86-2034

THE SURE RELIABILITY ANALYSIS PROGRAM

A86-47423

Ricky W. Butler
NASA Langley Research Center
Hampton, Virginia

Abstract

The SURE program is a new reliability analysis tool for ultrareliable computer system architectures. The program is based on computational methods recently developed for NASA Langley Rusearch Center. These methods provide an efficient means for computing accurate upper and lower bounds for the death state probabilities of a large class of semi-Markov models. Once a semi-Markov model is described using a simple input language, the SURE program automatically computes the upper and lower bounds on the probability of system failure. A parameter of the model can be specified as a variable over a range of values directing the SURE program to perform a sensitivity analysis automatically. This feature, along with the speed of the program, makes it especially useful as a design tool.

Introduction

A reliability analysis of a reconfigurable fault-tolerant computer system adequate for an advanced integrated flight system inevitably requires the determination of the death-state probabilities of a stochastic reliability model. For more than a decade, automated tools (e.g., ARIES, SURF, CARE III, etc.) have been developed to analyze such models. [1] This research has been motivated by the fact that the traditional methods of analysis of redundant system configurations cannot be used to compute the reliability of a reconfigurable system. For example, the traditional fault-tree analysis method cannot compute the probability of failure due to the occurrence of coincident faults (i.e. the arrival of a second fault before the system can remove the first fault). In order to analyze a reconfigurable system the more powerful Markov (or semi-Markov) modeling analysis technique is necessary. Unfortunately, calculation of the probability of system failure using a Markov model requires the solution of a set of coupled differential equations and a semi-Markov reliability model requires the solution of a complex system of convolution integrals. Furthermore, because of the large disparity between the rates of fault arrivals and system recoveries, models of fault-tolerant architectures inevitably lead to numerically stiff integral/ differential equations. This problem, along with the large computational cost of solving large state space problems, have led to the use of decomposition/aggregation techniques in recent reliability analysis tools such as CARE III and HARP. [2] In such programs, the problem is decomposed into a fault-handling model and a fault-occurrence model. Coverage parameters derived from the solution of the fault-handling model are inserted by various aggregation techniques into the fault-occurrence model in order to compute the system reliability. The coverage parameters are computed based on the assumption that critical-pair failures are the dominant failure mode in the system. Unfortunately, such strategies reduce the set of architectures that can be modeled.

Recently, a new mathematical theorem was proved which provides bounds for the probability of entering death states of semi-Markov models (a general class of stochastic models which includes pure Markov models). [3] [4] The upper and lower bounds of this theorem are algebraic functions of simple parameters of the model such as the means and variances of the transitions. This theorem is the basis of a new reliability analysis tool named the Semi-Markov Unreliability Range Evaluator (SURE). The SURE program processes semi-Markov models described in a simple input language, and computes the upper and lower bounds on system unreliability defined in the theorem. Although an exact answer is not produced by the SURE program, the calculated bounds are close together for reliability models of ultrareliable systems - usually within 5 percent of each other. The advantage of the SURE technique is that the bounds are algebraic in form and, consequently, are computationally efficient. Very large and complex models can be analyzed by the program. This is important since future integrated aircraft electronics systems will be much larger and far more complex than the non-integrated systems seen today. Because SURE does not rely on the solution of integral/differential equations, stiffness is not a problem. In fact, the "stiffer" the model, the closer the upper and lower bounds are. Furthermore, the technique applies to a general class of semi-Markov models (i.e. models containing exponential fault arrivals and absorbing death-states) and thus does not impose restrictions on the type of architecture that can be analyzed. A parameter of the model can be specified as a variable over a range of values directing the SURE program to perform a sensitivity analysis automatically.

Since SURE can handle a general distribution of recovery time, the overall fault-handling process of a fault-tolerant computer system can be captured in a single transition. It is unnecessary to assume some underlying parametric form or a special model of fault-handling behavior. The results of experimentation can be directly utilized. However, if desired, detailed fault-handling models can be incorporated into the system reliability model and analyzed by SURE.

The SURE program is currently running under VMS 3.7 on VAX-11/750 and VAX-11/780 computers at the NASA Langley Research Center. The program is implemented in Pascal and has been designed with minimal usage of VMS-specific constructs. An optional graphical display and plotting module is available. This module is written in FORTRAN and uses a special graphics library named TEMPLATE (available only from Megatek Corporation). The SURE program can be installed with or without this module.

Reliability Modeling of Computer System Architecture

Highly reliable systems must use parallel redundancy to achieve their fault tolerance since current manufacturing techniques cannot produce circuitry with adequate reliability. Furthermore, reconfiguration is often utilized in an attempt to increase the reliability of the system without the overhead of even more redundancy. Such systems exhibit behavior that involves both slow and fast processes. When these systems are modeled stochastically, some state transitions are many orders of magnitude faster than others. The slower transitions correspond to fault arrivals in the system. If the states of the system are delineated properly, then the slow transitions can be obtained from field data and/or by using the MIL-STD-217D Handbook calculation. These transitions are assumed to be distributed exponentially. (Electronic component failure is known to follow the exponential distribution very closely after the infant mortality region has passed). [5] The faster transition rates correspond to the system response to fault arrivals and can be measured experimentally using fault injection. (Experiments made by the Charles Stark Draper Laboratory, Inc., on the Fault-Tolerant Multiprocessor (FTMP), computer architecture have demonstrated that these transitions are not exponential). [6]

A semi-Markov model of a triad of processors with one spare is given in figure 1. The outputs of the processors in the triad are voted in order to mask faults. (In this model it is assumed that the spares do not fail while inactive.)

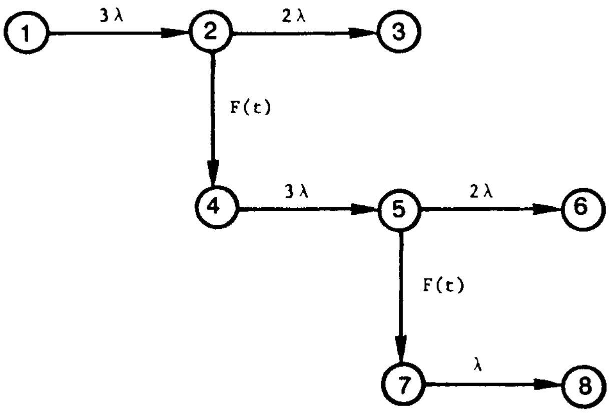

Fig. 1 Model of a triad with one spare.

The horizontal transitions represent fault arrivals. These occur with exponential rate λ. The coefficients of λ represent the number of processors in the configuration that can fail. The vertical transitions represent recovery from a fault. A recovery transition typically is not exponentially distributed and, consequently is described by some general distribution F(t). Since the system uses three-way voting for fault masking, there is a "race" between the occurrence of a second fault and the removal of the first. If the second fault wins the race, then system failure occurs.

Example SURE Session

Probably, the easiest way to learn the SURE input language is by example. The input to the SURE program for the above model is:

```
LAMBDA = 1E-4;
MU = 2.7E-4;
SIGMA = 1.3E-3;

1,2 = 3*LAMBDA;
2,3 = 2*LAMBDA;
2,4 = <MU,SIGMA>;
4,5 = 3*LAMBDA;
5,6 = 2*LAMBDA;
5,7 = <MU,SIGMA>;
7,8 = LAMBDA;
```

The first three statements equate values to identifiers. The first identifier LAMBDA represents the processor failure rate. The next two identifiers MU and SIGMA are the mean and standard deviation of the recovery time. Conveniently, the only information SURE needs about non-exponential recovery processes are the means and standard deviations. The final seven statements define the transitions of the model. If the transition is a slow fault arrival process then only the exponential rate must be provided. For example, the last statement defines a transition from state 7 to state 8 with rate LAMBDA (or 1 x 10^{-4}/ hour). If the transition is a fast recovery process then the mean and standard deviation of the recovery time must be given. For example, the statement 2,4 = <MU,SIGMA> above defines a transition from state 2 to state 4 with mean recovery time MU and standard deviation SIGMA.

The following is an illustrative interactive session using SURE to process the above model. The above model description has been stored in a file named TRIADP1. The user input is underlined.

```
$ SURE

  SURE V4.1     NASA Langley Research Center

  1? READ TRIADP1*;

  2: LAMBDA = 1E-6 TO* 1E-2 BY 10;
  3: MU = 2.7E-4;
  4: SIGMA = 1.3E-3;
  5: 1,2 = 3*LAMBDA;
  6: 2,3 = 2*LAMBDA;
  7: 2,4 = <MU,SIGMA>;
  8: 4,5 = 3*LAMBDA;
  9: 5,6 = 2*LAMBDA;
 10: 5,7 = <MU,SIGMA>;
 11: 7,8 = LAMBDA;
 12: TIME = 10;

 13? RUN;

   LAMBDA        LOWERBOUND     UPPERBOUND
 -----------    -----------    -----------
 1.00000E-06    1.12127E-14    1.77002E-14
 1.00000E-05    2.44035E-12    3.12024E-12
 1.00000E-04    1.56084E-09    1.66224E-09
 1.00000E-03    1.45010E-06    1.51644E-06
 1.00000E-02    1.21116E-03    1.50186E-03

  3 PATH(S) PROCESSED
  0.130 SECS. CPU TIME UTILIZED

15? PLOT XYLOG
```

```
16? EXIT
```

The first statement uses the READ command to input the model description file. It should be noted that LAMBDA is defined as a variable over a range of values in this file. This directs the SURE program to automatically perform a sensitivity analysis as a function of this parameter over the specified range. Statement 12 defines the mission time to be 10 hours. Figure 2 shows the model as displayed on the graphics device after the input file is processed. The SURE program displays all greek-word identifiers (e.g. SIGMA) as a single greek character to make the display more readable. Statement 15 directs the program to plot the output on the graphics device. Figure 3 shows the graph generated by this command. The XYLOG keyword indicates that the X-axis and Y-axis should be logarithmic.

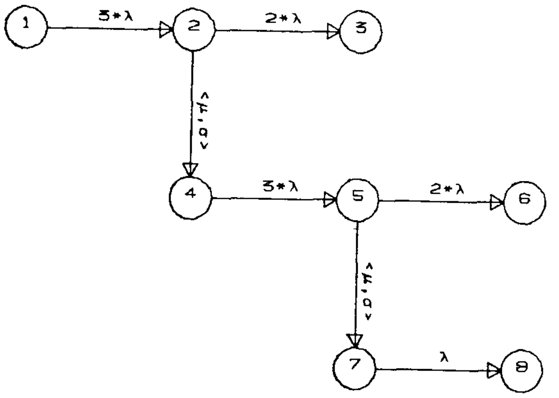

Fig. 2 SURE's graphical display of model.

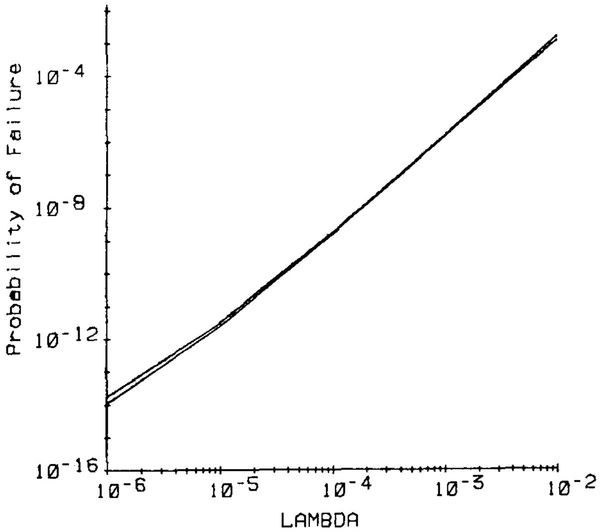

Fig. 3 SURE program's plot of output.

When specifying a transition, a parameter may be defined using arbitrary expressions of the constants and the variable. The standard operators +, -, *, /, ** and the standard functions EXP(X), LN(X) , SIN(X), COS(X), etc. may be used. Both ( ) and [ ] may be used for grouping in the expressions. The following are permissible SURE statements:

```
ALPHA = 1E-4; LAMBDA = 2E-4;
E3A = 1.2*EXP(-3*ALPHA);
1,2 = 7*ALPHA + 12*E3A;
2,3 = ALPHA*(1+LAMBDA) + ALPHA**2;
3,7 = 2*LAMBDA + (1/ALPHA)*[LAMBDA +(1/ALPHA)];
```

The time required to analyze a large model can often be greatly reduced by model pruning. It is essential that this be done carefully in order to maintain accuracy. The SURE user specifies the depth of pruning desired using the PRUNE constant. A path is traversed by the SURE program until the probability of reaching the current point on the path falls below the pruning level. Clearly, the probability of reaching a death state by continuing along this pruned path must be less than the pruning level. The error resulting from this pruning method is therefore less than the product of the number of paths pruned (NPP) and the value of the PRUNE constant. The SURE program will warn the user if this product is great enough to lead to less than a user-specified number of digits accuracy. Typically, the accuracy is far greater than is guaranteed by this test.

Mathematical Basis

The SURE program is based on a new theorem which provides bounds on the probability of entering a death state within a specified time. This theorem must be applied to every path in a semi-Markov model from the start state to the death states. By summing the unreliability due to each path, total system unreliability can be calculated.

Path-step Classification

Once a particular path has been isolated for analysis, each state along the path must first be classified into one of three classes. These classes are distinguished by the type of transitions leaving the state. A state and the transitions leaving it will be referred to as a "path step." The transition on the path currently being analyzed will be referred to as the "on-path transition." The remaining transitions will be referred to as the "off-path transitions." The classification is made on the basis of whether the on-path and off-path transitions are slow (and hence also exponential) or fast. If there are no off-path transitions, the path step is classified as if it contained a slow off-path transition. The classes of path steps along with the information required by the SURE program follows:

Class 1: slow on-path, slow off-path. - All transitions from a state in this class are slow (exponential).

λ_i = the rate of the on-path exponential transition leaving state i.

γ_i = the sum of the off-path exponential transition rates leaving state i.

Class 2: fast on-path, arbitrary off-path. This class includes all states where the on-path transition is fast. There may be an arbitrary number of slow or fast off-path transitions.

ε_i = the sum of the off-path exponential transition rates leaving state i

$F_{i,k}$ = The distribution of time for a fast transition from state i to state k

$\rho(F_{i,k})$ = the probability that the fast transition from state i to state k succeeds over the other fast transitions from state i. (If the competing fast recovery transitions were observed experimentally, this parameter would correspond to the fraction of time that transition i --> k is successful)

$\mu(F_{i,k})$ = the conditional mean transition time from state i to state k given that this transition is successful

$\sigma^2(F_{i,k})$ = the conditional variance of the transition time from state i to state k given that this transition is successful

Class 3: slow on-path, fast off-path. - This class includes path steps where the on-path transition is slow but at least one off-path transition is fast

α_j = the slow on-path transition rate from state j

β_j = the sum of the slow off-path transition rates from state j

$\rho(G_{j,k})$ = the probability that the fast transition from state j to state k succeeds over the other fast wodeQmwmneQ Konj Qwdwa j

$\mu(G_{j,k})$ = the conditional mean transition time from state j to state k given that this transition is successful

$\sigma^2(G_{j,k})$ = the conditional variance of the transition time from state j to state k given that this transition is successful

It should be noted that the parameters $\rho(F_{i,k})$, $\mu(F_{i,k})$, $\sigma^2(F_{i,k})$, $\rho(G_{j,k})$, $\mu(G_{j,k})$, and $\sigma^2(G_{j,k})$ are defined independently of the competing slow exponential transitions. This was done so that experimental measurement of these parameters could be independent of the actual failure rates of the hardware being tested. Consequently, the sum of the fast off-path transition probabilities at each state is 1. In particular, if there is only one recovery transition from a state, the transition probability is 1 and the conditional mean is equivalent to the unconditional mean recovery time. Although, the recovery time distributions are specified without consideration of the competing slow exponential transitions, the bounding theorem gives bounds that are correct in the presence of such exponential transitions.

The Semi-Markov Bounding Theorem

Preliminary Notation - For convenience, when referring to a specific path in the model, an on-path recovery distribution will be indicated by using only one subscript indicating the source state. For example, if the transition with distribution $F_{j,k}$ is the on-path transition, then it can be referred to as F_j:

$F_{j,k}$ = the k^{th} recovery transition from state j

F_j = the on-path recovery transition from state j

The theorem is also expressed in terms of the mean and variance of the "recovery holding time" defined below:

$$\mu(H_j) = \sum_{k=1}^{n_j}\rho(G_{j,k})\ \mu(G_{j,k})$$

$$\sigma^2(H_j) = \{\ \sum_{k=1}^{n_j}\rho(G_{j,k})[\sigma^2(G_{j,k})+\mu^2(G_{j,k})]\}-\mu^2(H_j)$$

Theorem. The probability D(T) of entering a particular death state within the mission time T, following a path with k class 1 path steps, m class 2 path steps, and n class 3 path steps, is bounded as follows:

$$LB \leqq D(T) \leqq UB$$

where

$$UB = E(T)\ \prod_{i=1}^{m}\rho(F_i)\ \prod_{j=1}^{n}\alpha_j\mu(H_j)$$

$$LB = E(T-\Delta)\ \prod_{i=1}^{m}\rho(F_i)\ [1-\ \varepsilon_i\mu(F_i)\ -\ \frac{\mu^2(F_i)+\sigma^2(F_i)}{r_i^2}]$$

$$\prod_{j=1}^{n}\alpha_j\{\mu(H_j)-[(\alpha_j+\beta_j)/2+1/s_j][\mu^2(H_j)+\sigma^2(H_j)]\}$$

and

$$\Delta = r_1 + r_2 + \dots r_m + s_1 + s_2 + \dots + s_n$$

E(T) = the probability of traversing a path consisting of the k class 1 path steps within time T.

for all $r_i > 0$ and $s_j > 0$ such that $\Delta < T$.

The SURE program uses the following values of r_i and s_j:

$$r_i = \mu^{1/2}(F_i)$$

$$s_j = \mu^{1/2}(H_j)$$

Two simple algebraic approximations for E(T) were given by White[3] - one that overestimates and one that underestimates; respectively:

$$E(T) < E_u(T) = \frac{\lambda_1\lambda_2\lambda_3\cdots\lambda_k\ T^k}{k!}$$

$$E(T) > E_l(T) = E_u(T)[1 - T/(k+1) \sum_{i=1}^{k} (\lambda_i + \gamma_i)]$$

Both $E_u(T)$ and $E_\ell(T)$ are close to E(T) as long as $T \sum (\lambda_i + \gamma_i)$ is small. Optionally, the SURE user may specify that a matrix exponential solver be used to calculate E(T) and $E(T - \Delta)$. This is necessary for long mission times where these algebraic bounds separate significantly.

An alternate formulation of the bounding theorem in terms of means and percentiles has been developed.[7] The SURE program also implements these bounds but the details will not be presented in this paper.

Transient and Intermittent Fault Models.

The mathematical theorem on which SURE is based does not directly apply to models that are not pure death processes. The problem with non-pure death process models is that the circuits in the graph structure of the model lead to an infinite sequence of paths of increasing length. However, the longer the path, the less significant is its contribution to the probability of entering a death state. The SURE program automatically unfolds a circuit into a sequence of paths. The truncation point is user-specifiable via the TRUNC command. If TRUNC = 4 then the sequence of paths is terminated after unfolding the loop four times. It is recommended that the user try several values of TRUNC until convergence is certain. Convergence typically occurs in transient fault models after unfolding a circuit two or three times.

Models of systems subject to intermittent faults also contain circuits and thus, like transient faults, lead to a infinite sequence of paths. Computationally, however, the problem is different. Since the circuit in an intermittent model contains only fast transitions, the rate of convergence can be very slow. In fact, the truncation point typically cannot be set to less than 100 unfoldings.

An alternative approach is recommended when convergence is slow. Suppose the intermittent fault oscillates between the benign and active states with rates α and β respectively. If the system's unconditional recovery rate is δ, then it can be shown that the conditional mean μ and variance σ^2 of the recovery time are

$$\mu = \frac{\alpha + \beta}{\beta\delta}$$

$$\sigma^2 = \frac{(\alpha+\beta)^2 + 2\alpha\delta}{\beta^2\delta^2}$$

Using these formulas, the α β loop does not have to be explicitly entered into the model. If the recovery transition is defined using the above mean and variance, the effect of the intermittent is implicitly included.

SURE Session Using Multiple Run Plotting

In this section an example SURE session is presented, where a degradable quadruplex system subject to transient faults is analyzed. The arrival rate of the transient faults, LAMBDA_T, is 10 times the arrival rate of permanent faults, LAMBDA_P. The transients are assumed to disappear with exponential rate BETA. For simplicity, the latency of a transient fault is assumed to be zero. The operating system waits OMEGA units of time before reconfiguring a faulty processor. If the fault disappears prior to this time, it is assumed to be transient and the processor is not removed. The mean and standard deviation of the recovery time in the presence of a permanent fault is assumed to be OMEGA. The variance in this recovery time comes from permanent fault latency. The SURE session follows:

```
$ SURE

  SURE V4.1    NASA Langley Research Center

   1? PLOTINIT BETA
   2? READ QUAD*

   3: LAMBDA_P = 1E-4;
   4: LAMBDA_T = 10*LAMBDA_P;
   5: INPUT BETA;
      BETA? 1E5
   6: OMEGA = 2E-5 TO* 2E-2;
   7: PROBTREC = EXP(-BETA*OMEGA);
   8:
   9: 1,2 = 4*LAMBDA_P;
  10: 2,3 = 3*LAMBDA_P;
  11: 1,4 = 4*LAMBDA_T;
  12: 4,1 = FAST BETA;
  13: 2,5 = <OMEGA, OMEGA>;
  14: 4,5 = < OMEGA, 0.0, PROBTREC >;
  15: 4,8 = 3*LAMBDA_T + 3*LAMBDA_P;
  16: 5,6 = 3*LAMBDA_P;
  17: 6,7 = 2*LAMBDA_P + 2*LAMBDA_T;
  18: 6,10 = <OMEGA, OMEGA>;
  19: 5,9 = 3*LAMBDA_T;
  20: 9,5 = FAST BETA;
  21: 9,10 = < OMEGA, 0.0, PROBTREC >;
  22: 10,11 = LAMBDA_P + LAMBDA_T;
  23: 9,12 = 2*LAMBDA_T + 2*LAMBDA_P;
  24: START = 1;
  25: TIME = 10;
  26: POINTS = 25;
  27: TRUNC=2;

  28? LIST = 0
  29? RUN

  30? PLOT+ XYLOG
  31? ECHO=0;
  32? READ QUAD
```

```
   BETA? 1E6

55? RUN
56? PLOT+ XYLOG
57? EXIT
```

The first statement initializes the SURE program for multiple run plotting. The next command initiates a read of a file named QUAD containing the reliability model. The INPUT statement in the file causes the SURE program to prompt for the value of BETA when the file is read. The model is displayed on the graphics terminal as the file is read (see Fig. 4). The TRUNC=2 command (line 27) specifies that the loops in the model be unfolded two times. The output of the RUN command is suppressed via the LIST=0 command (line 28). The upper and lower bounds are plotted via the PLOT+ XYLOG command (See Fig. 5).

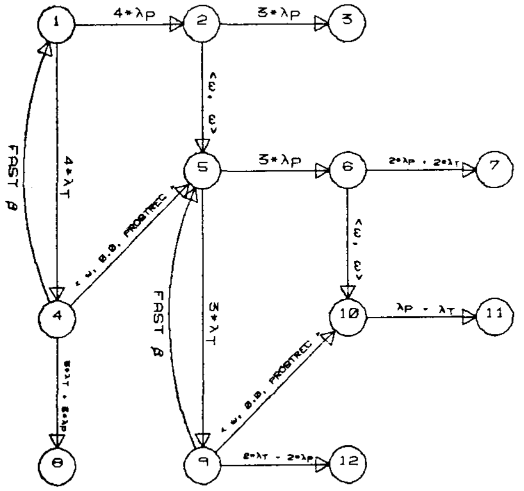

Fig 4. Model display of a quad subject to transient and permanent failure.

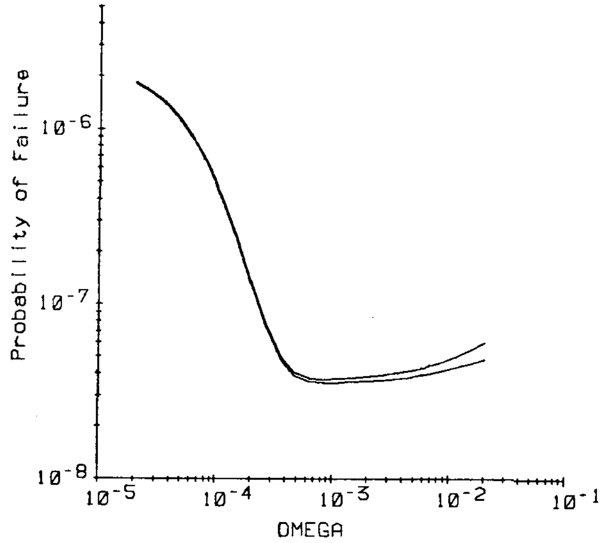

Fig. 5 Plot of system failure probability as a function of OMEGA.

At statement 32, the file is re-read and another value for BETA is specified. Since prior to this command (line 31) ECHO=0 is specified the contents of the file are not echoed back to the terminal. After the RUN command, both sets of run data are plotted on the graphics device simultaneously (see Fig. 6). The plot reveals the effect of different values of OMEGA (the time the operating system waits to see if a fault is transient) and BETA (the rate of disappearance of a transient fault) on the probability of system failure. As expected, if OMEGA is too small, then system reliability decreases because the system incorrectly reconfigures processors with transient faults too often. If OMEGA is too large, system reliability decreases because the system does not reconfigure permanent faults fast enough. The optimal point can be seen to be strongly dependent on BETA. By examining a model in this manner, the SURE program can be used to optimally select the design parameters of a system.

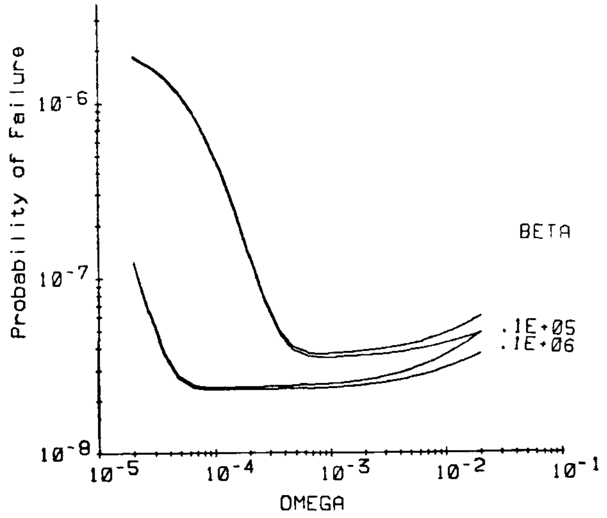

Fig. 6 Plot of system failure probability for two values of BETA.

Concluding Remarks

The SURE program is a flexible, user-friendly, interactive design/validation tool. The program provides a rapid computational capability for semi-Markov models useful in describing the fault-handling behavior of fault-tolerant computer systems. The only modeling restriction imposed by the program is that general recovery transitions must be fast in comparison to the mission time. For systems with recovery times greater than the mission time, the bounds are still correct, but they are not close together. The SURE reliability analysis method utilizes a fast approximation theory developed by Allan L. White of PRC Kentron, Inc., and later generalized by Larry D. Lee of the Langley Research Center and White. This approximation theory enables the calculation of upper and lower bounds on system reliability. These upper and lower bounds are typically within about 5 percent of each other. Since the computation method is extremely fast, large state space models can be analyzed.

Although the approximation theory does not explicitly deal with models that are not pure death processes, the SURE program utilizes simple path truncation strategies to enable the analysis of such models. Consequently, transient and intermittent behavior of fault-tolerant computer systems can be investigated with the SURE program.

References

1. Robert M. Geist and Kishor S. Trivedi: Ultrahigh Reliability Prediction in Fault-Tolerant Computer Systems. IEEE Trans. Comput., C-32, no. 12, Dec. 1983, pp. 1118-1127.

2. Kishor Trivedi; Joanne Bechta Dugan; Robert Dugan; and Mark Smotherman: Modeling Imperfect Coverage in Fault-Tolerant Systems. The Fourteenth International Conference on Fault-Tolerant Computing - FTCS 14, Digest of Papers, 1984, pp. 77-82.

3. Allan L. White: Upper and Lower Bounds for Semi-Markov Reliability Models of Reconfigurable Systems. NASA CR-172340, 1984.

4. Allan L. White: Synthetic Bounds for Semi-Markov Reliability Models. NASA CR-178008, 1985.

5. Daniel P. Siewiorek and Robert S. Swarz: The Theory and Practice of Reliable System Design, Digital Press, 1982, pp. 31-57.

6. Jaynarayan H. Lala and T. Basil Smith, III: Development and Evaluation of a Fault-Tolerant Multiprocessor (FTMP) Computer. Volume III - FTMP Test and Evaluation. NASA CR-166073, 1983.

7. Larry D. Lee: Reliability Bounds for Fault-Tolerant Systems With Competing Responses to Component Failures. NASA TP-2409, 1985.