NASA-CR-192405

Lan...
IN-61-CR
147557
P- 120

Polytechnic UNIVERSITY

COMPUTER AIDED RELIABILITY, AVAILABILITY, AND SAFETY MODELING FOR FAULT-TOLERANT COMPUTER SYSTEMS WITH COMMENTARY ON THE HARP PROGRAM

N93-19981

Unclas

G3/61 0147557

(NASA-CR-192405) COMPUTER AIDED RELIABILITY, AVAILABILITY, AND SAFETY MODELING FOR FAULT-TOLERANT COMPUTER SYSTEMS WITH COMMENTARY ON THE HARP PROGRAM Final Report (Polytechnic Univ.) 120 p

FINAL REPORT RESEARCH GRANT NAG-1-1001

NASA LANGLEY RESEARCH CENTER

April 1, 1991

by

Martin L. Shooman

School of Electrical Engineering and Computer Science
Department of Computer Science
Long Island Center
Route 110
Farmingdale, NY 11735

TABLE OF CONTENTS

Page

Report Documentation Page iii

1.0 INTRODUCTION 1

1.1 The Need for Computer Aided Modeling 1

1.2 Target Machines and Languages 2

1.3 Uses for a Modeling Program 2

1.4 The Need for Institutional Support 3

1.5 Viewpoint of This Research Study 4

2.0 EFFECTIVENESS OF SYSTEM DESIGN TOOLS 5

2.1 Ease of Use 5

2.2 Quality of Documentation 5

2.3 Correctness of Result 5

2.4 Modeling Time Required, Computational Speed, Memory Requirements 6

2.5 Generality of Program 6

2.6 Ease of Installation 6

3.0 EXPERIENCES WITH HARP 7

3.1 Basis of Evaluation 7

3.2 Experience with Installation 7

3.3 Comments on Modelling with HARP 7

3.4 Experiences with Using HARP 8

4.0 SUMMARY AND CONCLUSIONS 9

5.0 REFERENCES 10

Continued

APPENDIX A	CLASS SURVEY FORM - HARP AND ARIES	A-1
APPENDIX B	DETAILS OF THE INSTALLATION OF HARP ON POLYTECHNIC COMPUTERS	B-1
APPENDIX C	DETAILED COMMENTS ON PROBABILISTIC MODELING WITH HARP	C-1
C.1	Review of the Paper by Hayhurst et al.	C-2
C.2	An Example which Illustrates the Problems with HARP's Treatment of Near-Coincident Faults	C-9
C.3	An Approximate Solution for a Triad with Cold Spare	C-11
C.4	Review of Paper by White	C-13
C.5	Modeling Fault-Tolerant Systems with an Instantaneous-Jump Model	C-19
C.6	Comments by White	C-22
APPENDIX D	SIGNIFICANCE OF MODELING ERRORS	D-1
APPENDIX E	PUBLISHED PAPERS SUPPORTED BY THE GRANT	E-1

NASA
National Aeronautics and Space Administration

Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Computer Aided Reliability, Availability, and Safety Modeling for Fault-Tolerant Computer Systems with Commentary on the HARP Program		5. Report Date April 1, 1991
		6. Performing Organization Code
7. Author(s) Martin L. Shooman		8. Performing Organization Report No.
		10. Work Unit No.
9. Performing Organization Name and Address Polytechnic University School of Electrical Engineering & Computer Science Dept. of Computer Science Route 110, Farmingdale, NY 11735		11. Contract or Grant No. NAG-1-1001
		13. Type of Report and Period Covered Final Report
12. Sponsoring Agency Name and Address		14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

Many of the most challenging reliability problems of our present decade involve complex distributed systems such as interconnected telephone switching computers, air traffic control centers, aircraft and space vehicles, and local area and wide area computer networks. In addition to the challenge of complexity, modern fault-tolerant computer systems require very high levels of reliability and availability, e.g. avionic computers with MTTF goals of one billion hours. Most analysts find that it is too difficult to model such complex systems without computer aided design programs. In response to this need, NASA has developed a suite of computer aided reliability modeling programs beginning with CARE III and including a group of newer programs such as: HARP, HARP-PC, Reliability Analysts Workbench (Combination of model solvers SURE, STEM, PAWS and common front-end model ASSIST), and the Fault Tree Compiler.

This study focuses on the HARP program and investigates how well the user can model systems using this program. One of the important objectives will be to study how user friendly this program is, e.g. how easy it is to model the system, provide the input information, and interpret the results. This report describes the experiences of the author and his graduate students who used HARP in two graduate courses. Some brief comparisons were made with the ARIES program which the students also used. Theoretical studies of the modeling techniques used in HARP are also included. Of course no answer can be any more accurate than the fidelity of the model, thus an Appendix is included which discusses modeling accuracy.

A broad viewpoint is taken and all problems which occurred in the use of HARP are discussed. Such problems will include: computer system problems, installation manual problems, user manual problems, program inconsistencies, program limitations, confusing notation, long run times, accuracy problems, etc.

17. Key Words (Suggested by Author(s)) HARP, Probabilistic Modeling, Markov Models, Fault Trees		18. Distribution Statement	
19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of pages	22. Price

NASA FORM 1626 OCT 86

1.0 INTRODUCTION

1.1 The Need for Computer Aided Modeling

The subject of fault-tolerant computing deals with the reliability and availability of computer hardware, software, and systems. Because of the rapid advances in the computer field, there has been an enormous amount of work done in this area in the last two decades.

Many of the most challenging reliability problems of our present decade involve complex distributed systems such as interconnected telephone switching computers, air traffic control centers, aircraft and space vehicles, and local area and wide area computer networks. In addition to the challenge of complexity, modern fault tolerant computer systems require very high levels of reliability and availability. For example, NASA's SATURN V launch computer (circa 1964) had a reliability requirement equivalent to a mean time to failure, MTTF, of 25,000 hours. In the late 1970's the SIFT and FTMP avionic computers designed to control dynamically unstable aircraft had a MTTF goal of one billion hours (Pradhan 1986, p. xiii, 421.) We can contrast this with the observed mean time to system crash MTTC for several commercial computers (Pradhan 1986, p. 420): Burrows B5500 14.7h, Univac 1108 17h, Dual 370/165 system 8.9h, PDP-10 10h, CRAY-1 4h. Clearly no ordinary design can meet such goals. The aerospace computers discussed above employ several hardware and software reliability enhancement techniques to achieve acceptable levels of system reliability and availability. The ready availability of low cost microprocessors and electronic memory has encouraged the use of multiple processors to achieve high reliability and often leads to distributed computing systems.

The fault-tolerant systems described above are very complex due to the large number of processors, various versions of the software, complex schemes for redundancy management, and error recovery. Also, there are many competing approaches to fault tolerance. One of the most important design criteria is the system reliability (and availability), thus the reliability analyst or system designer is faced with the task of building and solving a complex reliability model or a group of complex models if a design trade-off study is underway. Another characteristic of fault-tolerant systems is that they must respond rapidly when a redundant unit fails and switch in a good spare unit or combine the output with correct outputs from redundant units. The net result is that one has a mixture of fast response to failures and slower "natural response" of the system. This leads to a mixture of time constants and results in so-called "stiff" differential equations which are hard to solve. Most analysts find that it is too difficult to model and solve such complex systems without computer aided design programs. This has led to the suite of computer aided reliability modeling programs developed by NASA, and specifically the HARP program, which is the subject of this proposal.

1.2 Target Machines and Languages

In devising a computer aided design program, some thought should be given to which computer systems and languages the prospective users will find most convenient. If we consider the primary users to be the modest sized group of large aerospace manufactures, then in all likelihood the company possesses an ample assortment of large mainframes, supermini computers, workstations, and personal computers. Thus, if the programs are to be used only occasionally, a standard host computer can be supported as long as the program is written in one of the modern languages such as C, Ada, Pascal, Fortran 77, etc. However, if the modeler is to use the program frequently, then it is important that he have ready access to the host computer in his own work area, which generally favors a PC or work station. Many of the older reliability analysis programs including HARP, evolved over a period of time and the language in which they are written was fixed a long ago without regard to their evolution. In the case of HARP, FORTRAN was the chosen language and one would have expected that the program design concepts and syntax would be compatible with a modern version of Fortran 77, however this was not the case, and many sections were written in older versions of Fortran.

1.3 Uses for a Modeling Program

A reliability modeling program can serve many uses:

1. It allows one to predict the reliability and availability of a complex system during the proposal/early design stages, and evaluate if proposed design improvements will help meet reliability requirements.
2. It allows an analyst to do a comparative study among various proposed design alternatives so that the final decision will weigh realistic quantitative reliability estimates.

3. If the program becomes widely used, then its nomenclature and approach become a standard in the reliability analysis area, and provide a uniform basis of comparison in contracting, the technical literature, and education.
4. The standardization referred to in 3 above could serve as a standard for reliability data gathering, during development and field deployment.

1.4 The Need for Institutional Support

Most large computer programs which achieve success are used over a much longer period of time then their designers initially envisioned. This often leads to problems with regard to maintenance and enhancement. There is always a need to correct errors discovered during the lifetime of a program and port the code to various other computers and correct minor inconsistencies, correct maintain, and extend manuals, etc. In addition, as new computer systems and facilities become available, there is a need to modify and enhance programs. Lastly, as the field of fault-tolerant, distributed systems evolves, there will probably be a need for additional modeling capability which is not present in contemporary programs. For example, it is not clear whether presently available computer programs can easily model all the different techniques of software fault tolerance which have been proposed.

The long term support of the various fault-tolerant modeling programs developed by NASA is a mater of some concern. Such support and maintenance can require a substantial amount of annual funding without which such programs disappear. Such was the fate of the first availability modeling program (known to the author) GEM, developed by the NAVY in the late 1960's [Orbach 1970]. Consider electronic circuit analysis programs as one example of how such long term support develops. In the 1960, the first widely used electronic circuit analysis program, ECAP, was developed by IBM [Jensen 1968]. Initially IBM provided versions of this program which ran on their 1620, 7094, and 360 computers. In the mid 1970's, the SPICE program was developed with public funds at the University of California Berkeley by Prof. D.O. Pederson and his colleagues and students, [Antognetti 1988]. Supermini, fileserver, and PC versions of this program are now available from commercial sources which provide the necessary support. However, before the reader draws to hasty a parallel, we should add that the base of electronic circuit modelers is much larger than that of reliability analysts. For example the membership of the IEEE is now 315,000 members, 32 technical societies, and the Reliability Society has about 4,300 members whereas there are 29,300 members who belong to the Electronic Devices, the Circuits and Systems Society, or the Components and Manufacturing Society. The 7:1 difference in members between these two groupings may not accurately reflect the relative number of users of fault-tolerant programs and circuit analysis programs; however, a strategy which works for electronics may not work for reliability. Thus, one must consider various governmental, professional, university, and commercial strategies (individually or cooperatively) for support of fault-tolerant programs over their extended lifetimes.

One must also realize that a suite of reliability modeling programs which have been developed for use within the aerospace community have a much wider range of applicability. Such programs could, for example, be used to analyze a fault-tolerant air traffic control system, a redundant communications network, or any complex digital system with many operating states. This is important to keep in mind with regard to NASA's goals for technology transfer. Also if these programs are to be maintained over a long period of time it may be necessary to explore a larger user base to help pay for continued maintenance of such a suite of programs.

1.5 Viewpoint of This Research Study

It should be apparent to the reader that this introduction discusses many factors and much more breath then were investigated under the modest budget which submitted for this grant. This is done for a number of reasons. First, the author hopes that such discussions will be helpful to NASA in planning and coordinating their future research in this area. Second, this will help further define the viewpoint for the study.

It seemed clear at many points during this study that there was not a clear set of detailed instructions on how to use HARP. In addition there is a need for a clear plan as to how the newer fault-tolerant programs will supplant or complement the existing programs. Unfortunately, this is the rule and not the exception with most projects, thus we must constantly work hard to eliminate this problem. Furthermore, because of the mathematical nature of the problems to be solved, there must be strong and constant coordination between the math modelers and the software developers, which at times seemed lacking.

2.0 EFFECTIVENESS OF SYSTEM DESIGN TOOLS

The best overall measure of goodness for a system design tool is how frequently the user employs the tool, and how much it helps him in his analysis and design tasks. This overall measure can be expressed in terms of a number of components which are discussed below.

2.1 Ease of Use

The three key goals in any computer aided design program are ease of use, correctness of the result, and generality of the problem set which can be modeled. Ease of use depends on:

(a) how well the program is described (documentation)

(b) how easy it is to input the structure of the problem

(c) how easy it is to choose the various modeling modes

In addition a superior program generally provides graphic output showing the system model as well as tabular and graphic output.

2.2 Quality of documentation

An important aspect of the documentation is the insight provided into the model limitations. Most computer models will have limitations, i.e. situations in which the answers are wrong due to round-off errors, modeling anomalies, errors in the algorithm, etc. In some cases, it may be easier to describe the type of problems and situations in which such problems occur. And clear statement of such problems improves the documentation.

2.3 Correctness of result

The primary measure of the correctness of the result is how well the algorithms upon which the program is based compute the correct result, i.e.. the program accuracy. Computational algorithms are not always correct for all modes or all conditions. Even if we assume that the computational algorithm is correct, one can still obtain wrong results if the round-off errors in computation cause significant inaccuracies.

Another measure of the correctness of the result is how well we can verify the program (the implementation of the algorithm). A computer aided design program must be treated like any other program, i.e. it will contain errors once developed and must be verified. One program will have an advantage over others if it is easier to verify.

Of course we should not overlook the correctness of the basic model which we use as our program input. A wrong model can lead to wrong results even if our program is flawless. This is discussed in more detail in Appendix D. Also, one must realize that the data used to compute failure and repair rates is often noisy on a small sample. Thus, the various Markov model parameters should really be stated as point and interval values. If the range of the interval estimate is broad, then the accuracy of the modeling results may be more heavily dependent on model parameter values than program fidelity.

2.4 Modeling Time Required, Computational Speed, Memory Requirements

All computer modeling techniques are resource limited and the amount of analysts time required to set up the problem and interpret the results is one basis of comparison. In addition, one must be concerned with the amount of computer time and memory used in computation and the amount of disk space needed to store the program, the model, and the results of runs which must be retained for study.

2.5 Generality of Program

A reliability program generally has many different behaviors which must be modeled. The following is a partial list of factors which should be modeled:

(a) Constant failure rate.

(b) Non-constant failure rate (Weibull, other)

(c) Constant repair rates.

(d) Non-constant repair rates.

(e) Series-parallel structures supported.

(f) Non-series-parallel structures (bridges, etc.) supported.

2.6 Ease of Installation

Most computer programs are developed in a given language using a particular compiler, and a host computer. A measure of how easy the program is to install can be obtained when we try to move the program to other compilers and other host computers. Robust programs will install with little problem. An example of this is the ease with which a personal computer program can be installed on an MS DOS compatible computer.

3.0 <u>EXPERIENCES WITH HARP</u>

3.1 Basis of Evaluation

In order to evaluate the ease of installing HARP on a computer system, the following "ground rules" were adopted:

* Only minor help from system programmer and NASA should be requested.
* System to be used by Poly instructor and students in graduate courses in reliability and fault-tolerant systems.
* Comparisons made with other programs limited to those already available at Poly, namely ARIES.

Thus, the installation was attempted by this author, who has only elementary knowledge of the Unix operating system with minor help from our Unix expert and some help from a graduate student with an intermediate knowledge of the Unix.

3.2 Experiences with Installation

A number of difficulties were encountered in attempting to install HARP on the Poly Gould computer. These could not be easily resolved without significant systems programming (not covered by this Grant). (See Appendix B.) Thus the tests of HARP were via network connection to a Langley Airlab computer.

3.3 Comments on Modeling With HARP

In late 1989 and early 1990 two reports were written by five NASA researchers which thoroughly investigated some of the mathematical and modeling basis of HARP and other fault tolerant computing programs:

"A Critical Assessment of the HARP Program," Kelly J. Hayhurst, Rickey W. Butler, and Sally C. Johnson, Draft NASA Technical Memorandum 102607, NASA Langley, Feb. 1990.

"A Review of the HARP Program: Approach and Mathematics," Allan L. White, Draft NASA Technical Memorandum, NASA Langley, Nov. 1989.

The comments (reviews) on these two reports contributes to the detailed understanding of HARP and appears in Appendix C.

3.4 Experiences with Using HARP

This section is based on the authors own experiences in using HARP and those of his two graduate classes who used HARP.

During the spring semester of 1990 Prof. Shooman taught two graduate courses at the Polytechnic Farmingdale Campus which involved probabilistic analysis. The 18 students in these classes used both the ARIES and HARP programs in the courses. Twelve of the students completed a questionnaire on their experiences with ARIES and HARP and the details are given in Appendix A.

The students were predominantly Part Time MSEE students, working in industry, with some experience in reliability and quite a bit of computer experience. Overall, the students found both ARIES and HARP useful and the documentation provided to the class was satisfactory. Scoring the results using a 4 point scale similar to a University grading system (Excellent = 4 =A, Good = 3 = B, Fair = 2 = C, Poor = 1 = D), we obtain overall satisfaction scores of 2.9 for ARIES and 2.8 for HARP. The scores for documentation were 2.8 for ARIES and 3.4 for HARP (Tutorial was distributed). In summary, both programs received B - scores, however due to the detail provided by the HARP Tutorial, the documentation score for HARP was B +. Details of the responses are given in Appendix A.

Prof. Shooman will be using SHURE and ASSIST in his future courses on fault-tolerant computing and will provide NASA Langley with feedback on their effectiveness and student acceptance on an informal basis in the future.

4.0 SUMMARY AND CONCLUSIONS

HARP has played a transitional role in the development of fault-tolerant computer programs as has ARIES. Neither of these programs is mature enough in terms of basic modeling ease, validity, and generality of the algorithms to serve as a standard. Also the implementation is not user friendly enough in terms of input/output or in terms of a standard transportable version.

NASA should investigate their newer programs such as SHURE, PAWS, STEM, ASSIST and FAULT-TREE to see if some subset of these can serve as the standard program. Further it is suggested that there be three implementations:

(a) For MS-DOS 286, 386, 486 machines.

(b) For Unix systems such as SUN workstations, etc.

(c) For Macintosh computers.

Each of these implementations should be carefully planned to be functionally equivalent, easily transportable, and user friendly (standard modeling terms and ease of input and output). Proper documentation at a systems and users level is required. Considerable thought should be given as to how a standard program can be maintained since such a program is truly a "national resource" for fault-tolerant (and general reliability) analysts.

REFERENCES

Anderson, T. and P. A. Lee: "Fault Tolerance: Principles and Practice," Prentice-Hall, New York, NY, 1981.

Anderson, T.: Resilient Computing Systems, Vol. 1, John Wiley, New York, NY, 1985.

Antognetti, P. and G. Massobrio, Semiconductor Device Modeling with Spice, McGraw-Hill, New York, NY, 1988.

Bateman, K. and E. Cortes: "Availability Modeling of FDDI Networks," Proc. Ann. Reliability and Maintainability Symp., IEEE, New York, 1989, pp. 389-395.

Bavuso, S. J.: "A User's View of CARE III," Proc. Ann. Reliability and Maintainability Symp., IEEE, Jan. 1984, pp. 382-389.

Bavuso, S. J. et. al.: "CARE III Hands-On Demonstration and Tutorial," NASA Technical Memo. 85811, NASA Langley Research Center, Hampton, VA., May 1984.

Bavuso, S. J. et. al.: "CARE III Model Overview and User's Guide," NASA Technical Memo. 85810, NASA Langley Research Center, Hampton, VA., June 1984, NASA Technical Memo. 86404, April 1985.

Bavuso, S. J. et. al.: "Analysis of Typical Fault-Tolerant Architectures Using HARP," IEEE Trans. on Reliability, Vol. R-36, No. 2, June 1987.

Blemel, K. G.: "Quality Assurance - A Total Systems Approach," Proc. Ann. Reliability and Maintainability Symp., IEEE, Jan. 1986, pp. 367-369.

Breuer, M. A. and A. D. Friedman: Diagnosis & Reliable Design of Digital Systems, Computer Science Press, Inc., Woodland Hills, CA, 1976.

Bridgman, M. S. and W. G. Ness: "Automated Ultrareliability Models: A Review," Proc. Ann. Reliability and Maintainability Symp., IEEE, Jan. 1984, pp. 396-402.

Brilliant, S. S., J. C. Knight and N. G. Leveson: "The Consistent Comparison Problem in N-Version Software," ACM SIGSOFT Software Engineering Notes, Vol. 12, No. 1, January 1987, pp. 29-34

Bryant, L. A. and J. J. Stiffler: "CARE III. Version 6 Enhancements," NASA Contractor Report 177963, Langley Research Center, Hampton, VA., Nov. 1985.

Butler, R. W. and P. H. Stevenson, "The PAWS and STEM Reliability Analysis Program," NASA Technical Memo. 100572, Langley Research Center, Hampton, VA., March 1988.

Butler, R. W.: "An Abstract Language for Specifying Markov Reliability Models," IEEE Trans. on Reliability, Vol. R-35, No. 5, Dec. 1986.

Butler, R. W. and A. L. White: "SHURE Reliability Analysis - Program and Mathematics," NASA Technical Paper 2764, Langley Research Center, Hampton, VA., March 1988.

Butler, R. W. and A. L. Martensen: "The FTC Fault-Tree Program," DRAFT, NASA Technical Memo., Langley Research Center, Hampton, VA., Dec. 1988.

Chen, L. and A. Avizienis: "N-Version Programming: A Fault-Tolerance Approach to Reliability of Software Operation," Digest of Eighth International Fault-Tolerant Computing Symp., IEEE Computer Society, Toulouse, France, 1978. pp. 3-9.

Conroe, D. et. al: "R/M/T Design for Fault Tolerance," RADC-TR-87, Grumman Corp., Bethpage, NY, Dec. 1987.

"COSMIC - A Catalog of Selected Computer Programs," 112 Barrow Hall, The University of Georgia, Athens, Georgia, 30602.

Cristian, F.: "Fault-Tolerant Computing," Lecture Notes for Seminar, University of California at Santa Cruz, July 13-15 1987.

Davies, D. W. et. al.: Distributed Systems - Architecture and Implementation, Lecture Notes in Computer Science, Springer-Verlag, New York, 1981, Chaps. 8,10,13,17,20.

Edfors, H. C.: "Proc. 1985 Computerized Reliability Predictions," IEEE Reliability Society, Chicago Chapter, Nov. 16, 1986, 6 So. 315, New Castle Rd., Naperville, IL 60540.

Estes, G. E.: "Bounds for Reliability Program (Cut Set/Tie Set)," Polytechnic University, MS Computer Science Project, Dec. 1980.

Flemming, R. E., J. Josselyn, et al: "Fault-Tolerant C^3I System A_0, A_I, MTBF Allocations," and "Application of Markov Models for RMA Assessment," Proc. Ann. Reliability and Maintainability Symp., 1986, pp. 1-6, 352-357. Also see papers by same authors in RAM Symposia: 1983 pp. 417-422; 1984 pp. 403-408; 1985 pp. 125-130.

Garman, J. R.: "The "Bug" Heard Round The World," ACM SIGSOFT Software Engineering Notes, Oct. 1981, pp. 3-10.

Grace, K. Jr.: "Approximate System Availability Models," Proc. of the 1969 Ann. Symp. on Reliability, Jan. 1969, pp. 146-152.

Grace, K. Jr.: "Repair Queing Models for System Availability," Proc. of the 1969 Ann. Symp. on Reliability, Feb. 1970, pp. 331-336.

Hayhurst, K. J.: "Testing of Reliability-Analysis Tools," Proc. Ann. Reliability and Maintainability Symp., IEEE, New York, 1989, pp. 487-490.

Hill, F. J. and G. R. Peterson: Introduction to Switching Theory and Logical Design, Third Edition, John Wiley & Sons, New York, NY, p. 219, 1981.

IEEE Computer Society: "Digest of Papers, Int'l. Symposia on Fault-Tolerant Computing," IEEE, yearly, 1971-1987, (Symposiums 1 through 17).

IEEE Computer Society: "Tutorial on Fault-Tolerant Computing," IEEE 1987.

Ingogly, W. F. et. al.: "CARE3MENUE Maintenance Manual," NASA Contractor Report 172609, Langley Research Center, Hampton, VA., May. 1985.

Johnson, S. C: "ASSIST Users Manual," NASA Technical Memo. 87735, Langley Research Center, Hampton, VA., Aug. 1986.

Johnson, S. C: "Reliability Analysis of Large, Complex Systems Using Assist," AIAA/IEEE 8th Digital Avionics Systems Conf., San Jose, CA, Oct 17-20, 1988.

Jensen, R. W. and M. D. Lieberman: IBM Electronic Circuit Analysis Program, Prentice-Hall Englewood Cliffs, N.J. 1968

Kaplan, G.: "The X-29: Is it Coming or Going?," IEEE Spectrum, June 1985, pp. 54-60.

Knight, J. C. and N. G. Leveson: "An Experimental Evaluation of Independence in Multiversion Programming," IEEE Trans. on Software Engineering, Vol. SE-12, No. 1, January 1986, pp. 96-109.

Kohavi, Z.: Switching and Finite Automata Theory, McGraw-Hill, New York, NY, 1978, Chap. 8.

Laemmel, A. E.: "Bounds on Markov State Probabilities", CS Research Memo, Polytechnic University, June 1988.

Lala, P. K.: Fault Tolerant & Fault Testable Hardware Design, Prentice-Hall, Englewood Cliffs, NJ, 1985.

Laviron, A., et al: "ESCAF - A New and Cheap System for Complex Reliability Analysis and Computation, IEEE Trans. on Reliability, Vol. R-31, No. 4, Oct. 1982.

Laviron, A., et al: "S.ESCAF: Sequential Complex Systems are Analyzed with ESCAF Through an Add-on Option," IEEE Trans. on Reliability, Aug. 1985.

Makam, S. V. and A. A. Avizienis: "Modeling and Analysis of Periodically Renewed Closed Fault-Tolerant Systems," IEEE Publication No. CH1600-6/81/0000/0134, 1981.

Makam, S. V., A. A. Avizienis, and G. Grusas: "UCLA ARIES 82 Users Guide," Report No. CSD-820830/UCLA-ENG-8262, Computer Science Department, UCLA, Aug. 1982

Martensen, A. L. and R. W. Butler,: "The Fault-Tree Compiler," NASA Technical Memo. 89098, Langley Research Center, Hampton, VA., Jan. 1987.

Martensen, A. L.: "CARE III User Friendly Interface Users Guide," NASA Contractor Report 178251, Langley Research Center, Hampton, VA., Jan. 1987.

Martensen, A. L. and S. J. Bavuso: "Tutorial and Hands-On Demonstration of a Fluent Interperter for CARE III," NASA Technical Memo. 4011, Langley Research Center, Hampton, VA., Nov. 1987.

McCormick, N. J.: Reliability and Risk Analysis, Academic Press, New York, NY, 1981

McCluskey, E. J.: "Logic Design Principles with Emphasis on Testable Semicustom Circuits," Prentice-Hall, Englewood Cliffs, NJ, 1986.

Messinger, M. and M. Shooman: "Approximations for Complex Structures," Proc. 1967 Ann. Symp. on Reliability, IEEE, New York.

NASA: "Practical Reliability - Volume II Computation," NASA Contractor Report, NASA CR-1127, Research Triangle Institute, Aug. 1968.

Nelson, V. P. and B. D. Carrol: "Tutorial: Fault-Tolerant Computing," IEEE Computer Society, New York, 1987.

Ng, Y. W., and A. A. Avizienis: "ARIES - An Automated Reliability Estimation System for Redundant Digital Structures," Proc. Ann. Reliability and Maintainability Symp., 1977, pp. 108-113. Also see paper in 1975 RAM Symp. by Avizienis, pp. 333-339.

Ng, Y. W., and A. A. Avizienis: "A Unified Reliability Model for Fault-Tolerant Computers," IEEE Trans. on Computers, Vol. C-29, No. 11, Nov. 1980, pp. 1002-1011.

Orbach, S.: "Talk on GEM Reliability and Availability Analysis Program," Presented at the 6th. Monmouth Conf. on Statistics and Quality Assurance, ASQC and American Statistical Association, Holiday Inn, Hazlet, NJ, April 24, 1970.

O'Rorke, D. and A. Lamanna: "NASA Reliability Program for the PC," Polytechnic University, Senior Computer Science Project, Dec. 1985.

O'Rorke, D.: "Solving Markov Models with TUTSIM," Polytechnic University, Senior Computer Science Project, May 1986.

Osaki, S. and T. Nishio: Reliability Evaluation of Some Fault-Tolerant Computer Architectures, Lecture Notes in Computer Science, Springer-Verlag, New York, NY, 1980

Pham, H. and S. Upadhyaya: "Reliability Analysis of A Class of Fault Tolerant Systems [Digital Data Communications]," Proc. Ann. Reliability and Maintainability Symp., IEEE, New York, 1989, pp.114-118.

Pierce, J. R.: An Introduction to Information Theory, Second, Revised Edition, Dover Publications, New York, NY, 1980.

Pierce, W. H.: Fault-Tolerant Computer Design, Academic Press, New York, NY, 1965.

Pradhan, D. K.: Fault-Tolerant Computing Theory and Technique, Prentice-Hall, Englewood Cliffs, NJ, 1986.

Research Triangle Institute: "CARE III User's Workshop," NASA Conf. Publication 10011, Oct 6-7, 1987.

Rothman, E. et al.: "HARP: The Hybrid Automated Reliability Predictor Tutorial HARP Version 6.1," Department of Computer Science, Duke University, Dec. 1989.

Shooman, M.L.: Software Engineering: Design, Reliability, and Management, McGraw-Hill Book, Co., New York, NY, 1983.

Shooman, M.L.: Probabilistic Reliability: An Engineering Approach, First Edition, McGraw-Hill Book Co., New York, NY, 1968, Second Edition, Krieger, Melborne, FL, 1990.

Shooman, M. L. and A. E. Laemmel: "Simplification of Markov Models by State Merging", Proc. 1987 Ann. Reliability And Maintainability Symp., IEEE, New York.

Siewiorek, D. P. and R. S. Swarz: The Theory and Practice of Reliable System Design, Digital Press, Bedford Mass, 1982.

Siewiorek, D. P.: "The Art of Fault-Tolerant Computers," Chap. 6 in Pradhan 1986.

Spencer, J. L.: "The Highs and Lows of Reliability Predictions," Proc. Ann. Reliability and Maintainability Symp., IEEE New York, 1986, pp.156-162.

Smith, H. F.: "Data Structures Form and Function," Harcourt Brace Jovanovich, New York, NY, 1987, pp. 400-404.

Smith, J. M.: Mathematical Modeling and Digital Simulation for Engineers and Scientists, John Wiley & Sons, New York, NY, 1977.

Texas Instruments TTL Data Book, Dallas, TX, 1988, pp. 2-597 to 2-599.

Trivedi, K. S. and R. M. Geist: "A Tutorial on the CARE III Approach to Reliability Modeling," NASA Contractor Report 3488, NASA Langley Research Center, Hampton, VA., 1981.

Trivedi, K. S. et. al.: "HARP:The Hybrid Automated Reliability Predictor, Introduction and Guide for Users," NASA Langley Research Center, Sept. 1986.

Trivedi, K. S. et. al.: "HARP Programmer's Maintenance Manual," NASA Langley Research Center, April 1988.

Vesley, W. E., et al: "PREP and KITT: Computer Codes for the Automatic Evaluation of a Fault Tree," Idaho Nuclear Corp., Report for U.S. Atomic Energy Commission, No. IN-1349, Aug. 1970.

White, A.: "Motivating the SURE Bounds," Proc. Ann. Reliability and Maintainability Symp., IEEE, New York, 1989, pp. 277-282.

Wolf, J. K., M. L. Shooman, and R. R. Boorstyn: "Algebraic Coding and Digital Redundancy," IEEE Trans. on Reliability, Vol. R-18, No. 3, Aug. 1969, pp. 91-107.

APPENDIX A

CLASS SURVEY FORM - HARP AND ARIES

A.1 INTRODUCTION

During the spring semester of 1990 Prof. Shooman taught two graduate courses at the Polytechnic Farmingdale Campus which involved probabilistic analysis, CS907 Fault-Tolerant Computing and EL617/IE685 System Reliability. Descriptions of CS907 and EL617 appear in Tables A-1 and A-2 respectively. Course CS907 is given by the Computer Science Department and about half the students who take the course are EE students and half CS students. Course EL617/IE685 is jointly sponsored by the Electrical Engineering and Industrial Engineering Departments and the majority of the students are EE's. Since the courses were given in the evening at the Farmingdale Long Island Campus, most of the students were working in industry and pursuing an MS degree part time in the evening. (There were a few full time Senior Undergraduates who were taking these courses for elective credit.) Both the ARIES and HARP programs were used in the classes and the students agreed to fill out the questionnaire shown in Table A-3. The ARIES program was installed on the Poly Gould supermini computer and was accessed from the terminals in the Farmingdale Computer Room or via a modem from work or home. The HARP program at the Langley Research Airlab computer was accessed via the TELNET network from terminals in the Farmingdale Computer Room or via a modem from work or home.

A.2 RESULTS

There were 11 students in CS 907 and 10 in EL617 and two students took both classes. Out of the 19 students, 12 returned a completed survey form. A summary of the responses to the various questions appears in Table A-4. As can be seen from Table A-4, the students were predominantly Part Time MSEE students, working in industry, with some experience in reliability and quite a bit of computer experience. Overall, the students found both ARIES and HARP useful and the documentation provided to the class was satisfactory. Scoring the results using a 4 point scale (similar to University grading) where Worked well (good) is 4, Satisfactory is a 3, Fair is a 2 and Poor is a 1, we obtain an overall scores of 2.9 for ARIES and 2.8 for HARP. The scores for documentation were 2.8 for ARIES and 3.4 for HARP.

TABLE A-1 Description of Course CS907 - Fault Tolerant Computing

COMPUTER SCIENCE DEPARTMENT

BROOKLYN CAMPUS

CS903 - FAULT-TOLERANT COMPUTERS

MISSION TO MARS

CONTROL OF NUCLEAR REACTORS

UNSTABLE AIRCRAFT: BOEING 767-400, AIRBUS A320

AIR-TRAFFIC CONTROL

NON-STOP BANK FUNDS TRANSFER

RELIABLE AND AVAILABLE LANS

Much of modern society depends on reliable, safe, available and fault-free computing. Digital computers allow a wide range of redundant approaches to insure continuous operation, i.e, tolerance to individual hardware component or software faults. This course introduces and studies a variety of hardware and software techniques for designing and modeling Fault Tolerant computers. Topics will include:

- Coding techniques (Hamming, SECSED, SECDED, etc.) to detect and correct memory and data transmission errors
- Parallel processors and majority voting schemes (TMR)
- Software redundancy (N-Version programming, dependency models)
- Software checks and recovery schemes

The course will introduce the architectures, approaches, and probabilistic models needed to compare the reliability and availability of various techniques. Examples will be drawn from space fault-tolerant approaches, LAN network examples, and commercial non-stop systems such as TANDEM and STRATUS. The HARP and ARIES Fault-Tolerant modeling tools will be used for system reliability and availability computations for homework and the term design project.

PREREQUISITES:	CS237 and MA 223 or approval of instructor*
INSTRUCTOR:	Professor Martin Shooman (Ext. 4290)
TEXTBOOK:	The Theory and Practice of Reliable System Design, Siewiorek and Swarz, Digital Press, 1982 plus supplementary notes by M. Shooman.
DAY AND TIME:	Wednesday - 5:55 - 8:10 PM (Brooklyn, Spring 1991, ISIS No. 20844)

*Suitable for use as a technical elective for EE and CS Graduate Students or for Seniors with Advisor's approval.

TABLE A-2   Description of Course EL617/IE685 - System Reliability

EL 617   System Reliability* **2½:0:3**
Structural reliability, redundancy, bounds on reliability of complex systems. Repairable systems: Markov models, maintainability and availability. Optimization of spare parts inventories, inspection intervals and replacement times. Failure models: accumulated shocks and stress-strength-time. Marginal failures, dependent failures. Prerequisite: EL 531 or MA 561 or equivalent.
Also listed under IE 685

TABLE A-3

USE OF ARIES AND HARP QUESTIONAIRE FOR:

CS907 FAULT-TOLERANT COMPUTING
EL617 SYSTEM RELIABILITY

SPRING TERM FARMINGDALE APRIL 25, 1990

When completed please return this questionaire to Prof. M. L. Shooman, Farmingdale Room 222 or Ms. JoAnn McDonald Room 250.

The purpose of this questionaire is to gather information on the ease of use, accuracy, and modeling flexibility of the ARIES and HARP fault-tolerant computing programs. Sometimes you will be asked to comment from the viewpoint of a graduate student and sometimes as a practicing professional.

1. **Education:**

Degree	Year Completed (or expected)	School	Concentration Area

____Full Time ____Part Time ____EE ____CS ____MS ____Ph.D. Other:________

2. **Employment:**

____Full Time ____Part Time ____Full Time Student ____Work Summers

Company:____

Job Title:____

3. **Past Experience with reliability or probabilistic analysis:**

Course work: ____Very Experienced ____Experienced ____Some Knowledge ____First Course

Describe academic experience:____

____

____

Professional Experience:

____Very Experienced ____Experienced ____Some Knowledge ____First Course

Describe work experience:____

____

4. **Computer Experience:**

Hardware: ____Main Frame ____Mini ____PC/XT/AT ____Work Station ____Special

Level: ____Very Experienced ____Experienced ____Some Knowledge ____Beginner

Software: ____DOS ____UNIX ____VMS ____Xwindows ____Other____________________

Level: ____Very Experienced ____Experienced ____Some Knowledge ____Beginner

5. **Experiences with ARIES:**

____Worked well ____Satisfactory ____Fair ____

6. **Experiences with HARP:**

____Worked well ____Satisfactory ____Fair ____

7. **How useful was the Documentation on ARIES:**

The Users Manual: ____Good ____Satisfactory ____Fair ____Poor

The material from Shooman's Notes: ____Good ____Satisfactory ____Fair ____Poor

8. **How useful was the Documentation on HARP:**

The tutorial: ____Good ____Satisfactory ____Fair ____Poor

The material from Shooman's notes: ____Good ____Satisfactory ____Fair ____Poor

9. **Describe any errors or problems you had with aries:**

____

____

____

____

10. **Describe any errors or problems you had with HARP:**

____

____

____

____

11. **How could ARIES be improved?**

____

____

____

____

12. **How could HARP be improved?**

____

____

____

____

13. **Would you be interested in doing a project or thesis for credition evaluating and improving ARIES and HARP:**

____Yes ____Maybe ____No

14. **Optional**

Name:______________________________ Address:______________________________

Phone Number:__________

15. **Additional Comments:**

Use additional sheets if necessary.

TABLE A-4 Photocopies of Students Responses to Questionnaire

USE OF ARIES AND HARP QUESTIONAIRE FOR:

CS907 FAULT-TOLERANT COMPUTING
EL617 SYSTEM RELIABILITY

SPRING TERM — FARMINGDALE — APRIL 25, 1990

- -

When completed please return this questionaire to Prof. M. L. Shooman, Farmingdale Room 222 or Ms. JoAnn McDonald Room 250.

The purpose of this questionaire is to gather information on the ease of use, accuracy, and modeling flexibility of the ARIES and HARP fault-tolerant computing programs. Sometimes you will be asked to comment from the viewpoint of a graduate student and sometimes as a practicing professional.

1. **Education:**

Degree	Year Completed (or expected)	School	Concentration Area
BSEE	May86	SUNY @ StonyBrook	EE/AMS
MSEE	May90	polytech	EE

___Full Time X Part Time X EE ___CS X MS ___Ph.D. Other:_______

2. **Employment:**

X Full Time ___Part Time ___Full Time Student ___Work Summers

Company: HArris GSSD

Job Title: SR. Engineer

3. **Past Experience with reliability or probabilistic analysis:**

Course work: ___Very Experienced X Experienced ___Some Knowledge ___First Course

Describe academic experience: I have a second major in applied math

Professional Experience:

___Very Experienced ___Experienced X Some Knowledge ___First Course

Describe work experience: Minor work on prediction of Relay Reliability

4. **Computer Experience:**

Hardware: X Main Frame X Mini X PC/XT/AT X Work Station X Special

Level: X Very Experienced ___Experienced ___Some Knowledge ___Beginner

Software: X DOS X UNIX X VMS ___Xwindows X Other Rf, Apple, ~

Level: X Very Experienced ___Experienced ___Some Knowledge ___Beginner

5. **Experiences with ARIES:**

____Worked well [✓ Satisfactory] ____Fair ____

6. **Experiences with HARP:**

____Worked well [✓ Satisfactory] ____Fair ____

7. **How useful was the Documentation on ARIES:**

The Users Manual: ____Good ____Satisfactory [✓ Fair] ____Poor

The material from Shooman's Notes: ____Good ____Satisfactory [✓ Fair] ____Poor

8. **How useful was the Documentation on HARP:**

The tutorial: ____Good [✓ Satisfactory] ____Fair ____Poor

The material from Shooman's notes: [✓ Good] ____Satisfactory ____Fair ____Poor

9. **Describe any errors or problems you had with aries:**

parameters poorly Explained

10. **Describe any errors or problems you had with HARP:**

never showed how to Enter Repair into markov model (I figured it out)

11. **How could ARIES be improved?**

add graphic interface or at least clean up parameter Entry

12. **How could HARP be improved?**

add graphic Interface (fault tree picture) clean up output to be more table like

13. **Would you be interested in doing a project or thesis for credition evaluating and improving ARIES and HARP:**

____Yes ____Maybe ____No N/A

14. **Optional**

Name: R. WEISS Address: 14 HENRY St CENTEREACH NY 11720

Phone Number: 467-2314

15. **Additional Comments:**

ARIES DATA output was good while data Entry was poor, Larry had a good data Entry Less Lousy output

Use additional sheets if necessary.

USE OF ARIES AND HARP QUESTIONAIRE FOR:

CS907 FAULT-TOLERANT COMPUTING
EL617 SYSTEM RELIABILITY

SPRING TERM FARMINGDALE APRIL 25, 1990

When completed please return this questionaire to Prof. M. L. Shooman, Farmingdale Room 222 or Ms. JoAnn McDonald Room 250.

The purpose of this questionaire is to gather information on the ease of use, accuracy, and modeling flexibility of the ARIES and HARP fault-tolerant computing programs. Sometimes you will be asked to comment from the viewpoint of a graduate student and sometimes as a practicing professional.

1. **Education:**

Degree	Year Completed (or expected)	School	Concentration Area
MSEE	'90	Polytechnic Univ.	Digital Comput.
BEEE	'87	Pratt institute	

Full Time [✓ Part Time] EE CS [✓ MS] Ph.D. Other:

2. **Employment:**

[✓ Full Time] Part Time Full Time Student Work Summers

Company: Standard Microsystems Co.

Job Title: Logic Design / Test Eng.

3. **Past Experience with reliability or probabilistic analysis:**

Course work: Very Experienced Experienced Some Knowledge [✓ First Course]

Describe academic experience: one course in Fault Tolerant Computers.

Professional Experience:

Very Experienced Experienced Some Knowledge First Course

Describe work experience: NONE

4. **Computer Experience:**

Hardware: Main Frame [✓ Mini] [✓ PC/XT/AT] [✓ Work Station] Special

Level: Very Experienced [✓ Experienced] Some Knowledge Beginner

Software: [✓ DOS] [✓ UNIX] [✓ VMS] Xwindows Other

Level: Very Experienced [✓ Experienced] Some Knowledge Beginner

5. **Experiences with ARIES:**

[✓ Worked well] ____Satisfactory ____Fair ____

6. **Experiences with HARP:**

____Worked well ____Satisfactory ____Fair [✓ ____]

7. **How useful was the Documentation on ARIES:**

The Users Manual: [✓ Good] ____Satisfactory ____Fair ____Poor

The material from Shooman's Notes: [✓ Good] ____Satisfactory ____Fair ____Poor

8. **How useful was the Documentation on HARP:**

The tutorial: [✓ Good] ____Satisfactory ____Fair ____Poor

The material from Shooman's notes: [✓ Good] ____Satisfactory ____Fair ____Poor

9. **Describe any errors or problems you had with aries:**

No errors found.

10. **Describe any errors or problems you had with HARP:**

No errors found.

11. **How could ARIES be improved?**

Enter data in a file format similar to that of SPICE Decks and not interactivly.
A major draw back.

12. **How could HARP be improved?**

Would be nice to provide calculated data in table format so it could easily be used. Enter data in the file like SPICE and not interactivly

13. **Would you be interested in doing a project or thesis for credition evaluating and improving ARIES and HARP:**

[✓ Yes] ____Maybe ____No

14. **Optional**

Name: Vladislav Feldman Address: 71-46 162nd st Flushing, N.Y. 11366

Phone Number: (718) 380-0907.

15. **Additional Comments:**

Use additional sheets if necessary.

USE OF ARIES AND HARP
QUESTIONAIRE FOR
CS907 FAULT-TOLERANT COMPUTING
EL617 SYSTEM RELIABILITY
SPRING TERM, FARMINGDALE, APRIL 25, 1990

When completed please return this questionaire to Prof. M. L. Shooman, Farmingdale RM222 or Ms. Jo Ann McDonald RM250.

The purpose of this questionaire is to gather information on the ease of use, accuracy, and modeling flexibility of the ARIES and HARP fault-tolerant computing programs. Sometimes you will be asked to comment from the viewpoint of a graduate student and sometimes as a practicing professional.

1. Education

Degree	Year Completed (or expected)	School	Concentration Area
B.S.E.E	1988	New York Tech	Electrical Engineering
M.S.E.E.	1992	Polytechnic	Digital Signal Processing

___ Full Time ___Part Time ___EE ___CS ___MS ___Phd. Other ________

2. Employment

[✓] Full Time ___Part Time ___Full Time Student ___Work Summers

Company: Grumman Aircraft Systems
Job Title: Associate Engineer / Fellow

3. Past Experience with reliability or probabilistic analysis.

Course work: ___Very Experienced ___Experienced ___Some Knowledge [✓] First Course

Describe academic experience: ________

Professional Experience: ___Very Experienced ___Experienced ___Some Knowledge ___First Course

Describe work experience: None

4. Computer Experience

Hardware: [✓] Main Frame ___Mini [✓] PC/XT/AT [✓] Work Station [✓] Special

Level: ___Very Experienced ___Experienced [✓] Some Knowledge ___Beginner

Software: [✓] DOS ___UNIX [✓] VMS ___Xwindows ___ Other ________

Level: ___Very Experienced ___Experienced [✓] Some Knowledge ___Beginner

5. Experiences with ARIES
__Worked well [✓ Satisfactory] __Fair __Poor
6. Experiences with HARP
__Worked well __Satisfactory __Fair [✓ Poor]
7. How useful was the Documentation on ARIES
The users manual __Good [✓ Satisfactory] __Fair __Poor
The material from Shooman's Notes: __Good [✓ Satisfactory] __Fair __Poor
8. How useful was the Documentation on HARP
The tutorial: --Good [✓ Satisfactory] __Fair __Poor
The material from Shooman's notes: --Good [✓ Satisfactory] __Fair __Poor
9. Describe any errors or problems you had with ARIES:
I was unsure of the truncation error on the output (ie. does 0 = .0000 or .0001 truncated?) - for single element subsystems, they must be input first - no-where is this stated.

10. Describe any errors or problems you had with HARP:
Very slow to run - the format of failure rate is not explicitly stated - the meaning of "include faults", "truncate @ low [illegible] failures", and "compute bounds" is not stated either - Output was not clear -- what does the format of the output mean? (i.e. state name, etc...) "truncated model redundancy [illegible]"

11. How could ARIES be improved?
Better documentation.

12. How could HARP be improved?
Better method of input - fault tree with all of its nodes is very cumbersome, better output data format - better documentation

evaluating and improving ARIES and HARP: __Yes [✓ Maybe] __No
14. OPTIONAL
Name: Marianne Fort Address: 18 Royal Oak Drive, Huntington, NY 11743
Phone Number: 385-1389
Please use additional sheets with your name on them if necessary!

USE OF ARIES AND HARP
QUESTIONAIRE FOR
CS907 FAULT-TOLERANT COMPUTING
EL617 SYSTEM RELIABILITY
SPRING TERM, FARMINGDALE, APRIL 25, 1990

When completed please return this questionaire to Prof. M. L. Shooman, Farmingdale RM222 or Ms. Jo Ann McDonald RM250.

The purpose of this questionaire is to gather information on the ease of use, accuracy, and modeling flexibility of the ARIES and HARP fault-tolerant computing programs. Sometimes you will be asked to comment from the viewpoint of a graduate student and sometimes as a practicing professional.

1. Education

Degree	Year Completed (or expected)	School	Concentration Area
BEEE	MANHATTAN COLLEGE	'84	ELEC. ENGG
MSEE	POLYTECHNIC	90	

___ Full Time ___Part Time ___EE ___CS ___MS ___Phd. Other ________

2. Employment

[✓] Full Time ___Part Time ___Full Time Student ___Work Summers

Company: FERRANT-VENUS
Job Title: DESIGN ENGINEER

3. Past Experience with reliability or probabilistic analysis.

Course work: ___Very Experienced ___Experienced [✓] Some Knowledge ___First Course
Describe academic experience: PROBABILITY, STOCHASTIC PROCESSES

Professional Experience: ___Very Experienced ___Experienced [✓] Some Knowledge ___First Course
Describe work experience: Quoting for increased MTBF

4. Computer Experience

Hardware: [✓] Main Frame ___Mini [✓] PC/XT/AT [✓] Work Station ___ Special

Level: ___Very Experienced ___Experienced [✓] Some Knowledge ___Beginner

Software: [✓] DOS [✓] UNIX [✓] VMS ___Xwindows ___ Other __________

Level: ___Very Experienced ___Experienced [✓] Some Knowledge ___Beginner

. Experiences with ARIES
__Worked well __Satisfactory [✓ Fair] __Poor

. Experiences with HARP
__Worked well [✓ Satisfactory] __Fair --Poor

. How useful was the Documentation on ARIES
The users manual __Good __Satisfactory __Fair [✓ Poor]
The material from Shooman's Notes: __Good __Satisfactory [✓ Fair] __Poor

. How useful was the Documentation on HARP
The tutorial: --Good __Satisfactory [✓ Fair] __Poor
The material from Shooman's notes: --Good __Satisfactory [✓ Fair] __Poor

. Describe any errors or problems you had with ARIES:
confounding in editing; wouldn't accept input as I am led to believe by notes in editing subsystem parameters

0. Describe any errors or problems you had with HARP:
tied up first GERK then STIFF SOLVER when I input λ=1 in an effort to temporarily (for a particular mission that is) remove a component from the model (a paralleled component)

11. How could ARIES be improved?
I can't judge that since my bad experiences may have been alleviated by more thorough documentation

12. How could HARP be improved?
I would like to be able to eliminate a component by making it fail immediately - i.e. λ=1 as mentioned above

evaluating and improving ARIES and HARP: __Yes __Maybe [✓ No]

4. OPTIONAL
Name:________________ Address:________________
Phone Number:________________
Please use additional sheets with your name on them if necessary!

B: are you interested in doing a project for credit in

USE OF ARIES AND HARP QUESTIONAIRE FOR:

CS907 FAULT-TOLERANT COMPUTING
EL617 SYSTEM RELIABILITY

SPRING TERM FARMINGDALE APRIL 25, 1990

When completed please return this questionaire to Prof. M. L. Shooman, Farmingdale Room 222 or Ms. JoAnn McDonald Room 250.

The purpose of this questionaire is to gather information on the ease of use, accuracy, and modeling flexibility of the ARIES and HARP fault-tolerant computing programs. Sometimes you will be asked to comment from the viewpoint of a graduate student and sometimes as a practicing professional.

1. **Education:**

Degree	Year Completed (or expected)	School	Concentration Area
MSEE	1979	POLYTECHNIC UNIV. OF BUCHAREST — ROMANIA —	ELECTRONICS & TELECOMMUNICATIONS

[✓] Full Time ___ Part Time ___ EE [✓] CS [✓] MS 2 [✓] Ph.D. Other: ______

2. **Employment:**

[✓] Full Time ___ Part Time [✓] Full Time Student ___ Work Summers

Company: POLYTECHNIC UNIVERSITY - NEW YORK

Job Title: TEACHING FELLOW

3. **Past Experience with reliability or probabilistic analysis:**

Course work: ___ Very Experienced [✓] Experienced ___ Some Knowledge ___ First Course

Describe academic experience:
EL 530, EL 630

Professional Experience:

___ Very Experienced [✓] Experienced ___ Some Knowledge ___ First Course

Describe work experience: Hardware & Software Design with control / Communication Systems

4. **Computer Experience:**

Hardware: ___ Main Frame ___ Mini [✓] PC/XT/AT ___ Work Station ___ Special

Level: ___ Very Experienced [✓] Experienced ___ Some Knowledge ___ Beginner

Software: [✓] DOS [✓] UNIX ___ VMS ___ Xwindows ___ Other ______

Level: ___ Very Experienced [✓] Experienced ___ Some Knowledge ___ Beginner

5. **Experiences with ARIES:**

____Worked well [✓ Satisfactory] ____Fair ____

6. **Experiences with HARP:**

____Worked well [✓ Satisfactory] ____Fair ____

7. **How useful was the Documentation on ARIES:**

The Users Manual: ____Good [✓ Satisfactory] ____Fair ____Poor

The material from Shooman's Notes: [✓ Good] ____Satisfactory ____Fair ____Poor

8. **How useful was the Documentation on HARP:**

The tutorial: [✓ Good] ____Satisfactory ____Fair ____Poor

The material from Shooman's notes: [✓ Good] ____Satisfactory ____Fair ____Poor

9. **Describe any errors or problems you had with aries:**

Does not compute availability for type 3 systems.

10. **Describe any errors or problems you had with HARP:**

Is not able to compute MTTF
Cannot handle spares.

11. **How could ARIES be improved?**

Include ESCAPE feature.
Add PRINT capability.

12. **How could HARP be improved?**

see item 10 above.

13. **Would you be interested in doing a project or thesis for credition evaluating and improving ARIES and HARP:**

____Yes [✓ Maybe] ____No

14. **Optional**

Name: IOANA BANICESCU Address: 23 HILLVALE RD., SYOSSET, NY 11791

Phone Number: (516) 822-2669

15. **Additional Comments:**

Use additional sheets if necessary.

USE OF ARIES AND HARP QUESTIONAIRE FOR:

CS907 FAULT-TOLERANT COMPUTING
EL617 SYSTEM RELIABILITY

SPRING TERM FARMINGDALE APRIL 25, 1990

When completed please return this questionaire to Prof. M. L. Shooman, Farmingdale Room 222 or Ms. JoAnn McDonald Room 250.

The purpose of this questionaire is to gather information on the ease of use, accuracy, and modeling flexibility of the ARIES and HARP fault-tolerant computing programs. Sometimes you will be asked to comment from the viewpoint of a graduate student and sometimes as a practicing professional.

1. **Education:**

Degree	Year Completed (or expected)	School	Concentration Area
M.S.E.E.	1986	Polytechnic	Information Science

___Full Time ✓ Part Time ✓ EE ___CS ___MS ✓ Ph.D. Other:________

2. **Employment:**

✓ Full Time ___Part Time ___Full Time Student ___Work Summers

Company: LNR Communications

Job Title: Consultant

3. **Past Experience with reliability or probabilistic analysis:**

Course work: ___Very Experienced ✓ Experienced ___Some Knowledge ___First Course

Describe academic experience: EL-530, EL-630

Professional Experience:

✓ Very Experienced ___Experienced ___Some Knowledge ___First Course

Describe work experience: Communications systems, Microwave satellite communications

4. **Computer Experience:**

Hardware: ___Main Frame ___Mini ✓ PC/XT/AT ___Work Station ___Special

Level: ___Very Experienced ✓ Experienced ___Some Knowledge ___Beginner

Software: ✓ DOS ✓ UNIX ___VMS ✓ Xwindows ___Other________

Level: ___Very Experienced ✓ Experienced ___Some Knowledge ___Beginner

5. **Experiences with ARIES:**

____Worked well [✓ Satisfactory] ____Fair ____

6. **Experiences with HARP:**

[✓ Worked well] ____Satisfactory ____Fair ____

7. **How useful was the Documentation on ARIES:**

The Users Manual: ____Good [✓ Satisfactory] ____Fair ____Poor

The material from Shooman's Notes: [✓ Good] ____Satisfactory ____Fair ____Poor

8. **How useful was the Documentation on HARP:**

The tutorial: [✓ Good] ____Satisfactory ____Fair ____Poor

The material from Shooman's notes: [✓ Good] ____Satisfactory ____Fair ____Poor

9. **Describe any errors or problems you had with aries:**

Must enter elements in ascending order of number of elements.
Will not compute availability for type 3 system

10. **Describe any errors or problems you had with HARP:**

No problems but cannot compute MTTF and cannot handle spares

11. **How could ARIES be improved?**

Correct problems noted in 9.
Add print capability.

12. **How could HARP be improved?**

Add MTTF and spares capability

13. **Would you be interested in doing a project or thesis for credition evaluating and improving ARIES and HARP:**

____Yes [✓ Maybe] ____No

14. **Optional**

Name:______________________________ Address:______________________________

Phone Number:__________

15. **Additional Comments:**

____

____

____

____

____

____

____

____

____

____

____

____

Use additional sheets if necessary.

USE OF ARIES AND HARP QUESTIONAIRE FOR CS907 FAULT-TOLERANT COMPUTING EL617 SYSTEM RELIABILITY SPRING TERM, FARMINGDALE, APRIL 25, 1990

When completed please return this questionaire to Prof. M. L. Shooman, Farmingdale RM222 or Ms. Jo Ann McDonald RM250.

The purpose of this questionaire is to gather information on the ease of use, accuracy, and modeling flexibility of the ARIES and HARP fault-tolerant computing programs. Sometimes you will be asked to comment from the viewpoint of a graduate student and sometimes as a practicing professional.

1. Education

Degree	Year Completed (or expected)	School	Concentration Area
BS	JAN, 1990		ELECTRICAL ENGINEERING

[✓ Full Time] ___Part Time [✓ EE] ___CS ___MS ___Phd. Other ________

2. Employment

___ Full Time ___Part Time ___Full Time Student ___Work Summers

Company: ________
Job Title: ________

3. Past Experience with reliability or probabilistic analysis.

Course work: ___Very Experienced ___Experienced [✓ Some Knowledge] ___First Course

Describe academic experience: ThdIntroductory to Probability

Professional Experience: ___Very Experienced ___Experienced ___Some Knowledge ___First Course

Describe work experience: ________

4. Computer Experience

Hardware: [✓ Main Frame] ___Mini [✓ PC/XT/AT] [✓ Work Station] ___ Special

Level: ___Very Experienced ___Experienced [✓ Some Knowledge] ___Beginner

Software: [✓ DOS] [✓ UNIX] [✓ VMS] ___Xwindows ___ Other ________

Level: ___Very Experienced ___Experienced [✓ Some Knowledge] ___Beginner

. Experiences with ARIES
__Worked well __Satisfactory [✓ Fair] __Poor

. Experiences with HARP
__Worked well [✓ Satisfactory] __Fair --Poor

. How useful was the Documentation on ARIES
The users manual __Good [✓ Satisfactory] __Fair __Poor
The material from Shooman's Notes: __Good [✓ Satisfactory] __Fair __Poor

. How useful was the Documentation on HARP
The tutorial: [✓ Good] __Satisfactory __Fair __Poor
The material from Shooman's notes: --Good [✓ Satisfactory] __Fair __Poor

. Describe any errors or problems you had with ARIES:
1) When a subsystem has no active module and no failure, it must be entered first. 2) When I created two identical subsystem with the same parameters, two different results are created

0. Describe any errors or problems you had with HARP:
Missing data 2- the output from solver when interlin[illegible]

11. How could ARIES be improved?
Aries' reliability can be improved.

12. How could HARP be improved?
-[illegible] improvements to HARP would be implementing the problem with graphic and the problem of missing blocks of output of the solver between time interval

evaluating and improving ARIES and HARP: __Yes __Maybe __No

.4. OPTIONAL
Name:____________ Address:____________
Phone Number:____________
Please use additional sheets with your name on them if necessary!

I am planning to complete Senior Project

USE OF ARIES AND HARP QUESTIONAIRE FOR CS907 FAULT-TOLERANT COMPUTING EL617 SYSTEM RELIABILITY SPRING TERM, FARMINGDALE, APRIL 25, 1990

When completed please return this questionaire to Prof. M. L. Shooman, Farmingdale RM222 or Ms. Jo Ann McDonald RM250.

The purpose of this questionaire is to gather information on the ease of use, accuracy, and modeling flexibility of the ARIES and HARP fault-tolerant computing programs. Sometimes you will be asked to comment from the viewpoint of a graduate student and sometimes as a practicing professional.

1. Education

Degree	Year Completed (or expected)	School	Concentration Area
EE/CS	6/1990		Software / System Engineering

[✓ Full Time] ___Part Time [✓ EE] ___CS ___MS ___Phd. Other ________

2. Employment

___ Full Time ___Part Time ___Full Time Student [✓ Work Summers]

Company: NYNEX
Job Title: SERVICE TECHNICIAN

3. Past Experience with reliability or probabilistic analysis.

Course work: ___Very Experienced [✓ Experienced] ___Some Knowledge ___First Course

Describe academic experience: Quite experienced taken courses in D.E., Linear Algebra, Probability, Computational Theory

Professional Experience: ___Very Experienced [✓ Experienced] ___Some Knowledge ___First Course

Describe work experience: Phone installation & Repair work In Main Office Connections

4. Computer Experience

Hardware: ___ Main Frame ___Mini [✓ PC/XT/AT] ___Work Station ___ Special

Level: ___Very Experienced ___Experienced ___Some Knowledge [✓ Beginner]

Software: [✓ DOS] [✓ UNIX] ___VMS ___Xwindows ___ Other ________

Level: ___Very Experienced [✓ Experienced] ___Some Knowledge ___Beginner

. Experiences with ARIES
__Worked well __Satisfactory [✓ Fair] __Poor

. Experiences with HARP
__Worked well [✓ Satisfactory] __Fair --Poor

. How useful was the Documentation on ARIES
The users manual __Good [✓ Satisfactory] __Fair __Poor
The material from Shooman's Notes: [✓ Good] __Satisfactory __Fair __Poor

. How useful was the Documentation on HARP
The tutorial: [✓ Good] __Satisfactory [✓ __Fair] __Poor
The material from Shooman's notes: [✓ Good] __Satisfactory __Fair __Poor

. Describe any errors or problems you had with ARIES:
Hard inputing values for D, CS, Mu, Lamda
Not Accepting (always — even if correct)

0. Describe any errors or problems you had with HARP:
VERY SLOW
CANNOT GET RUNS
HARD TOO BREAK OUT OF PROGRAM IF INPUT IS WRONG

11. How could ARIES be improved?
Make it more user friendly; Let user know what inputs could be put in if the input of values are incorrect.

12. How could HARP be improved?
Make it user friendly although it is more user friendly than Aries, Make it run faster, Give us the option to print out systems evaluation.

~~evaluating and improving ARIES and HARP~~: __Yes __Maybe [✓ No]

4. OPTIONAL
Name: Sunil Katwala Address: 14 Ribbon Lane
Phone Number: 516-579-8727
Please use additional sheets with your name on them if necessary!

Are you interested in Doing a Project in Aries or Harp for credit? No.

USE OF ARIES AND HARP
QUESTIONAIRE FOR
CS907 FAULT-TOLERANT COMPUTING
EL617 SYSTEM RELIABILITY
SPRING TERM, FARMINGDALE, APRIL 25, 1990

When completed please return this questionaire to Prof. M. L. Shooman, Farmingdale RM222 or Ms. Jo Ann McDonald RM250.

The purpose of this questionaire is to gather information on the ease of use, accuracy, and modeling flexibility of the ARIES and HARP fault-tolerant computing programs. Sometimes you will be asked to comment from the viewpoint of a graduate student and sometimes as a practicing professional.

1. Education

Degree	Year Completed (or expected)	School	Concentration Area
BSE	Princeton Univ '82		Mech and Aerospace Eng.
MSE	'90 - EE		Systems Engineering

___ Full Time [✓ Part Time] [✓ EE] ___CS ___MS ___Phd. Other ________

2. Employment

[✓ Full Time] ___Part Time ___Full Time Student ___Work Summers

Company: Grumman Corp.
Job Title: Master's Fellow

3. Past Experience with reliability or probabilistic analysis.

Course work: ___Very Experienced [✓ Experienced] ___Some Knowledge ___First Course

Describe academic experience: In systems concentration but have taken many probability and applied mathematics courses

Professional Experience: ___Very Experienced ___Experienced [✓ Some Knowledge] ___First Course

Describe work experience: On programs some discussion of reliability was involved on the job.

4. Computer Experience

Hardware: [✓ Main Frame] [✓ Mini] [✓ PC/XT/AT] [✓ Work Station] [✓ Special]

Level: [✓ Very Experienced] ___Experienced ___Some Knowledge ___Beginner

Software: [✓ DOS] [✓ UNIX] [✓ VMS] [✓ Xwindows] ___ Other ________

Level: [✓ Very Experienced] ___Experienced ___Some Knowledge ___Beginner

. Experiences with ARIES
  [✓ Worked well] __Satisfactory __Fair __Poor
. Experiences with HARP
  __Worked well [✓ Satisfactory] __Fair --Poor
. How useful was the Documentation on ARIES
The users manual __Good [✓ Satisfactory] __Fair __Poor
The material from Shooman's Notes: [✓ Good] __Satisfactory __Fair __Poor
. How useful was the Documentation on HARP
The tutorial: --Good [✓ Satisfactory] __Fair __Poor
The material from Shooman's notes: --Good __Satisfactory [✓ Fair] __Poor
. Describe any errors or problems you had with ARIES:
- Allows you to generate a system which contains repair elements
but will not let you do availability analysis.
- help file did not work.

0. Describe any errors or problems you had with HARP:
- sequential manner it operated was [illegible]
meaning incorporate [illegible], [illegible] merging into 1 program.
- too slow ; because of the linking to NASA.

11. How could ARIES be improved?
- Improve performance and documentation on line.
- Plotting routines as an extra.
- Draw model before analyses.

12. How could HARP be improved?
- Graphical output
- Hard copy procedure
* make a drawing of the model before it is run on screen

evaluating and improving ARIES and HARP: __Yes [✓ Maybe] __No
4. OPTIONAL
Name: Thomas A Dolan Address: 120-12 85th Ave, Kew Gardens NY 11415
Phone Number: 718-849-9836
Please use additional sheets with your name on them if necessary!

Are you interested in doing a project for credit on

USE OF ARIES AND HARP QUESTIONAIRE FOR:

CS907 FAULT-TOLERANT COMPUTING
EL617 SYSTEM RELIABILITY

SPRING TERM FARMINGDALE APRIL 25, 1990

When completed please return this questionaire to Prof. M. L. Shooman, Farmingdale Room 222 or Ms. JoAnn McDonald Room 250.

The purpose of this questionaire is to gather information on the ease of use, accuracy, and modeling flexibility of the ARIES and HARP fault-tolerant computing programs. Sometimes you will be asked to comment from the viewpoint of a graduate student and sometimes as a practicing professional.

1. **Education:**

Degree	Year Completed (or expected)	School	Concentration Area
M.S.	FALL 1990	N.Y. Polytechnic University	Computer Science

___Full Time [✓ Part Time] ___EE ___CS ___MS ___Ph.D. Other:________

2. **Employment:**

[✓ Full Time] ___Part Time ___Full Time Student ___Work Summers

Company: SYLTEL INFORMATION SYSTEMS INC.

Job Title: SOFTWARE DESIGN ENGINEER

3. **Past Experience with reliability or probabilistic analysis:**

Course work: ___Very Experienced ___Experienced [✓ Some Knowledge] ___First Course

Describe academic experience: GRAPH / Queueing Theory, Network Design,

Professional Experience:

___Very Experienced ___Experienced [✓ Some Knowledge] ___First Course

Describe work experience: PROBABALISTIC MODELS for Queuing Application M/M/1 Queue

4. **Computer Experience:**

Hardware: ___Main Frame ___Mini [✓ PC/XT/AT] [✓ Work Station] ___Special

Level: ___Very Experienced [✓ Experienced] ___Some Knowledge ___Beginner

Software: [✓ DOS] [✓ UNIX] [✓ VMS] [✓ Xwindows] [✓ Other] Aegis

Level: ___Very Experienced [✓ Experienced] ___Some Knowledge ___Beginner

5. **Experiences with ARIES:**

____Worked well [✓ Satisfactory] ____Fair ____

6. **Experiences with HARP:**

____Worked well [✓ Satisfactory] ____Fair ____

7. **How useful was the Documentation on ARIES:**

The Users Manual: [✓ Good] ____Satisfactory ____Fair ____Poor

The material from Shooman's Notes: ____Good [✓ Satisfactory] ____Fair ____Poor

8. **How useful was the Documentation on HARP:**

The tutorial: ____Good [✓ Satisfactory] ____Fair ____Poor

The material from Shooman's notes: ____Good [✓ Satisfactory] ____Fair ____Poor

9. **Describe any errors or problems you had with aries:**

Any single point Failure must be entered first. There is No way to escape from subsystem input until you are finished successfully inputting input parameters. No way to get comprehensible output to file.

10. **Describe any errors or problems you had with HARP:**

No way to get comprehensible output to file.

11. **How could ARIES be improved?**

Better Human interface, More info. when using help facility.

12. **How could HARP be improved?**

——

13. **Would you be interested in doing a project or thesis for credition evaluating and improving ARIES and HARP:**

____Yes ____Maybe [✓ No]

14. **Optional**

Name:______________________________ Address:______________________________

Phone Number:__________

15. **Additional Comments:**

____

____

____

____

____

____

____

____

____

____

____

____

Use additional sheets if necessary.

USE OF ARIES AND HARP QUESTIONAIRE FOR:

CS907 FAULT-TOLERANT COMPUTING
EL617 SYSTEM RELIABILITY

SPRING TERM FARMINGDALE APRIL 25, 1990

When completed please return this questionaire to Prof. M. L. Shooman, Farmingdale Room 222 or Ms. JoAnn McDonald Room 250.

The purpose of this questionaire is to gather information on the ease of use, accuracy, and modeling flexibility of the ARIES and HARP fault-tolerant computing programs. Sometimes you will be asked to comment from the viewpoint of a graduate student and sometimes as a practicing professional.

1. **Education:**

Degree	Year Completed (or expected)	School	Concentration Area
MSCS	Fall-1990	Polytechnic	Computer Science
BSEE	1980	INDIA	Digital Computer

____Full Time [✓ Part Time] ____EE [✓ CS] ____MS ____Ph.D. Other:________

2. **Employment:**

[✓ Full Time] ____Part Time ____Full Time Student ____Work Summers

Company: GULL ELECTRONICS SYSTEM DIV.

Job Title: SENIOR ENGINEER

3. **Past Experience with reliability or probabilistic analysis:**

Course work: ____Very Experienced ____Experienced ____Some Knowledge [✓ First Course]

Describe academic experience:________________

Professional Experience:

____Very Experienced ____Experienced ____Some Knowledge [✓ First Course]

Describe work experience:________________

4. **Computer Experience:**

Hardware: ____Main Frame ____Mini [✓ PC/XT/AT] ____Work Station ____Special

Level: [✓ Very Experienced] ____Experienced ____Some Knowledge ____Beginner

Software: [✓ DOS] ____UNIX ____VMS ____Xwindows ____Other________________

Level: [✓ Very Experienced] ____Experienced ____Some Knowledge ____Beginner

5. **Experiences with ARIES:**

___Worked well [✓ Satisfactory] ___Fair ___

6. **Experiences with HARP:**

___Worked well ___Satisfactory [✓ Fair] ___

7. **How useful was the Documentation on ARIES:**

The Users Manual: ___Good ___Satisfactory [✓ Fair] ___Poor

The material from Shooman's Notes: ___Good [✓ Satisfactory] ___Fair ___Poor

8. **How useful was the Documentation on HARP:**

The tutorial: ___Good [✓ Satisfactory] ___Fair ___Poor

The material from Shooman's notes: [✓ Good] ___Satisfactory ___Fair ___Poor

9. **Describe any errors or problems you had with aries:**

If you try to Configure system with two parallel unit and a single unit, than Ariel only allows to configure, If you Configure single unit first, otherwise it gives Error on parameter Inputs.

10. **Describe any errors or problems you had with HARP:**

—

11. **How could ARIES be improved?**

- Better documentation is desired for the parameter Explanation.
- Allow to store result in file

12. **How could HARP be improved?**

- Harp documentation does not specify how to input markov mode with repair.
- Run in Unix Environment

13. **Would you be interested in doing a project or thesis for credition evaluating and improving ARIES and HARP:**

___Yes ___Maybe [✓ No]

14. **Optional**

Name: JAGDISHKUMAR SHAH Address: 28 Henry Ave. Selden. N.Y. 11784

Phone Number: 516-696-6839

15. **Additional Comments:**

Use additional sheets if necessary.

USE OF ARIES AND HARP QUESTIONAIRE FOR:

CS907 FAULT-TOLERANT COMPUTING
EL617 SYSTEM RELIABILITY

SPRING TERM FARMINGDALE APRIL 25, 1990

When completed please return this questionaire to Prof. M. L. Shooman, Farmingdale Room 222 or Ms. JoAnn McDonald Room 250.

The purpose of this questionaire is to gather information on the ease of use, accuracy, and modeling flexibility of the ARIES and HARP fault-tolerant computing programs. Sometimes you will be asked to comment from the viewpoint of a graduate student and sometimes as a practicing professional.

1. **Education:**

Degree	Year Completed (or expected)	School	Concentration Area
MS	1991	Polytechnic University	Computer Science
BS	1985	Hofstra University	Computer Science

___Full Time [✓ Part Time] ___EE ___CS ___MS ___Ph.D. Other:________

2. **Employment:**

[✓ Full Time] ___Part Time ___Full Time Student ___Work Summers

Company: Grumman Aircraft Systems

Job Title: Engineer - Software Development

3. **Past Experience with reliability or probabilistic analysis:**

Course work: ___Very Experienced ___Experienced [✓ Some Knowledge] ___First Course

Describe academic experience: Two courses in probabilities

Professional Experience:

___Very Experienced ___Experienced ___Some Knowledge ___First Course

Describe work experience: None

4. **Computer Experience:**

Hardware: [✓ Main Frame] [✓ Mini] [✓ PC/XT/AT] ___Work Station ___Special

Level: [✓ Very Experienced] [✓ Experienced] ___Some Knowledge ___Beginner

Software: [✓ DOS] [✓ UNIX] [✓ VMS] ___Xwindows ___Other________

Level: [✓ Very Experienced] ___Experienced ___Some Knowledge ___Beginner

5. **Experiences with ARIES:**

____Worked well [✓ Satisfactory] ____Fair ____

6. **Experiences with HARP:**

____Worked well [✓ Satisfactory] ____Fair ____

7. **How useful was the Documentation on ARIES:**

The Users Manual: ____Good ____Satisfactory [✓ Fair] ____Poor

The material from Shooman's Notes: ____Good [✓ Satisfactory] ____Fair ____Poor

8. **How useful was the Documentation on HARP:**

The tutorial: ____Good [✓ Satisfactory] ____Fair ____Poor

The material from Shooman's notes: [✓ Good] ____Satisfactory ____Fair ____Poor

9. **Describe any errors or problems you had with aries:**

The software was a little unclear as to when spares information should be entered.

10. **Describe any errors or problems you had with HARP:**

11. **How could ARIES be improved?**

Eliminate prompts for variables which are irrelevant to the particular model.

12. **How could HARP be improved?**

Some models that are placed in the system are too tedious to enter. Simplify this.

13. **Would you be interested in doing a project or thesis for credition evaluating and improving ARIES and HARP:**

____Yes ____Maybe [✓ No]

14. **Optional**

Name: Jim Boehmler Address: 3400 Jerusalem Ave. Wantagh

Phone Number: 826-4260

15. **Additional Comments:**

Use additional sheets if necessary.

TABLE A-5 Summary of Responses to the Questionnaire

Question 1 Degree level

Expecting an MSEE this year - 4

Expecting an MSEE in the next few years - 2

Expecting an MSCS this year - 2

Working toward a PhdEE - 2

Expecting a BSEE this year - 2

Question 2 Employers

Full time Poly Undergraduates - 2

Full time Poly Fellow

Companies:

Harris
Standard Microsystems
Grumman Corporation - 3
Ferrant-Venus
NYNEX
Syltel Industrial Systems
Gull Electronic Systems

Questions 3,4 Experience with Reliability and Computers

	Very Exp.	Experienced	Some Know.	First Course
Reliability/ Prob. Courses	0	5	4	3
Reliability/ Prob. Work	1	2	4	5
Computers	4	5	3	0

Questions 5,6 Overall Experience with ARIES and HARP

	Worked Well	Satisfactory	Fair	Poor
ARIES	2	7	3	0
HARP	1	8	2	1

Questions 7,8 Documentation for ARIES and HARP

	Good	Satisfactory	Fair	Poor
ARIES				
Manual[1]	2	6	3	1
Notes[2]	5	5	2	0
HARP				
Manual[3]	4	7	1	0
Notes[4]	8	2	2	0

[1] Prof. Shooman wrote a 20 page summary of ARIES.
[2] Prof. Shooman supplied course notes including a few simple ARIES examples.
[3] Photocopies of the HARP Tutorial (Rothman 1989) were distributed to all students.
[4] Prof. Shooman supplied a script of a session showing how to describe a simple problem to ARIES.

APPENDIX B

DETAILS OF THE INSTALLATION OF HARP ON POLYTECHNIC COMPUTERS

APPENDIX B

The test of installation ease was to be carried out by installing the HARP program on the Poly Gould computer operating under the Unix operating system. This test was carried out by Martin Shooman (who has no system programming experience) with the occasional help of a senior systems programmer (a unix expert and a VMS beginner) and a junior systems programmer (an intermediate in unix and a beginner in VMS).

It was assumed that the initial tapes for HARP which were received in June 1989 were compatible with a "general Unix" system and only minor modifications would be required to make it compatible with the Poly Gould computer.

Our initial tests went slowly because of some differences in the file names between the HARP manual and the names used on the tape. These difficulties were resolved by determining the equivalencies via reading the heading text in each file and determining equivalencies for the three major programs. The files were then renamed care3.f, carein.f, and covrge.f to correspond to the manual.

The next step was to compile these three major programs using the Gould Fortran compiler. Compilation took 20-30 minutes for each program and resulted in several warnings and one major error. The error turned out to be an illegal in line comment in Gould Fortran (apparently it was legal in VAX Fortran) which was "!ANNA should this be KP". This was changed to a comment on the next line. The program then compiled.

Attempts to execute the compiled files resulted in system errors which could not be resolved. Perhaps the program was looking for VMS libraries which did not exist on the Poly computer.

A call to Rudy Williams of NASA Langley suggested that the tapes I had were probably not Unix versions but VMS versions which were made Unix readable but not compatible with Unix. The conversion of the HARP programs from VMS to Unix had been assigned as a full time task to one of the Airlab system programmers.

Subsequent calls to Sal Bavuso verified that I did not have a Unix version of the program and he promised to send me a Unix version when available.

In the interim Shooman obtained a HARP account on the NASA Langley Airlab computer and connected to HARP from Poly terminals and PC's via the internet network. Shooman considered trying the PC version of HARP, however, this would require the following:

(a) Lahey Fortran Compiler 3.01
(b) MS-DOS Linker
(c) Lattice MS-DOS C Compiler Ver. 3.0
(d) MS-DOS Ver. 3.2
(e) GSS*GKS Graphics Package 2.02

We had copies at Poly of item (d) and probably (c) but not (a), (b), and (e) thus no further investigations of PC HARP were attempted.

Once I received the Unix HARP version (uxharp-61), I attempted (without success) to compile, link, and load the three major programs, fiface, harpeng, and tdrive. The problem was to do an error (incompatibility) between the different versions of Fortran (see Table B.1 for complete details). Once the Fortran incompatibility was solved all three programs compiled and linked in 5-12 minutes. Execution of tdrive checked with the results of the same execution on the Langley computer. Unfortunately fiface and harpeng gave system errors which were never resolved and testing continued via a network connection to the Langley computer. The procedure was to enter Shooman's account (or the student's accounts on the Gould computer), connect to the Langley computer via the Telnet network and then log into the Langley computer and run HARP. This worked well except for some delays in the network connections.

The conclusion of these tests is that the versions of HARP which were then available were not easily transportable to a Unix machine unless the services of a dedicated and experienced systems programmer were available. Shooman called a colleague who is a reliability analyst at Grumman Aircraft and was told that their initial installation of HARP (VMS version) on a VAX running VMS took about one day of time for an experienced system programmer.

Table B-1 (E-Mail from John Buck)

```
From john Fri Mar  9 12:59:30 1990
Received: by polyof.poly.edu (4.12/UTX/32 1.2)
      id AA10886; Fri, 9 Mar 90 12:54:43 est
Date: Fri, 9 Mar 90 12:54:43 est
From: john ( John Buck )
Message-Id: <9003091754.AA10886@polyof.poly.edu>
Re: fortran problem you have been having
Apparently-To: shooman
Status: RO
```

The problem is with the fortran compiler, but there is a workaround...
(It is gonna take some time for me to fix the compiler (a couple of
days at least)), so you may want to consider working around the problem.

Any piece of code like:

```
      IF(condition)THEN
label             statement
            statement
            IF(condition)THEN
                  statement
                  GOTO label
            ENDIF
      ENDIF
```

will cause the problem. The compiler maps the "label" (whatever number
the user chose) into some internal LABEL, like L32 or L545 for example.
The problem is, if a label appears immediately after something like
a THEN, ELSE, WHILE, DO, etc. the compiler forgets to generate a label
for it, and when you do a GOTO later, it is (obviously) undefined.

The workaround is to put a CONTINUE before the label statement, like:

```
      IF(condition)THEN
            CONTINUE
label             statement
            statement
            IF(condition)THEN
                  statement
                  GOTO label
            ENDIF
      ENDIF
```

This works. The problem is finding all the places in your code
where this happens... I fixed one in nxt.f, around lines 166-175
I added the CONTINUE and the L32 message (one of them) went away.

I will let you know as soon as I fix (or find) the problem with the
compiler.

APPENDIX C

DETAILS COMMENTS ON PROBABLISTIC MODELING WITH HARP

C.1 REVIEW OF THE PAPER BY HAYHURST ET AL.

"A Critical Assessment of the HARP Program," Kelly J. Hayhurst, Rickey W. Butler, and Sally C. Johnson, Draft NASA Technical Memorandum 102607, NASA Langley, Feb. 1990.

This is a very interesting report which provides much valuable material on how the HARP program works, how to model fault-tolerant systems with the program and the limitations of the program. The following comments are offered to improve the clarity and expand some sections of the report. The major comments deal with the mathematics, structure, or understanding of the program, whereas, the minor comments are mainly editorial in nature.

MAJOR COMMENTS

1. p3 "1.3 The Assessment Approach" The 4 stages of verification mentioned are good, however, they all seem to imply that HARP will be critiqued on an absolute basis. I think there should also be some relative basis to the discussion. For example, suppose that the only present alternatives to using HARP are to use CARE III or to make various rough analytical approximations. If these are truly the "generally available" alternatives, then HARP, even with its warts may be more attractive on a relative basis than on an absolute one. You can comment better than I on what the "practical" alternatives are and we can both comment on how well one can do with analytical approximations.

2. p5 To really understand how HARP works, it is necessary to briefly define the following terms and give a simple example of each: Fault-Occurrence Model, Fault Error Handling Model, Interfering Components Specification, Instantaneous Jump Model, Behavioral Decomposition. These definitions and explanations may be in White's report, however, some explanation is needed in this report unless the reader is expected to have read White's report first.

3. p5 (a) The term conservative is not well defined. I prefer to use the terms optimistic and pessimistic and apply them to the system reliability. Then optimistic means an upper bound on the system reliability and pessimistic means a lower bound on the system reliability. Since the HARP answer is an upper bound on the probability of failure, (UB_{sys})*, it is also a lower bound (pessimistic) on the reliability which is good.

   * The notation UB_{sys} is confusing since it refers to the system probability of failure rather than the system success.

   (b) It is always nice to have in addition to the pessimistic bound an optimistic upper bound on the system reliability, which would be a lower bound on the system probability of failure, and I assume this would be called LB_{sys} to be consistent with the HARP notation. The existence of both upper and lower bounds on reliability allows one to bracket the true reliability and calculate error bounds.

4. p6 (a) The term unreliability is used here as a synonym for system probability of failure, which is correct, however, there are enough problems with notation, thus system probability of failure should be used instead.

   (b) It is hard to understand exactly what bounds on the Instantaneous Jump Model, (UB_{ijm} and LB_{ijm}), represent since the instantaneous jump model has not been defined, nor has the behavioral decomposition process. For example:

   (1) If there are NO approximations involved in the decomposition process, then an upper bound on the decomposed model reliability is an upper bound on the system reliability.

(2) If the decomposition process produces a decomposed model which is itself an upper bound on the actual model, then an upper bound on the decomposed model reliability is an upper bound (perhaps a loose one, perhaps reasonably tight), on the system reliability.

(3) If the decomposition process produces a decomposed model which is itself a lower bound on the actual model, then an upper bound on the decomposed model reliability may or may not be a bound on the system reliability.

(4) If the decomposition process produces an approximate model which is not necessarily an upper or lower bound on the actual model, then an upper bound on the decomposed model reliability may or may not be a bound on the system reliability.

(c) "The UB_{ijm} is based on the same concept as UB_{sys} and is CLAIMED* to be an upper bound on system unreliability." Why do we need a second bound on system unreliability? Is UB_{ijm} a sharper bound? is it easier to calculate?

* Why use the word CLAIMED and cast doubt on the validity of UB_{ijm} when two sentences below the authors state "the validity of the bounds .... are not in dispute?"

(d) The following wording pertaining to the issues raised in (b) and (c) above seems clearer to me, is it correct?

(1) There are some bounds which establish the validity of the HARP model, however, there is not a complete set of bounds.

(2) A upper bound on the system failure probability, UB_{sys}, (lower bound on the system reliability), has been developed by McGough and Trivedi, unfortunately a flaw has been found in his proof, (is Trivedi working on removing the flaw?), however, Alan White has found an independent proof.

(3) The instantaneous jump model involves the following assumptions and approximations ....???

(4) Upper and lower bounds on the probability of failure associated with the instantaneous jump model, UB_{ijm} and LB_{ijm} have been developed by Trivedi et al. They have the following advantages when compared with UB_{sys}: ....???

(5) It is difficult to convert a bound on the instantaneous jump model to a system bound because of the assumptions and approximations which arise in the modeling process, however Trivedi claims that UB_{ijm} is also an upper bound on the system probability of failure, however, no detailed proof is given. Also there are no proofs relating LB_{ijm} to the the system reliability.

(e) In foot note 4, the term near-concident failures should be defined and explained.

5. p7 The term single critical-pair N-plex must be defined and examples must be given. The explanation in footnote 6 is too short.

6. p7 If I understand the given explanation of critical pair, then the example given in Appendix C.2 may help explain the nature of the approximation. This example could represent two redundant CPU's and two redundant memories where the "switching is perfect."

$$\begin{tikzcd}
0 \arrow[r, bend left, "x1"] \arrow[r, bend right, "x2"'] & 0 \arrow[r, bend left, "y1"] \arrow[r, bend right, "y2"'] & 0
\end{tikzcd}$$

7. p7 Based on the example which is analyized in Appendix C.2 the discrepancies in unreliability are quite serious in even this simple case. The percentage errors are 137% and 179%

8. p7,8 I agree that if this is how HARP treats "critical-pairs" then it is limited in modeling and comparing fault isolation strategies.

9. p7,8 If all the bounds require the assumption that the system components are critically coupled, then the bounds will only be valid for critical coupling and for any other situation the HARP answer will be an approximation of unknown (mathematically unbounded) accuracy.

10. p8 If you have upper and lower bounds, $A - e1 < C < A + e2$, they don't have to be symmetrical ($e1 = e2$) as long as both are reasonably sharp. In fact, if we are talking about system reliability I would accept just a lower (pessimistic bound) if that was all that was available.

11. p8,9 Perhaps the ICS=ALL and ICS=SAME would be of some help even with the problems cited. This needs more study.

12. p10 Explain how the recovery and redundancy management unit works in the model of Fig. 1. It seems you are saying that the recovery unit is perfect (never fails to detect a fault and never never produces a false alarm, i.e., detects a nonexistent fault) except when a second fault occurs before the first has been handled (recovery completed). If this is the model, state it and explain.

13. p11 Lines 9,10,11 are not clear, explain.

14. p11 Give Tables with 1-2 line explanations describing the alternative fault/error-handeling models supported by HARP and the options for ICS.

15. p12 Define interfering components.

16. p13 Explain in words the example of Fig. 3, and why the different types of failure rates, or is it just a hypothetical model with no physical counterpart.

17. p14 Since in some problems HARP infers the wrong failure rates for near-coincident faults, is there any way for the analyst who knows of this problem to force the program to use the correct rates via input?

18. p14 In Table 2 you compare HARP results with the "Correct" solution. By the correct solution I assume you mean the results of another computer program which you trust more, or do you mean an analytical solution?

19. An approximate analytical solution is given for the example in Figure 4 in Appendix C.3. The solution is based on failure modes M1, M2, M3, and M4 which correspond to the probabilities of states 3,6,9, and 11 of Figure 4. The results for Mission time =10 are:

$$P(M1) = 5.9901008 \times 10^{11}$$

$$P(M2) = 1.7946 \times 10^{13}$$

P(M3)	$= 1.8 \times 10^{16}$
P(M1) + P(M2) + P(M3)	$= 6.008972 \times 10^{11}$
P(M4)	$= 1.7919187 \times 10^{11}$
P(M1) + P(M2) + P(M3) + P(M4)	$= 7.8008907 \times 10^{11}$

Based upon these approximate analytical results and the unknown source of the column labeled Correct in Table 2, I am not sure which set of results I would label correct. (NOTE: See comment 39!)

20. p15 Next to last line: "unless two faults occur coincidently." What happens then?

21. p16 Footnote 8. I agree that when you have reliability values where R is close to one you should compare U = 1-R. The percentage difference between U1 and U2 is what is used in the report, [|U1-U2|/U1] x 100%. However, if U1 and U2 differ by an order of magnitude or more, the ratio U1/U2 is probably better to use. Note in no case would I use the term incorrect for either U1 or U2 unless I had an analytical solution that I was sure of or a correspondence of several independent (algorithms and code) computer solutions.

22. p17 Middle of page. "the problem manifests itself with less extreme values of the parameters." Reword for more clarity.

23. p17 Last line. "letting P_t = Probability that T exit (Transient) is taken, we have:" Explain what this means.

24. p18 I agree with the reasoning of Section 3.5

25. p18 Next to last line. "consisting of three independent triads to the HARP program. What does this mean. Is the example shown below a triad? Does independence mean that CPU, memory and I/O failures are all independent?

```
         z1              x1              y1
   /----->----\    /----->----\    /---->-----\
  /     z2     \  /     x2     \  /    y2      \
0------->------0-------->------0------->-------0
  \            /  \            /  \            /
   \---->-----/    \----->---/     \---->-----/
         z3              x3              y3

     CPU Triad     Memory Triad    I/O Channel Triad
```

26. p20 Line 1,2 Change words? "Clearly theorem(s) or bounds are needed ...."

27. p20 Middle of page. Not clear what you mean by design flaws. Does that mean the difference between input and math domains?

28. p20 Last line. "that have an intimate knowledge of the HARP tool." I strongly agree. This is made difficult by incomplete and confusing documentation.

29. p21 Middle of page "Two simultaneously failures can be accommodated." Why is this so? Is the system a 3 out of 6 system? But that doesn't agree with Fig. 13? Explain.

30. p22 Fourth line. What does 3. A majority of .... mean? Not clear explain.

31. p23 Fig. 15, the transition rate from (3,0,1) back to (4,0,0) is not labeled.

32. p24 The description of FTMP is too terse. More words plus a system block diagram (graph) using the <u>type of symbols</u> shown below (or an equivalent type of diagram) with additional words annotating the diagram would help a lot. This same comment applies to the discussion at the top of page 27. Show a model of a 6-plex, redundant power supplies.

```
      /|\                   x1             y1
       |             /----->----\     /---->-----\
       |            /            \   /            \
 On line system    0              0                0
       |            \     x2     / \      y2      /
      \|/            \----->----/   \---->-----/
                            .              .
                            .              .
                            .              .
                            .              .
                            x3             y3
      /|\              0---->----0    0---->----0
    Spares                            0---->----0
      \|/                                  y4
```

33. p27 Explain the terms NMR, multiple triads, multiple quads, critical pair N-plex.

34. p27 Last paragraph and footnote. Isn't this the sum of the number of combinations of 71 taken 0,1,2,and 3 at a time? I think this yields 1 + 71 + 4,970 + 57,155 = 62,190 = 62K?

35. p27 Would anyone ever want to solve a model with over 1,000 states? This would require the input of several hundred values for transition rates (very laborious) and how would one ever get data for so many parameters? Perhaps I am overlooking some classes of models. Can anyone suggest a practical problem of very large size?

36. p28 Fig. 19 Add additional labels in ( ) to 2000, 4000, 6000 seconds points, (33 min.), (67 min.), (100min.).

37. p28 Fig. 19. One can very crudely fit a model to the curve. Choose the form $T = KS^2$. Fit at 100 states and 700 seconds, which yields K = .07. Then at 200 states the formula predicts 2800 sec., and at 300 states 6300 seconds. Both the predicted points are approximately on the curve. A better fit could be obtained by plotting the data on semilog paper.

38. p29 Perhaps one could get analytical solutions for some of these problems.

39. p30 You state that these models have an exponential recovery rate. This seems different than the assumptions used in Chap. 3 and 4, where I assumed you were talking about a fixed recovery time. It makes a difference in the analytical solution. For example in Appendix B, one would have to compute the probability that the random variable time to next failure exceeds the random variable recovery time. (This is a standard computation which involves the convolution integral or any of several other change of random variable techniques). Stating the mean recovery time may imply an exponential or a normal recovery distribution. You have to be specific. All three variations could be analytically computed in the simpler cases.

40. p30 Give a graph for Example 5.1 and the other examples similar to that in comment 32.

41. p31. One can solve the Markov model for Fig. 20. It is not too difficult but long and requires solving 8 first order differential equations with drivers. Laplace transforms help organize the algebra.

42. p31 The results in Table 7 are quite convincing and lead one to believe that SURE, PAWS, and STEM agree and give the right answer, and that often HARP differs from the right answer. However, if SURE, PAWS, and STEM all share the same basic model formulation philosophy and differ mainly in the way numerical approximations are made, then the results are not 3 versus one but 3 dependent ones reducing to one versus one. Of course if the modeling philosophy and algorithms of SURE, PAWS, and STEM are really all different then it is 3 versus one. Maybe an analytical solution for a few of these problems is worth the effort?

43. p32 Figure 21 describes a situation with two types of faults, "hard" faults and transient faults? Explain. In 5.3 you discuss permanent faults. Are these yet a third category? Explain.

44. p33 Fig. 21, the "feedback paths" created by the transient fault-recovery rate couples the differential equations (the same way repair does) and makes solution harder, but still possible. (See comment 41.)

45. p34 Can we show why HARP differs so much for small failure rates? Would comparison with an analytic solution help? (See Tables 7, 9, 10).

46.p35,40 80,000 % error? This is inconsistent with what you say on p16. Also see comment 21. Also holds for Table 16,17,18 and others.

47. p41 Equation given only holds if E_1 and E_2 are independent. probably a good assumption except for electrostatic discharge, common power supplies, etc.

48. p47 "the answers given by the other analysis tools were in complete agreement." See comment 42.

49. p47 "Consequently, even an experienced user of HARP unaware of all the subtleties of the program could easily generate wrong answers .... This is especially true because the documentation is sparse, hard to understand, incomplete, and has some errors.

50. p.47ff. The new "nonstandard" gates. These gates are designed to deal with dependent failures and sequenced events. Since, the definition and use of these gates is not clear it would seem they are of limited usefulness. All the situations where these gates would be used can be easily formulated using a Markov model. For an example of such Markov dependency models see Shooman (1990, pp. 235, 236, 243-251).

MINOR COMMENTS

1. p1 "Several different types of fault-tolerant computing systems are currently in use ..." I doubt that these programs are as widely used as either the authors or I would wish.

2. p1 "Mathematical techniques such as k-out-of-n .....". Give some standard textbook references to the techniques.

3. p2 "... HARP has been widely publicized ..." I am not sure that your beta-test sites use it that widely?

4. p3 "Would it be clearer if the fourth problem area was called (4) reliability modeling with HARP fault-trees

5. p4 In footnote 1, differential equations solvers are dismissed as competitors to HARP because they only handle Markov models with constant transition rates. I am not sure whether this is the case with modern math packages. Also one would have to investigate the capabilities of modern math analysis programs such as MACSYMA, MATHEMATICA, and MATLAB.

6. p6 Typo in footnote 3? SIMLE should be SIMPLE?

7. p10 Add a clarifying sentence after line 10? " States 3,6,8 are failed states, and F_r the recovery time distribution.

8. p10 Define the terms: near-coincident, triad, system recovery, duplex, simplex.

9. p11 Typo, change bahavior to behavior on lines 4 and 5.

10. p22 Last paragraph. Somewhere earlier in the report you should explain that the models you are using are based upon two major classes of failure modes (you might even say you are ignoring other possible failure modes such as a false-alarm attempt to recover the system) , one where all the parts fail which you call exhaustion of parts, and one where while a recovery to a failure is in progress another failure occurs and confounds the recovery process, which you call two coincident failures. You really say this at the top of page 10, but you don't use the term exhaustion of components till later. Be a little more explicit.

11. p18 Note it is better if all major sections start on a new page.

C.2 AN EXAMPLE WHICH ILLUSTRATES THE PROBLEMS WITH HARP'S TREATMENT OF NEAR-COINCIDENT FAULTS

The following example may help explain the nature of the HARP near-coincident faults approximation. This example could represent two redundant CPU's and two redundant memories, where the "switching is perfect."

```
            x1                  y1
      /----->----\      /---->-----\
     /            \    /            \
    0              0                 0
     \            /    \            /
      \----->----/      \---->-----
            x2                  y2
```

The reliability of this system in terms of probability of success or failure is given by:

$$R_1 = P[x_1y_1 + x_1y_2 + x_2y_1 + x_2y_2] = 1 - P[x_1'x_2' + y_1'y_2'] \qquad (1)$$

where P[x] is the probability that x is successful and P[x'] is the probability that x fails.

The correct reliability expression is given by Eq. (1).

HARP would evaluate this expression as:

$$R_2 = 1 - P[x_1'x_2' + y_1'y_2' + x_1'y_1' + x_1'y_2' + x_2'y_1' + x_2'y_2'] \qquad (2)$$

If we assume all element failures are independent and $P(x_1') = P(x_2') = q_1$ and $P(y_1') = P(y_2') = q_2$ then Eq. (1) yields:

$$R_1 = 1 - q_1^2 - q_2^2 + (q_1^2)(q_2^2) \qquad (3)$$

The evaluation of Eq. (2) depends on how HARP works. The correct evaluation is to expand all terms (6 singles, 15 pairs, 20 triplets, 15 quadruplets, 6 quintuplets and 1 sixtuplet) taking care to reduce terms such as $P(x_2'y1'x_2'y_2')$ to $P(x_2'y_1'y_2')$. This yields:

$$R_2 = 1 - q_1^2 - q_2^2 - 4q_1q_2 + 3(q_1^2)(q_2^2) + 6(q_1^2)(q_2) + 6(q_1)(q_2^2) - [3 \text{ at a time}]$$
$$+ [4 \text{ at a time}] - [5 \text{ at a time}] \qquad (4)$$

It is well known that if we drop all the bracketed terms then we bound R_2:

$$R_2 \geq 1 - q_1^2 - q_2^2 - 4q_1q_2 + 3(q_1^2)(q_2^2) + 6(q_1^2)(q_2) + 6(q_1)(q_2^2) \qquad (5)$$

Of course HARP may choose to ignore the intersection reductions such as $P(x_2'y_1'x_2'y_2') = P(x_2'y_1'y_2')$, which would yield $15(q_1^2)(q_2^2)$ instead of the last 3 terms in Eq. 5 (many programs do this). HARP could also ignore anything beyond the single terms, yielding Eq. 5 with the last 3 terms deleted which now changes the inequality from a lower to an upper bound.

Evaluation of Eqs. (3) and (5) yields:

	R and 1-R	R and 1-R
$q_1 = q_2$	Correct Model Eq. (3)	Assumed HARP Model Eq. (5)
.1	0.9799/0.0201	0.9523/0.0477
.01	0.99979999/0.0002111	0.99941203/0.00058797

Based on the above example, the discrepancies in unreliability are quite serious in even this simple case. The percentage errors are 137% and 179%.

C.3 AN APPROXIMATE ANALYTICAL SOLUTION FOR A TRIAD WITH A COLD SPARE

We can develop an approximate model for the probability of failure $P_f = 1 - R$ for the system model given in Fig. 4, a triad with a cold spare. We model the the probability of system failure as a union of the four different modes of system failure, (M_1, M_2, M_3, M_4), which correspond to states 3, 6, 9, 11 in the transition diagram of Fig. 4 as follows:

$$P_f = 1 - R = P(M_1 + M_2 + M_3 + M_4) \tag{6}$$

where

M_1 = One failure and a second failure before recovery is finished = $x'_a.y'_a$

M_2 = Two failures and a third failure before recovery is finished = $x'_b.y'_b$

M_3 = Three failures and a fourth failure before recovery is finished = $x'_c.y'_c$

M_4 = Four failures = x'_d

Substitution into Eq. 6 yields:

$$P_f = P(x'_a.y'_a + x'_b.y'_b + x'_c.y'_c + x'_d) \tag{7}$$

It is clear that the various modes are mutually exclusive (disjoint) since it is impossible for example to have simultaneously one failure and four failures. Thus, we can write Eq. (7) as:

$$P_f = P(x'_a.y'_a) + P(x'_b.y'_b) + P(x'_c.y'_c) + P(x'_d) \tag{8}$$

Evaluation of Eq. (8) involves conditional probabilities, for example:

$$P(x'_b.y'_b) = P\ (\text{Two failures and a third failure before recovery is finished}) \tag{9}$$

and this should be expanded to read:

$P(x'_b.y'_b)$ =
P(One failure) x

P(A second failure|that recovery of the first failure is successful) x

P(A third failure before recovery from the second failure is finished) (10)

If we approximate this expression by assuming independence, we ignore the conditioning, and write Eq. (8) as:

$$P_f = P(x'_a) \times P(y'_a) + P(x'_b) \times P(y'_b) + P(x'_c) \times P(y'_c) + P(x'_d) \tag{11}$$

Given exponential failure and recovery distributions (constant hazards) where λ is the failure rate rate we can write for the case where $\lambda = 10^{-4}$ failures/hr., the system operating time t = 10 hr., and A fixed recovery time $t_r = .36/60x60 = 10^{-4}$ hr., (NOTE: If recovery time is not fixed but has an exponential or normal distribution, then a different computation must be used. See comment 39)!

$P(x'_a) = 1 - e^{-3\lambda t} = 1 - e^{-3 \times 10^{-3}} = 0.002995504$

$P(x'_b) = (1 - e^{-3\lambda t})^2 = (1 - e^{-3 \times 10^{-3}})^2 = (0.002995504)^2$

$P(x'_c) = (1 - e^{-2\lambda t})P(x'_b) = 1 - e^{-2 \times 10^{-3}}P(x'_b) = 0.001998001 \times P(x'_b)$

$P(x'_d) = (1 - e^{-\lambda t})P(x'_c) = 1 - e^{-10^{-3}}P(x'_c) = 0.0009995 \times P(x'_c)$

$P(y'_a) = 1 - e^{-2\lambda t_r} = 1 - e^{-2 \times 10^{-8}} = 0.00000002$
$= 2 \times 10^{-8}$

$P(y'_b) = 1 - e^{-2\lambda t_r} = 1 - e^{-2 \times 10^{-8}} = 0.00000002$
$= 2 \times 10^{-8}$

$P(y'_c) = 1 - e^{-\lambda t_r} = 1 - e^{-10^{-8}} = 0.00000001$
$= 10^{-8}$

Substituting these values into Eq. 11 yields:

$$P_f = .002995504 \times 2 \times 10^{-8} + (.002995504)^2 \times 2 \times 10^{-8} + (.002995504)^2 \times (.001998001) \times 10^{-8} + (.002995504)^2 \times (.001998001) \times (.0009995) \quad (12a)$$

$$P_f = 5.991008 \times 10^{-11} + 1.7946 \times 10^{-13} + 1.8 \times 10^{-16} + 1.7919187 \times 10^{-11} \quad (12b)$$

Thus, from Eq. (12b) we can write:

$P(M1)$	$= 5.9901008 \times 10^{-11}$
$P(M2)$	$= 1.7946 \times 10^{-13}$
$P(M3)$	$= 1.8 \times 10^{-16}$
$P(M1) + P(M2) + P(M3)$	$= 6.008972 \times 10^{-11}$
$P(M4)$	$= 1.7919187 \times 10^{-11}$
$P(M1) + P(M2) + P(M3) + P(M4)$	$= 7.8008907 \times 10^{-11}$

C.4 REVIEW OF PAPER BY WHITE

"A Review of the HARP Program: Approach and Mathematics," Allan L. White, Draft NASA Technical Memorandum, NASA Langley, November 1989.

This paper provides insight into the mathematical background of the HARP program, and is a very good review of the capabilities and limitations of the HARP fault-tolerant computing program. In addition there are valuable discussions of the instantaneous-jump model, which has importance as a modeling technique even when divorced from the HARP program. Other valuable insights into the modeling process for fault-tolerant systems are also included. The following comments are not offered as criticism but as suggestions for further improving the draft of a good report. Allan White briefly responded to some of these comments by E-mail and his note is included in Appendix C.6.

MAJOR COMMENTS

1. Section 2.1. Some technical terms are used in this section without definition. Although many of the readers will know these terms, some will not. Also there are probably different definitions or interpretations of some of these terms, therefore please define: Markov discrete state time model, Markov continuous state time model, Semi-Markov model, and instantaneous-jump model.

2. Figs. 2.1 and 2.2 and analysis in between. I think you need more detail here, this is a very important point. The instantaneous-jump model explains a lot about what is special about fault-tolerant computing systems modeling: Recovery from transient errors, large and small time constants leading to "stiff equations" (if you use this term it must be defined and explained), etc. I have tried to discuss some of these matters in Appendix C.5.

3. Note: In Appendix C.5, I have assumed that the transient mode of failure is due to the fact that " the system can not recover from two soft failures because the recovery unit becomes "confused" if the second soft failure occurs before the recovery process (which operates at a rate δ) has been completed. Is this the correct failure mode, or is it a failure because the system eventually recovers from two transient failures but takes too long to complete recovery and the system is down for an unacceptable time?

4. Section 2.3. Why do you want to convert a standard Markov model to an instantaneous jump model? To avoid the problem of solving stiff differential equations? How about formulating and solving a simple example of two parallel elements with recovery by using a standard Markov model. Give actual parameters for failure rate λ and system recovery rate δ. Show the solution problem with stiff equations. Make a instantaneous-jump model and show how the problem is eased.

5. I have checked the validity of the instantaneous-jump transformation by deriving the partial Markov model with and without the transformation and the results check. See discussion in Appendix C.5.

6. Section 2.3. Define the following terms in your discussion:

    a. Reconfigurable fourplex.

    b. Fault-free state.

    c. Recovery-mode state.

    d. Coincident-fault failure state.

    e. Exhaustion-of-parts failure state.

7. Two sentences below Fig. 2.4(a). Which differential equation package?

8. The analytical expressions derived in Appendix C were evaluated in a small BASIC program (see Table 1) for the numerical values given below Fig. 2.4. The results are given in Table 2. Note that they compare well with the values given in the report, however, the instantaneous-jump model gave identical results since the difference term, X4 in the program, was or the order exp(-1000) which is so small it is given as zero. Note the smallest exponential value which my scientific calculator will compute is exp(-227) = 2.60 x 10^{-99}. Either there is a discrepancy in the models derived in Appendix C, or there is computational error in the differential equation package.

9. Top of page above Sec. 2.4. If the computations in comment 8 above are correct, perhaps the word over estimation should be changed. Definitions of 4-plex (or fourplex, both are used?), and N-plex should be given earlier in this section.

10. Section 2.4, 10 lines from bottom: "The probability of system recovery is V, ..." This sentence is not clear. Is this the transition rate from state R to S shown in Fig. 2.5(a)?? Define words and notation carefully. (A real definition of V doesn't occur until one gets to Fig. 2.8. Some of this explanation is needed here as well).

11. Section 2.4, next to last paragraph. Define what you mean by an active and a benign fault? Not clear. Define all the transition parameters and explain.

12. Section 2.4, last paragraph. Define what you mean by a semi-Markov recovery model. Not clear. Define all the transition parameters and explain.

13. Figure 2.5. I think all these models must be discussed in much more detail. I would suggest derivations of each one using "before-and-after" Markov models similar to what is done in Appendix C. I think the result may be a more general statement about instantaneous jump models. I wonder if the people who deal with still differential equations and widely separated time constants have already done such an analysis by focusing on the roots of the differential equations. If so, it would be very helpful to carry this work over into the Markov modeling realm, where one has much more modeling insight.

14. Below Fig. 2.5 and Fig. 2.6. Needs more explanation how models in Fig. 2.5 are combined to yield Fig. 2.6.

15. Last sentence in Sec. 2.5. Is there some sort of a "Y to V" (turn the letters on their sides) transform going on here?

16. Section 2.6. Why do you use the term coverage here? Explain how its conventional meaning is related to your usage?

17. Section 2.6. Where is Fig. 5?

18. Section 2.6. When do we need the additional complexity of a Weibull model? If we use the correct values of failure and recovery rates, the steady state values of Markov model probabilities should be correct. Or is this not the case with fault-tolerant system models?

19. Section 2.6. More explanation of FORM needed.

20. Fig. 2.7. Definition of E. When you first discussed exhaustion-of-parts, did you say that you are assuming that the system fails when there is only one good system left because you don't know which of the two remaining systems is the good one and which is the bad one? This is only true if no human is in control, since there is a good chance that we could switch between the two remaining systems and determine which performs properly. This would be possible on a long term space flight, but not of course for a dynamically unstable aircraft.

21. Section 2.8. "The HARP construction does not always reproduce the intended reliability model." Is the problem with the instantaneous-jump theory, or with the way HARP implements it? Where is Fig. 8?

22. Fig. 2.9. It would be clearer if you labeled the figure "incorrect models produced by errors in the HARP look ahead procedure" and gave another figure labeled "correct models for the examples of Fig. 2.9. Doesn't ARIES avoid this problem by asking the user to tell the program how many active elements there are after a failure and reconfiguration?

23. Section 2.9. If you are willing to give the analyst the task of reducing the model by using an instantaneous-jump transformation you avoid some of the problems caused by asking the program to make all the decisions. Maybe what is needed is a modeling phase where the problem is "prepared" for solution?

24. Above Fig. 2.9. Define what is meant by a double dual.

25. Section 3.1. The comment "HARP is essentially a CARE IV." is significant. It belongs here but should also appear somewhere in the introduction as well.

26. Section 3.1. You are considering the occurrence of overlapping faults, a second fault occurring before the system has finished reconfiguring in response to a first fault. Is this an important point?
a. The probability of two such faults is low.
b. The system may be unable to handle two overlapping faults and system failure may be the correct model.
c. Can HARP artificially terminate the first fault and work on the second?
Please comment on these three points.

27. Section 3.3. "Unfortunately, there are no results in this area." Rephrase for clarity.

28. Section 3.3. Define sojourn times. Two lines later insert word and split word? (There ARE counter examples ...).

29. Section 3.4. Define the following terms: plurality voting fourplex, intermittent faults with an active failure condition, triad plus spare, majority voting sixplex (is this another term for 6-modular redundancy?), two triads.

30. I think a carefully written and comprehensive glossary of terms would be helpful in defining the many terms which need definition. In some cases it would be helpful to add additional definitions and discussions in the text even if a glossary is included.

31. Fig. 3.2. It would be helpful to add a legend to this figure (and possibly to other figures). For example:

LEGEND

S_1 = Initial system state all four elements are good
R_1 = One failure has occurred and recovery is in progress
S_2 = etc.
etc.

32. Above Fig. 3.2. Let us simplify and assume that each component has a single bit output which is either 0 or 1. Let us define the following "redundancy management rules:
a. If a failure occurs and one component disagrees with the other 3, (3-1), it is voted out of the system (essentially its output is disconnected,
and the remaining three elements continue, forming a TMR (triple-modular or 3-modular redundancy system, isn't this what you also call a triad?).

b. If a second failure occurs after the system has reconfigured to form a TMR system, (2-1) the disagreeing system is not disconnected and the TMR continues until there are two failures and system failure occurs. (Is this what you mean by 2-1-1 ?)
c. If there are initially two simultaneous failures, or two overlapping failures, (a second before the recovery to the first is complete), the system fails. (Is this what you mean by 2-2).
If the first failure is not disconnected, the above redundancy management rules are different. You should clarify.
Note that one can have even more complex redundancy management rules. One can build an adaptive voter which keeps a record of how many times each component disagrees with the majority. One the component exceeds a predecided error level, he is disconnected from the system. Records of disagreement are still kept, and there is a possibility that the disconnected component may score better in the future and be rehabilitated. Can HARP model such a system?

33. Fig. 3.3 does not seem complete. The caption is incomplete. Out of the 7 terminal states (right most states), 4 are unlabeled. The transition rate out of S_2 is not labeled.

34. Fig. 3.4. What is an unmonitored vs. a monitored spare?

35. Fig. 3.5 and the preceding text. Define a sixplex and explain the model.

36. Fig. 3.6 and the preceding text. Define a two triads system and explain the model.

37. Section 3.5. Define what is meant by fault containment regions.

38. Section 3.6. Define what is meant by adjusting the critical pair list.

39. Fig. 3.7 and above. A system consists of ten dual processors. What does this mean? A series model (chain model) of ten sub systems, each of which is a dual processor? Explain the system with a reliability block diagram or a reliability graph.

40. Fig. 3.7 and above. Are the transition rates of 10^{-4} failures per hour (about one per year) and 10^{+4} recoveries per hour (about three per second) typical rates?

41. Fig. 3.8 and above. You should give reliability graph and system redundancy management rules to define this system.

42. Fig. 3.9 and above. You should give reliability graph and system redundancy management rules to define this system, and explain how it differs from Fig. 3.8.

43. Section 3.7. Give a reference to SIFT, eg. Siewiorek & Swarz?

44. Section 3.7. Change of wording from "HARP developers are trying to implement a cold-spare gate [1,2]." to "HARP developers have developed a new gate called a cold-spare gate which should allow one to model unpowered system spares [1,2]. Unfortunately there are presently problems in the implementation of this new gate [1] "

45. Below Fig. 3.12. If the results in the table have been checked and are correct, this is not a very encouraging example of HARP's ability to deal with a practical problem. The best agreement is three orders of magnitude to large, and the worst six orders of magnitude!

46. Section 3.8. You should give reliability graph and system redundancy management rules to define this system.

47. Section 3.8. Why were the results so good for this example but so poor for SIFT?

48. Section 3.10. Even if HARP has unacceptable limitations, isn't the instantaneous-jump transformation (simplification) rule which was previously developed a useful technique

for use by a reliability analyst in simplifying a model or as an algorithm to be used in some future fault-tolerant program?

49. Above Fig. 4.1. Define terms first coincident-fault failure, second coincident- fault failure, etc.

50. Section 4.2 and Fig. 4.1. Won't you get the same result if you argue that in the Markov model the state with the highest probability is S, the next highest probabilities are R_1 .... R_n, lower still are C_1 .... C_n, etc. If λT is small than the probabilities decrease rapidly as we go to the right in the Markov model, and states C_1 .... C_n, and higher order states can be truncated with little loss in computational accuracy.

51. Above Section 4.4. The examples to be discussed have 25,000 and 100,000 states. It seems to me that common sense, without any mathematics, would argue that it is impossible to determine and input failure rates for any model so big unless it was some sort of a repetitive or periodic structure, in which case states could be merged (aggregated).

52. Above Section 4.4. How did you determine that the truncated model gave only a error of .003. By running both the full model and the truncated model and comparing? Explain.

53. Last sentence above Section 4.4. "use a λT of value 10" Do you mean $10^{1 \text{ or } 2}$?? How did you determine this? by running the same example for smaller values of λT??

54. Section 4.4. "elementary methods would produce an accurate estimate of reliability." What elementary methods? A Markov model solving package? (which one?) A program such as Mathematica, Matlab, or Macsyma? A simple fault-tolerant solving program?

55. Section 5.2. Define what is meant by the term maximum-coverage-leakage.

56. Section 5.3. The definition of critical pair N-plex belongs in a Glossary of Terms and also in some of the early introductory material.

57. Section 5.3. Define what is meant by the term higher level voting.

58. Section 6. Some experts on control theory should be asked to comment on the quoted result from control theory which is used in the HARP bounds proof.

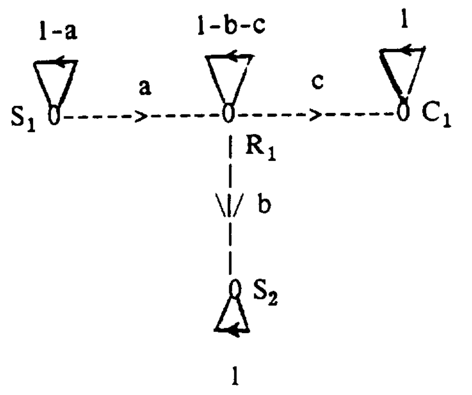
MINOR COMMENTS

1. Pages need numbering.
2. Three lines below Fig. 2.1. Should the phrase read: .... from state S to state I followed by ..." ? [Interchange states S and I ?].
3. There seems to be a typo in Fig. 2.4(a), the transition between states R_1 and S_2 should be $\delta/(3\lambda + \delta)$
4. Typo on line 8 of Section 2.8... the the .....
5. Above Fig. 3.4, second line. Word has should be as??
6. Below Fig. 3.9. "As shown in section 1.8" Where is section 1.8?
7. Section 3.8, lines 4 & 5. "Sections *****"??
8. Reference No. 1, "Langley TM-******" ??.
9. Also this report needs a formal cover page and TM number.

C. 5 MODELING FAULT-TOLERANT SYSTEMS WITH AN INSTANTANEOUS-JUMP MODEL

One can show how the instantaneous-jump model works by making a Markov model for the first 4 states S_1, R_1, S_2, C_1 of Fig. 2.3 and another Markov model for the first 3 states in Fig. 2.4(a), S_1, S_2, C_1, and comparing the results.

The first Markov model is shown in Fig. 1 below:



a = 4λ
b = δ
c = 3λ

Fig. 1 Markov model for first 4 states of Fig. 2.3

The differential equations for the Markov model of Fig. 1 is given below:

$$\dot{P}_{S_1} = - 4\lambda P_{S_1} \tag{1}$$

$$\dot{P}_{R_1} = 4\lambda P_{S_1} - (3\lambda + \delta)\, P_{R_1} \tag{2}$$

$$\dot{P}_{C_1} = 3\lambda\, P_{C_1} \tag{3}$$

$$\dot{P}_{S_2} = \delta\, P_{C_1} \tag{4}$$

$$P_{S_1}(0) = 1;\ P_{R_1}(0) = P_{C_1}(0) = P_{S_2}(0) = 0 \tag{5}$$

A convenient way to solve equations (1) - (5) is to use Laplace transforms. Where we use the notation that the Laplace transform of f(t) is denoted by F(s) or $\{f(t)\}^*$. The Laplace transforms of Eqs. (1) - (5) become:

$$\{P_{S_1}\}^* = 1/(s + 4) \tag{6}$$

$$\{P_{R_1}\}^* = 4\lambda/(s + 4\lambda)(s + 3\lambda + \delta) \tag{7}$$

$$\{P_{C_1}\}^* = 12\lambda^2/s(s + 4\lambda)(s + 3\lambda + \delta) \tag{8}$$

$$\{P_{S_2}\}^* = 4\lambda\delta/s(s + 4\lambda)(s + 3\lambda + \delta) \tag{9}$$

The simplest approach to solving the above transform equations is to expand the right hand sides in partial fractions and identify each corresponding constant and exponential term. The results are:

$$P_{S_1}(t) = \exp(-4\lambda t) \tag{10}$$

$$P_{R_1}(t) = [4\lambda/(\delta - \lambda)]\exp(-4\lambda t) - [4\lambda/(\delta - \lambda)]\exp[-(3\lambda + \delta)]t \tag{11}$$

$$P_{C_1}(t) = [3\lambda/(3\lambda + \delta)] - [3\lambda/(\delta - \lambda)]\exp(-4\lambda t) + [12\lambda^2/(3\lambda + \delta)(\delta - \lambda)\exp[-(3\lambda + \delta)]t \tag{12}$$

$$P_{S_2}(t) = [\delta/(3\lambda + \delta)] - [\delta/(\delta - \lambda)]\exp(-4\lambda t) + [4\lambda\delta/(3\lambda + \delta)(\delta - \lambda)\exp[-(3\lambda + \delta)]t \tag{13}$$

One can check the validity of these results by summing the left and right hand sides of Eqs. (10) - (13). By definition, the sum of the 4 probabilities should be one, and the right hand sides of the four equations sum to unity.

The second Markov model is shown in Fig. 2:

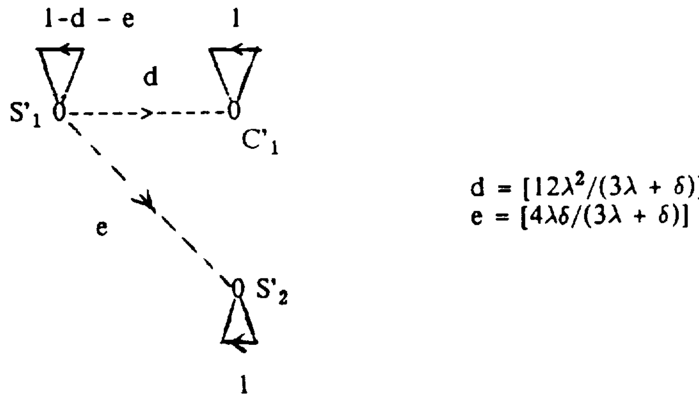


Fig. 2 Markov model for first 3 states of Fig. 2.4 (b)

Our objective is to derive the probabilities S'_1, C'_1, S'_2 and show under what conditions these values check with those of S_1, C_1, S_2. From the model of Fig. 2, the equations become:

$$\dot{P}_{S'_1} = -[(12\lambda^2 + 4\lambda\delta)/(3\lambda + \delta)]P_{S'_1} = -\,4\lambda P_{S'_1} \tag{14}$$

$$\dot{P}_{C'_1} = [(12\lambda^2)/(3\lambda + \delta)]\ P_{C'_1} \tag{15}$$

$$\dot{P}_{S'_2} = [4\lambda\delta/(3\lambda + \delta)]P_{C'_1} \tag{16}$$

$$P_{S'_1}(0) = 1\ P_{C'_1}(0) = P_{S'_2}(0) = 0 \tag{17}$$

Taking Laplace transforms yields

$$\{P_{S'_1}\}^* = 1/(s + 4\lambda) \tag{18}$$

$$\{P_{C'_1}\}^* = [12\lambda^2/(3\lambda+\delta)]/s(s + 4\lambda) \qquad (19)$$

$$\{P_{S'_2}\}^* = [4\lambda\delta/(3\lambda+\delta)]/s(s + 4\lambda) \qquad (20)$$

As previously, we expand the right hand sides of the above equations in partial fractions and identify each corresponding constant and exponential term. The results are:

$$P_{S'_1}(t) = \exp(-4\lambda t) \qquad (21)$$

$$P_{C'_1}(t) = [3\lambda/(3\lambda + \delta)] - [3\lambda/(3\lambda + \delta)]\exp(-4\lambda t) \qquad (22)$$

$$P_{S'_2}(t) = [\delta/(3\lambda + \delta)] - [\delta/(3\lambda + \delta)]\exp(-4\lambda t) \qquad (23)$$

Again, the validity check, summing the left and right hand sides of the above equations works.

Comparing the results, we see that the two models give equivalent results if the value of $\delta >> 4\lambda$, since in this case the transient terms associated with the δ time constants die out almost instantly and leave the remainder of the solution.

TABLE 1

BASIC Program to Evaluate Equations (12) and (22)

```
LIST
10 REM Program to check White report below Fig. 2.4
20 REM Abreviations for greek letters: lambda = LL, delta = DD
30 REM EQUATION IS IN NOTES
100 INPUT "Values of LL,DD,T";LL,DD,T
110 A = (3*LL)/(3*LL + DD)
120 B = ((3*LL)/(DD - LL))*(EXP(-4*LL*T))
122 X1 = 3*LL
124 X2 = (3*LL - DD)
126 X3 = (LL - DD)
128 X4 = EXP(-(3*LL + DD)*T)
130  C = ((X1)/(X2*X3))*(X4)
140 PRINT "A=";A, "B=";B, "C=";C
142 PRINT "X1,X2,X3,X4=";X1;X2;X3;X4
150 PRINT "PC=", (A - B -C)
160 PRINT "PC1 = ", (A - B)
200 END
Ok
```

TABLE 2

Results of Running BASIC Program Given in Table 1

```
RUN


Values of LL,DD,T? 5E-4,1000,1
A= 1.499998E-06              B= 1.497004E-06            C= 0
X1,X2,X3,X4= .0015 -999.9985 -999.9995  0
PC=            2.99417E-09
PC1 =          2.99417E-09
Ok
```

C.6 COMMENTS BY WHITE

Allen White responded briefly by E-mail to the comments by Shooman in Appendices C.4 and C.5. His comments are included here because they illustrate some of the intricacies of the fault-tolerant modeling art and should be of interest to the analyst.

You recently read my "Review of the HARP Program: Approach and Mathematics" (White) and asked for a reply to some of your comments and questions. I am trying a quick reply to some of the comments.

For comment #8 on the accuracy of the instantaneous jump approximation, I get the same equations that you present in Appendix C, but I think there is a typo in line 160 of the BASIC program.

For comment #26 on the importance of "overlapping faults". Depending on the system and its parameters, this category of faults can range from an insignificant contribution to system failure to the dominant cause of system failure. Computing this probability of "overlapping faults" is the major concern of several reliability estimation packages produced by NASA Langley.

For comments #47 and #48 on the usefulness of some of the HARP techniques and their domain of application. Essentially, we already know this. The basic theory for the conservativeness of instantaneous jumps, error bounds for instantaneous jumps, and what a FEHM needs is contained in

> A. L. White, Reliability Estimation for Reconfigurable Systems with Fast Recovery. Journal Microelectronics and Reliability, Volume 26, No. 6, pp.1111-1120, 1986.

We will attempt to publish extensions of this theory to address some of the questions and issues raised by the HARP developers.

For comment #50 which gives a plausibility argument. It can be seen that, in general, systems are capable of devious behavior. Try a system that has the reconfiguration sequence

majority-voting five-plex
majority-voting four-plex
majority-voting three-plex
simplex.

For most choices of parameters, the second coincident-fault failure state (for the four-plex) will dominate the first coincident-fault failure state (for the five-plex) even though the second one occurs later in the model. The reason is that the first coincident-fault failure needs three faults present in the system while the second coincident-fault failure only needs two faults present in the system.

For comment #51. I agree with this observation that the systems must have a lot of symmetric structure or they will be impossible to handle. A local model generation program (ASSIST) is designed to take advantage of the symmetry that is found in most large systems.

Comment #52 on the error bound of .003. This error bound is obtained from the theorem on the dominance of the first coverage failure (compared to other coverage failures) for a certain class of systems. This theorem along with another theorem (which I hope to publish soon) show that reliability estimation is easy for a popular class of systems.

Sincerely,
Allan White

APPENDIX D

SIGNIFICANCE OF MODELING ERRORS

APPENDIX D
SIGNIFICANCE OF MODELING ERRORS

Various factors which affect the correctness of HARP model solutions (and other modeling programs) were discussed in Sec. 2.3. In this appendix we treat the problem of modeling errors - "sins of omission and commission." By omission we mean that the model for some circuit or system neglects one or more factors which are significant and result in a modeling error. Errors of commission mean that an effect is identified as a factor in the model, however, it is included in an incorrect manner which leads to a modeling error. Either of these errors may lead to an optimistic or pessimistic reliability or availability prediction. Furthermore, the size of such an error may be larger then any errors in the program algorithm, the code implementation, or round-off errors in computation. Such modeling errors become "common mode" errors in that they will be present regardless of the computational program which is to be employed.

As an illustration of an omission modeling error, consider the reliability analysis of a parity bit coding circuit. (See Shooman 1990, pp. 566-569). This is a simple computation which we can do analytically and obtain a closed form result. Such a checking circuit is used in many fault-tolerant applications such as reading and writing from computer memory, transferring data to and from a computer bus, and transmission of data between computers over telephone lines. We will model the later application and consider it typical of the class of application.

Let us consider the addition of a ninth parity bit to an 8 bit message byte. The parity bit adjusts the number of 1's in the word to an even (odd) number and is computed by a parity bit generator circuit which calculates the exclusive or function of the 8 message bits. Similarly, an exclusive or detecting circuit is used to check for errors. If there are one, three, five, seven, or nine errors in the received word, the parity is violated, and the checking circuit will detect and error. This can lead to several consequences such as "flagging" the error byte, retransmission of the byte until no errors are detected, etc. The probability of interest is the probability of an undetected error, P_{ue}, which is the probability of two, four, six, or eight errors. These can be simply calculated using the binomial distribution and if we let q = the probability of an error per transmitted bit then we obtain:

General $\quad B(r{:}9,q) = \binom{9}{r} q^r(1-q)^{9-r}$ (D-1)

Two Errors $\quad B(2{:}9,q) = \binom{9}{2} q^2(1-q)^{9-2}$ (D-2)

Four Errors $\quad B(4{:}9,q) = \binom{9}{4} q^4(1-q)^{9-4}$ (D-3)

etc.

For q relatively small (10^{-4}), it is easy to see that Eq. D-3 is much smaller than Eq. D-2, thus only Eq. D-2 need be considered and the probability of an undetected error with parity bit coding becomes

$$P_{ue}' = B(2{:}9,q) = 36q^2(1-q)^7 \quad \text{(D-4)}$$

We wish to compare this with the probability of an undetected error for an 8 bit transmission without any checking which is given by

$$1-P\text{ (zero errors)} = 1-B(0{:}8,q) = 1 - \binom{8}{0} q^0(1-q)^{8-0}$$

$$= P_{ue} = 1 - (1-q)^8 \quad \text{(D-5)}$$

The ratio of Eqs. D-5 and D-4 yields the improvement ratio due to the parity bit coding

$$P_{ue}/P_{ue}' = [1 - (1-q)^8]/[36q^2(1-q)^7] \quad \text{(D-6)}$$

For small q we can approximate Eq. D-6 by replacing $(1-q)^n$ by $1-nq$ and $[1/(1-q)]$ by $1 + q$, which yields

$$P_{ue}/P_{ue}' = [2(1 + 7q)/9q] \quad \text{(D-7)}$$

The parameter q, the probability of failure per bit transmitted, is quoted as 10^{-4} in Hill and Peterson 1981, was 10^{-5} or 10^{-6} in the 1960's and 70's and now may be as low as 10^{-7} for the best lines. Equation D-7 is evaluated for this range of q values and the results appear in Table D-1 and in Fig. D-1.

Table D-1 Evaluation of Eq. D-6

q	P_{ue}/P_{ue}'
10^{-4}	2.223×10^{3}
10^{-5}	2.222×10^{4}
10^{-6}	2.222×10^{5}
10^{-7}	2.222×10^{6}
10^{-8}	2.222×10^{7}

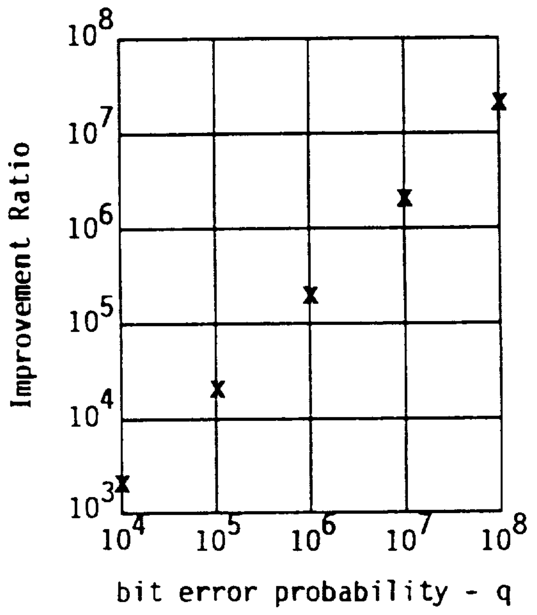


Fig. D-1 Improvement Ratio of Undetected Error Probability Due to Parity Bit Coding

In the above analysis we have assumed that the coder and decoder are perfect. We now examine the validity of that assumption by modeling the reliability of the coder and decoder. Assume that we are using a commercial device, the SN74180, a 9-bit odd/even parity generator/checker, (See Texas Instruments 1988). We will use two such devices since the same chip can be used as a coder and a decoder (generator/checker). The logic diagram of this device is shown in Fig. D-2.

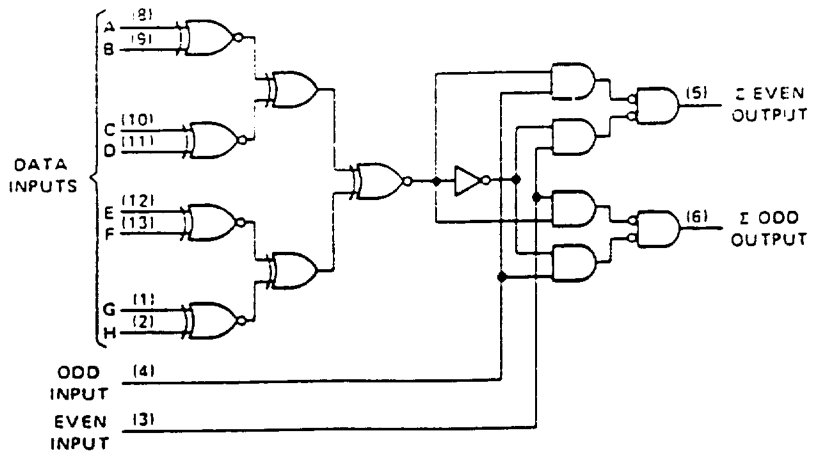


Fig. D-2 Logic Diagram for SN74180 (Texas Instruments, 1988)

A simple model for IC reliability is given in Shooman (1990), p. 641-644. The model gives the failure rate per million hours as $\lambda_b = C(g)^{1/2}$, where C was evaluated as 0.004 for 1985 IC failure rate data. In Fig. D-2, there are four AND, one NOT and two NOR (note $x'y' = (x + y)'$) gates. Since the seven EXOR gates in Fig. D-2 use about 1.5 times as many transistors to realize their function as a simple gate, we will consider them as equivalent to 10.5 gates. Thus we have 17.5 equivalent gates and $\lambda_b = 0.004(17.5)^{1/2}$ failures per million hours = 1.67×10^{-8} failures per hour. Since we will be using one chip for coding and one for decoding, we must double this rate, yielding 3.35×10^{-8} failures per hour. Our previous computation was in terms of failures per bit and this is in terms of failures per hour, thus, we will convert all terms to failures per hour. The baud rate, B, of a transmission is essentially the number of bits per second, and there are 3600 seconds per hour. An additional parameter is needed, namely D the duty cycle, which is the fraction of time there are bit transmissions. The result is that we must multiply probability of failure per bit by 3600BD to obtain failure rate per hour. For infrequent events, the failure rate per hour multiplied by t = 1 hour gives the probability of failure. Thus we obtain from Eqs. D-6

$$P_{ue}/P_{ue}' = \{[1 - (1-q)^8] \times 3600BD + [3.35 \times 10^{-8}]\} \Big/ \{[36q^2(1-q)^7] \times 3600BD + [3.35 \times 10^{-8}]\} \quad \text{(D-8)}$$

simplifying,

$$P_{ue}/P_{ue}' = \{[2.88 \times 10^4 BDq] + [3.35 \times 10^{-8}]\} \Big/ \{[1.296 \times 10^5 BDq^2] + [3.35 \times 10^{-8}]\} \quad \text{(D-9)}$$

Equation D-8 is evaluated for some typical numerical values in Table D-2. Note that at the higher values of q, there is a significant difference between the values of the improvement ratio and the results given in Table D-2 for perfect coders and decoders. Of course, the effects of coder/decoder reliability will become negligible for high baud rates and duty cycles closer to unity. The results given in Tables D-1 and D-2 are compared in Fig. D-2.

Table D-2 Evaluation of Eq. D-8

q	P_{ue}/P_{ue}' B= 1200 D = 0.1	P_{ue}/P_{ue}' B= 1200 D= 0.01
10^{-4}	2.223×10^3	2.223×10^3
10^{-5}	2.222×10^4	2.222×10^4
10^{-6}	2.217×10^5	2.175×10^5
10^{-7}	1.828×10^6	7.046×10^5
10^{-8}	9.859×10^5	1.027×10^5

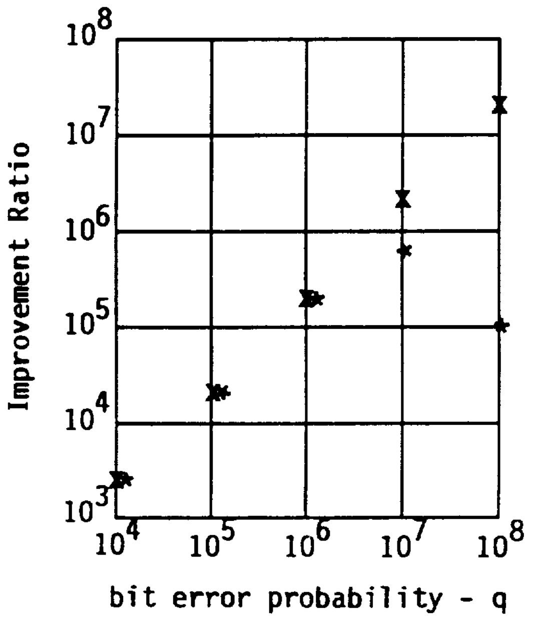


Fig. D-2 Improvement Ratio of Undetected Error Probability Due to Parity Bit Coding

x = Assumes perfect coder/decoder, Eq. D-6

* = Assumes the coder/decoder can fail, Eq. D-8 (Baud Rate = 1200, Duty Cycle = 0.01)

Clearly, for certain values of the parameters, the effect of neglecting the coder/decoder reliability is much greater than the differences one might encounter due to errors caused by the modeling program itself.

Consider a second example. Suppose that we have a system of two elements in parallel with a third element as a spare. Assume no repair and that the failure of either of the first two elements is immediately sensed and the spare element is switched in. The question now arises whether the spare element is a "hot" or "cold" standby, i.e.. whether it is powered up in standby and can fail or whether it is not powered while a standby and can not fail until it is powered up to replace one of the original two units. We will solve the resulting Markov models for both situations and compare the results.

The Markov model for two parallel elements with a hot spare is shown in Fig. D-3, and that for a two elements plus a cold spare is given in Fig. D-4.

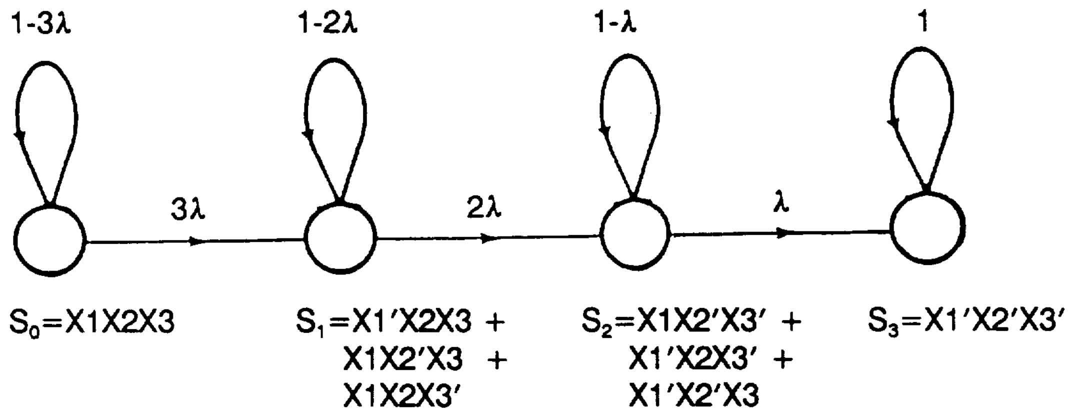

1-3λ
1-2λ
1-λ
1
3λ
2λ
λ
S₀=X1X2X3
S₁=X1'X2X3 + X1X2'X3 + X1X2X3'
S₂=X1X2'X3' + X1'X2X3' + X1'X2'X3
S₃=X1'X2'X3'

Fig. D-3 Markov Model for Two Elements With a Hot Spare

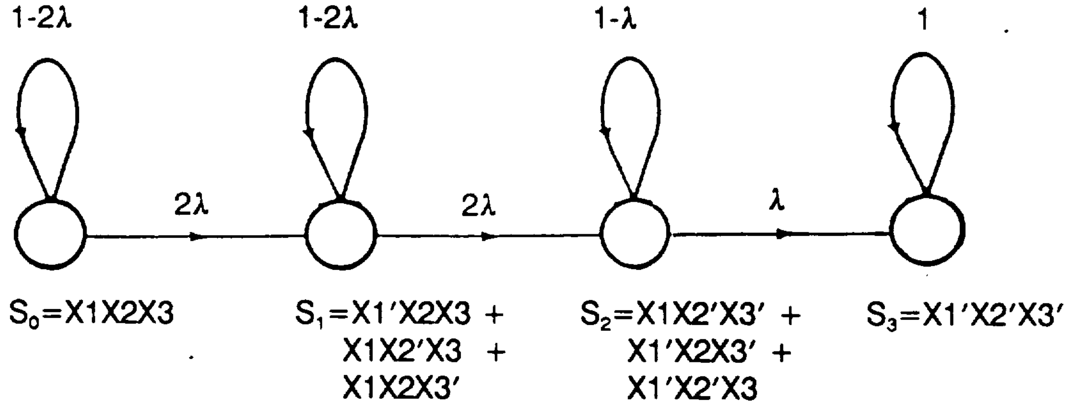

1-2λ
1-2λ
1-λ
1
2λ
2λ
λ
S₀=X1X2X3
S₁=X1'X2X3 + X1X2'X3 + X1X2X3'
S₂=X1X2'X3' + X1'X2X3' + X1'X2'X3
S₃=X1'X2'X3'

Fig. D-4 Markov Model for Two Elements With a Cold Spare

Since there is no repair coupling the states, the state probabilities can be solved independently in order from left to right in the diagram. The equations for the first two states, which we will need to compute the availability, are given below for each model:

Hot Spares

$$\dot{P}_{S_0} = -3\lambda P_{S_0} \tag{D-10}$$

$$\dot{P}_{S_1} = 3\lambda P_{S_0} + 2\lambda P_{S_1} \tag{D-11}$$

Cold Spares

$$\dot{P}_{S_0} = -2\lambda P_{S_0} \tag{D-12}$$

$$\dot{P}_{S_1} = 2\lambda P_{S_0} + 2\lambda P_{S_1} \tag{D-13}$$

$$\dot{P}_{S_1} = 2\lambda P_{S_0} + 2\lambda P_{S_1} \tag{D-13}$$

If we assume that the system is placed in use at t=0 with all 3 items working, then the initial conditions are $P_{S_0} = 1$, $P_{S_1} = 0$. Using Laplace transforms or conventional differential equation theory to solve Eqs. D-10 to D-13 yields for the state probabilities and the availability, A(t):

Hot Spares

$$P_{S_0} = e^{-3\lambda t} \tag{D-14}$$

$$P_{S_1} = 3(e^{-2\lambda t} - e^{-3\lambda t}) \tag{D-15}$$

$$Ah(t) = P_{S_0} + P_{S_1} = 3e^{-2\lambda t} - 2e^{-3\lambda t}) \tag{D-16}$$

Cold Spares

$$P_{S_0} = e^{-2\lambda t} \tag{D-17}$$

$$P_{S_1} = (1 + 2\lambda t)e^{-2\lambda t}) \tag{D-18}$$

$$Ac(t) = P_{S_0} + P_{S_1} = (3 - 2e^{-\lambda t})e^{-2\lambda t}) \tag{D-19}$$

In Table D-3 we compare Eqs. D-16 and D-17 for a few values of λt.

λt	Ac	Ah	Ac/Ah
0	1	1	1
0.5	0.736	0.657	1.12
1	0.404	0.306	1.32
2	0.091	0.05	1.83

As we can see by the ratio of the two availabilities there are significant differences in these two solutions. Thus, a mistake in the modeling phase which confuses one model with another will make a considerable difference which may be much greater than the differences one might encounter due to errors caused by the modeling program itself.

APPENDIX E

PUBLISHED PAPERS SUPPORTED BY THE GRANT

Route 110. Farmingdale, New York 11735 516-755-4400 / FAX5167554404

SCHOOL OF
ELECTRICAL ENGINEERING
AND COMPUTER SCIENCE

Polytechnic
UNIVERSITY

RELIABILITY MODELS OF FAULT-TOLERANT SYSTEMS FOR CONTROL OF ELECTRIC POWER GENERATION

by

Martin L. Shooman

Presented at

INTERNATIONAL MEETING

on

THE SAFETY AND RELIABILITY OF ENERGY SYSTEMS

July 2-4, 1990

Sopron, Hungary

New York City:
333 Jay Street
Brooklyn, NY 11201
718-260-3600
FAX 7182603136

Long Island:
Route 110
Farmingdale, NY 11735
516-755-4400
FAX 5167554404

Westchester:
36 Saw Mill River Road
Hawthorne, NY 10532
914-347-6940
FAX 9143476939

1.0 INTRODUCTION

The control and safety systems for power generating stations have traditionally been designed using relay logic and analog electronic components. With the advent of the digital electronics revolution, we are now able to build controllers with great power and flexibility of modest cost out of a handful of microprocessor and memory chips. A host of new and challenging problems accompany the introduction of digital controllers for power generating stations [Lauber, 1980]

1.1 Digital Controllers

Because of the stringent safety and reliability requirements digital controllers use several levels of redundancy (as do the earlier analog systems). However, the flexibility and power of digital computation give us several ways in which to realize redundancy. The term fault-tolerant computing is generally used to describe the various techniques which are employed to implement redundancy in digital systems.

1.2 Fault-Tolerant Systems

The term fault-tolerant systems was probably coined in the 1960's in conjunction with long duration space missions, where some redundant elements could fail (develop faults) but the other elements would take over and the overall space craft would be tolerant to (some number of) faults. Today the term is used to describe a wide variety of techniques:

1. Coding techniques involving extra check bits (redundant) to detect and correct transmission errors on busses, lines,memory units, etc.
2. Majority voting schemes (N-modular redundancy), compare the outputs of redundant units.
3. Parallel redundancy (or majority voting) combined with spares and the possibility of repair/replacement.
4. Recovery from transient hardware or software faults.

This paper will focus on the use of majority voting to enhance the reliability of the control system for electric power generation. The models of the hardware are well known [Siewiorek 1982 and Shooman 1990], however, we must also consider how to model the reliability of the software in a majority voting configuration, [Shooman 1983]. This paper will focus on the modeling of the software reliability in a majority voting scheme.

1.3 N-Modular Redundant Hardware

The simplest voting scheme, n-modular redundancy scheme, is where n = 3 and is called triple modular redundancy or simply, TMR. One of the advantages of TRM is that no special technique is required to implement redundancy. The logic of the voter is simple and easy to build, and a block diagram of a TMR configuration is shown in Fig. 1.

The system succeeds if two or more of the circuits succeed. If we assume that the voter does not fail, then the reliability is given by

$$R = P(AB + AC + BC) \qquad (1)$$

If all the digital circuits are independent and identical with probability of success p, then this equation can be rewritten in terms of the binomial theorem, where B(3:3) and B(2:3) mean that 3 out of 3 and 2 out of 3 circuits succeed.

$$\begin{aligned} R &= B(3:3) + B(2:3) \\ &= \binom{3}{3} p^3(1-p)^0 + \binom{3}{2} p^2(1-p)^1 \qquad (2) \end{aligned}$$

p.1.

C-2

$$=3p^2 - 2p^3 = p^2(3-2p) \tag{3}$$

This is, of course, the reliability expression for a two-out-of-three system.

If a single circuit has a reliability $p=0.95$, then substitution in Eq. (3) shows that R for the TMR system increases to 0.993, a 700% reduction in the probability of failure.

Of course in deriving Eq. (3) we have assumed that the voter does not fail. If we include the reliability of the voter, p_v, then Eq. (3) becomes

$$R = p_v p^2(3-2p) \tag{4}$$

Clearly, the voter must have a high reliability, otherwise it can negate most or all of the gains due to the TMR scheme. Using the numerical values given above, if the voter reliability is 0.98, then Eq. (4) yields a system reliability of 0.973 and the probability of failure improvement ratio drops from a factor of 7 to a factor of 1.86. Clearly, these results which were derived for TMR in Eqs. (1-4) extend to N-Modular redundancy. Because of cost, TMR is the most popular version. In more advanced implementations one can vote at a lower (subcircuit) level, incorporate redundant voters by using a voter at the output of each subcircuit, and achieve a higher reliability at the cost of increased hardware. [Siewiorek 1982 and Shooman 1990].

2.0 SOFTWARE RELIABILITY

The analysis of the TMR system in the preceding section excluded the effects of software failure. There are always some residual errors in software which are not found because exhaustive testing is impossible. As the software is subject to a wide variety of inputs during operation, some of these errors are excited, and the system fails. The rate at which such software caused errors occur is called the software failure rate. Various software reliability models have appeared in the literature, [Shooman 1983, and Musa 1987], which lead to a model for the reliability of the software, p_{sw}.

If the system can fail due to the hardware as well as due to the software, and these failures are independent, then the simplest TMR model which includes the effect of software is to introduce a multiplicative term into Eq. (4).

$$R = p_{sw} p_v p^2(3-2p) \tag{5}$$

Thus, both the voter reliability and the software reliability appear as series terms, and both lower the potential improvement which TMR can provide. If we use the same numerical values as previously, and include a software reliability of 0.99 in Eq. (4), then we obtain a system reliability of 0.963. At this point we might just as well use a single system with a hardware reliability of 0.95 and a software reliability of 0.99 yielding a system reliability of 0.94 since the failure improvement ratio (.06/.037) is only 1.62.

The initial approach to dealing with the software reliability problem is to try and develop the best possible software, and make it nearly error free so that p_{sw} approaches unity and can thus be neglected in Eq. (5). This involves using all the techniques of high quality software development which have evolved over he past 20 years:

- Formal specifications
- Design reviews
- Controlled design
- Top-down
- Two stage

        * rapid prototype

        * final top-down

    - Modular design

    - Structured programming

    - Formal module testing

    - Integration testing

    - Software reliability modeling

    - Simulation testing

Despite such strenuous efforts, the software reliability is seldom so high as to be insignificant, and system designers began to consider he possibility of making the software redundant using the technique of N-Version Programming which is discussed in the next section. (Some authors call this software diversity.)

3.0 RELIABILITY OF N-VERSION SOFTWARE

For many years fault-tolerant designers have thought of using more than one independent version of a program to provide software redundancy, Chen and Avizienis [1978] chose the name "N-Version Programming" to refer to the use of multiple independent versions of software. If we assume that the hardware and software are independent, that we are using TMR, and that we have three independent versions of the software, (even though this is costly), then the system reliability model is as shown in Fig. 2. The reliability expression for Fig. 2 becomes

$$R = p_v(p_h p_{sw})^2(3 - 2p_h p_{sw}) \qquad (6)$$

where p_h = reliability of the hardware

and p_{sw}= reliability of the software.

Using our previous numerical values, $p_h = 0.95$, $p_{sw} = 0.99$, and increasing the voter reliability to 0.99, we obtain after substitution in Eq. (6): $R = 0.98$. This represents an improvement factor of 2.5. Clearly, the voter reliability must be increased, either by designing a high reliability voter or by voting at a lower level. If the voter reliability is made to approach unity, then the last computation predicts a reliability value of $R = 0.989$, and the improvement factor becomes about 5. Thus, the use of TMR hardware and software has been very effective in improving reliability.

There is also another factor which is more difficult to deal with, the effect of common mode software failures.

4.0 EFFECT OF COMMON MODE FAILURES

In the past analysts have found that at times hardware redundancy schemes are compromised by common mode failures. There can be similar problems with software, and in the past n-version programming has been subject to the following common mode failures:

1. Wrong or incomplete requirements.

2. Identical or equivalent misinterpretation of the requirements.

3. Identical or equivalent incorrect treatment of boundary problems.

4. Identical or equivalent incorrect designs for difficult portions of the problem.

This section models and evaluates the effect of software common mode failures.

If we assume that there are two different ways in which common mode dependencies exist, requirements and program, then we can model a TMR system as shown in Fig. 3, where

p_{sw}	= 1 - probability of an independent mode software fault
p_{cmr}	= 1 - probability of a common mode requirements error
p_{cms}	= 1 - probability of a common mode software fault

The reliability expression for the model given in Fig. 3 is given by:

$$R = p_{cmr}p_{cms}[p_{sw}^2(3-2p_{sw})] \tag{7}$$

We can evaluate the effect of software common modes by studying Eq. (7). In the next section, we will substitute numerical values and evaluate the effect of common mode failures. However, inspection of Eq. (7) suggests the calculation of a simple bound on the common mode probabilities. If we observe that the effect of the common mode probabilities is to decrease the reliability, we can bound the common mode probability by setting R = p, the reliability of a single unit.

We set

$$p_c = p_{cmr}p_{cms} \tag{8}$$

and substitute in Eq. (7)

$$R = p_c[p_{sw}^2(3-2p_{sw})]$$

Now we set $R = p_{sw}$

$$p_{sw} = p_c[p_{sw}^2(3-2p_{sw})]$$

$$1 = p_c[3p_{sw}-2p_{sw}^2] \tag{9}$$

$$p_{sw}^2 - \frac{3}{2}p_{sw} + \frac{1}{2p_c} = 0 \tag{10}$$

Solving the above quadratic equation for the larger root yields

$$p_{sw} = \frac{3/2+\sqrt{(9/4)-(4/2p_c)}}{2} = \frac{3}{4} + \frac{1}{2}\sqrt{\frac{9}{4}-\frac{2}{p_c}} \tag{11}$$

The smallest value of p_c which satisfies Eq. (11) is that which drives the radical to zero. The radial cannot be negative since, p_{sw} is a probability and can not have an imaginary part.

$$\frac{9}{4} = \frac{2}{p_c}$$

$$p_c \geq \frac{8}{9} \tag{12}$$

Eq. (12) is a well known result, since the function (form of Eq. (7)) is the same as Eq. (4) and the same minimum holds for the reliability of a voter [see Shooman 1990, Table 6.7].

5.0 NUMERICAL DATA FOR COMMON MODE ERRORS

In order to Evaluate Eq. (6) we must have some typical data values for p_c. Unfortunately, experiments to determine p_c are difficult to set up and costly to run. Two sets of data will be discussed below, that by Chen and Avizienis [1978] and by Knight and Leveson [1986].

The data gathered by Chen and Avizienis [1978] is reported in Pradhan, [1972, p. 665]. Seven versions of an algorithm for solving a partial differential equation for temperature over a two-dimensional region were used. Twelve three-version sets were constructed, and each set was subjected to 32 test cases, yielding a total of 384 tests. The results are shown in Table 1.

Inspection of Table 1 shows that 91% of the test cases resulted in zero or one failure of the three versions of the program, and thus the reliability was given by $R = 0.91$. Assuming no common mode, Eq. (3) holds for the 3-version software, and we can solve the quadratic equation for the value of individual program reliability. Another approach is to solve the equation by simple interpolation as is done in Table 2. The resulting value is $p = 0.815$.

Clearly, using TMR to raise the reliability of a single set of software from a reliability of 0.815 to a system reliability of 0.91 is not a very attractive design. The decrease in $1 - R$ was from .185 to .09, a factor of 2. The problem is that the basic software is not reliable enough. Consider a second set of data discussed below.

Consider a second set of data gathered by Knight and Leveson 1986: 27 different versions of a program were subjected to 200 acceptance tests and once these versions were accepted, each program was run 1 million times and compared with a standard program (the "gold" program) which had received several million tests and had been subjected to extensive structured walkthroughs. The data from the Knight and Leveson experiment is summarized in Tables 3 and 4.

Assume that we will use three independent programs which have the worst reliability, No. 22 in Table 3, where we assume that for each program version

$$p_c p_{sw} = p_{cmr} p_{cms} p_{sw} = 0.990344 \tag{13}$$

If there were no dependency, then $p_c = 1$, and substituting the value of p_{sw} in Eq. (7) would yield a TMR reliability of 0.99972. The improvement in $1 - R$ would be from .0097 to .00028, or a factor of 34.6.

Since we have decided to use the data for Version No. 22 given in Table 3, we now need the data for common mode failures associated with Version No. 22. Unfortunately, this data is not reported in the paper, however, in Table 4 we see the summary data for common failures among all the programs. There were 1255 occurrences of common mode failures among all the different programs in the group of 27 which were each tested 1,000,000 times. Thus, an upper bound on the common mode failure probability can be obtained from the ratio 1255/27,000,000 = 0.000046, and the probability of no common mode failure, $p_c = 0.999954$. Using the value given in Eq. (12), we can solve for $p_{sw} = 0.990344/0.999954 = 0.99039$. Substituting these values in Eq. (7) yields:

$$R = 0.999954[0.99039^2(3 - 2x0.99039)] = 0.99968$$

The value of $1 - R$ becomes 0.00032 and the improvement factor is reduced to 30.2. This is a minor reduction, (also remember that we have used an upper bound on the common mode probability), and the effect of common mode failures in the case is not significant.

We repeat our above calculation for another set of data. Consider case No. 14 of Table 3 where the program reliability is higher.

$$p_c p_{sw} = 0.99862$$

If there were no dependency, $p_c = 1$ and substituting the value of p_{sw} in Eq. (7) would yield a TMR reliability of 0.99999433. The improvement in 1-R would be from 0.001386 to 0.0000057, or a factor of 243. We use the same estimate for common mode probability as previously, p_c = 0.999954. Using the above value for p_c and $p_c p_{sw}$ we can solve for p_{sw}:

$$p_{sw} = 0.99862/0.999954 = 0.99867.$$

Substituting these values into Eq. (7) yields:

$$R = 0.999954[0.99867^2 (3 - 2x0.99867)] = 0.999945$$

The value of 1 - R becomes 0.00005533 and the improvement factor is reduced to 25. This is a significant reduction in this case, however, there is still a large improvement due to TMR.

Clearly, it is important to study further the data from the Knight and Levison experiment, and the Chen and Avizienis data as well as any other available data to better determine the relevant probabilities.

6.0 ACHIEVING INDEPENDENT SOFTWARE VERSIONS

There are impediments to achieving independent software versions:

1. Cost
2. Insuring independence of the versions
3. Synchronization of different versions

The following procedures have been suggested as guidelines in developing independent software versions:

1. All programmers work from the same requirements.
2. Each programmer or programming group works independently of the others and communication among the various groups is not permitted.
3. Each version of the software is subjected to the same comprehensive acceptance test.
4. Specification languages can be used to write two or more sets of equivalent specifications, the formal specifications can be analyzed and verified, and these can be used as separate independent problem specifications. This tends to reduce the probability of specification common mode failures.

In running large scale tests on software, we can take advantage of some features of TMR. Since we will be producing at least 3 versions of the software, using TMR, we can run a large scale test by comparing the outputs of the three programs. This is cheaper than an ordinary test since we are relieved of the cost of computing the correct output for each test case, because agreement of 3 outputs is deemed correct operation. Disagreement of two or three of the systems must be analyzed, and the number of identical wrong outputs can be statistically analyzed to yield an estimate of the common mode probability.

For a discussion of the problems in synchronizing the inputs and outputs of N versions and the problems in comparing the outputs of different versions due to roundoff error see Brilliant 1987.

7.0 SUMMARY AND CONCLUSIONS

N-version programming has been applied or proposed in a number of applications:

1. The flight control software for space shuttle has a primary system with 4 computers, a voter and, a primary software system. The backup system has a fifth computer running a second independent software version, written in a different language, by a separate contractor.

2. The slat and flap control system of the 310 airbus industry aircraft has redundant software.

3. The point switching, signal control, and traffic control in the Gothenberg area of the Swedish state railways uses redundant software.

4. Several authors have proposed the use of redundant software for nuclear reactor safety systems.

ACKNOWLEDGEMENT

This research was in part supported by NASA Langley Research Center under NASA Research Grant NAG-1-1001, "Computer Aided Reliability, Availability, and Safety Modeling for Fault-Tolerant Computer Systems".

REFERENCES

Bhargava, B. K.: Concurency Control and Reliability in Distributed Systems, Van Nostrand Reinhold Co., New York, NY, 1987.

Brilliant, S. S., J. C. Knight and N. G. Leveson: "The Consistent Comparison Problem in N-Version Software," ACM Sigsoft Software Engineering Notes, Vol. 12, No. 1, pp. 29-34, January 1987.

Chen, L. and A. Avizienis: "N-Version Programming: A Fault-Tolerance Approach to Reliability of Software Operation," Digest of Eighth Int'l. Fault-Tolerant Computing Symp., IEEE Computer Society, pp. 3-9., Toulouse, France, 1978.

Garman, J. R.: "The "Bug" Heard Round the World," ACM Sigsoft Software Engineering Notes, pp. 3-10., Oct. 1981.

Johnson, B. W.: Design and Analysis of Fault Tolerant Digital Design, Addison Wesley, Reading, MA, 1989.

Knight, J. C. and N. G. Leveson: "An Experimental Evaluation of Independence in Multiversion Programming," IEEE Trans. on Software Engineering, Vol. SE-12, No. 1, pp. 96-109, Jan. 1986,

Lala, P. K.: Fault Tolerant and Fault Testable Hardware Design, pp. 103-107, Prentice Hall, Englewood Cliffs, NJ, 1985.

Lauber, R. Ed.: Safety of Computer Control Systems, Pergamon Press, New York, NY, 1980.

Musa, J. D., et. al.: Software Reliability - Measurement, Prediction, Application, McGraw Hill, New York, 1987.

Pradhan, D. K.: Fault Tolerant Computing - Theory and Technique, pp. 664-677, Prentice Hall, Englewood Cliffs, NJ, Vol. I and II, 1986.

Shooman, M. L.: Software Engineering: Design, Reliability, and Management, McGraw Hill, New York, 1983.

Shooman, M. L.: Probabilistic Reliability: An Engineering Approach, First Edition, McGraw Hill, New York, 1968, Second Edition, Kreiger, Melbourne, FL, 1990.

Siewiorek D. P. and R. S. Swarz: The Theory and Practice of Reliable System Design, pp. 119-121; 169-175, Digital Press, Bedford, MA, 1982.

Zebrowski, E. L.: "Sources of Common Cause Failures in Decision Making Involved in Man-Made Catastrophes," Risk Assessment in Setting National Priorities, Edited by Bonin and Stevenson, Plenum Press, New York, NY, 1989.

TABLE 1

EXPERIMENT IN THREE-VERSION PROGRAMMING**

Number of faulty versions in the set	Number of tests	Correct executions of the set		Incorrect executions of the set	
		Number	Percent*	Number	Percent*
0	290	290	76	0	0
1	71	59	15	12	3
2	18	0	0	18	5
3	5	0	0	5	1
	384	349	91 .	35	9

* Percentage of the total cases
**From Chen 1978

TABLE 2

SOLUTION OF EQ. (3) FOR p BY INTERPOLATION GIVEN THAT R=0.91

p	$R=p^2(3-2p)$
1	1
0.9	0.972
0.8	0.896
0.85	0.939
0.82	0.914
0.818	0.912
0.816	0.911
Solution p=0.815	0.910
0.814	0.909

TABLE 3

VERSION FAILURE DATA, NUMBER OF FAILURES IN 1,000,000 TESTS [KNIGHT 1986]

Version	Failures	Pr(Success)	Version	Failures	Pr(Success)
1	2	0.999998	15	0	1.000000
2	0	1.000000	16	62	0.999938
3	2297	0.997703	17	269	0.999731
4	0	1.000000	18	115	0.999885
5	0	1.000000	19	264	0.999736
6	1149	0.998851	20	936	0.999064
7	71	0.999929	21	92	0.999908
8	323	0.999677	22	9656	0.990344
9	53	0.999947	23	80	0.999920
10	0	1.000000	24	260	0.999740
11	554	0.999446	25	97	0.999903
12	427	0.999573	26	883	0.999117
13	4	0.999996	27	0	1.000000
14	1368	0.998632			

TABLE 4

OCCURRENCES OF MULTIPLE FAILURES IN 1,000,000 TESTS [KNIGHT 1986]

Number	Probability	Occurrences
2	0.00055100	551
3	0.00034300	343
4	0.00024200	242
5	0.00007300	73
6	0.00003200	32
7	0.00001200	12
8	0.00000200	2
TOTAL		1255

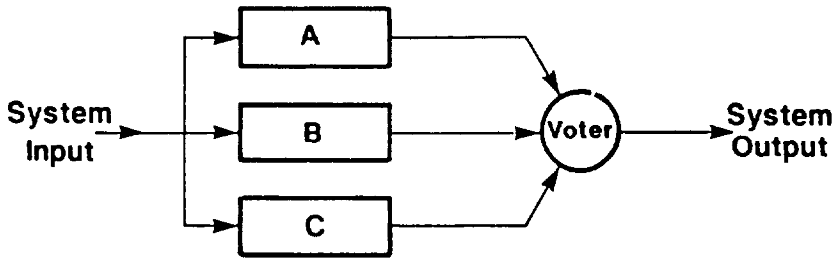


Fig. 1 TMR Reliability Model

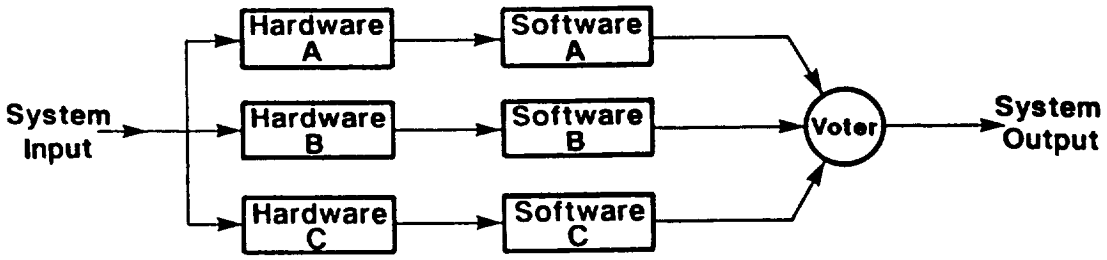


Fig. 2 A TMR System with Three Independent Software Versions

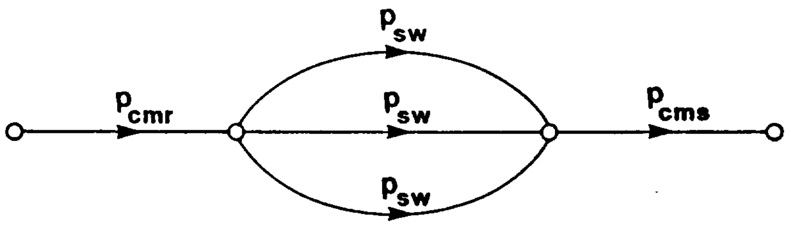


Fig. 3 Model of TMR Software (3-Version Programming) Including Common Mode Software Faults

P.K.

See page viii

Bounding and Approximating Markov Models

Arthur E. Laemmel, Polytechnic University, Farmingdale
Martin L. Shooman, Polytechnic University, Farmingdale

KEY WORDS Markov Model, Simplification, Availability, Merging, Bounds, Approximations

Abstract

Markov models are frequently used to analyze the reliability and availability of complex systems. For example, modern fault-tolerant computer systems can lead to hundreds or thousands of system states. Even if a computer program is used, simplification leads to reduced solution time, greater insight, and sometimes checks on the validity of the solution.

In a 1987 paper, the authors introduced criteria for merging states in a Markov model to reduce the complexity. Also, a method was involved for decomposing suitable processes into two or more subprocesses which are more easily solved. This papers develops simple expressions for upper and lower bounds on the Markov state probabilities obtained from inspection of the terms in the Markov probability matrix. These bounds can be used for quick paper, pencil, and calculator estimates. The bounds are also combined with the merging and decomposition methods in several examples. Use of these bounds can aid the well known simplification technique of truncation (deleting the low probability states), by quickly obtaining upper bounds on the state probabilities which insures that only low probability states are truncated.

Introduction

Problem Complexity

The standard technique for studying the availability of a repairable system is to begin by formulating a Markov probability model for the system. (Shooman, 1989). The solution of a Markov model for a system of n elements (with two states good and bad) involves m system states where $m \leq 2^n$. The term element applies to components, subsystems, replaceable units, modules, etc. The probabilities of being in a particular state at some time t are computed by formulating and solving a system of m coupled first order differential equations. The system availability, A, is then computed by summing the probabilities, (all Markov model states are mutually exclusive), for the subset of states where the system is UP.

Most modern systems are digital in nature, extremely complex, and often distributed and fault-tolerant. (Shooman 1989, Shooman and Laemmel 1987). In addition the hardware and software are coupled, which contributes additional complexity, and the system reliability is modeled by a many-state Markov model. If the system is repairable, additional states must be added to characterize realistically the logistics aspects of repair, including the number of available spares and the effect of shared repairmen.

At present, the approach to solving such large and complex Markov models is to use one or more of the available computer modeling programs. Many of these programs have been developed by NASA to model fault-tolerant computing systems, (Bavuso 1984, Ng 1977, Trivedi 1986). Such programs are quite effective; however, design insight is often lacking due to the complexity of the problem. Thus, a large number of computer runs are needed to establish sensitivities, as to how the various parameters affect the solution.

In most cases of Markov availability modeling, it is desirable to have simplifying analytical techniques. These can be used for initial studies to provide solution checks, design insight, and to reduce the number of computer runs, or to reduce the solution time for each run.

Several methods for simplifying such problems will be discussed and compared. These are:
1. State merging
2. Process decomposition
3. The use of upper and lower bounds
4. Approximations based on combinations of these methods

Our 1987 paper focused on the first and second techniques (see Shooman and Laemmel 1987), whereas this paper concentrates on the third and fourth methods and applies these techniques to examples of a fault-tolerant computing system (Siewiorek 1982) and a local area communications network (Bateman 1989).

Solution Complexity

Any sizeable problem will eventually utilize a computer for solution, thus modeling simplification techniques may be viewed as a preliminary step in the modeling and solution process. For a medium sized problem, a prior complexity reduction often allows one to perform a pencil, paper, and calculation solution. If the initial problem is large, the solution must eventually be done by computer; however, complexity reduction can reduce the solution time by a considerable amount. In the remainder of this section, we study the limiting complexity of computer solutions.

There is another benefit to complexity reduction. One can generally synthesize a large hypothetical problem which can be collapsed to a simple closed form analytical solution. Such an example can be used to test the validity of accuracy of a reliability modeling program by "feeding" the computer the large scale problem to solve, and comparing the results with the closed form analytical solution. Clearly, large reliability modeling programs are subject to the same types of residual "bugs" which all software experiences and it is difficult to invent large comprehensive test programs.

A Markov model results in a system of first order differential equations. If the analyst is interested in the time solution for the state probabilities, this system of equations must be solved numerically. We can analyze the order of complexity of such a solution by considering a simple standard solution technique. Consider the following set of first order differential equations which represent the Markov equations of an m state Markov model

$$\begin{aligned}\dot{p}_1(t) &= A_{11}p_1(t)+A_{12}p_2(t)+\cdots A_{1m}p_m(t)\\ \dot{p}_2(t) &= A_{21}p_1(t)+A_{22}p_2(t)+\cdots A_{2m}p_m(t)\\ &\cdots\cdots\cdots\cdots\cdots\cdots\\ \dot{p}_m(t) &= A_{m1}p_1(t)+A_{m2}p_2(t)+\cdots A_{mm}p_m(t)\end{aligned} \qquad (1)$$

where

$\dot{p}_i(t)$ = the first derivative of $p_i(t)$

$p_i(t)$ = the probability of being in state i at time t

A_{ij} = a coefficient which is the transition rate into state i from state j, for $i \neq j$. A_{ii} is defined to make column sums zero

0149-144X/90/0000-0254$01.00 © 1990 IEEE

Consider that we will use the simple Euler approximation to change the system of differential equations into difference equations

$$\frac{dp_i}{dt} \approx \frac{p_i(t+\Delta t) - p_i(t)}{\Delta t} \qquad (2)$$

Substitution of Eq. (2) into Eq. (1) yields

$$p_1(t+\Delta t) = \Delta t\left[(A_{11}+\frac{1}{\Delta t}) p_1(t) + A_{12}p_2(t) + \cdots A_{1m}p_m(t) \right]$$
$$p_2(t+\Delta t) = \Delta t\left[A_{21}p_1(t) + (A_{22}+\frac{1}{\Delta t}) p_2(t) + \cdots A_{2m}p_m(t) \right]$$
$$\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots$$
$$p_m(t+\Delta t) = \Delta t\left[A_{m1}p_1(t) + A_{m2}p_2(t) + \cdots (A_{mm}+\frac{1}{\Delta t}) p_m(t) \right] \qquad (3)$$

We begin the solution for the system of equations [Eq. (3)] at time t=0 by substituting the initial state probabilities (initial conditions) into the first equation (p_1) of Eq. (3). This requires m substitution operations, m additions, and m multiplications in the p_i equation, or we can lump the various arithmetic step times and just say m operations. Now since we must repeat this task m times (for each of the equations), we have m^2 operations. Of course this is just to compute the new state probabilities at $t=\Delta t$. We must repeat this for $2\Delta t$, $3\Delta t$, etc. Assume that we know the natural frequencies (eigenvalues) of the system of equations given in Eq. (1). If we let τ_{max} be the time constant associated with the lowest natural frequency (real part only if any are complex conjugate pairs), then we can say that the computation must continue until $t=5\tau_{max}$. At this time the corresponding exponential term in the solution is reduced to e^{-5} which is less than 1% of its initial value. Thus, we will need to repeat the computation $5\tau_{max}/\Delta t$ times. The choice of Δt is often a complex problem in numerical analysis, however, if we know the minimum time constant in the solution, τ_{min}, we can approximate Δt by $\tau_{min}/100$. Thus, an approximate expression for the number of operations to solve Eq. (1) is given by

$$\left(\frac{500\tau_{max}}{\tau_{min}}\right) m^2 \qquad (4)$$

In many cases, the analyst requires only the steady state values of the probabilities, i.e. $p_i(t)$ as $t\to\infty$, which we denote by p_i. To solve for the steady state values, all the time derivatives on the left hand side of Eq. (1) are set to zero, and any one of the m equations is deleted to cure the fact that the m equations are dependent. Then to replace the deleted equation we use

$$1 = p_1 + p_2 + \cdots p_m \qquad (5)$$

which yields a nonsingular set of m algebraic equations. We can compute the solution complexity of the m equations if we consider the Gaussian elimination method (Strang 1980). To perform Gaussian elimination we take the first and second equations, multiply equation two by A_{11}/A_{21} and subtract the modified second equation from the first to obtain a zero instead of A_{21}. This requires m operations and is repeated (m-1) times, i.e., for all equations but the first, resulting m(m-1) operations. The elimination procedure has to be repeated for each of the m variables, skipping the columns which have been made zero until a triangular matrix is obtained. This results in $1/3m(m^2-1)$ operations, and the overall order is approximately $m^3/3$. Comparing this with Eq. (4) we see that the time solution requires more steps than the steady state solution if $(500\ \tau_{max}/\tau_{min})>m$, which will often be the case.

In the future, we hope to analyze the order of complexity of some of the popular Markov model solving programs. (CARE III, Bauro 1984; ARIES, NJ 1977, HARP, Trivedi: 1986, etc.)

Previous Work

Merging

Our 1987 paper reports a set of rules which can be used to determine under what conditions the states in a Markov model can be combined (merged) to reduce the model complexity (Shooman and Laemmel 1987). As an example of the large reduction in complexity which can sometimes be obtained, we refer to the 189-state Markov model of a local area network given in Bateman (1989). The original model was solved using a computer program. We have reduced the 189-state model to an equivalent two-state model (all components UP, versus one or more components completely or partially DOWN) by applying the merger conditions, although not all state transitions satisfied are criteria. Simple analysis with pencil and paper and a pocket calculator yielded virtually the same probability as the computer program for the UP state probability. Of course solution for some of the other state probabilities would require further analysis; however, this result does provide some insight and an independent check on the computer program results. Our analytical techniques should also yield approximate solutions to more sophisticated figures of merit for the communications system such as expected channel capacity. (A weighted sum of the various failure mode probabilities and the reduced channel capacity of each partial failure mode.)

Process Decomposition

Process decomposition was discussed in our 1987 paper and involves decomposing the Markov model in question into two or more simpler models. The simpler models are solved and the state probabilities of the original model are obtained by a product of the decomposed model probabilities. Example I is discussed to illustrate this means of simplification. It is a decomposable four-state Markov model as shown in Fig. 1.

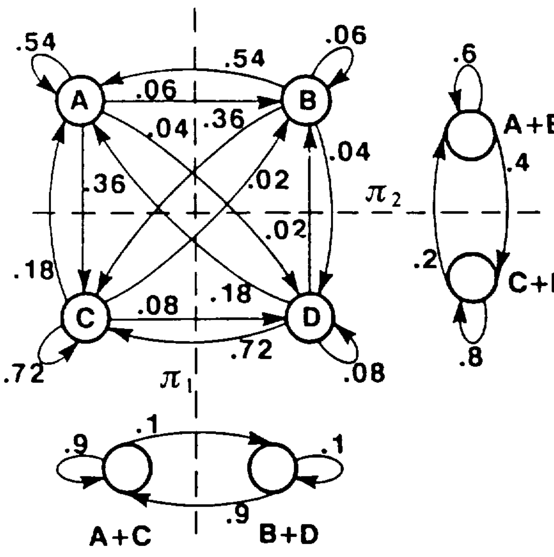


Fig. 1 - Example 1, a four-state Markov model which can be decomposed into two-state models. (Shooman and Laemmel, Volume IV, p. 6).

The original model has 4 states A,B,C,D which can be decomposed into two components (sub-Markov models) π_1 and π_2. As shown in the figure, the sub-model π_1 has two states which are related to the states in the original model by (A or C), (B or D). Similarly for sub model π_2, the two states are (A or B), (C or D). Assume that all the transition probabilities in this example can be viewed as $r_{ij}\Delta t = p_{ij}$ (transition rate times time interval), and the time interval can be assumed to be unity. We now solve the two sub-Markov models. These two models were solved using a computer program. The results for the steady state values are:

$$\begin{aligned} P(A+C) &= 0.9 \\ P(B+D) &= 0.1 \end{aligned} \qquad (6)$$

The probabilities of being in the four states of the original model are:

$$\begin{aligned} P(A) &= P(A+C) \times P(A+B) = 0.9 \times 0.333 = 0.300 \\ P(B) &= P(B+D) \times P(A+B) = 0.1 \times 0.333 = 0.0333 \\ P(C) &= P(A+C) \times P(C+D) = 0.9 \times 0.667 = 0.600 \\ P(D) &= P(B+D) \times P(C+D) = 0.1 \times 0.667 = 0.0667 \end{aligned} \qquad (8)$$

A separate solution of the four-state Markov model verifies the numerical results given in Eq. 8.

Upper and Lower Bounds

A search of the literature has revealed few bounds for the values of the steady state Markov model probabilities (eigenvector components for the stationary probabilities in a Markov process). (See White 1989, Butler 1988, Seneta 1981.) We discuss several bounds on the steady state Markov probabilities, p_i, for the i'th system state. The following derivations will utilize matrix notations for the Markov processes.

A Markov process can be described either by a rate matrix $\underline{A}$, (note the under bar used to denote a matrix). The expanded form of the equations is given in Eq. (1) or more compactly by

$$\dot{p}_i(t) = \sum_{j=1}^{m} A_{ij}p_j(t) \qquad (9)$$

and in matrix form this becomes

$$\dot{p}(t) = \underline{A}p(t) \qquad (10)$$

Note that the left hand side of Eq. (10) is a rate (per unit time), thus, $\underline{A}$ must also be composed of rates. Another way to formulate a Markov model is in terms of a probability matrix $\underline{P}$. The expanded form for such a formulation is

$$\begin{aligned} p_1(t) &= P_{11}p_1(t) + P_{12}p_2(t) + \cdots P_{1m}p_m(t) \\ p_2(t) &= P_{21}p_1(t) + P_{22}p_2(t) + \cdots P_{2m}p_m(t) \\ &\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ p_m(t) &= P_{m1}p_1(t) + P_{m2}p_2(t) + \cdots P_{mm}p_m(t) \end{aligned} \qquad (11)$$

and the alternate forms of Eq. (11) are

$$p_i(t+\Delta t) = \sum_{j=1}^{m} P_{ij}p_j(t) \qquad (12)$$

$$p(t+\Delta t) = \underline{P}p(t) \qquad (13)$$

The relation between the two matrices $\underline{P}$ and $\underline{A}$ is

$$\underline{P} = \exp(\Delta t\underline{A}) \qquad (14)$$

If Δt is very small, we can write

$$\underline{P} \approx \underline{I} + \Delta t\ \underline{A} \qquad (15)$$

We can denote the steady state value of p by p_∞. In the steady state, we can also write that $p_\infty = \underline{P}p_\infty$ which substituted into Eq. (12) leads to:

$$p_\infty = \underline{P}p_\infty = \underline{I}p_\infty + \Delta t\underline{A}p_\infty \qquad (16)$$

and in the steady state $\dot{p}=0$ which forces $\sum_{j=1}^{m} A_{ij}p_j$ to be zero (see Eq. (9)).

A more detailed derivation shows that Eq. (16) is exact rather than approximate. We can now use the above notation in our derivation of the bound equations.

Since P_{ij} is the probability of the system going to state i from state j, then we can write[1]

$$\sum_{i=1}^{m} P_{ij} = 1 \qquad 0 \le P_{ij} < 1 \qquad (17)$$

From Eq. (12) we can write an expression for the steady state where $p_i(t)$ is not changing

$$p_i = \sum_{j=1}^{m} P_{ij}p_j \qquad (18)$$

where

$$\sum_{i=1}^{m} p_i = 1 \qquad 0 \le p_i < 1 \qquad (19)$$

and (18) can be rearranged yielding

$$(1-P_{ii})p_i = \sum_{j\ne i} P_{ij}p_j \qquad (20)$$

We can now develop bounds on the state probabilities, p_i, based on Eq. (20). Define m_i and M_i as the minimum and maximum values, respectively, of P_{ij} with j varying over all values except i. That is, m_i and M_i are the minimum and maximum <u>off-diagonal</u> values of the ith row of matrix P_{ij}. All the p_i are non-negative, since they are probabilities. Thus, we can bound each of the P_{ij} terms on the right hand side of Eq. (20) by M_i from above and m_i from below yielding

$$m_i \sum_{j\ne i} p_j \le \sum_{j\ne i} P_{ij}\ p_j \le M_i \sum_{j\ne i} p_j \qquad (21)$$

Substituting the left hand side of Eq. (20) for the center term of Eq. (21) yields

$$m_i \sum_{j\ne i} p_j \le (1-P_{ii})p_i \le M_i \sum_{j\ne i} p_j \qquad (22)$$

Using Eq. (19), the left and right hand terms in Eq. (22) can be written as

$$m_i(1-p_i) \le (1-P_{ii})p_i \le M_i(1-p_i) \qquad (23)$$

We can solve for a lower bound on p_i from the left and center terms in Eq. (23) by solving the inequality for p_i yielding

$$m_i - m_ip_i \le p_i(1-P_{ii}) \qquad (24)$$

Since $(1-P_{ii}) + m_i$ is positive, solving for p_i results in the lower bound

$$\frac{m_i}{1-P_{ii} + m_i} \le p_i \qquad (25)$$

Similarly, using the right and center terms in Eq. (23) gives an upper bound

[1]We avoid the trivial cases where some P_{ij} is unity.

$$p_i \le \frac{M_i}{1-P_{ii} + M_i} \qquad (26)$$

Unless a transition is possible to state i from every other state, Eq. (25) gives zero as the lower bound for, p_i, the probability of being in state i. Since a lower bound of zero is of little help, we seek a closer bound. Let state k be some state from which a transition to state i is possible. Then from Eq.20, since all terms are non-negative, we can write that the single term is ≤ to the summation

$$P_{ik}\, p_k \le (1-P_{ii}) p_i$$

Now use Eq. 25 to obtain a lower bound on p_k yielding

$$\frac{P_{ik} m_k}{1-P_{kk}+m_k} \le (1-P_{ii}) p_i \qquad (27)$$

which gives

$$\frac{P_{ik} m_k}{(1-P_{ii})(1-P_{kk}+m_k)} \le p_i \qquad k \ne i \qquad (28)$$

Of course, this new lower bound might well be 0 if m_k is also 0, no matter which k is chosen.

A somewhat more involved variation on the above methods will always yield a non-zero lower bound for a state which actually has a non-zero probability. The process can be most easily explained by reference to a particular example described in the form of a state diagram.

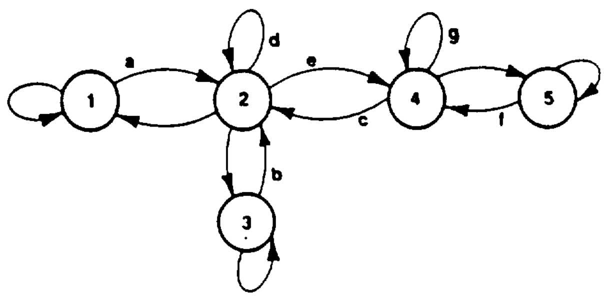


Fig. 2 - Example 2, a five-state Markov model.

Suppose a lower bound on p_2 is desired. There is no path from 5 to 2, so that p_5 does not appear in the expression for p_2, thus using Eq. 18 we obtain

$$p_2 = ap_1 + dp_2 + bp_3 + cp_4 \qquad (29)$$

rearranging terms yields

$$(1-d)p_2 = ap_1 + bp_3 + cp_4 \qquad (30)$$

We wish to manipulate Eq. 30 into a form where all the p_i terms appear. We observe that p_4 does depend on p_5, and an expression for p_4 can be substituted in Eq. (30).

$$p_4 = ep_2 + gp_4 + fp_5$$
$$(1-d)p_2 = ap_1 + bp_3 + c(ep_2 + gp_4 + fp_5)$$
$$(1-d-ce)p_2 = ap_1 + bp_3 + cgp_4 + cfp_5 \qquad (31)$$

Let r be the minimum of the coefficients a,b,cg,cf. Then

$$(1-d-ce)p_2 \ge r(p_1+p_3 + p_4 + p_5) = r(1-p_2)$$

and solving for p_2 yields a nonzero lower bound

$$\frac{r}{1-d-ce+r} \le p_2 \qquad (32)$$

Note that the term in parentheses in Eq. 31 is positive since the right hand side is positive, hence the division of an inequality is again legitimate. The method just described can be applied to any state which can be reached from every other state, and the tree of paths showing these transitions can be used as a guide in arriving at an equation such as Eq. 31. For the example of Fig. 2, the tree is shown in Fig. 3.

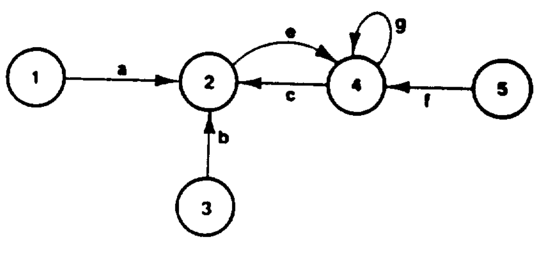


Fig. 3 - Spanning tree for Example 2

The numerical values in Example 2 are:

$$P = \begin{bmatrix} .6 & .02 & 0 & 0 & 0 \\ .4 & .95 & .6 & .3 & 0 \\ 0 & .02 & .4 & 0 & 0 \\ 0 & .01 & 0 & .3 & .2 \\ 0 & 0 & 0 & .4 & .8 \end{bmatrix}$$

The lower bound on p_2 from Eq. 25 is 0, but the upper bound from Eq. 26 is

$$p_2 \le \frac{.6}{1-.95+.6} = .923$$

The values of the coefficients in Eq. 31 are

$$a = .4 \qquad b = .6 \qquad cg = .09 \qquad cf = .06$$

Thus r=.06 and Eq. 32 yields

$$.561 = \frac{.06}{1-.95-.003+.06} \le p_2$$

The actual p_i and bounds are summarized in Table 1.

TABLE 1 - State Probabilities and Bounds for Example 2

States	1	2	3	4	5
UB (Eq. 26)	.048	.923	.032	.222	.667
p_i	.042	.845	.028	.028	.057
LB "tree"		.561			

Approximations Based on Combinations of the Above Methods

The bounds which are discussed in the preceding section are useful in their own right, as well as when combined with the merging and decomposition processes. For example, suppose that intuition tells us that several states are of low probability and can be dropped to simplify a model. We can use the upper bounds to guide us in identifying states which can be eliminated by merging them into other model states.

Use of Bounds as a Guide for Merging Low Probability States

Suppose it is desired to simplify a Markov process by dropping states with a probability of .033 or less. These states will not really be eliminated, rather they are to be merged with other states. In Example 2, the upper bounds from Eq. 26 and the exact state probabilities are shown in Table 1. Only state 3 has a probability <.033 and should be dropped, i.e., merged with state 2. Since the whole object is to simplify computations, we do not yet know that state 4 could also be dropped. (The upper bound on state 4 is not sharp.) Simply increase P_{22} to keep the second column sum unity, and drop row and column 3:

$$P = \begin{bmatrix} .6 & .02 & 0 & 0 \\ .4 & .97 & .3 & 0 \\ 0 & .01 & .3 & .2 \\ 0 & 0 & .4 & .8 \end{bmatrix}$$

The exact stationary probabilities are now:

$$.044 \quad .869 \quad .029 \quad .058$$

which agree well with the exact terms in Table 1. Note that the new merged state probability is approximately equal to the sum for states 2 and 4

$$.869 \approx .845 + .028$$

The process of dropping states can be generalized as follows. If the upper bound for p_k indicates that state k is relatively unimportant, then drop the k'th row and the k'th column from the Markov probability matrix. This will require that certain remaining values must be increased to maintain the unit column sums.

In the following example we compare approximate decomposition and bounds. A Markov process of fourth order is given as

$$\begin{bmatrix} .55 & .2 & .54 & .16 \\ .36 & .7 & .38 & .74 \\ .05 & 0 & .06 & 0 \\ .04 & .1 & .02 & .1 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix} \quad (33)$$

The exact values of p_i together with the upper and lower bounds from Eq. (25) and Eq. (26) are

TABLE 2 - Bounds and State Probabilities for Example of Eq. 33

UB (Eq. 26)	.545	.712	.051	.100
exact p_i	.311	.592	.017	.080
LB (Eq. 25)	.262	.545	0	.022

The worst bound, as far as ratios go, is the lower bound on p_3. However, the error of .017 actually makes this the best bound in absolute error. This example was chosen to be "approximately decomposable" into components

$$\begin{bmatrix} .9 & .9 \\ .1 & .1 \end{bmatrix} \quad \begin{bmatrix} .6 & .2 \\ .4 & .8 \end{bmatrix}$$

These have stationary state probabilities .9, .1, and .333, .667. Multiplying these, we get

$$.300 \quad .600 \quad .033 \quad .067$$

Conclusions

We conclude that the bounds which have been developed along with the various approximation techniques provide a valuable set of tools for solving complex Markov models. In some cases, a closed form analytical solution can be obtained. In other cases, bounds are available. Lastly, if a computer modeling program is required, our computations can be used as a check and to provide insight. Recent work on the solution of large Markov models which are approximately decomposable is described in Iazeolla 1984 and 1988, Courtois 1977 and Boxma 1986.

Acknowledgement

This research was in part supported by NASA Langley Research Center under NASA Research Grant NAG-1-1001.

References

1. M. L. Shooman and A. E. Laemmel: "Research on Combined Hardware and Software Reliability Models," Final Report, Polytechnic University, Oct. 1984. Polytechnic Report No. 84-003,004,005,006.

2. M. L. Shooman and A. E. Laemmel: "Simplification of Markov Models by State Merging," M.L.Shooman and A.E.Laemmel, Proc. 1987 Annual Reliability and Maintainability Symp.

3. A. E. Laemmel: "Bounds on Markov State Probabilities", being submitted to SIAM Journal. 1989.

4. A. E. Laemmel: "Simplification of Markov Models of Availability," Polytechnic Institute of New York, Research Report, POLY 84-006, Oct. 1984.

5. M. L. Shooman: Probabilistic Reliability: An Engineering Approach, McGraw-Hill, NY, 1968, Reprint Edition, Krieger, Melbourne, FL, 1989.

6. M. L. Shooman: "Approximating the Availability of Complex Systems with Built-In-Test and Hardware-Software Interaction", Polytechnic Report No. POLY84-004, Oct. 1984.

7. K. A. Bateman and E. R. Cortes: "Availability Modeling of FDDI Networks," Proc. Annual Reliability and Maintainability Symp., Jan. 1989, pp. 389-395.

8. S. J. Bavuso, et. al.: "CARE III Model Overview and User's Guide," NASA Technical Memo. 85810, NASA Langley Research Center, Hampton, VA., June 1984.

9. K. Grace, Jr.: "Approximate System Availability Models," Proc. of the 1969 Annual Symp. on Reliability, Jan. 1969, pp. 146-152.

10. Y. W. Ng and A. A. Avizienis: "ARIES - An Automated Reliability Estimation System for Redundant Digital Structures," (Proc. Annual Reliability and Maintainability Symp.), 1977, pp. 108-113. Also see paper in 1975 Reliability and Maintainability Symp. by Avizienis, pp. 333-339.

11. D. P. Siewiorek and R. S. Swarz: The Theory and Practice of Reliable System Design, Digital Press, Bedford, MA, 1982.

12. K. S. Trivedi, et. al.: "HARP: The Hybrid Automated Reliability Predictor, Introduction and Guide for Users," NASA Langley Research Center, Sept. 1986.

13. G. Strang: Linear Algebra and Its Applications, Second Edition, Academic Press, NY., 1980, pp. 1-5.

14. A.L. White: "Motivating the Sure Bounds", Proc. of the 1989 Annual Symp. on Reliability, Jan. 1989, pp. 277-282.

15. R.W. Butler and A.L. White: "Sure Reliability Analysis: Program and Mathematics", NASA Technical Paper 2764, March 1988.

16. E. Seneta: Non-Negative Matrices and Markov Chains, Springer-Verlag, p. 65.

17. G.I. Iazello, et al.: "Mathematical Computer Performance and Reliability", North-Holland, New York 1984.

18. G.I. Iazeolla, et al.: "Computer Performance and Reliability", North-Holland, New York 1988.

19. P.J. Courtois: "Decomposability Queueing and Computer System Applications, Academic Press, NY 1977.

20. O.J. 'Boxma et al.: "Teletraffic Analysis and Computer Performance Evaluation", North-Holland, 1986.

Biographies

Martin L. Shooman
Professor of Electrical Engineering and Computer Science
Polytechnic University
Route 110
Farmingdale, NY 11735 USA

Martin L. Shooman is a Professor of Electrical Engineering and Computer Science at Polytechnic University. He received his BS and MS degrees from MIT and his DEE (Ph.D. in EE) from Polytechnic Institute of Brooklyn. He has been active in research and teaching in the fields of probabilistic modeling, software engineering, and computers. He has won five best papers awards from the IEEE Reliability and Computer Societies. Dr. Shooman is author of "Probabilistic Reliability", second edition, Krieger 1990, and "Software Engineering", McGraw-Hill, 1983; (Japanese and Chinese editions published in 1989).

Arthur E. Laemmel
Professor of Electrical Engineering and Computer Science
Polytechnic University
Route 110
Farmingdale, NY 11735 USA

Arthur E. Laemmel is a Professor of Electrical Engineering and Computer Science at Polytechnic University. He received his BEE degree from Polytechnic Institute of Brooklyn. He has been active in research and teaching in the fields of probabilistic modeling, automata, electromagnetic theory, communications theory and computers. He won the best paper award from the IEEÉ Computer Society for his 1977 COMPCON paper "Statistical Theory of Computer Programs".

N16

Reliability and Availability Modeling of Coupled Communication Networks

A Simplified Modeling Approach

P.K.

See page viii

Martin L. Shooman, Polytechnic University, Farmingdale, NY
Eladio Cortes, Digital Equipment Corp., Littleton, MA

Key Words: Networks, LAN. Availability, Modeling, Approximation, Simplification

ABSTRACT

Modeling the reliability of Local Area Networks (LAN) and LANS interconnected by bridges and routers is a challenging problem because of the network complexity. If repair is included, the problem becomes even more complex and the general approach is to formulate a Markov model. The difficulty is that the several components of a LAN and their various modes of failure lead to a very complex Markov model with hundreds or thousands of states.

A basic approach is to simplify the problem by reducing the number of states by truncation and state merging.

The example used in this paper is a LAN described by Bateman and Cortes [7] with a Fiber Distributed Data Interface (FDDI), dual (redundant) counter rotating rings and two wiring concentrators which provide connections between the 16 computing stations and the two network rings. The primary ring carries communication until a failure takes place and then the secondary ring is activated. The Markov model which Bateman and Cortes formulated for this system required 189 states and solution of this model took about 1/2 minute of VAX cpu time.

This paper consider various simplification methods which have recently been developed by Shooman and Laemmel [14, 15] and applies them to this problem to illustrate their power in reducing the complexity of otherwise nearly intractable communications network models.

Through the use of state merging, we were able to reduce the Bateman-Cortes 161 state model to a two state model with a closed form solution which yielded an up state complexity of Pup = 0.999097895 compared with P(88) = 0.9990973 which was computed in the Bateman and Cortes paper. Other approximations are also discussed.

In the case of coupled networks, we must employ a technique which allows for problem decomposition. We begin in a top down fashion and assume that the network coupling elements (bridges) fail in a few simple modes, initially just good or bad. We can then decouple (decompose) the problem into a combination of bridge and LAN subsystems and formulate a reliability block diagram or fault tree to represent the entire system.

INTRODUCTION

Modeling the reliability of Local Area Networks (LAN) and LANS interconnected by bridges[1] and repeaters (see Note 1) is a challenging problem because of the network complexity [1], [2]. Furthermore, as users depend more heavily on LANs, many will regard reliability as the paramount goal, [3], p. 165. Previous network approaches have dwelt on the combinatorics of the problem [4], and much of the work has not included equipment repair. In the case of no repair, combinatorial cut-set and tie-set methods can be used [5].

If repair is included, the problem becomes more complex and one relies on the general approach to modeling repairable systems, i.e. formulation of a Markov model [6], [5]. The difficulty with such an approach is that the several components of a LAN (dual attachment stations, single attachment stations[2], wiring concentrators, station ports, station links, bridges, and routers) and their various modes of failure lead to very complex Markov models with hundreds or thousands of states [7], [8]. Contemporary approaches to the solution of complex Markov models involve the use of modern, efficient Markov model solvers. Several such programs were developed to solve fault-tolerant computing models, [9], [10], [11], [12], during the 1980's with support from NASA Langley Research Center.

Other approaches are to simplify the problem by reducing the number of states through truncation (deleting low probability multiple failure states), [13] and state merging [14], [15]. Of course, a configuration involving more than one LAN with associated bridges and repeaters, (an extended LAN), is even more complex and a combination of various modeling strategies and approximations is necessary.

The logical structure of the bridges is quite complex, and the algorithms which they implement in building network interconnections, handling the routing of messages, recovery, etc. needs further study, [16], [17]. These details can also be modeled using a Markov model, however, the complexity of the problem increases with the number of bridge failure modes.

This paper will consider various simplification methods which have recently been developed [14], [15] and apply them to various LAN and extended LAN models to illustrate their power in reducing the complexity of otherwise nearly intractable communications network models. (Even if the models are tractable, the computation times are long and the results are difficult to interpret because of the complexity of the model.)

COMPLEXITY

The standard model for analyzing the reliability and availability of a system with repair is a Markov model. If a Markov model has M states, then the related equations are a set of M coupled first order differential equations. If only steady state probabilities are required, the M differential equations can be converted into M simultaneous linear equations. For any sizable practical problem, such as the type of networks we are considering, M becomes very large, solution times become very long, and design insight and parameter sensitivity computations become very difficult. If there are n elements in a system, the Markov model may contain as many as 2^n states, which results in a set of 2^n coupled first order differential equations. Even if steady state values are acceptable, the resulting set of 2^n coupled linear equations is formidable.

Consider the following example. Bateman and Cortes [7] modeled a Fiber Distributed Data Interface (FDDI) ring, with dual (redundant) counterrotating rings and two wiring concentrators (which provide connections between stations and rings), see Fig. 1. The primary ring carries communication until a failure takes place and then the secondary

[1]A repeater connects two LANS and passes all messages on LAN 1 to LAN 2 and vice-versa. A bridge also connects two LANS but is smarter. It knows who is connected to each LAN and ignores intralan messages but passes on interlan messages in both directions. A bridge also has a "learning mode" which allows it to automatically build a "membership" table as the network is initially configured.

[2]A station is an addressable device attached to the network such as a computer, a terminal server, a workstation, etc.

/.

counter rotating ring is activated.

Each of the two concentrators has 8 stations (single attachment stations, SAS) which can communicate with each other yielding 2 out of 16 = 120 different combinations of connections. One could formulate a Markov model based upon the number of communication pairs which are working in the system, yielding a 120 state model. Bateman and Cortes [7] focused on the number of stations which could communicate through each concentrator and the states were labeled accordingly, e.g., state 88 meant 8 stations on concentrator one could communicate with any of the 8 stations on concentrator two or with each other (inter as well as intra concentrator communication), and state 87 meant that one station on concentrator two could not communicate. Their model is shown in part in Fig. 2, and the total number of states is delineated in Table 1. Note that there are 81 total states shown in Table 1. The 20 states shown above the dashed line in Table 1 are represented in Fig. 2. Note that states 75 and 57 are lumped together since the two permutations are assumed to be equivalent in a reliability sense, and the 16 such states are marked with ' in Table 1. The remaining 45 states represent states where half (8) or fewer of the stations are connected, and all these are merged into State SS in Fig. 2. Since 8 or more disconnected stations represent major degredation, State SS is considered to be a failure state.

Each of the twenty states represented by circles in the Fig. 2 has additional associated failure modes which we can visualize as additional Markov model states appearing beneath the paper along an axis perpendicular to the plane of the paper. If the various failure modes are considered, each state in Fig. 2. has 7 additional associated states, [8], and we have 20 x 8 = 160 states, plus state SS, which yields 161 states. Solution of this model took about 1/2 minute of VAX cpu time using the SHARPE program, [18].

Another more complex example involves an FDDI ring used as a backbone[3] to connect four other LANS (two Ethernets, one Token ring, and a Token bus), via a packet forwarding device such as a gateway[4] or bridge and 3 computers. See Fig. 3.

In the system of Fig. 3 there are 22 different stations, including the computers shown and a 4 gateway/bridge connector, yielding a total of 26 elements (assuming that link reliability is ignored because it is much higher than station reliability). If we use a brute force Markov modeling approach we have 2^{26} = 64 million states! Clearly, we need to apply some modeling expertise if we are to solve this problem, and two general approaches will be discussed in the following section to guide one in model simplification.

APPROACHES TO MODELING AND COMPLEXITY REDUCTION

Bottom-Up Approach

One approach to modeling a complex problem is to identify all the elements and all the possible failure modes and visualize the complete state space of the problem. One can then formulate a reduced model. The basic approaches to model reduction are:

1. State merging
2. Probability bounds
3. State truncation
4. Problem decomposition

State Merging

State merging refers to the combining of states (also called aggregation), generally to simplify a problem. For example in the model given in Fig. 2, the authors merged state 57 with state 75. This was based on arguments of system symmetry and reasoning that communication between 5 stations on concentrator 1 with 7 stations on concentrator 2 is no different than communication between 7 stations on concentrator 1 with 5 stations on concentrator 2. In this example, this is true, however, there are general theorems which can be used to prove the validity of such mergers in general, [14]. In fact, such theorems can be applied to further simplify the model of Fig. 2.

A more complete version of the Markov model given in Fig. 2 in the vicinity of state 88 is given in Fig. 4. In the event of a station, station port, or station link failure, a failure state (wrapped, bypassed, or partitioned) is entered. A failure in a station, station port or station link will result in a bypassed station, i.e., the model will move from state 88 to state 87. There are two types of ports in the wiring concentrator: station ports and ring ports [8]. A station port connects to an end station. A ring port is connected to the backbone ring and interconnects the two wiring concentrators. A wrapped network (state 88W) will only occur when there is a failure in the backbone ring or in one of the ring ports. A second ring port failure will result in a "partitioned" network (state 88 P). Multiple station, station port or station link failures will result only in "bypassed" states like state 87, 86, 77, and so forth (Fig. 2).

In Fig. 4, state 88 represents complete operation of the system and all other states represent degraded operation or complete failure. An an illustration of the power of the state merging method, we will reduce the model represented by Figs. 2 and 4 to an equivalent two state model composed of two blocks of merged states. The first block will consist of only state 88 and will be called Bup and the second block will contain all the other states in the model and will be called Bdown, see Fig. 5. In Fig. 4 we illustrate these two blocks by the interior and exterior of the shaded circle. The mathematical conditions under which various types of merging can take place are developed in [14] and are given in Appendix A. For this type of merger, all the transition rates from the states within Bdown to Bup must be the same. The repair rates indicated in Figs. 2, and 4 are given in Table 3.

For the merger conditions to hold exactly, $\mu_n = \mu_s = \mu_p = \mu_r$, and the Bateman and Cortes paper assumes that μ_n, μ_s, and $\mu_p = \mu$, however, μ_r = 0 (no repair shown) and the condition is violated. Since the probability of "all the other states" as defined in Table 3 is small, [7], their effect will be small, and the violation of the condition will be insignificant. Thus, we proceed as if the conditions were all met, and set all repair rates to μ. The merger results in the two state model of Fig.5.

The equivalent repair rate y in Fig. 5 is set = μ, which in the paper [7] was assumed to be 0.25 repairs/hr. which is the reciprocal of the assumed mean time to repair, MTTR = 4 hours. The mathematical conditions for the merger also require that the equivalent failure rate, x, be set equal to the sum of all the transition rates from Bup to Bdown, which are given in Table 3. Thus, x = 225.726×10^{-6}.

For the two state model of Fig. 5, it is a well known result that the steady state value of the probability of being in the up state is given by

$$P(Bup) = P(88) = y/(y + x) = 0.25/(0.25 + 225.73 \times 10^{-6}) \quad (1)$$

$$= 0.999097879$$

From the paper, [7], the value calculated using the SHARPE program was P(88) = 0.9990973, and the two state approximation yields very accurate results since the percentage difference in [1 - P(88)] is .066%

Probability Bounds

One can establish various upper and lower bounds and approximations on Markov model state probabilities by simple computations involving the terms in the probability transition matrix [19], [15]. For example, a simple upper bound on a Markov steady state probability is given by:

$$p_i \leq M_i/(1 - P(i,i) + M_i) \quad (2)$$

where:	p_i	= the steady state probability for the i'th model state
	M_i	= the maximum off diagonal transition
	$P(i,i)$	= the diagonal element for row i of the transition matrix

As an example of how these bounds are applied we consider the Markov model of Fig. 2. First we will use Eq. (2) to calculate an upper bound for state 88. The top row in the transition matrix shown in Table 4 contains a diagonal term as the first term which is equal to unity minus the sum of all the transitions leaving the state which was previously computed in Table 3, thus

$$P(i,i) = 225.73 \times 10^{-6} \quad (3)$$

For state 88, (the top row in the transition matrix), the terms following the diagonal term are zero except for the repair transitions from various states back to state 88, which are all the same and equal to 0.25, thus the maximum off diagonal term is 0.25 and

$$M_i = 0.25 \quad (4)$$

[3]A backbone is the major structure to which other subsidiary networks and devices connect.

[4]A gateway forwards packets at a higher level than a bridge and is often used to connect wide area networks, WANS.

Substitution of Eq. (3) and (4) into Eq. (2) yields the result p88 = 0.999097895, thus our previous approximation is also an upper bound.

It is useful to use the interpretations of the previous computation to rewrite Eq. (2) in a form which can yield insight when the Markov model is studied. The nondiagonal terms in the transition matrix are the transition rate on the branches entering a node in the associated Markov graph. Thus, if we wish to calculate an upper bound on p_i, then M_i is the maximum transition rate associated with all the branches which enter node i. The diagonal term in each row of the transition matrix is given by 1-Σ leaving branch rates. Thus, $1-P_i = \Sigma$ leaving branch rates, and Eq. (2) becomes

$$p_i \le \frac{\text{Maximum Entering Branch Rate}}{\Sigma \text{ Leaving Branch Rates} + \text{Max. Entering Branch Rate}} \qquad (5)$$

State Truncation

Practitioners have used state truncation as a technique for solving complex Markov models for many years. In general the analyst identifies states in the initial Markov diagram which seem to be of low probability based on intuition or some rough computation, and these are deleted from the model. This process continues until a manageable size model is obtained which is then solved. The problem with this practical approach is that one can never be sure about the size of the error produced by neglecting the truncated states. The availability of the simple bounds of the previous section supplies a general technique for computing the significance of the truncated states. For example, [7] reduced the 21 state Markov model shown in Fig. 2 to a simplified 6 state model (see Fig. 6).

This problem provides a good example for application of the state probability bounds. The simplified model in Fig. 6 will hold if we can show that states 85, 75, 65, and 55 have negligible small probabilities. Our approach is to apply Eq. (5) to each of these states and show that the upper bound probabilities are all negligibly and thus can be deleted. Bateman and Cortes [7] showed this was so by solving the complete model first. Unfortunately, when we apply Eq. (5) to compute the upper bounds, they all turn out to be 0.5 which are not sharp enough bounds to provide the desired proof. The reason why this happens can be easily understood if we consider Eq. (5), and Table 4. The maximum off-diagonal terms in rows 85, 75, 65, and 55 are the repair rates μ_n which numerically dominates the sum, thus the ratio given by Eq.(5) is approximately $\mu_n/(\mu_n+\mu_n) = 1/2$.

Another technique for obtaining the solution is to use a more advanced set of bounds given in [19]. Also a third approach is to merge states 88, 87, 86, 77, 76, and 66 into block B1 and the remainder of the states into block B2. If we can show that the probability for block B2 is negligible then all the states in B2 can be neglected.

Problem Decomposition

In some situations it is possible to decompose (disaggregate) a Markov model into two simpler models, where the original model reliabilities are simply related to those of the decomposed parts. since it is easier to solve the parts and combine the results than to solve the original model, this represents a simplification [14].

A Top-Down Approach

Another approach to reliability modeling of complex interconnected communication networks is to view the interconnected networks as a system of independent elements and each LAN, gateway/bridge, or computer as a subsystem. A reliability/availability fault-tree or block diagram can then be used to decompose the system reliability/availability into subsystem reliabilities/availabilities. Of course we must formulate and solve separate Markov models for each of the subsystems and then substitute these probabilities into the combinatorial expressions for the fault-tree or block diagram, however, this "divide and conquer" approach is easier. The top down approach which we have just described is similar mathematically to the decomposition approach of the preceding section, however, the motivation is based upon system considerations rather than mathematical properties of the problem.

As an example of situations where the repair organization can lead to coupling and decoupling, let us consider two hypothetical distributed networks, one where the top-down approach will work, and one where it will fail. Assume we have an organization with a distributed network similar to Fig. 3. Suppose the backbone LAN and each of the 4 connected LAN's are run by different groups often in different buildings which perform their own repair and maintenance, (e.g. a university with largely independent departments and research laboratories). Such a situation does not have repairman coupling and should fit the necessary assumptions for top-down decomposition. However, suppose the Model of Fig. 3 is for a large insurance company where each network represents a division of the company located on one or more different floors of a large office building. In such a case, it is likely that a central organization will administer and service all the networks, and the repair actions of this group will couple all the systems. Thus, the networks would not be independent, and top-down decomposition might not be appropriate.

We can further illustrate the method using the example of Fig. 3.

We will use the following notation for the elements of the distributed network shown in Fig. 3, which is referred to as the system, SY:

FDDI Ring: (BACKBONE)	N_1 with computers C_1, C_2, C_3 (labeled clockwise starting at the top).
Token Bus:	N_2 with stations S_{21}, S_{22}, S_{23}, S_{24} (labeled from left to right).
Ethernet (on right)	N_3 with stations S_{31}, S_{32}, S_{33}, S_{34} (labeled from left to right)
Ethernet (on bottom)	N_4 with stations S_{41}, S_{42}, S_{43}, S_{44}, S_{45}
Token Ring:	N_5 with stations S_{51}, S_{52}, S_{53}, S_{54}, S_{55}, S_{56} (labeled clockwise from 12 o'clock position)
Gateways:	G_1 from N_1 to N_2, G_2 from N_1 to N_3, G_3 from N_1 to N_4, and G_4 from N_1 to N_5

If we assume that there is no repairman coupling between networks, (i.e. each network has its own repairman), and that the gateways G_1-G_2 are considered part of N_1, then we can write the following expression for the system reliability assuming that success means that all stations can communicate with each other.

$$R(SY) = P(N_1 \cdot N_2 \cdot N_3 \cdot N_4 \cdot N_5) \qquad (6)$$

where N_i = the event-all items in network i are working.

If the networks are independent, as we are assuming, then Eq. (1) becomes:

$$P(SY) = P(N_1) \times P(N_2) \times P(N_3) \times P(N_4) \times P(N_5) \qquad (7)$$

Each of the probabilities $P(N_i)$, are obtained from a separate Markov model as described below.

Model for N_2, N_3, N_4	Includes 4 or 5 stations plus ethernet attachments and network control units.
Model for N_5	Includes 6 stations plus token ring attachment and network control units.
Model for N_1	Includes 3 computers, 4 gateways, and FDDI attachment and network control units.

Each of the 5 Markov models is solved for the network success probabilities and these are substituted into Eq. (2). Although 5 separate Markov models are involved, the computations are still much simpler than solving a single 26 element model.

If a single repairman services all the networks there is coupling, and the above decomposition does not hold exactly. However, suppose that we compute the probability that a service queue is greater than one (the coupling condition). If this probability is sufficiently small, then decoupling is still a good approximation.

The probability that a station, say station S_{34}, can communicate with another station, say station S_{42}, can be accomplished by a method very similar to the calculation of P(SY) in Eqs. (6) and (7). In order for S_{34} to communicate with S_{42}, everything in the path from S_{34} to S_{42} has to be available. This can be visualized by drawing a Reliability Block Diagram (RBD), as shown in Fig. 7.

If all the blocks in the diagram are up and available, then there can be communication between the two terminals.

If we assume independence within the subsystems:

$$A = A_{e1} \times A_{be1} \times A_{tr} \times A_{be2} \times A_{e2} \qquad (8)$$

where

A	= Probability that S_{34} can communicate with S_{42} (availability)
$A_{s1(2)}$	= Probability that S_{34} (S_{42}) is up
$A_{bs1(2)}$	= Probability that bridge (gateway) 1 (2) is up
A_{br}	= Probability that the backbone ring is up

A Markov model can be developed for each one of the subsystems to find their "UP" probabilities.

This example results in a simple series RBD. however, if there had been two different bridges (bs1 and bs1') attaching N3 to the backbone (a very common occurrence), they would have been represented in the RBD by two blocks in parallel. This would result in the RBD of Fig. 8, and the availability would be given by:

$$A = A_{s1} \times (1-(1-A_{bs1})(1-A_{bs1'})) \times A_{br} \times A_{bs2} \times A_{s2} \qquad (9)$$

Notice that a model for each pair of users could potent:·lly result in a different RBD. However, most of the time one model will be representative of a group of stations, so that only a few different RBD's would need to be developed.

The analysis just performed will provide us with a "user perceived" availability of the network. If user S_{34} can communicate with S_{42}, then user S_{34} conside.s the network up, regardless of what is happening with the rest of the network. The analysis for P(SY) as described in the paper, on the other hand, will provide us with a "network view" of availability (or reliability). If a model is created to aid in network product design decisions, a network view of availability is sufficient. However, if we decide that a user view is most important, the problem could be much larger, as seen in the above example. Another interesting, but difficult problem, is that of determining the probability that say 90% of the stations can communicate.

NEW RESEARCH AND APPROACHES

The development of a Markov model for a big, complex network will always result in a big, intractable and unmanageable state space. The two approaches available to deal with this problem are to either tolerate the largeness ("largeness tolerance"), or avoid it ("largeness avoidance") [20]. In this paper both approaches have been studied.

The bottom up approach, is an example of the "largeness tolerance" approach. A complete model for the network has to be formulated, followed by the applications of the appropriate reduction techniques. The approach has the advantage of providing relatively accurate results, and once the model reduction has been accomplished, analysis will require very short computation times [7]. However, the model for the complete network has to be developed first. In a big and complex network, generation of the Markov model could be very difficult, if not impossible. Automatic generation of the state space may solve this problem. The use of Stochastic Petri Nets is an effective tool that can be used to model the system [21]. Once the network has been described with a Petri Net, several packages exist to generate and analyze it's associated Markov chain [22]. For very complex models, however, hugh files will be generated, vast amounts of memory will be required, and the analysis could take many hours of CPU time to complete.

The combinatorial methods, shown in Section 3.2 as a top down approach,are examples of the "largeness avoidance" approach. The technique will not require the development of a large, intractable models of the complete network. There are, however, several problems that still need to be solved. The accuracy of the model, for example, is greatly dependent on the assumption of independent repair facilities for each one of the different subsystems. More research is needed if the method is to be applied to cases where this assumption is not valid. A possible trade-off could be a mixed approach, where the bottom up approach can be used for portions of the network where the assumption cannot be made, together with a top down approach linking all the portions where the assumption can be safely made.

Two more factors should be considerᵉd when modeling systems like those discussed in this paper.

- Intermittents: Intermittent failures are those that clear themselves or only require a reboot of the system without repair being involved. It has been shown that intermittent failures can account for more than 90% of all the failures experienced by a system. The introduction of intermittents will make the model and analysis of the problem much more complex.
- Performance: Extremely low performance (either low throughput or high delay, or both) can be regarded, from a user perspective, as an unavailable or marginally available system. A decision should be made regarding what is considered a "down" system. The model, for example, could be extended to assume that the system is down if the delay is above a certain threshold or the throughput is below a certain threshold.

ACKNOWLEDGEMENTS

This research was in part supported by NASA Langley Research Center, Grant NAG-1-1001 and by the Digital Equipment Corp., External Research Program Agreement No. 887.

REFERENCES

1. J. McQuillan, "Will LAN Technology Replace WANs," Newsletter, Communications Networks Conference, Jan. 1989, p. 10.
2. Network - The Magazine of Computer Communications, (Special Issue on Bridges and Rosters), Vol. 2, No. 1, Jan. 1988.
3. T. Brooks, Ed., *The Local Area Network Reference Guide*, Prentice-Hall, Englewood Cliffs, NJ, 1985.
4. C. J. Colbourn, *The Combinatorics of Network Reliability*, Oxford University Press, NY, 1987.
5. M. L. Shooman, *Probabilistic Reliability: An Engineering Approach*, First Edition, McGraw-Hill Book, Co., New York, 1969, Second Edition, Kreiger, Melborne, FL., 1990.
6. K. S. Trivedi, *Probability & Statistics with Reliability, Queuing, and Computer Science Applications*, Prentice-Hall, Englewood Cliffs, NJ, 1982.
7. K. A. Bateman and E. R. Cortes, "Availability Modeling of FDDI Networks," *Proc. Annual Reliability and Maintainability Symp.*, Jan. 1989, pp. 389-395.
8. E. R. Cortes, "An Availability Comparison of Three Concentrator Alternatives in a Fiber Distributed Data Interface Ring Network," *Proc. Annual Reliability and Maintainability Symp.*, Jan. 1990, pp. 171-176.
9. S. J. Bavuso, et. al., "CARE III Model Overview and User's Guide," NASA Technical Memo. 85810, NASA Langley Research Center, Hampton, VA., June 1984.
10. Y. W. Ng and A. A. Avizienis, "ARIES - An Automated Reliability Estimation System for Redundant Digital Structures," *Proc. Annual Reliability and Maintainability Symp.*, 1977, pp. 108-113. Also see paper in 1975 RAM Symp. by Avizienis, pp. 333-339.
11. K. S. Trivedi, et. al., "HARP: The Hybrid Automated Reliability Predictor, Introduction and Guide for Users," NASA Langley Research Center, Sept. 1986.
12. R. W. Butler and A. L. White, SHURE Reliability Analysis - Program and Mathematics," NASA Technical Paper 2764, Langley Research Center, Hampton, Va., March 1988, and S. C. Johnson, "ASSIST Users Manual," NASA Technical Memo. 87735, Langley Research Center, Hampton, VA., Aug. 1986.
13. K. Grace, Jr., "Approximate System Availability Models," *Proc. of the 1969 Annual Symp. on Reliability*, Jan. 1969, pp. 146-152.
14. M. L. Shooman and A. E. Laemmel, "Simplification of Markov Models by State Merging", *Proc. Annual Reliability And Maintainability Symp.*, IEEE, New York, Jan. 1987.
15. A. E. Laemmel and M. L. Shooman, "Bounding and Approximating Markov Models", *Proc. Annual Reliability and Maintainability Symp.*, Jan. 1990.
16. W. Stallings, "When One LAN is Not Enough - Bridges, Routers and Gateways Can 'Network the Networks'," Byte, Jan. 1989, pp. 293-298.
17. Digital, *Digital's Wide Area Networking Solutions*, Digital Equipment Corp. Handbook, 1988.
18. R. A. Sahner and K. S. Trivedi, "SHARPE: Symbolic Hierarchical Automated Reliability and Performance Evaluator, Introduction and Guide for Users," Sept. 1986.
19. A. E. Laemmel, "Bounds on Markov State Probabilities", Computer Science Research Memo, Polytechnic University, June 1988, revised March 1989.
20. O. Ibe, et. al., "Stochastic Petri Net Modeling of a VAX Cluster System Availability," *Proc. of the Third Int'l. Workshop on Petri Nets and Performance Models*, Kyoto, Japan, Dec. 1989.
21. J. L. Peterson, *Petri Net Theory and the Modeling of Systems*, Prentice-Hall, Englewood Cliffs, NJ, 1981.
22. G. Ciardo, et. al., "Manual for the SPNP Package, Version 2.0," Duke University, Durham, NC, 1989.

23. A. S. Tanenbaum, *Computer Networks*, Second Edition, Prentice-Hall, Englewood Cliffs, NJ, 1988.

BIOGRAPHIES

Dr. Martin L. Shooman
Professor of Electrical Engineering & Computer Science
Polytechnic University
Route 110
Farmingdale, NY 11735 USA

Dr. Shooman received his BS and MS degrees from MIT and his Doctorate in Electrical Engineering from Polytechnic Institute of Brooklyn. He has been active in research and teaching in the fields of probabilistic modeling, software engineering, and computers. He has won five best papers awards from the IEEE Reliability and Computer Societies. Dr. Shooman is author of *Probabilistic Reliability*, second edition, Krieger 1990, and *Software Engineering*, McGraw-Hill, 1983; (Japanese and Chinese translations published in 1989).

Dr. Eladio R. Cortes
Digital Equipment Corporation
153 Taylor St. TAY 2-2/J14
Littleton, MA 01460 USA

Dr. Cortes is a Principal Reliability Engineer for Digital Equipment Corporation. He has worked for Digital for the past ten years in various manufacturing and engineering positions. Prior to Digital, Dr. Cortes worked five years as an instructor at the University of Puerto Rico. He holds a BSEE and an MSEE from the University of Puerto Rico, Mayaguez, and a Doctorate in Electrical Engineering from Tufts University, Medford, MA. He is a registered professional engineer, an ASQC certified reliability engineer and a member of the IEEE, ASQC and Tau Beta Pi.

APPENDIX A - State Merger Conditions

The rules for state merger are given in Table A-1 below. (See [5], pp. 529-533.)

As an example of the application of these rules consider the 8 state Markov model given in Fig. A-1.

We wish to simplify by merging states 011, 101, 110 into block B1 and states 001, 010, 100 into block B2. The results appear in Fig. A-2.

The result of the merging has been to reduce an 8 state model to an equivalent and much simpler 4 state model.

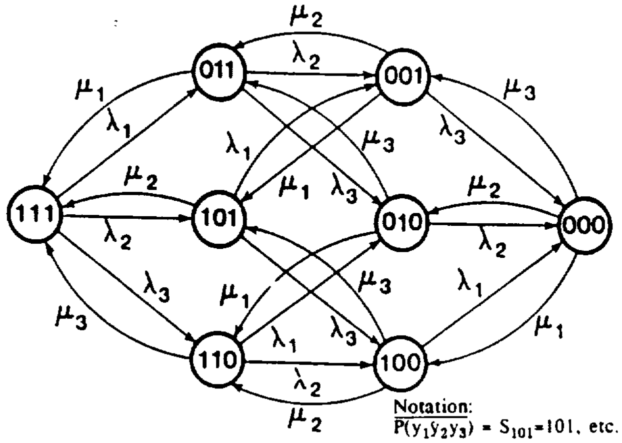


Fig. A-1 An 8 State Markov Model for Three Distinct Elements

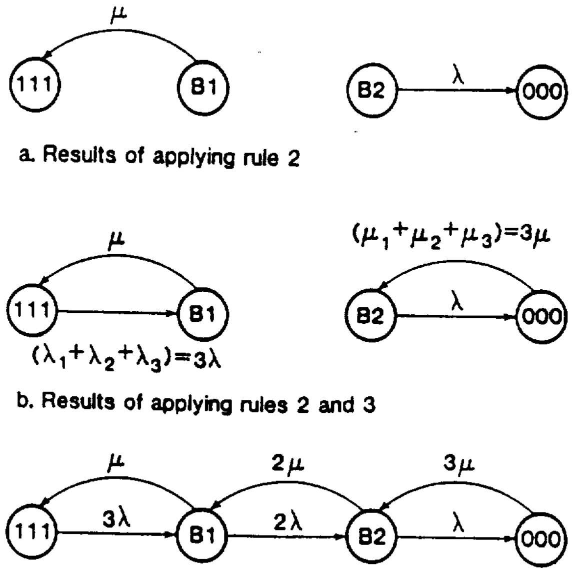

μ
111
B1
B2
λ
000
a. Results of applying rule 2
$(\mu_1+\mu_2+\mu_3)=3\mu$
$(\lambda_1+\lambda_2+\lambda_3)=3\lambda$
b. Results of applying rules 2 and 3
2μ
3μ
3λ
2λ
c. Results of applying rules 2, 3, 4

Fig. A-2 A Model Obtained from Fig. A-1 by Merging States

TABLE A-1 - State Merger Conditions

STATE MERGER CONDITIONS

The Following are a Necessary and Sufficient Set of Conditions for Merging States in a Markov Model

1. Transition rates between two nonmerged states are unchanged.
2. Transitions from a block of merged states to an unmerged state must all have the same transition rate. Note: If the original model has no transition branch from one of the merged states in the block to the unmerged state, then the transmission probability for the missing branch must be treated as zero.
3. Transitions from an unmerged state to a block of merged states are replaced by a single transition with a rate equal to the sum of the transition rates in the original model.
4. Transitions between any block of merged states B1 and any other block of merged states B2 must satisfy the following two rules:
   a. The transition rates from each node in block B1 to all nodes in block B2 must sum to the same constant $C_{1,2}$.
   b. The transition rates from each node in block B2 to all nodes in block B1 must sum to the same constant $C_{2,1}$.
5. Transition rates to the same block (self loops) are determined by summing the rates for all branches leaving the block and setting this sum to unity. The self loop turns out to be unity minus the sum of the transition rates of branches leaving the block.

TABLE 1 States for the Model of Fig. 2

88	87	86	85	84	83	82	81	80
78'	77	76	75	74	73	72	71	70
68'	67'	66	65	64	63	62	61	60
58'	57'	56'	55	54	53	52	51	50
48'	47'	46'	45'	44	43	42	41	40
38'	37'	36'	35	34	33	32	31	30
28'	27'	26	25	24	23	22	21	20
18'	17	16	15	14	13	12	11	10
08	07	06	05	04	03	02	01	00

TABLE 2 Transition Rates from Block Bdown to Bup

Initial State	Final State	Transition Rate (Repair)
87	88	μ_n
00	88	μ_a
88W	88	μ_p
SS	88	μ_s
All Other States	88	$\mu_x = 0$

TABLE 3 Transition Rates from Block Bup to Bdown

Initial State	Final State	Transition Rate (Repair)	Numerical value x 10^{-6}
88	87	$16\lambda n+16\lambda cp$	191.104
88	00	$16\lambda nu+20\lambda pu+2\lambda cu$	13.406
88	88W	$4\lambda cp$	9.044
88	80	$2\lambda c$	12.172
88	All Other States	0	0
		Total	225.726

Note: Bateman and Cortes [7] assume that $\lambda nu = 0.680$, $\lambda cu = 0.603$, $\lambda pu = 0.066$, $\lambda c = 6.086$, $\lambda cp = 2.261$. All failure rates are given as failures per million hours.

Table 4 A Transition Matrix for the Markov Model of Fig. 2

PRESENT STATE	NEW STATE																				
	88	87	86	85	84	83	82	81	77	76	75	74	73	72	66	65	64	63	55	54	00
88	$1-\lambda_{88}$	μ_n	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	μ_n
87	$16\lambda_a$	$1-\lambda_{87}$	μ_n	0	0	0	0	0	μ_n	0	0	0	0	0	0	0	0	0	0	0	0
86	0	$7\lambda_a$	$1-\lambda_{86}$	μ_n	0	0	0	0	0	μ_n	0	0	0	0	0	0	0	0	0	0	0
85	0	0	$6\lambda_a$	$1-\lambda_{85}$	μ_n	0	0	0	0	0	μ_n	0	0	0	0	0	0	0	0	0	0
84	0	0	0	$5\lambda_a$	$1-\lambda_{84}$	μ_n	0	0	0	0	0	μ_n	0	0	0	0	0	0	0	0	0
83	0	0	0	0	$4\lambda_a$	$1-\lambda_{83}$	μ_n	0	0	0	0	0	μ_n	0	0	0	0	0	0	0	0
82	0	0	0	0	0	$3\lambda_a$	$1-\lambda_{82}$	μ_n	0	0	0	0	0	μ_n	0	0	0	0	0	0	0
81	0	0	0	0	0	0	$2\lambda_a$	$1-\lambda_{81}$	0	0	0	0	0	0	0	0	0	0	0	0	0
77	0	$8\lambda_a$	0	0	0	0	0	0	$1-\lambda_{77}$	μ_n	0	0	0	0	0	0	0	0	0	0	0
76	0	0	$8\lambda_a$	0	0	0	0	0	0	$1-\lambda_{76}$	μ_n	0	0	0	μ_n	0	0	0	0	0	0
75	0	0	0	$8\lambda_a$	0	0	0	0	0	0	$1-\lambda_{75}$	μ_n	0	0	0	μ_n	0	0	0	0	0
74	0	0	0	0	$8\lambda_a$	0	0	0	0	0	0	$1-\lambda_{74}$	μ_n	0	0	0	μ_n	0	0	0	0
73	0	0	0	0	0	$8\lambda_a$	0	0	0	0	0	0	$1-\lambda_{73}$	μ_n	0	0	0	μ_n	0	0	0
72	0	0	0	0	0	0	$8\lambda_a$	0	0	0	0	0	0	$1-\lambda_{72}$	0	0	0	0	0	0	0
66	0	0	0	0	0	0	0	0	0	$7\lambda_a$	0	0	0	0	$1-\lambda_{66}$	μ_n	0	0	0	0	0
65	0	0	0	0	0	0	0	0	0	0	$7\lambda_a$	0	0	0	$12\lambda_a$	$1-\lambda_{65}$	μ_n	0	0	0	0
64	0	0	0	0	0	0	0	0	0	0	0	$7\lambda_a$	0	0	0	$5\lambda_a$	$1-\lambda_{64}$	μ_n	0	μ_n	0
63	0	0	0	0	0	0	0	0	0	0	0	0	$7\lambda_a$	0	0	0	$4\lambda_a$	$1-\lambda_{63}$	0	0	0
55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	$6\lambda_a$	0	0	$1-\lambda_{55}$	μ_n	0
54	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	$6\lambda_a$	0	$10\lambda_c$	$1-\lambda_{44}$	0
00	0	0	0	0	0	0	0	α	0	0	0	0	0	α	0	0	0	α	0	α	$1-\lambda_{00}$

where $\lambda_n + \lambda_{cp} = \lambda_a$

λ_{ij} = Σ transition rates leaving state ij

$\alpha = 9\lambda_a+2\lambda_c$

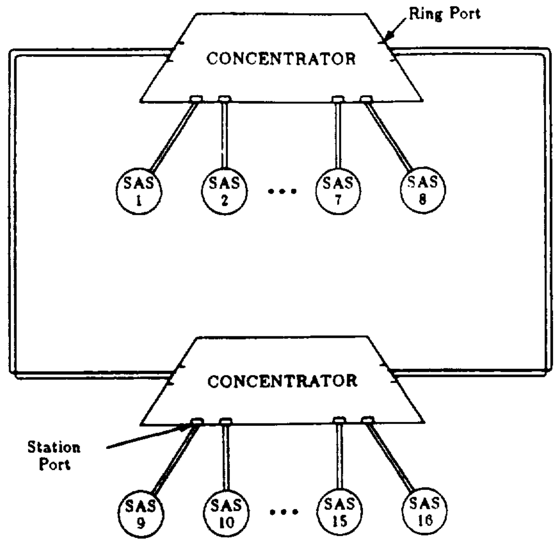


Fig. 1 FDDI Ring Network with Dual Counter Rotating Rings and Two Wiring Concentrators [7]

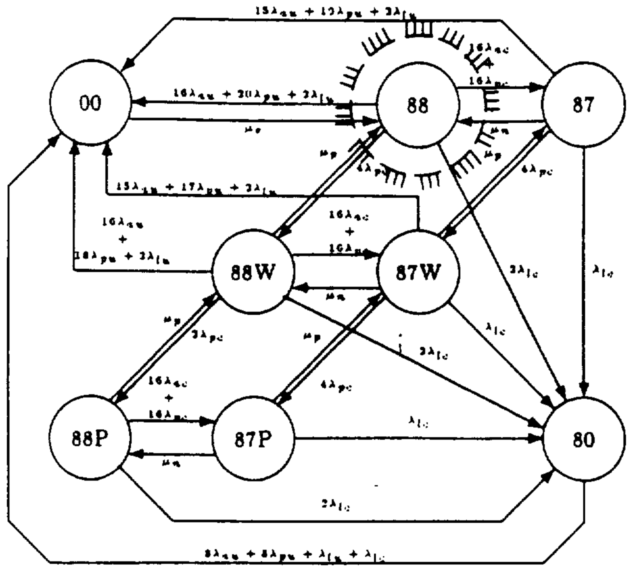


Fig. 4 Detailed Markov Model of FDDI Network (see Fig. 2) in the Vicinity of State 88 [7]

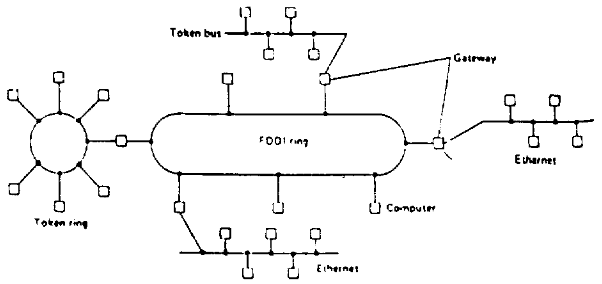
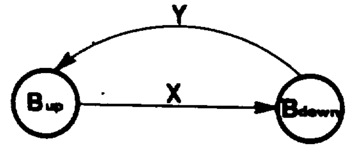


Fig. 3 A System Composed of 5 Coupled LANS. (Tutorial example adapted from Fig. 3-26, [23]

Note, this figure contains the four, most popular types of local area networks which all differ in geometry (ring or linear/bus), transmission media (fiberoptics, coaxial cable, twisted copper wires), and network control methods (message broadcast or control token passing). the acronym FDDI stands for fiber distributed data interface which is often used as a backbone to connect other networks (as illustrated in the figure) because of its high transmission speed. (See [23].)



Bup = P88 Bdown = All other states

Fig. 5 Simplified Two Block Markov Model for Figs. 2, 4.

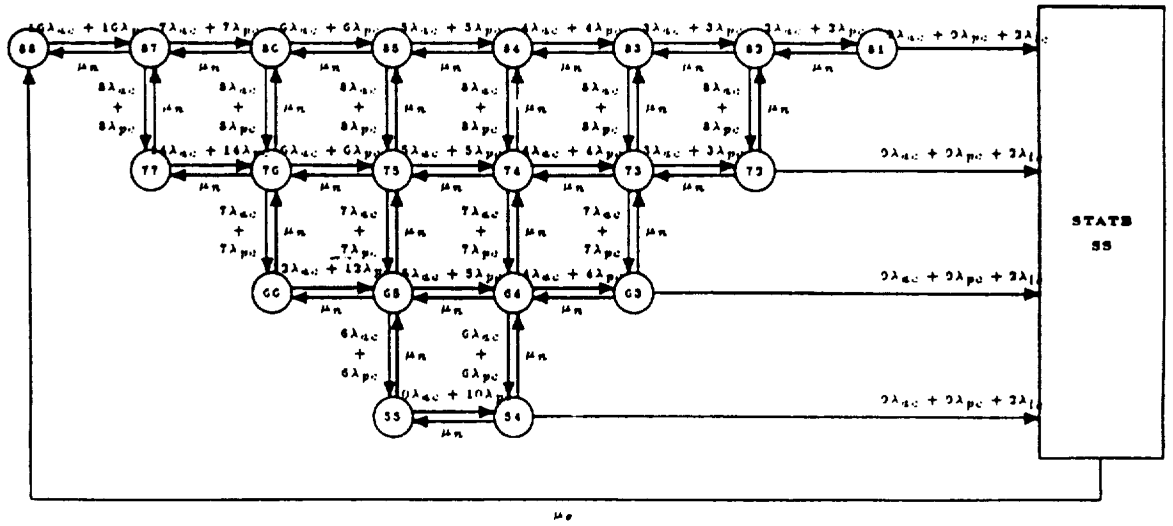
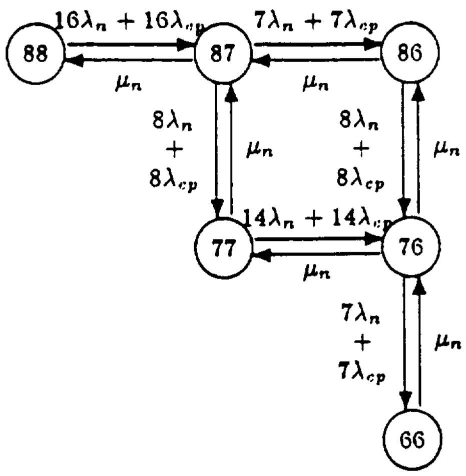


Fig. 2 Markov Model of FDDI Dual Ring, Two Concentrator Network [7]

ORIGINAL PAGE IS OF POOR QUALITY



88
87
86
77
76
66
$16\lambda_n + 16\lambda_{cp}$
$7\lambda_n + 7\lambda_{cp}$
μ_n
μ_n
$8\lambda_n + 8\lambda_{cp}$
μ_n
$8\lambda_n + 8\lambda_{cp}$
μ_n
$14\lambda_n + 14\lambda_{cp}$
μ_n
$7\lambda_n + 7\lambda_{cp}$
μ_n

Fig. 6 A Simplified 6 State Model Approximating the Model of Fig. 2.

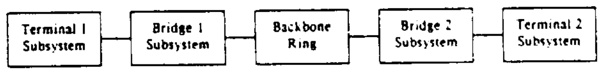


Fig. 7. A RBD for Communication Between S34 and S42

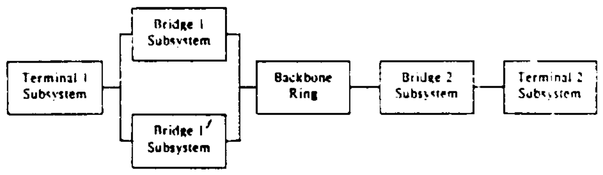


Fig. 8 A RBD for Communication Between S34 and S42 for a Two-Bridge Connection Between LAN N3 and the Backbone