Modeling a Highly Reliable Fault-Tolerant Guidance, Navigation, and Control System for Long Duration Manned Spacecraft

Mark A. Boyd
NASA Ames Research Center
M/S 269-3
Moffett Field, CA 94035

Salvatore J. Bavuso
NASA Langley Research Center
M/S 478
Hampton, VA 23665-5225

Abstract

We describe the use of a reliability modeling simulation tool to evaluate the reliability of a hypercube multiprocessor which is a candidate architecture for guidance, navigation, and control (G,N,& C) systems for long duration manned spacecraft. Our study focuses on the effect of assuming Weibull decreasing component failure rates compared to the usual assumption of constant component failure rates. We also examine the effect of the use of cold spares on system reliability under the assumptions of both constant and Weibull decreasing failure rates.

Introduction

NASA is currently exploring the feasibility of a manned mission to the Moon or Mars under the Space Exploration Initiative program. A vital element for the success of this mission is the development of a highly reliable guidance, navigation, and control (G,N,& C) system. Many such systems have been researched and developed for short duration aircraft applications[1, 2, 3], but few have addressed the long duration mission. This paper explores the system reliability that can be expected from the use of a hypercube architecture which includes cold and hot spares. A unique aspect of this study is the incorporation of a decreasing failure rate (DFR) model for active and spare modules. Recently acquired empirical data provide convincing evidence that decreasing failure rates are common in spacecraft applications[4]. Some studies that have used the traditional constant failure rate model indicate that a highly reliable long duration G,N,& C system may not be obtainable[5, 6, 7]. The use of the DFR model in conjunction with hot or cold spares may provide evidence that such desired systems may be feasible after all.

The inclusion of cold or hot spares with DFRs 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. An alternate approach involves the use of simulation applied to the non-Markovian model. However, the use of conventional analog simulation techniques to arrive at a reliability prediction for highly reliable long duration missions requires an excessive number of simulation trials to arrive at a confident result. This problem may be addressed through the use of a variance reduction technique called importance sampling, which recently has been receiving increased attention from the reliability modeling community[8, 9, 10]. Recent efforts to develop software tools capable of solving non-Markovian reliability models with this approach now make feasible the investigation of the reliability and performance of systems like the hypercube using cold and hot spares with DFRs that we initially mentioned. The tool which we use for our present study is compatible with the Hybrid Automated Reliability Predictor (HARP) modeling tool[11], which is itself a component of the HiRel package of reliability modeling tools[12].

In this paper we describe the use of a reliability modeling simulation tool to perform a number of analyses intended to explore the effect of differing usage of spares (hot vs. cold) and component failure rate behavior (constant vs. Weibull decreasing) on long duration reliability. The hypercube architecture is being studied at NASA's Jet Propulsion Laboratory for future deep space missions and was selected for this study because of its potential for high reliability and because the selected hypercube architectures have been previously studied by the authors using conventional Markovian models[13]. We begin with a discussion of simulation applied to discrete-state reliability models and the role that importance sampling can play. We then describe the hypercube multiprocessor architecture under study. We next present the results obtained from the models under the differing assumptions about failure rate behavior and comment on the resulting implications for the design of manned spacecraft intended for long duration missions.

Reliability Prediction Using Simulation

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. An alternate approach, which we use for our analysis, 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. We found this to better suit our purposes because it allowed us to capitalize on previous work which had already developed analytical models of the system we wanted to study[13]. We found that these analytical models were too large and involved behavior too complex to be solved by traditional analytical means. So the natural next step was to turn to simulation to perform our analysis. We needed to evaluate both Markovian and non-Markovian discrete-state models, which represent the system in terms of a number of discrete states between which the system makes transitions from time to time.

Simulation of Markovian and non-Markovian Models

Markovian and non-Markovian discrete-state models can be evaluated by simulation in the following way. A sequence of failure events denoting a "trial" is generated which represents a 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 the failure rates of the components still active. 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 [9] and [14]. During each trial, successive interstate 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.

Importance Sampling

Since failure events are extremely rare for highly reliable systems, a large majority of the trials are likely to end by the mission time expiring rather than through a system failure. This means that a very large total number of trials must be run before a sufficient number of system failures occur to provide a meaningful 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 [15]. The basic concept of importance sampling applied to discrete-state reliability models is to force and bias transitions along the rare event paths in an underlying Markovian or non-Markovian model while dynamically maintaining a record of the forcing and biasing that allows post simulation construction of an unbiased estimator of the event of interest, (e.g., system failure) with low variance. The importance sampling techniques implemented in the simulator used for this study are described in [9]. They have the effect of emphasizing component failure events in order to increase the number of trial terminations due to system failure, This reduces the total number of trials needed to accumulate a sufficient number of system failure terminations to provide an acceptable estimate of the system unreliability.

System Model

The hypercube multiprocessor system and it's model used in this study are described in [13] and [16] 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 hot or cold. 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. The message routing protocol restricts messages between each pair of processing nodes in the system to only one path. This restriction is mitigated somewhat by permitting messages to be routed through the spare processing node (even if it has not yet been activated) in a fault-tolerant module if needed to bypass a failed direct link between two active processing nodes in the module. 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*[16]. 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 model is inherent in the simulation process and the dynamic fault tree is used to determine whether a state which has been entered is a failure state. Dynamic fault tree models for the hypercube system appear in [16] and [13].

Analysis Results

We now present the results of our analysis of the hypercube multiprocessor. The mission time was assumed to be 10 years throughout. We are interested in the effect of assuming components having a Weibull DFR instead of the usual constant failure rate that is characteristic of time-homogeneous Markov models. We are also interested in assessing the improvement in system reliability, if any, that can be achieved by using a cold spare processor in the processing nodes instead of a hot spare. Note that the choice of cold vs. hot spare affects power consumption as well as reliability and hence may affect the duration of a feasible mission. Evaluations of the system were made assuming Weibull DFRs for various subsets of components (with remaining components assumed to have constant failure rates). Table 1 gives the constant failure rates λ_{exp} initially assumed for the components in the processing nodes (taken from our previous study of this system[13]). For Weibull failure rates, the failure rate $\lambda_{weib}(t)$ was taken to be equal to the constant failure rate λ_{exp}

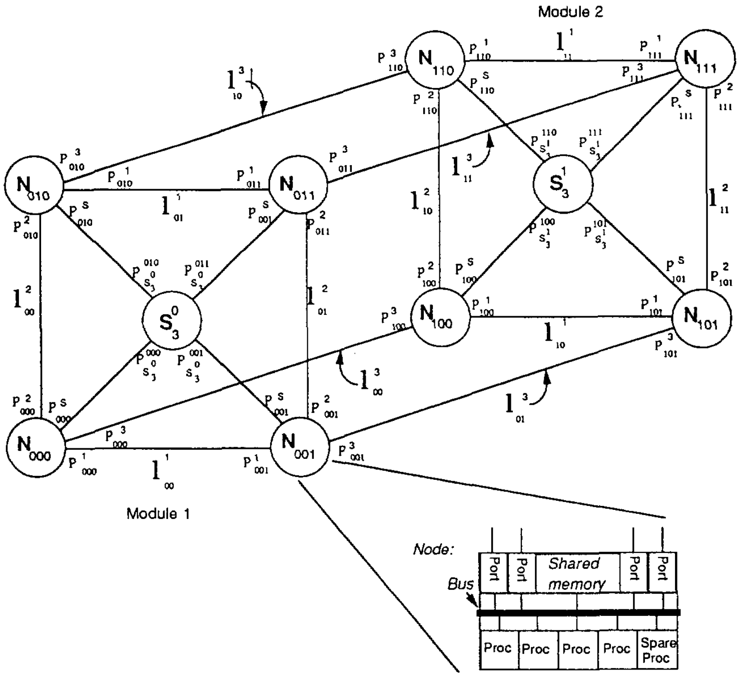

Figure 1: Hypercube Multiprocessor System

at $t = 0$ and assumed to decrease monotonically throughout the mission according to the expression:

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

where α is the Weibull shape parameter[17]. Note that when $\alpha = 1$ the Weibull failure rate reduces to a constant failure rate. When $\alpha < 1$ the Weibull failure rate is monotonically decreasing.

Component	Initial constant failure rate
Shared Memory	3.477×10^{-7}
Intra-node bus	1.147×10^{-7}
Processor	1.990×10^{-6}
Hyperswitch and I/O port	3.477×10^{-7}

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

Effect of Weibull DFRs

Previous studies have indicated that the use of constant failure rates to model highly reliable spacecraft systems did not yield acceptable system reliability for long duration missions of 5 - 10 years[5, 6, 7]. If the assumption of constant component failure rates over such long missions is valid then the conclusion must be that such systems will not be adequate for missions of such long duration. However, Hecht et al.[4] cite evidence that component failure rates for spacecraft on long missions may follow a Weibull DFR model rather than a constant failure rate model. In view of this, we wish to determine whether assuming a Weibull DFR model makes enough of a difference to indicate acceptable system reliability may be obtained from a fault tolerant spacecraft system (specifically, the hypercube system described in the previous section). We used the HARP reliability prediciton program[11] to evaluate the reliability of a single processing node of the hypercube with a hot spare. Figure 2 shows the effect of assuming a Weibull DFR for processors (all other components assumed to have constant failure rates) for varying values of the shape parameter α. The Weibull failure rate is shown to make a very significant difference in the processing node unreliability over the 10 year mission time. Node unreliability drops significantly after only a very modest reduction in α below 1, with most of the unreliability reduction achieved for $0.5 \leq \alpha \leq 1.0$. Hecht et al. found that spacecraft failure rates attributable to parts/quality and operation were consistent with values of α in the range of $0.25 - 0.5$[4]. With this in mind, a value of $\alpha = 0.5$ was selected as a representative value for α for the remainder of the study.

When all processing node components have the constant failure rates given in table 1, the unreliability of the processing node at 10 years is 0.2199. Assuming a Weibull DFR for the processors only with $\lambda = 1.990 \times 10^{-6}$ and shape parameter $\alpha = 0.5$ (i.e. with $\lambda_{weib}(t)$ declining over time from an initial value of 1.990×10^{-6}), the unreliability of a processing node at 10 years falls to 0.04468. Following Chau and Liestman[18], we observe that the expression giving the combinatorial reliability (i.e. considering processing node failures only, not interconnection failures) is: $R_S(t) = [r(t)^4 + 4r(t)^4(1 - r(t))]^2$, where $R_S(t)$ is the sys-

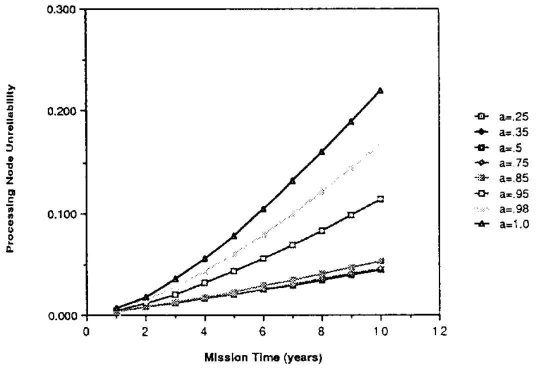

Figure 2: Effect of Processor Weibull DFRs on Processing Node Unreliability

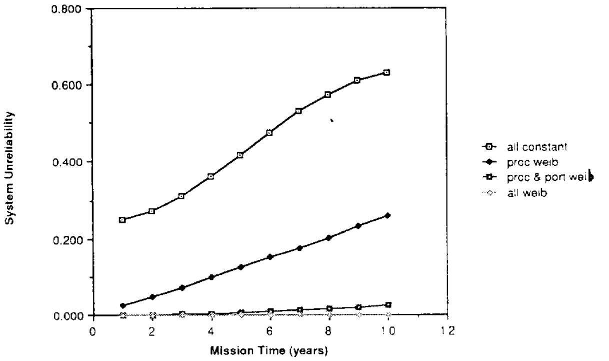

Figure 3: Effect of Weibull DFRs on System Unreliability (Hot Spares)

tem reliability and $r(t)$ is the reliability of the processing node. Using a quick combinatorial calculation with this expression, the above processing node unreliabilities translate into system unreliabilities at 10 years of 0.5155 (constant FR's) compared to 0.03614 (Weibull DFR). This confirms that unacceptable unreliability estimates ($>$ 50% chance of system failure) can be obtained by using a constant failure rate model, whereas an acceptable level of unreliability ($<$ 4%) can be obtained from the same system architecture under the assumption of Weibull DFRs. The values calculated above were confirmed with our simulator program by applying it to a model of the system which considered only processing node failures.

We next considered the effect of Weibull DFRs on the overall system unreliability. To do this, we used our simulator program to evaluate the model of the full system. Table 2 and figure 3 show the effect of assuming Weibull DFRs for various subsets of components. The results reported in table 2 are averaged over 10 runs. 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).

From the data in table 2 we may also see the important effect of the functional dependencies (in the form of interconnection failures) on system unreliability. The unreliability at 10 years when all components have constant failure rates is seen to be 0.631 ± 0.013 which is greater than the value of 0.5155 calculated above accounting for processing node failures only. The difference can be attributed to the functional dependencies: when an interconnection failure occurs, two ports (each on different processing nodes) are rendered unusable and consequently both are considered to be failed. This has the effect of raising the unreliability of the system from that obtained by considering only processing node failures. This effect is even more pronounced when processors are assumed to have a Weibull DFR and the ports are still assumed to have a constant FR: the 10 year unreliability of the full system is in the neighborhood of 0.257 ± 0.036, about a seven-fold increase over the value for the combinatorial calculation of 0.03614. This may be explained by the following reasoning: the processors are by far the major source of failures initially, but their influence on system failure declines dramatically over time due to the Weibull FR's, whereas the ports (with constant FR's) retain the same influence over the system's failure and become the dominant factor near the end of the mission. The functional dependencies involve only the ports and have the effect of increasing the influence of the ports on system reliability. Hence the huge increase in system unreliability of the full system over the combinatorial prediction after a 10 year mission. These results illustrate dramatically an important issue in all systems modeling work which is often easy to overlook: the modeling assumptions used can have a big effect on the results obtained from the model!

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)

Mission Time (Years)	All Components Constant FRs	Processors Weibull DFRs	Processors and Ports Weibull DFRs	All Components Weibull DFRs
1	.0245 ± .0049	.0249 ± .0033	.000526 ± .00021	.000258 ± .00010
2	.0529 ± .0070	.0486 ± .0049	.00140 ± .00029	.000382 ± .00013
3	.0882 ± .0090	.0730 ± .0064	.00282 ± .00041	.000461 ± .00014
4	.134 ± .011	.0978 ± .0078	.00476 ± .00065	.000526 ± .00016
5	.189 ± .012	.124 ± .010	.00705 ± .00087	.000593 ± .00019
6	.255 ± .014	.150 ± .012	.00993 ± .0013	.000660 ± .00022
7	.329 ± .015	.178 ± .015	.0133 ± .0021	.000704 ± .00024
8	.408 ± .015	.202 ± .016	.0170 ± .0026	.000760 ± .00030
9	.490 ± .015	.227 ± .018	.0216 ± .0038	.000781 ± .00031
10	.568 ± .014	.252 ± .020	.0266 ± .0059	.000795 ± .00031

Table 3: Effect of Weibull DFRs on System Unreliability (Cold Spares)

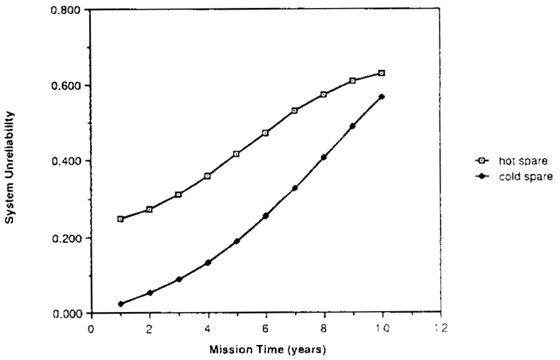

Figure 4: Effect of Hot vs. Cold Spares on System Unreliability (Constant FRs)

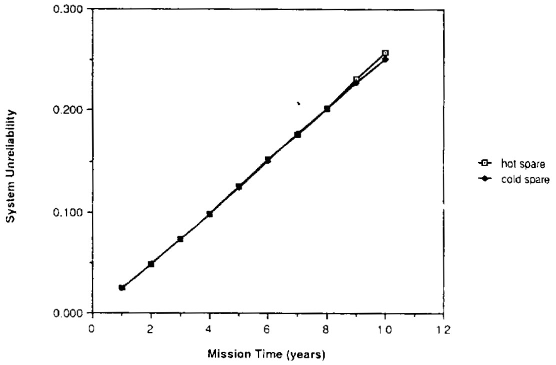

Figure 5: Effect of Hot vs. Cold Spares on System Unreliability (Weibull DFRs for Processors)

Effect of Cold Spares for Processors

We next consider the effect of using a cold spare processor in the hypercube processing nodes. Table 3 shows the effect of assuming Weibull DFRs for various subsets of components. The table shows that the effect of assuming Weibull DFRs for increasing numbers of components is similar to that when hot spares exclusively are used: decreasing system unreliability. We next compare usage of hot spares vs. cold spares for specific combinations of constant and Weibull FRs among the components. When all components have constant FRs, the use of a cold spare processor in the processing nodes instead of a hot spare yields a lower system unreliability as shown in figure 4. This effect holds whenever the components with Weibull DFRs are not the components for which cold spares exist in the system (this general result was observed during our study but we omitted data supporting it in this paper).

One might expect that using a cold spare with Weibull DFRs might be even more effective in lowering the system unreliability. However, an interesting contrary effect is observed in figure 5, which shows the system unreliability for hot vs. cold spare processors when Weibull DFRs are assumed for the processors in the hypercube processing nodes. There is no improvement in system unreliability over using hot spares, and in fact the system unreliability may be even slightly higher than if hot spares are used. This can be accounted for in the following manner: components with Weibull FR's begin with an initial FR equal to the constant FR given in table 1. As the mission progresses the FR of the component decreases steadily from the initial value according to the Weibull failure rate expression in equation 1. For active processors and hot spares, the component FR has its inital value at t = 0 and decays from there throughout the mission. For cold spares, the FR is 0 while the processor is cold, but then takes on the initial value at the time the spare is activated and begins decaying from then on. Hence at a point in time after the spare is brought online, it will have a higher operating FR than the other processors that were operating from mission start (i.e. it has the same Weibull decreasing rate progression as the others, but trails them in time by t_{act}, where t_{act} is the time the cold spare was activated assuming the mission started at time $t = 0$). The higher FR of the activated spare may be enough to negate the benefit of it being cold until needed, and (for appropriate values of alpha) can even outweigh that benefit, resulting in a worse (i.e. larger) system unreliability over the entire mission life. This can be considered counter-intuitive in a sense, and serves as an example of what modeling can do for design engineers to give them insights into the system that would not be initially apparent or expected.

Summary and Conclusion

We have reported on a reliability analysis of a hypercube multiprocessor which is a candidate architecture for a guidance, navigation, and control system for manned spacecraft intended for long duration missions. The analysis was performed using a simulation tool to evaluate homogeneous Markovian, non-homogeneous Markovian, and non-Markovian models of the hypercube. The simulation tool is part of the HiRel package of reliability analysis programs and is compatible with the HARP reliability analysis tool, which was also used in the study. The goal of our study was to determine whether the assumption of Weibull decreasing failure rates for system components, rather than the usual constant failure rates, will lead to estimates of system reliability high enough to indicate adequate system performance for the intended missions. Recent studies employing constant component failure rates suggest that acceptable levels of system reliability could not be acheived. We found that assuming Weibull DFRs for components does indeed lead to dramatically lower system unreliability estimates that should be more than adequate for missions as long as 10 years in duration. We also wanted to determine the extent of improvement in system reliability offered by the use of cold spares over hot spares. Our analysis results show that when constant failure rates are assumed for the components for which cold spares exist in the system, a significant reduction in system unreliability results from the use of cold spares. However, when Weibull DFRs are assumed for components that have cold spares, improvement in system reliability due to the use of cold spares is uncertain. In fact, we observed that the use of cold spares with Weibull DFRs may slightly *increase* the system unreliability if the cold spares truely have Weibull DFRs that behave uniformly beginning when the component is activated (which in the case of a cold spare generally is not the beginning of the mission). This implies that system engineers designing such a system must carefully weigh the benefits of power consumption against potential adverse effects on long term system reliability that the cold spares offer when deciding whether to use hot or cold spares.

References

[1] S. G. Corps, "A320 flight controls," in *The Society of Experimental Test Pilots, Twenty-Ninth Symposium Proceedings*, September 1985.

[2] J. Dannenhoffer, "Development of hardware for the X-29A flight control system," in *NAECON*, vol. 1, pp. 440-445, 1985.

[3] C. J. Walter, "MAFT: An architecture for reliable fly-by wire flight controls," in *Proceedings of the AIAA/IEEE Digital Avionics Systems Conference, San Jose, CA*, October 1988.

[4] H. Hecht and E. Florentino, "Reliability assessment of spacecraft electronics," in *Proceedings of the Reliability and Maintainability Symposium*, pp. 341-346, January 1987.

[5] M. L. Johnson, "Long duration mission reliability via multiprocessing," in *American Astronautical Society, 17th Annaual Meeting the Outer Solar System*, June 1971.

[6] J. Kim and e. a. Chita R. Das, "Reliability evaluation of hypercube multicomputers," *IEEE Transactions on Reliability, Special Issue on Parallel/Distributed Computing Networks*, 1988.

[7] L. Tien and e. a. Chita R. Das, "Reliability evaluation of butterfly multiprocessors," in *ACM SIGMETRICS*, 1989.

[8] A. Goyal, P. Shahabuddin, P. Heidelberger, and V. Nicola, "A unified framework for simulating Markovian models of highly dependable systems," *IEEE Transactions on Computers*, vol. 41, pp. 36-51, January 1992.

[9] E. E. Lewis and F. Boehm, "Monte Carlo simulation of Markov unreliability models," *Nuclear Engineering and Design*, vol. 77, pp. 49-62, 1984.

[10] V. F. Nicola, M. K. Nakayama, P. Heidelberger, and A. Goyal, "Fast simulation of dependability models with general failure, repair, and maintenance processes," in *Proceedings of the Twentieth International Symposium on Fault Tolerant Computing*, pp. 491-498, June 26-28 1990.

[11] J. B. Dugan, K. S. Trivedi, M. K. Smotherman, and R. M. Geist, "The hybrid automated reliability predictor," *AIAA Journal of Guidance, Control and Dynamics*, vol. 9, no. 3, pp. 319-331, May-June 1986.

[12] S. J. Bavuso and J. B. Dugan, "HiRel: Reliability/availabilty integrated workstation tool," in *Proceedings of the Reliability and Maintainability Symposium*, pp. 491-500, January 21-23 1992.

[13] M. A. Boyd, "Fault tree models for fault tolerant hypercube multiprocessors," in *Proceedings of the Reliability and Maintainability Symposium*, January 1991.

[14] E. E. Lewis and T. Zhuguo, "Monte Carlo reliability modeling by inhomogeneous Markov processes," *Reliability Engineering*, vol. 16, pp. 277-296, 1986.

[15] C. E. Clark, "Importance sampling in Monte Carlo analyses," *Operations Research*, vol. 9, pp. 603-620, September-October 1961.

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

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

[18] S. C. Chau and A. Liestman, "Proposal for a fault-tolerant binary hypercube architecture," in *Proc. IEEE Int. Symp. on Fault-Tolerant Computing, FTCS-19*, pp. 323-330, June 1989.