p. 31.
This paper describes the principles of operation of this class of random number generators and shows the results of comparisons between other types.

16. T. Ito and C. Kavaguchi
Reliability of Special Redundant Systems Considering Exchange Time and Repair Time.
I.E.E.E. Trans Reliability, Vol. R.20, February 1971.
The paper examines the maths of a 2 out of 3 redundant system with limited repair which is not utilised instantaneously.

17. G. James
Mathematics Dictionary
Van Nostrand.

18. B. Koen and A. Carnino
Pattern Recognition and List Processing Methods for Systems Represented by Fault Trees.
I.E.E.E. Trans Reliability, Vol. R23, April 1974.
This paper describes a novel way of handling the large matrices generated by the Markov method. However it is restricted to systems of independent elements.

19. H.G. Kuehn
A 48 Bit Pseudo-Random Number Generator.
Comm. A.C.M. Vol. 4, p. 350 (1961)
This paper describes a random number generator and gives the results of tests performed using it.

20. B. Lowrey
Interim report on the use of the Reliability program RELY.
Rex Thompson and Partners.
This report describes the experience gained from two years use of the Reliability program RELY. A comparison is made with an internal Admiralty computer program.

21. L.W. Lukis
Reliability Assessment: Myths and Misuse of Statistics.
Microelectronics and Reliability, Vol. 11, 1972.
This paper argues about statistics in Reliability Assessment where it used to support bad data and wrong assumptions.

22. G. Menicou and L. Van Os
Comprehensive Computer Programs for Solving Complex Reliability Problems.
Electrical Communication, Vol. 48, No. 3, 1973.
The paper describes the ITT Suite of Reliability Programs.

23. E.J. Muth
A Method for predicting System Downtime.
I.E.E.E. Trans Reliability, Vol. R17, June 1968.
Equations are developed for prediction assuming exponential distributions.

24. A.C. Nelson, J.R. Batts and R.L. Beadles
A Computer Program for Approximating System Reliability.
I.E.E.E. Trans Reliability, Vol. R19, May 1970.
The algorithms used in a program that evaluates cut sets and success paths are described.

25. S. Osaki
Reliability Analysis of a Two Unit Standby-Redundant System with Preventive Maintenance.
I.E.E.E. Trans Reliability, Vol. R21, February 1972.
The maths of the above situation are worked through and equations developed that describe the situation.

26. S. Osaki
On a Two Unit Standby Redundant System with Imperfect Switch-over.
I.E.E.E. Trans Reliability, Vol. R21, February 1972.
The maths of the above situation are worked through and equations developed that describe the situation.

27. H.O. Posten
The Robustness of Reliability Predictions for Series Systems of Identical Components.
I.E.E.E. Trans Reliability, Vol. R22, October 1973.
Theoretical paper concerned with using an exponential distribution for elements described by the Weibull family of distributions.

28. B. Sayers
Reliability Assessment Techniques and their Use in Improving Reliability.
I.E.E.E. Colloquium on Reliability, April 1975.
The paper describes a case study of a chemical process using the Systems Reliability Service of the U.K.A.E.A.

29. R.S. Scowen
QUICKERSORT
Algorithm 271: Collected Algorithms from CACM

30. J.M. Sheppard and J.E. Fichling
A Discussion of Errors Arising in the Distributions Currently Used in the Kingston Reliability Program.
Rex Thompson and Partners, RTP/1019/01, November 1974.
This report examines the program RELY and gives user comments of the errors resulting from the program.

31. M.L. Shooman
Probablistic Reliability: An Engineering Approach.
McGraw Hill.
Basic reference book for Reliability Analysis.

32. M.L. Shooman
Equivalence of Reliability Diagrams and Fault Tree Analysis.
I.E.E.E. Trans Reliability, Vol. R19, May 1970.
This paper shows the boolean duality that exists between two representations of a reliability problem.

33. J.E. Staley and P.S. Sutcliff
Reliability Block Diagram Analysis.
Microelectronics and Reliability, Vol. 13, 1974.
This paper discusses the application of the Markov method with emphasis on simplification methods.

34. J.P. Vedel
La Conception de Projet Assistee par Ordinateur.
Proc AGARD Conference on C.A.C.D. Copenhagen, May 1973.
This paper describes the EPAC system developed by the French Army and the University of Paris.

35. G.A. Weber
State of Reliability Effort in Europe.
I.E.E.E. Trans Reliability, Vol. R23, August 1974.
The paper gives an overall view of reliability analysis in Europe.

36. G.A. Mihram
Simulation- Statistical Foundations and Methodology
Academic Press, 1972

37. J.M. Hammesley and D.C. Hardscomb
Monte Carlo Methods
Methuen, 1964

38. MSc Course Notes, Module III (Simulation), Vol 1, Part 1 and Part II.
Kingston Polytechnic School Of Electrical and Electronic Engineering, October 1975

Appendix i: Conferences and publications

Conferences

In the course of this research the opportunity arose to present aspects of this work at conferences. These occasions have allowed the assessment of interest in this work and have led to valuable contacts with potential users whose views and comments were utilised in subsequent planning of the project.

1. AGARD Conference on Computer Aided Circuit Design. May 1973, Lyngby, Denmark.
   AGARD is an advisory group of NATO. Because of the international nature of NATO, papers were presented by authors from most of the NATO countries. A report on the conference by the author appeared in C.A.D. magazine and is reprinted in the publications section of this appendix. The purpose of attending the conference was to read a joint paper by the two supervisors of this project and to answer technical queries about the project.

2. I.E.E. Conference on Computer Aided Design, Southampton University, April 1974.
   The paper describes the approach to CAD developed at the Kingston Polytechnic School of Electrical and Electronic Engineering and shows the use of the approach in the context of the development of RELY. An example of a communication system is shown, together with the reliability assessment achieved by RELY, demonstrating the main features of the program at the time of the presentation of the paper.

3. CAD 74 Conference, Imperial College, London, September 1974.
   The synopsis of a paper, jointly read at the Conference by the author and one of the supervisors, is reprinted in the publications section of this appendix. The paper presents a practical design example and illustrates the use of RELY in developing design strategies for evolving an acceptable compromise between reliability performance and cost of a system. The input - output data formats of RELY are illustrated.

Publications

Apart from the conference papers described earlier, the project gave risc to a series of research reports held in the library of ASWE and of the Kingston Polytechnic which indicate the development of the project over the years.

Computer-aided circuit design

21st - 24th May 1973, Lyngby, Denmark. 25th Technical Meeting of the Avionics Panel of AGARD (Advisory Group for Aerospace Research and Development)

150 representatives from the NATO countries met at the Technical University of Denmark at Lyngby, near Copenhagen. Of the 86 papers received, 35 were accepted. The opening address was given by Mr. R. Voles, the programme chairman, who outlined the theme of the conference, namely, to bridge the gap between the expert in C.A.D. and the prospective user.

The conference was divided into seven sessions covering topics in reliability, modelling, microwave, analogue, digital and layout. The first paper was given by Dr. Lindberg of the Institute of Circuit Theory and Communication at the Technical University of Denmark. He described the approach to the teaching of C.A.D. in Denmark and made reference to a number of programs developed in the Institute. Two in particular, ANP3 a linear circuit analysis program, and NAP2 a nonlinear analysis program, were demonstrated. Some time was made available on the 370/165 at the University for AGARD representatives to try out these programs.

Also given in the first session was a paper on the economics of c.a.d. by Mr. A. Llewelyn, the director of the CAD Centre, Cambridge. The system in operation was described and emphasis put on the needs of the industrial users. Mr. Llewelyn was asked if the limits imposed by PO land lines had affected the capacity of the system. He replied by saying that the proposed new trunk network should alleviate the situation.

The second session devoted to reliability contained two papers on reliability prediction, one on reliable design and one on reliability or yield. The prediction papers were both sponsored by defence establishments in conjunction with academic institutions. A users view of c.a.d. related to reliability, N.A. Walter, Admiralty Surface Weapons Establishment, and Dr. A.A. Kaposi, Kingston Polytechnic, described the particular problems encountered by the Navy. A modular prediction method was detailed.

Professor J.P. Vedel, University of Paris, described work of a similar nature being sponsored by the French Army.

Five papers were presented in the modelling session. The main theme was to try and improve the validity of parameters in existing models by different techniques. Dr. Mertens, University of Leuven, Belgium, described a method of taking physical parameters of a bipolar transistor and producing C.A.D. parameters from this. Captain Covello, USAF, gave details of a method of obtaining the nonlinear parameters of an n-p-n bipolar transistor by successive iteration on known linear parameters.

The analogue session was represented by papers on filters, i.e. measurements and large-scale circuit analysis. Dr. Van der Ouderaa, Philips Eindhoven, described a program called ICAN which predicts the a.c. behaviour of a bipolar i.c. Inputs used are the technological data, geometric dimensions and nodal connections. This program is aimed at device designers as the dimensions and diffusion techniques would not normally be known by circuit designers. Mr. W. Hochwald, Rockwall International USA, described a system of programs called SYSCAP/SELECT, currently available from CDC/Cybernet. The system is able to handle input in the form of lumped parameters, discrete parameters and functional blocks. The stimuli can be electrical, mechanical, hydraulic or aerodynamic. All the usual facilities are available together with some useful additions. These include the effect of component failure on each node, stress analysis and open and closed loop responses. When asked if there was any backup service for users of the system Mr. Hochwald said that local centres had analysts who were able to assist and as a last resort Rockwall International could be consulted. A stand-in paper was presented in this session by Mr. K.G. Nichols, Southampton University, who described a system which has been implemented on a minicomputer. The paper outlined the advantage of C.A.D. on a mini and stimulated a lively discussion on the subject.

The session devoted to digital started with a paper given by Williams, Scott and Beaumont of Bell, Ottawa, on a high level graphical language called Grapple. It is designed to enable users of a graphical display to interact with the device by means of a very simple command language. This removes all the tedium of writing graphical display software and also avoids the use of databases. The language can be extended as required by a compiler-compiler technique. Professor D. Lewin, Brunel University, presented a paper on design languages for logic systems and this was followed by a paper given by K. Juergensen on the NASA computer aided design and test system.

The CADAT system enables the design, layout, evaluation and testing of l.s.i.c.'s. It is designed to be used by contractors to NASA as well as NASA themselves. C.S. Meyer of Motorola Phoenix, USA, then described the design system used by Motorola for l.s.i.c.'s.

The final session was devoted to layout. W. E. Hillier, Redac Software, Tewkesbury, described the p.c.b. design package that is available from his company. Two papers were then given by representatives of ICL, West Gorton. One, by H. G. Adshead described the work of the design automation group within ICL. The techniques developed by this group are used in the design of multilayer p.c.b.'s. The SYSTEM 71 facility was described. This system is used to produce the master logic drawings and n.c. tape necessary for the production of p.c.b.'s. The paper gave an interesting insight into the use of design automation within a large company. The second paper, presented by Dr. R. W. McGuffin, was concerned with the placement and routing of high-density chip interconnections. A major consideration was the positioning of chips to produce short propagation times on critical paths. Dr. D. Oestreichter then gave a paper on a parallel p.c.b. design system. The system used a parallel method of solving the assignment, placement and layout problem associated with p.c.b. design.

Finally W. M. Gaddes, IBM New York, gave a paper on Design Automation. This described the guidelines used by IBM to implement a design automation system.

The conference proved to be very successful. The standard of the papers was high without becoming too detailed. From a C.A.D. point of view the majority of representatives felt very encouraged at the progress that has been made.

C. Partridge.

Reproduced from C.A.D. Magazine Oct.1973

CAD for reliability assessment

A.A. Kaposi & D.C. Partridge

This contribution discusses the problems associated with the assessment of reliability of complex technological systems and proposes a CAD method which has the potential of solving most of them. It presents the current state of development of such a method giving details of

- current capabilities of the method
- the structure of the software system
- the algorithms used
- the modes of implementation of these algorithms
- the computer facilities used
- the plans for future development

An example is included demonstrating the capabilities of the method, showing the input/output data format and the cost of computation.

Features of the method

The paper argues and aims to demonstrate that the difficult and diffuse problems associated with complex technological systems are most conveniently solved by modular software which affords, among other advantages

- choice of combination of modules to suit individual problems
- efficient use of storage space by partitioning the data and keeping in use only the relevant parts for operating modules
- ease of software system and program development
- ease of extension modification and maintenance of software
- ease of documentation

At its present state of development the CAD method may be used to solve reliability problems

- at the feasibility study, design and maintenance stage
- of complex technological systems at sub-system, equipment, assembly or component level
- of systems containing any form of redundancy (standby, n-out-of-m, parallel, etc)
- of systems liable to variable modes of failure and repair

- of assessment of the systems sensitivity to element reliability

At the present time reliability assessment is based upon statistical methods using Monte Carlo techniques. Future plans include the incorporation of probabilistic methods.

Background

The project, sponsored by the Admiralty, is part of the research at the School of Electrical and Electronic Engineering of the Kingston Polytechnic, directed at the development of a unified approach to the use of CAD.

IEE Conference on CAD, Southampton, 1974.

CAD for reliability improvement

A.A. Kaposi[*] & C.D. Partridge[**]

Most people concerned with devising or using CAD have experience with programs whose facilities fail to keep in touch with changing demands of a dynamic industrial environment. This paper seeks to describe the development of a user-oriented CAD program which has specifically been designed for ease of modification or expansion. The program, originally intended for the purpose of reliability prediction, has recently been extended to include facilities which help designers in improving system reliability. The paper illustrates the use of RELY in this context with the aid of a practical design problem.

* Kingston Polytechnic

** Kingston Polytechnic and A.S.W.E.

CAD 74, London, 1974.

Appendix ii: List of symbols and abbreviations

ACM:	Association for Computing Machinery
ASWE:	Admiralty Surface Weapons Establishment
BS9000:	British Standard Nine Thousand
CACM:	Communications of the Association for Computing Machinery
CAD:	Computer Aided Design
f:	failure rate
IEEE:	Institute of Electrical and Electronic Engineers
IREE:	Institute of Radio and Electronic Engineers
λ:	Failure rate of exponentional distribution
MIL 217:	United States Defence Standard
MRT:	Mean Repair Time
MTBF:	Mean Time Between Failure
MTFF:	Mean Time to First Failure
MTTF:	Mean Time to Failure
PCV:	Pressure Control Valve
ppmh:	parts per million hours
RAE:	Royal Aircraft Establishment
RJE:	Remote Job Entry
SFR:	System Failure Rate
TRRL:	Transport and Road Research Laboratory

Appendix iii: Definitions

The definitions given relate to the context in which the term is used in the text. The following references were used in compiling the definitions.

i) M. Shooman: Probabilistic Reliability: McGraw Hill.
ii) A.E. Green & A.J. Bourne: Reliability Technology: Interscience.
iii) G. James & R. James: Mathematics Dictionary: Van Nostrand.
iv) D.W. Ballentyre & D.R. Lovett: A Dictionary of named effects and laws in Chemistry, Mathematics and Physics: Chapman and Hall.
v) M.E. Kendall & W.R. Buckland: A Dictionary of Statistical terms: Oliver and Boyd.
vi) Chambers Dictionary of Science and Technology.
vii) P.B. Jordain: Condensed Computer Encyclopedia: McGraw Hill.
viii) IFIP Guide to Concepts and Terms in Data Processing: North Holland.

ALGOL 60: An algorithmic and procedure-oriented machine language used principally in the programming of scientific problems.

Assembler: The generic term for the low level language of a particular computer. Also used to denote the computer program that translates the language into machine executable code.

Availability: The percentage of time a system is functioning compared with the total time it could function if it were perfect.

Backing Store: A means of storing data external to the main computer itself,but accessible to the program.

Branch: A two terminal model representating a physical device, forming an element of a reliability block diagram.

Chi Squared Test: A test of compatibility of observed and expected frequencies of occurrence of events.

Compiler: A computer program designed to translate a high level language into a machine language.

Connection Matrix: A rectangular array that displays the information held in a connection table, see below, by assigning a column to each node of the network and a row to each branch of the network. Connection of a branch to a node is represented by an entry in the intersecting square.

Connection Table: A table describing the topology of a system as a list of the branches and their associated nodes.

Contact Network: A Boolean model, used in Boolean systems and reliability studies, of a physical system where each element is represented by a contact that can be in one of two states; made or broken.

Cumulative Distribution: A representation of the distribution function F(x) of a variate x where F(x) is the total frequency of members with variate values less than or equal to x.

Elapse Time: The total time taken by a computer system to complete a run of a program.

Element: One or more branches in a reliability block diagram.

Fault Tree: A model of a physical system that describes the possible fault conditions which may cause the failure of the system.

Failed: One of the two possible reliability states in which a system may exist, namely, when it fails to meet its required specification.

Free Failure: One of three failure modes in mechanical engineering systems, namely, where the output cannot be controlled by the input.

Free Running Clock: A pulse generator not dependent on any exterior timing signals.

Functioning: One of the two possible reliability states in which a system may exist, namely, when it meets its required specification.

High Level Language: A computer programming language that is machine independent, e.g. ALGOL 60.

Language (Programming): An artificial language specifically designed for expressing a schedule of actions proposed in order to achieve some desired result.

Library File: A collection of standard routines held on a storage medium.

Linear Interpolation: The process of finding an approximate value of a continuous function between two adjacent discrete values by assuming linearity in the interval.

Locked Failure: One of three failure modes in mechanical engineering systems, namely, where the output remains fixed and does not respond to input.

Logic Gate Model: A model of a physical system where the state of Boolean input signals represent the reliability state of each element and logic gates are used to generate an output signal representing the reliability state of the system.

Markov Process: A probabilistic process in which development of the process depends entirely on its state at any one time and not on how that state arose. This is known as a first order Markov Process.

Mean Repair Time: The mean of all times between a failure and subsequent repair.

Mean Time Between Failure: The mean of all times between a failure and the next failure.

Mean Time to First Failure: The mean of all times to the first system failure.

Mean Time to Failure: The mean of all times between a repair and subsequent failure.

Mill Time: The amount of central processor time taken by a computer in running a program.

Mission Time: A pre-determined real-time interval representing the working life of a system between subsequent renewals or overhauls and simulated in the course of a single simulation run.

Monte Carlo Analysis: A method involving repeated statistical sampling in order to obtain an estimate of the solution of a stochastic problem.

Multiprogramming: The interleaved execution of two or more programs by a computer.

Nest: A structure in reliability modelling composed of a number of branches grouped together for the purpose of performing a Monte Carlo analysis.

Node: A terminal common to two or more branches of a network or to a terminal of any branch.

Notional Time: A ratio of the elapse time, used for accounting purposes, that reflects the utilisation of a computer systems resources by a program.

Object Program: A computer program that can be executed by the computer hardware.

Operating System: A collection of computer programs that automate the running of computer programs by performing some of the tasks necessary for the desired result.

Overlaying: A technique for enabling a computer to run a program larger than the available store by using the same area of internal storage during the different stages of a program run.

PLAN: The low level assembler code of the ICL 1900 range of computers.

Peripheral: A device connected to a computer to enable communication with that computer.

Probability Density Function: An expression giving the frequency of a variate value x as a function of x or dx whose total is taken to be unity.

Random Access Device: A storage device in which the access time for any location is independent of the sequence in which references are made to locations.

Redundant Element: An element that enhances the reliability of a system but is not essential for the functioning of the system.

Remote Job Entry: A method of assessing a computer system from a remote terminal.

Run: A single use of a computer to carry out a defined piece of work.

Runaway Failure: One of the three failure modes in mechanical engineering systems, namely, where the output is in an unstable state and does not respond to the input.

Segmentation: The division of a program into self-contained parts called segments in order to be able to execute the program without necessarily maintaining it in its entirety in the internal store throughout the run.

Shelf Life: A finite real-time period for which the parameters of an element stay within the specification under storage conditions and hence the element would be in a functioning state if put into service.

Standby Redundancy: A mode of operation of an element where it is available for service should another element fail.

Stock Out Risk: The probability that a system cannot be repaired due to there being no spares available.

System Failure Rate: The rate at which failures occur to a system averaged over a whole simulation.