Simulation Modeling for Long Duration Spacecraft Control Systems

Mark A. Boyd
NASA Ames Research Center
Moffett Field, California

and

Salvatore J. Bavuso
NASA Langley Research Center
Hampton, Virginia

Simulation Modeling for Long Duration Spacecraft Control Systems

Mark A. Boyd □ NASA Ames Research Center □ Moffett Field
Salvatore J. Bavuso □ NASA Langley Research Center □ Hampton

Key Words: Fault Tolerant Hypercube Multiprocessors, Fault Trees, Weibull Decreasing Failure Rates, Cold Spares, HARP (Hybrid Automated Reliability Predictor), Simulation for Reliability Model Evaluation

Summary and Conclusions

We describe the use of simulation and contrast it to analytical solution techniques for evaluation of analytical reliability models. We also discuss the role importance sampling plays in simulation of models of this type. We next describe the simulator tool we use for our analysis. Finally, we demonstrate the use of the simulator tool by applying it to evaluate the reliability of a fault tolerant hypercube multiprocessor intended for spacecraft designed for long duration missions. We use the reliability analysis to highlight the advantages and disadvantages offered by simulation over analytical solution of Markovian and non-Markovian reliability models.

1. INTRODUCTION

Recent work in the development of reliability analysis tools has produced a number of software packages that allow complex system behavior to be expressed with analytical models. The systems to which these modeling methods are applied often are complex fault tolerant computing systems designed for very high reliability. However, these systems can exhibit certain types of system behavior that require analytical models for which feasible analytical (numerical) solution techniques are not currently available. In these situations the existing analytical modeling framework may be enhanced to allow simulation of the analytical model (i.e. a fault tree or Markov model) as a replacement solution method to the traditional analytical solution techniques for the model. This is the approach that we follow in this paper.

The very large number of trials needed to obtain statistically significant results historically has been a significant problem for the use of simulation to model complex, highly reliable fault tolerant systems. Recent efforts to overcome this problem have produced new modeling tools capable of obtaining acceptable results with a reasonable number of trials through the use of a variance reduction technique called *importance sampling*. New modeling tools which incorporate this technique have been designed to be compatible with the Hybrid Automated Reliability Predictor (HARP) modeling tool[9], which is itself a component of the HiRel package of reliability modeling tools[1]. HARP solves the same types of models as the simulator, but uses analytical (numerical) solution techniques instead of simulation.

As is often the case, the development of the new modeling tool we describe here was driven by the needs of a specific reliability analysis project: the use of hypercube multiprocessors for highly reliable guidance, navigation, and control (G,N,& C) systems for long duration manned spacecraft. We are interested in exploring the use of a fault tolerant hypercube architecture that can use either hot or cold spares. It is clear from preliminary studies that the use of hot and cold spares with the traditional constant failure rate model will not meet the high reliability requirement for long duration space missions without onboard repair[11, 12, 19]. Recently acquired empirical data provide convincing evidence that *decreasing* failure rates are common in spacecraft applications[10]. For these reasons, we want to be able to include decreasing failure rates in our reliability analysis. The inclusion of decreasing failure rates with cold spares requires the use of a non-Markovian reliability model which is substantially more difficult to solve analytically than a Markovian model that assumes constant failure rates. Given the current state of the art, analytical solution of such non-Markovian models generally is tractable only for very small simple models, whereas the model of the above hypercube system is very large. The cumulative effect of all of these factors led us to the use of simulation modeling.

In this paper we summarize the use of simulation as a modeling method and describe how it can be applied to the evaluation of analytical system models. We compare evaluation of analytical models by simulation to evaluation by analytical solution techniques and describe the role of importance sampling in our implementation of simulation. We next describe the simulator itself and the process of specifying a model for use with it. We then illustrate the use of the simulator by applying it to a hypercube architecture proposed for a G,N,& C system for long duration spacecraft. We explore the effect of assuming decreasing failure rates for active and cold processors within the hypercube instead of constant failure rates, and demonstrate the advantages that simulation provides over analytical solution methods for such system models.

2. SIMULATION MODELING FOR RELIABILITY PREDICTION

The usual method of using simulation to evaluate reliability and performance of systems involves building a computer model of the system, generating events of interest (i.e. component failures), and observing the response of the model to the generated events. The timing and types of events are generated using probability distributions which are assumed to govern event occurrence. Values are sampled from the appropriate probability distributions and are used to specify which type of event occurs next and when that occurrence will be. A sequence of events is generated in this manner until either the mission time expires or the system fails. Such a sequence of events provides one instance of how the system would be expected to behave in the environment characterized by the governing probability distributions and is referred to as a "history" or "trial". The model is evaluated at the end of a trial to determine measures of interest such as whether the system is still operating (reliability) or how much work was accomplished (performance), etc. This process is then repeated numerous times to obtain average values for the measures of interest and accompanying sample standard deviations. From probability theory it follows that as the number of trials increases, the average value obtained in the simulation approaches more closely the actual value that characterizes the long run behavior of the system as expressed by the model. The standard deviation, which is a measure of the expected closeness of the simulation average to the actual value, is proportional to $\frac{1}{\sqrt{n}}$ (where n is the number of trials)[18]. Hence obtaining a highly accurate value for a measure of interest may require a very large number of trials.

2.1 Analytic Solution Methods vs. Simulation

An alternate approach to reliability evaluation involves building an analytic (mathematical) model to express the relevant behavior of the system. A number of different analytical model types are in widespread use. One very successful analytical model type is the Markov chain and its generalizations (non-Markovian discrete state models). These models express system behavior by identifying a number of distinct states in which the system may be. The system can be in only one state at a time, and from time to time makes a transition from one state to another. The distribution of the time the system spends in individual states and the characteristics of the transition rates between states differentiate Markovian and non-Markovian models[13]. Analytical models are usually solved using either direct or numerical methods, so often they can give answers with greater accuracy than simulation methods for a comparable amount of computational effort. However, analytical solution methods suffer from requiring much more memory storage for data structures than simulation methods. As a result, models that become too large to be accommodated by analytical solution methods might still be within reach of simulation techniques. In addition, increasing behavioral complexity in analytical models requires analytical solution techniques with increasing computational requirements. Hence to solve a model of sufficient complexity, an analytical solution method could require more (rather than less) execution time than a simulation method for a comparable level of accuracy in the output. In cases like these where the limitations of analytical solution methods are exceeded, simulation provides a useful alternate approach.

The drawback to building a computer simulation model of a system under study is that constructing the model and validating it is often a complex, time consuming, and error-prone process. An alternative is to apply simulation not to a model of the system itself, but to an analytical model of the system such as a Markovian or non-Markovian model. With this approach there is of course the problem of constructing the analytical model. However, this tends to be easier than constructing a system-level simulation model. Also, the topic of analytical model construction has been addressed by a number of researchers in the past several years and tools have been created to assist in model construction (see [3] and [4] for brief surveys of tools for automated Markov model construction). The approach we have chosen for the current study applies simulation to Markovian and non-Markovian models of the hypercube multiprocessor system. This allows us to capitalize on previous work performed by the authors on the hypercube system using Markovian models[5] and permits us to extend the scope of that work.

2.2 Simulation for Evaluation of Markovian and non-Markovian Models

Markovian and non-Markovian discrete-state models can be evaluated by simulation in the following way. Each trial represents a single traversal path among the states of the model. The common beginning point for all trials is at an initial state in which all system components are assumed to be operating correctly. Upon entry into each state, the process is begun for determining the time of transition out of the current state and which state the system goes to next. The time to next transition is sampled from a probability distribution that depends upon

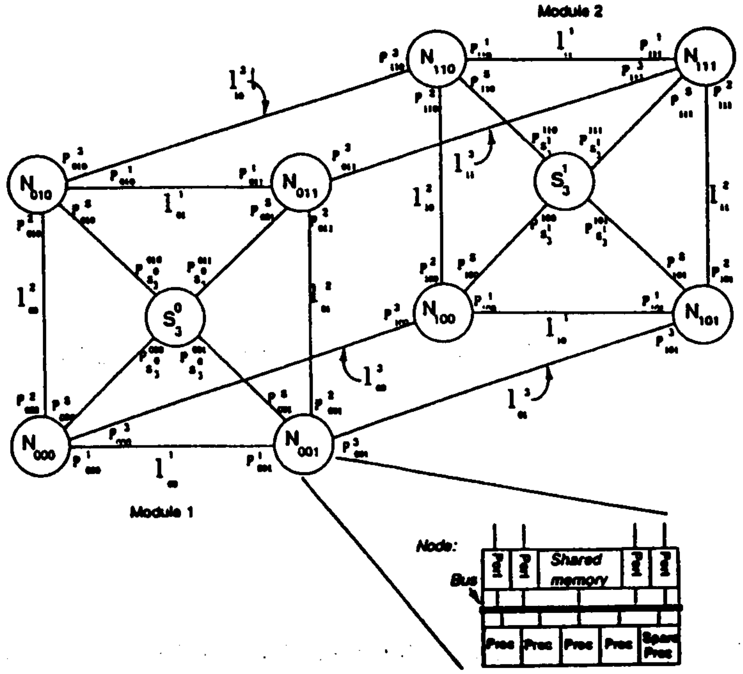

Figure 1: Hypercube Multiprocessor System

the failure rates of the components still active. If the failure rates of all components are constant, the model is a Markovian model. If the component failure rates are all functions of mission time (i.e. non-constant), the model is a non-homogeneous Markov model. If the component failure rates are individually functions of more than one time variable (i.e. there is more than one "clock" in the system upon which component failure rates may depend), the model is a non-Markovian model. We use all three types of models in the present study. Once the time to next transition has been determined, a sampling from a second distribution is done to determine which of the remaining operating components will experience the failure that is the cause of the transition out of the state. The determination of the sampling distributions is described in [14] and [16]. We note that this formulation of the simulation process can accommodate the use of Fault/Error Handling Models (FEHMs) to implement *behavioral decomposition* for incorporating imperfect fault coverage as is done in HARP[15]. Although that capability was available, we did not consider imperfect fault coverage in the present study. During each trial successive inter-state transitions are generated until either the mission time is exceeded or the system fails, causing the trial to end. The system unreliability is then estimated from the proportion of trials during which the system failed before the mission time was reached.

2.3 Importance Sampling

A major characteristic of highly reliable systems is that system failure events are extremely rare. This means that a large majority of the trials are likely to end by the mission time expiring rather than through a system failure. Since system failures are the events of interest, a very large total number of trials must be run before a sufficient number of system failures occur to provide a meaningful estimate from the proportion of failure trials to total trials (i.e. an estimate of the system unreliability). A variance reduction technique called *importance sampling* may be employed to reduce the total number of trials required. An excellent introduction to importance sampling may be found in [6]. The basic idea behind importance sampling is to select an alternate distribution from which to sample which has much higher probability density than the original distribution in the regions of interest where the original distribution's density is very small. Parity to sampling from the original distribution is maintained by weighting the observations sampled from the new distribution to reflect the relative difference in density magnitude between the two distributions. For example, if the density of the new distribution is four times greater than the density of the original distribution in a certain region, then a failure event observed in that region by sampling from the new distribution is counted as only $\frac{1}{4}$ of a failure. The importance sampling techniques implemented in the simulator we used for this study, called *forced transitions* and *failure biasing*, are described in [14]. Both have the effect of emphasizing component failure events in order to increase the number of trial terminations due to system

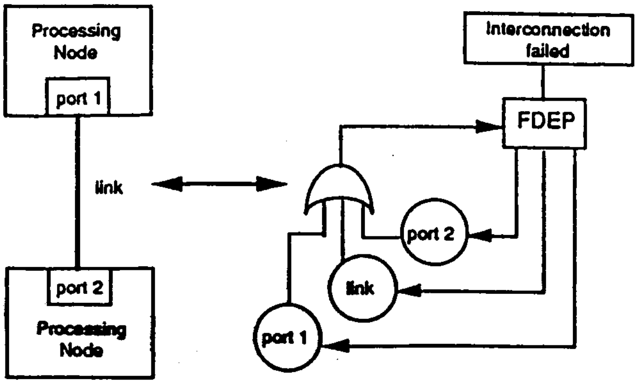

Figure 2: Modeling Functional Dependencies due to Processing Node Interconnections

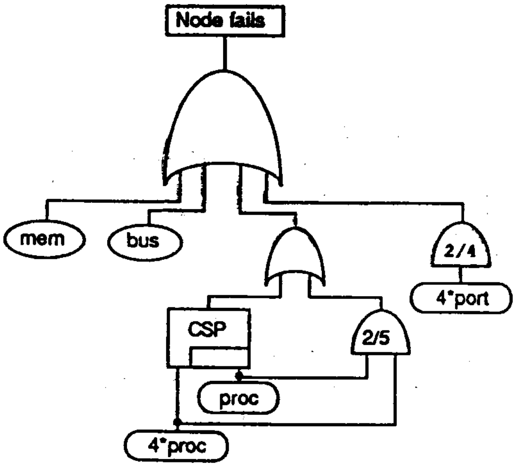

Figure 3: Fault tree model of Architecture 1 Processing Node with Cold Spares

failure, hence reducing the total number of trials needed in order to accumulate a sufficient number of system failure terminations to provide an acceptable estimate of the system unreliability.

2.4 Simulator Description

The original version of the simulator we used for our analysis was designed by Lewis[17] and implemented at Northwestern University. It required a system model to be described as a set of components arranged in groups. Each group could optionally have cold spares, and could have either a constant or a Weibull increasing failure rate. Each group could also have a Fault/Error Handling Model (FEHM) associated with it to allow the use of behavioral decomposition as is done in HARP. System failure criteria were specified in the form of a set of component cut sets which the analyst had to derive from a combinatorial model of the system (for example, a fault tree). For our study, we modified this simulator to enable it to use decreasing as well as increasing Weibull failure rates, and to allow it to accept the input model in the form of a *dynamic fault tree* (see below) rather than as a set of component cut sets. The resulting simulator program accepts its input model in the same form as the HARP program, and accepts input files with the same format as HARP. In addition, it is capable of evaluating all models that HARP is capable of evaluating, making it completely compatible with HARP. This is an important advantage because it allows the reliability analyst to develop his/her system model once and then input it to whatever evaluation program is most appropriate depending on the characteristics of the model and the programs. It also allows a comparative evaluation of the performance of the two programs by applying them both to the same model(s).

3. SYSTEM MODEL

The hypercube multiprocessor system and the model of it that we use in this study are described in [3] and [5] under the name of Architecture 1. We give a brief description of it here. The architecture is shown in figure 1. It consists of a 3-dimensional hypercube configured as two fault-tolerant 2-dimensional modules, each with a spare processing node. The processing nodes themselves are multiprocessors containing four active processors and a spare processor. The spare processor can be either a hot or cold spare. The structure of the processing nodes is also shown in figure 1. Each processing node communicates with other processing nodes in the system through four ports. For the system to be operational all eight processing nodes must be operational and must all be able to communicate with each other. Therefore, the system will be considered failed if any processing node fails and a spare processing node is unable to take over or if any two nodes in the hypercube are unable to communicate with each other.

Although the form of the analytical model that is actually evaluated is a Markovian/non-Markovian discrete-state model, it is specified by the reliability analyst in the form of a *dynamic fault tree*[3, 8]. When simulation is not used for model evaluation, the dynamic fault tree can be converted into a Markov chain which can then be solved numerically for state probabilities. When simulation is used for model evaluation, the discrete-state structure of the underlying Markovian model is inherent in the simulation process and the dynamic fault tree is used only to determine whether a state which has been entered is a failure state.

A dynamic fault tree is a generalized fault tree model in which the traditional set of combinatorial fault tree gates is extended to include several non-standard gates that are designed to express *sequence dependent* behavior. Sequence dependent behavior is behavior that depends in

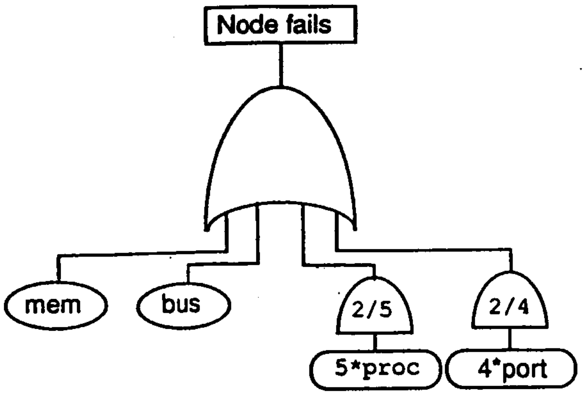

Figure 4: Fault tree model of Architecture 1 Processing Node with Hot Spares

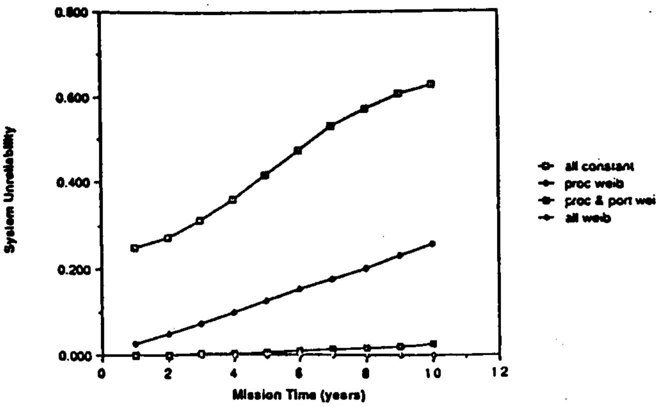

Figure 5: Effect of Weibull DFRs on System Unreliability (Hot Spares)

Component	Initial constant failure rate
Shared Memory	3.477×10^{-7}
Intra-node bus	1.147×10^{-7}
Processor	1.990×10^{-6}

Table 1: Initial Constant Hazard Rates (failures/hour) for Components in Processing Nodes

some way on the order in which events occur. The hypercube system under study exhibits two instances of this type of behavior: functional dependencies (the failure of one component causes one or more other components to either fail or become unavailable) and cold spares (a cold spare cannot fail while it is "cold"; it can fail only after it has been activated to substitute for a failed active component). The functional dependencies appear in the interconnections between the processing nodes; specifically, if either the internode link or one of the two ports on either side of the internode link fails, the remaining two components (link and/or port(s)) become useless to the remaining operation of the system and hence may be considered to be effectively failed themselves. These functional dependencies are modeled with *functional dependency gates*, as shown in figure 2. Cold spares are used within the processing nodes and are modeled using a *cold spare gate*, an example of which appears in figure 3.

Figure 4 models a processing node when the spare processor is hot (i.e. active and running from mission start just like the four initially active processors). The 2-out-of-4 gate for which the four ports are inputs reflects the effect of the message routing protocol[5]. Figure 3 models the processing node when the spare processor is cold. Diagrams of fault trees modeling the full architecture were omitted from this paper due to lack of space. The interested reader may find them in [3].

4. ANALYSIS RESULTS

We evaluated the system model for the cases where all components had constant failure rates with hot or cold spare processors (time homogeneous Markov models), various components had Weibull DFRs with hot spare processors (non-homogeneous Markov model), and various components had Weibull DFRs with cold spare processors (non-Markovian model). For this paper our primary purpose is to illustrate the use of simulation to evaluate the models and contrast it with analytical solution techniques. Therefore we will use here only selected results from our analysis to compare the advantages and disadvantages of simulation vs. analytical solution methods. A more complete reliability analysis of the hypercube system is found in [2]. Our primary analysis goal was to determine whether assuming Weibull decreasing failure rates (DFRs) for components instead of constant failure rates would result in a sufficient improvement in predicted system reliability to conclude that the architecture was adequate to successfully complete a 10 year mission. Results using constant failure rates[2] indicated that the proposed architecture would be inadequate, with the probability of system failure exceeding 60% after 10 years. Initial attempts to evaluate the model with HARP (which uses analytical solution techniques) were not successful due to the large size of the model. The dynamic fault tree model of the system contains 70 basic events (110 components total), and 175 fault tree nodes (basic events + gates). It produces a Markov model with many thousands of states. Furthermore, when Weibull DFRs are assumed together with cold spares, the size and complexity of the resulting non-Markovian model is well beyond the capability of any analytical solver tool that exists today, both in terms of memory and execution time required for its solution. In contrast, our simulator was able to evaluate the model with none of the problems experienced by HARP. Components with decreasing failure rates were assumed to have an initial failure rate λ_{exp} given in table 1 which declines monotonically over the mission time according to the Weibull failure rate expres-

Mission Time (Years)	All Components Constant FRs	Processors Weibull DFRs	Processors and Ports Weibull DFRs	All Components Weibull DFRs
1	.249 ± .016	.0250 ± .0031	.000519 ± .00022	.000255 ± .00013
2	.271 ± .016	.0489 ± .0048	.00147 ± .00031	.000361 ± .00015
3	.312 ± .017	.0738 ± .0065	.00286 ± .00044	.000439 ± .00017
4	.361 ± .018	.0988 ± .0091	.00481 ± .00078	.000504 ± .00019
5	.419 ± .018	.126 ± .014	.00729 ± .0013	.000550 ± .00020
6	.475 ± .018	.152 ± .017	.0102 ± .0018	.000638 ± .00031
7	.530 ± .018	.176 ± .019	.0135 ± .0024	.000673 ± .00033
8	.576 ± .017	.202 ± .023	.0173 ± .0037	.000718 ± .00036
9	.609 ± .016	.231 ± .031	.0208 ± .0045	.000766 ± .00041
10	.631 ± .013	.257 ± .036	.0257 ± .0091	.000777 ± .00041

Table 2: Effect of Weibull DFRs on System Unreliability (Hot Spares)

sion:

$$\lambda_{weib}(t) = \lambda_{exp}\alpha t^{\alpha-1} \qquad (1)$$

where α is the Weibull shape parameter[20] which is assumed to have the value $\alpha = 0.5$. All components not having DFRs were assumed to have constant failure rates given in table 1. Table 2 and figure 5 show the effect of assuming Weibull DFRs for various subsets of components. The results reported in table 2 are averaged over 10 runs of 10000 trials per run. The effect of assuming Weibull DFRs for increasing numbers of the components clearly results in decreasing system unreliability. The result of assuming Weibull DFRs for all components is a difference of about three orders of magnitude in the system unreliability (from 0.631 ± 0.013 when all components have constant FRs down to about $0.777 \times 10^{-3} \pm 0.41 \times 10^{-3}$ when all components have Weibull DFRs).

The above discussion illustrates the advantage that simulation can have over analytical techniques: simulation may be able to evaluate models that are beyond the reach of analytical techniques both in terms of memory and execution time. Furthermore, if only ballpark evaluations are desired, simulation may be able to produce the required results relatively quickly. Figure 6 contrasts the reliability predictions for the hypercube with hot spares assuming constant failure rates and Weibull DFRs for all components. The results are averaged over 10 runs, with each run consisting of only 1000 trials requiring approximately 4 minutes or less of clock time. With only 1000 trials per run, the standard deviations are relatively large. Nevertheless, the outcome of the comparison is clearly apparent.

However, simulation does have an important disadvantage compared to analytical solution techniques. If the accuracy of the evaluation is important, then the execution time required by simulation to achieve the required accuracy increases rapidly and can quickly become uncompetitive with that required by analytical solution techniques (provided the model is small enough for analytical solution techniques to be used). Table 3, which shows the reliability of a single processing node in the hypercube and the execution time required to obtain it, contrasts the values obtained using HARP to values obtained using the simulator with varying numbers of trials per run. Increases in the accuracy of the reliability estimate, as measured by the decreasing size of the standard deviation, require very significant increases in the execution time. The table clearly shows that it is better to use the analytical solver than the simulator, both in terms of execution time and accuracy of the reliability prediction. This result holds in general, and experience has shown that it is usually preferable to use an analytical solver whenever feasible rather than a simulator to evaluate a reliability model. In particular, whenever accuracy in results is important we feel that the use of a simulator generally should be a last resort to be pursued after analytical modeling techniques have been found to be infeasible.

5. SUMMARY

We have described a reliability analysis study which was performed to determine whether assuming of Weibull decreasing failure rates (DFRs) for components of a fault tolerant hypercube would significantly improve the 10 year system reliability estimate over that obtained assuming constant failure rates. Our results show that a substantial improvement in system reliability does result from assuming Weibull DFRs, indicating that a candidate architecture that would otherwise be considered inadequate instead could provide acceptable reliability after all. We also contrasted the use of simulation and analytical solution techniques to evaluate Markovian and non-Markovian reliability models. Observations made from our analysis indicate that analytical solution techniques are preferable whenever the model is small enough and when accuracy of the answer is important. Conversely,

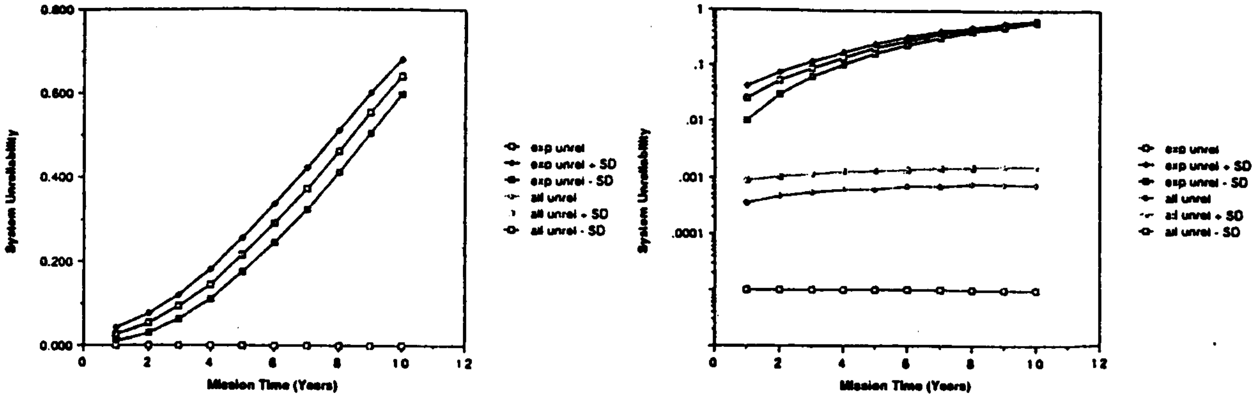

Figure 6: Ballpark Evaluation of the Effect of Weibull DFRs on System Unreliability (Hot Spares)

Solver	Reliability Estimate	CPU time required
HARP	0.04468	6.4 sec
Simulator, 10^3 trials/run	.04374 ± .0023	29.2 sec
Simulator, 10^4 trials/run	.04455 ± .00073	4 min 53.8 sec
Simulator, 10^5 trials/run	.04462 ± .00023	48 min 26.9 sec
Simulator, 10^6 trials/run	.04463 ± .000073	8 hrs 0 min 2.5 sec

Table 3: Processing Node Model Evaluation Accuracy vs. Execution Time

simulation is preferred whenever approximate ballpark answers for a large model are sufficient, or when the model is too large or exhibits system behavior too complex to be accommodated by analytical solution techniques. Finally, we have described a simulator tool for evaluating Markov and non-Markovian reliability models which is compatible with the HARP (analytical) reliability evaluation program and is part of the HiRel package of reliability evaluation tools. There is a great advantage to having analytical and simulation tools be compatible with each other in this way (i.e., both using the same input models and files, and both providing the same analysis capability) because it allows the reliability analyst a great deal of flexibility in conducting the analysis. Solution methods may be mixed and matched and applied in the most appropriate way to a single system model depending on the type and scope of the desired results.

References

[1] Salvatore J. Bavuso and Joanne Bechta Dugan. HiRel: Reliability/availabilty integrated workstation tool. In *Proceedings of the Reliability and Maintainability Symposium*, pages 491–500, January 21-23 1992.

[2] M. A. Boyd and Salvatore J. Bavuso. Modeling a highly reliable fault-tolerant guidance, navigation, and control system for long duration manned spacecraft. In *AIAA/IEEE Digital Avionics Systems Conference, Seattle, WA*, October 1992.

[3] Mark A. Boyd. *Dynamic Fault Tree Models: Techniques for Analysis of Advanced Fault Tolerant Computer Systems.* PhD thesis, Department of Computer Science, Duke University, 1990.

[4] Mark A. Boyd. *What Markov Modeling Can Do For You: An Introduction.* 1993 Reliability and Maintainability Symposium, Tutorial Notes, January 1993.

[5] Mark A. Boyd and Jesus O. Tuazon. Fault tree models for fault tolerant hypercube multiprocessors. In *Proceedings of the Reliability and Maintainability Symposium*, January 1991.

[6] Charles E. Clark. Importance sampling in Monte Carlo analyses. *Operations Research*, 9:603–620, September-October 1961.

[7] W. J. Dally and C. L. Seitz. Deadlock-free message routing in multiprocessor interconnection networks. *IEEE Transactions on Computers*, pages 547–553, May 1987.

[8] Joanne Bechta Dugan, Salvatore Bavuso, and Mark Boyd. Fault trees and sequence dependencies. In *Proceedings of the Reliability and Maintainability Symposium*, pages 286–293, January, 1990.

[9] Joanne Bechta Dugan, K. S. Trivedi, Mark K. Smotherman, and Robert M. Geist. The hybrid automated reliability predictor. *AIAA Journal of Guidance, Control and Dynamics*, 9(3):319–331, May-June 1986.

[10] Herbert Hecht and Eugene Florentino. Reliability assessment of spacecraft electronics. In *Proceedings of the Reliability and Maintainability Symposium*, pages 341–346, January 1987.

[11] M. L. Johnson. Long duration mission reliability via multiprocessing. In *American Astronautical Society, 17th Annual Meeting the Outer Solar System*, June 1971.

[12] Jong Kim and et al. Chita R. Das. Reliability evaluation of hypercube multicomputers. *IEEE Transactions on Reliability, Special Issue on Parallel/Distributed Computing Networks*, 1988.

[13] H. S. Larson and B. O. Shubert. *Probabilistic Models in Engineering Science*, volume II. John Wiley & Sons, NY, 1979.

[14] E. E. Lewis and F. Boehm. Monte Carlo simulation of Markov unreliability models. *Nuclear Engineering and Design*, 77:49–62, 1984.

[15] E. E. Lewis, F. Boehm, C. Kirsch, and B. Kelkhoff. Monte Carlo simulation of copmplex system mission reliability. In *Proceedings of the 1989 Winter Simulation Conference*, pages 497–504, Washington, D.C., December 4-6 1989.

[16] E. E. Lewis and T. Zhuguo. Monte Carlo reliability modeling by inhomogeneous Markov processes. *Reliability Engineering*, 16:277–296, 1986.

[17] M. E. Platt, E. E. Lewis, and F. Boehm. General Monte Carlo reliability simulation cold including common mode failures and HARP Fault/Error-Handling. Technical report, The Technological Institute of Northwestern University, January 1991.

[18] R. Y. Rubinstein. *Simulation and the Monte Carlo Method.* John Wiley & Sons, NY, 1981.

[19] Lei Tien and et al. Chita R. Das. Reliability evaluation of butterfly multiprocessors. In *ACM SIGMETRICS*, 1989.

[20] K. S. Trivedi. *Probability and Statistics with Reliability, Queueing and Computer Science Applications.* Prentice-Hall, Englewood Cliffs, NJ, 1982.

Biographies

Mark A. Boyd
NASA Ames Research Center
Mail Stop 269-4
Moffett Field, CA 94035 USA
(415) 604-3678
FAX: (415) 604-4036

Mark A. Boyd is a research scientist in the Information Sciences Division at NASA Ames Research Center. He was awarded a BA in Chemistry from Duke University in 1979, an MA in Computer Science from Duke University in 1986, and a Ph.D. in Computer Science from Duke University in 1991. His research interests include mathematical modeling of fault tolerant computing systems and the development of dependability modeling tools. He is a member of the IEEE, the ACM, and the MAA.

Salvatore J. Bavuso
NASA Langley Research Center
Mail Stop 478
Hampton, VA 23681 USA
(804) 864-6189
FAX: (804) 864-7891

Salvatore J. Bavuso is a senior researcher at NASA Langley Research Center in Hampton, VA. He received a BS degree in mathematics from the Florida State University in 1964 and a MS degree in applied mathematics from the North Carolina State University at Raleigh in 1971. He has been instrumental in the development of advanced reliability modeling technology for over a decade and is the NASA project manager for the HARP and CAREIII programs.