A state- and time-dependent error occurrence-rate software reliability model with imperfect debugging

by J. G. SHANTHIKUMAR
Syracuse University
Syracuse, New York

ABSTRACT

In this paper, assuming a state- and time-dependent software failure rate and imperfect debuggings, we develop a simple binomial model for software error occurrences. Maximum likelihood estimates for the required parameters of this model are also derived. It is established that the Jelinski-Moranda, imperfect debugging and non-homogeneous Poisson process models are all special cases of ours.

INTRODUCTION

In recent years, several statistical appraoches have been developed to measure and predict software quality. One of such approaches is to postulate a stochastic model, use its results and the data on error occurrences to estimate the model parameters and forecast the future behavior using the model and the estimated parameters.[1-22] In most of these models it is assumed that a software error once detected is perfectly debugged. Recently, in an article[6] in the proceedings of the National Computer Conference, Goel and Okumoto, however, considered a model in which imperfect debugging is allowed. Assuming a fixed number of initial error content and a constant failure rate for each error, they formulated a Semi-Markovian model for the software error occurrences. Using this model they derived expressions for software performance measures. Since these expressions seem complex, they also suggest some approximation.

In this paper, assuming a state- and time-dependent software failure rate and imperfect debuggings, we develop a simple binomial model for software error occurrences. We establish that the Semi-Markovian model developed by Goel and Okumoto[6] is a special case of our binomial model. It is also noted that the Jelinski-Moranda[7] and the Non-homogeneous Poisson Process model[5] are also special cases of our model.

The basic model and the assumptions are presented in section 2. System performance measures are derived in section 3. The parameter estimation is discussed in section 4 and the expressions for performance prediction are developed in section 5. The generality of our model is demonstrated in the appendix.

THE MODEL

The software reliability model developed here is based on the following assumptions.

1. The initial error content at the beginning of the observation phase, that is at time zero, is an unknown constant N.
2. The probability that an error will cause a software failure in a small time interval $(t,\ t+\Delta t)$ is equal to $\phi(t)\Delta t + 0(\Delta t)$, where $\lim_{\Delta t\to 0}\left(\frac{0\Delta t}{\Delta t}\right)=0$. These probabilities for all errors are independent of one another and ~~dependent of time. That is, if there are r errors in the~~ software at time t, the probability of a software failure in $(t,t+\Delta t)$ is $r\phi(t)\Delta t\ +\ 0(\Delta t)$. Note that this assumption, when restricted $\phi(t)$ to be a constant λ, is equivalent to assumption (2) of Goel and Okumoto,[6] page 769.
3. When an error occurs, it is corrected with probability p. That is, with probability $q\ (q=1-p)$, the error is imperfectly debugged (not eliminated).
4. No new errors are created, at most one error is removed at a correction time, and the time taken to correct an error is negligible.

With this set of assumptions we will now formulate our model. We shall do this by considering each error separately. Let us consider an error (out of those N) present in the software at time zero. Let T be the time by which this error is removed from the software. Suppose $F(\cdot)$ is the cumulative distribution function of T. That is $\Pr\{T\leqslant t\}=F(t),\ t>0$. Since

$$\Pr\{T>t+\Delta t\}=\Pr\{T>t \text{ and error is not removed during } (t,t+\Delta t)\},$$

we get

$$\bar{F}(t+\Delta t)=F(t)\{1-p\phi(t)\Delta t\}+0(\Delta t),\qquad(1)$$

where $\bar{F}(t)=1-F(t)\ =\ \Pr\{T>t\}$ and $p\phi(t)\Delta t+0(\Delta t)$ is the probability that the error is removed during $(t,\ t+\Delta t)$.

Now dividing (1) by Δt and taking the limit as $\Delta t \to 0$, we get

$$\frac{d}{dt}\{\bar{F}(t)\} = -p\phi(t)\bar{F}(t),\ t > 0. \tag{2}$$

Since $\Pr\{T > 0\} = 1$, we have the boundary condition $\bar{F}(0) = 1$. Solving (2) with this boundary condition we get

$$\bar{F}(t) = \exp\{-pG(t)\},\ t > 0, \tag{3}$$

where

$$G(t) = \int_0^t \phi(x)dx,\ t \geq 0. \tag{4}$$

Note that $\lim_{t\to\infty} \bar{F}(t)$ need not be zero since the limit $\lim_{t\to\infty} G(t)$ need not be infinity. This means that an error in the software may never be removed. This would then represent the situations in which an error is resident in a part of the code which is never processed or very scarcely processed. In almost all the software reliability models previously considered, it is assumed that all errors would be eventually eliminated. Our results thus represent a more realistic situation. From (3) we also have

$$F(t) = 1 - \exp\{-pG(t)\},\ t > 0. \tag{5}$$

Now let X be the time at which this error causes a software failure for the first time. Then, if $H(t) = \Pr\{X \leq t\}$, $t > 0$, using an analysis similar to the above we can show that

$$\bar{H}(t) = 1 - H(t) = \exp\{-G(t)\},\ t > 0 \tag{6}$$

and

$$H(t) = 1 - \exp\{-G(t)\},\ t > 0, \tag{7}$$

where $G(t)$ is as defined in (4).

With these results (3), (5), and (7), we have binomial distributions for the number of errors remaining at time t, for the number of errors perfectly debugged by time t, and for the number of distinct errors detected by time t with parameter sets $(N,\bar{F}(t))$, $(N, F(t))$, and $(N,H(t))$, respectively.

PERFORMANCE MEASURES

In this section, using results (3)–(7), we will derive expressions for software performance measures that are of interest to us.

Distribution of Number of Remaining Errors

Let $P_{N,n}(t)$ be the probability that there are n errors remaining at time t. From (3) we know that the probability that an error is not perfectly debugged by time t is $\bar{F}(t)$. Then $P_{N,n}(t)$ is binomial with parameters $(N,\bar{F}(t))$. That is,

$$P_{N,n}(t) = \binom{N}{n}(\bar{F}(t))^n(F(t))^{N-n}, n = 0,1,\ldots,N, \tag{8}$$

with mean

$$E(R(t)) = N\bar{F}(t), \tag{9}$$

where $R(t)$ is the number of errors remaining at time t. A software model satisfying conditions given in section 2 with $\phi(t) = \lambda$ a constant, should be identical to the imperfect debugging model of Goel and Okumoto.[6] Even though the results for $P_{N,n_0}(t)$ given there seems different from (8), we establish their equivalence in the appendix.

Distribution of Time to a Completely Debugged System

Let T^* be the time taken to completely debug the system. Define $G_{N,0}(t) = \Pr\{T^* \leq t\}$, $t > 0$. That is, by time T^*, the number of errors remaining should be zero. Then $G_{N,0}(t)$ should be equal to $P_{N,0}(t)$. So, from (8) and (3), we get

$$G_{N,0}(t) = (1 - \exp\{-pG(t)\})^N,\ t > 0. \tag{10}$$

It should be noted that, for reasons discussed earlier, $G_{N,0}(\cdot)$ may be defective. That is, $G_{N,0}(\infty)$ need not be 1.

Distribution of Time to n Remaining Errors

Let T_n^* be the time by which the number of remaining errors is n and define $G_{N,n}(t) = \Pr\{T_n^* \leq t\}$, $t > 0$. Noting that the events $\{R(t) = r,\ r < n\} \equiv \{T_n^* \leq t\}$ we have

$$G_{N,n}(t) = \sum_{r=0}^{n} P_{N,r}(t) = \sum_{r=0}^{n} \binom{N}{r}(\bar{F}(t))^r\,(F(t))^{N-r},$$
$$n = 0,1,\ldots,N. \tag{11}$$

Distribution of Time to Next Software Failure

Suppose there are r software errors remaining just after a recent software failure, say, at time t. Let $Y(r,t)$ be the time to next software failure. Then $Y(r,t)$ is the minimum of the r failure times, each of the r remaining errors. The unconditional cumulative distribution function of these failure times is given by equation (7). Now suppose X_i, $i = 1,2\ldots,r$, are the failure times corresponding to these r errors. Then knowing that $X_i > t$, $i = 1,2,\ldots,r$, we have from (7) and the laws of conditional probabilities,

$$\Pr\{X_i > t + x | X_i > t\} = \exp\{-(G(t+x) - G(t))\},\ x > 0,$$
$$i = 1,2,\ldots,r. \tag{12}$$

Then

$$\Pr\{Y(r,t) > x\} = \Pr\{X_i > t+x, i = 1,2,\ldots,r | X_i > t,$$
$$i = 1,2,\ldots,r\} = \exp\{-r(G(t+x) - G(t))\},\ x > 0. \tag{13}$$

Clearly (13) is the reliability function of the software when there are r errors remaining at time t. Now to use all these expressions, we need the model parameters N, p, and the function $\phi(t)$. We shall attend to this problem in the next section.

PARAMETER ESTIMATION

Suppose we have observed the software failure times caused by each of n errors for the first time. That is, we have the observations of the random variables X_i, $i = 1,2,\ldots,n$ (X_i as defined earlier). Let S_i, $i = 1,2,\ldots,n$, be the values of X_i, $i = 1,2,\ldots,n$ in the increasing order. Then from (13) it is easily verified that,

$$\Pr\{S_k \geqslant t + x | S_{k-1} = t\} = \exp\{-(N-k+1)(G(t+x) - G(t))\},\ x > 0. \quad (14)$$

Now suppose that $f(s_1, s_2,\ldots,s_n)$ is the joint probability density function of $S_1, S_2,\ldots,S_n$. Then from (14), the properties of the model, and the laws of conditional probabilities, we can show that

$$f(s_1,s_2,\ldots,s_n) = \prod_{k=1}^{n}\{(N-k+1)\phi(s_k)\exp\{-(N-k+1)(G(s_k) - G(s_{k-1}))\}\},\ s_i > 0,\ i = 1,2,\ldots,n, \quad (15)$$

where $s_0 = 0$. Then from (15), for a given sequence $s_1,s_2,\ldots,s_n$ of n software failure times caused by n distinct errors for the first time, the log likelihood function L is given by

$$L = \sum_{k=1}^{n} \ln(N-k+1) + \sum_{k=1}^{n} \ln\phi(s_k) - \sum_{k=1}^{n}(N-k+1)(G(S_k) - G(s_{k-1})). \quad (16)$$

To use (16) for parameter estimation, we need specific form of $\phi(t)$. We choose

$$\phi(t) = \alpha b \quad \exp(-bt)\ ,\ t \geqslant 0 \quad (17)$$

following Goel and Okumoto.[5] Note that several other forms may also be chosen for $\phi(t)$. From (4) and (17), we have

$$G(t) = \alpha(1 - \exp(-bt)),\ t \geqslant 0. \quad (18)$$

Using (16) and (18), it can be shown that (see Shanthikumar[20]) the maximum likelihood estimates $\hat{N}$, $\hat{\alpha}$, and $\hat{b}$ of N, α, and b, respectively, are the solution of

$$\sum_{k=1}^{n} \frac{1}{N-k+1} - \alpha(1 - \exp(-bs_n)) = 0 \quad (19)$$

$$\frac{n}{\alpha} - \sum_{k=1}^{n}(N-k+1)(\exp(-bs_{k-1}) - \exp(-bs_k)) = 0 \quad (20)$$

and

$$\frac{n}{b} - \sum_{k=1}^{n} s_k - \sum_{k=1}^{n}(N-k+1)\alpha(s_k\exp(-bs_k) - s_{k-1}\exp(-bs_{k-1})) = 0. \quad (21)$$

These equations (19), (20), and (21) can be numerically solved to obtain these estimates. Next we will look at an estimate for p. Let $u_i = 1,2,\ldots,n$, be the number of time error i (out of the n distinct errors observed) caused software failures during $(0,t_l)$. Then $u_i \geq 1$, $i = 1,2,\ldots,n$. Since the number of times an error causing software failures with imperfect debuggings can be represented by a Geometric random variable (see Shanthikumar[18]) with mean $1/p$, we can approximate $\hat{p}$ by

$$\hat{p} \cong \frac{1}{n}\sum_{i=1}^{n}\frac{1}{u_i}. \quad (22)$$

PERFORMANCE PREDICTION

Now that we have the estimates $\hat{N}$, $\hat{\alpha}$, $\hat{b}$, and $\hat{p}$, we can use equations (8)–(13) for performance prediction. We should note, however, that we have made some observations to estimate the parameters and therefore equations (8)–(13) have to be modified. This is because the current state of the software system has changed. This aspect has been ignored in an earlier paper.[6] Suppose we have observed n distinct errors during the time period $(0,t_l)$. Also, suppose we have observed a total of l software failures during this time period. Then $l \geq n$ since some or all of these n errors may have been imperfectly debugged. Hence if R_l is the number of remaining errors at time t_l, R_l is neither equal to $N - n$ nor equal to $N - l$. In fact, the exact value of R_l is unknown because of imperfect debuggings. If $R_l = r$, then equations (8)–(13) may be used when the time origin shifted to t_l and N replaced by r. Specifically, from (8),

$$P_{r,k}(t|R_l = r) = \binom{r}{k}(\bar{F}^*(t))^k(F^*(t))^{r-k},\quad k = 0,1,\ldots,r,\ t > t_l, \quad (23)$$

is the probability that there will be k errors remaining at time t given that there are r errors remaining at time t_l. In this equation (see (4), (6), and (7))

$$F^*(t) = 1 - \exp\{-pG^*(t)\},\ t > t_l$$

$$\bar{F}^*(t) = 1 - F^*(t)$$

and

$$G^*(t) = \int_{t_l}^{t}\phi(x)dx = G(t) - G(t_l),\ t \geq t_l$$

due to the shift in time origin.

Similarly, from (10),

$$G_{r,0}(t|R_l = r) = (1 - \exp\{-p(G(t_l))\})^r,\ T > t_l \quad (24)$$

from (11),

$$G_{r,k}(t|R_l = r) = \sum_{j=0}^{k}\binom{r}{j}(\bar{F}^*(t))^j(F^*(t))^{r-j},\quad k = 0,1,\ldots,r,\ t \geq t_l \quad (25)$$

and from (13), the reliability function

$$\Pr\{Y(r,t_l) > x\} = \exp\{-r(G(t_l + x) - G(t_l)\}, x > 0. \quad (26)$$

So, in order to use the above equations, we need an estimate for the number of errors remaining at time t_l. As mentioned earlier, the exact value of R_l is unknown and therefore we may develop a probability distribution for it.

Let z_i be the last time an error $i, i = 1,2,\ldots,n$, caused a software failure before time t_l. That is, this error did not reappear during (z_i, t_l). If the error is imperfectly debugged, the probability that it will not show up during (z_i, t_l) is from (7), equal to $\exp\{-(G(t_l) - G(z_i))\}$, $i = 1,2,\ldots,n$. This probability, when the error is perfectly debugged, is 1. Noting that the probability of perfect debugging is p, and using Bayes rule, we get

$$\beta_i = p / \{p + (1-p)\exp\{-(G(t_l) - G(z_i))\}\}, \; i = 1,2,\ldots,n,$$

where β_i is the probability that an error i (one of the n distinct errors) detected is perfectly debugged given that this error did not reappear during (z_i, t_l). Then

$$\Pr\{R_l = N - n + r\} = \sum_{A \in M_r} \prod_{i \in A'} \beta_i \prod_{i \in A} (1 - \beta i), \quad r = 0,1,\ldots,n, \quad (27)$$

where M_r is the set of all possible subsets of $\{1,2,\ldots,n\}$ with cardinality r, and A' is the complement of A. That is, $A' = \{1,2,\ldots,n\} - A$. It can be verified, after some algebra, that

$$E(R_l) = N - \sum_{i=1}^{n} \beta_i \triangleq \hat{r}. \quad (28)$$

Now either $\hat{r}$ can be used as an estimate for the number of remaining errors or use (27) along with (23)–(26) to predict software performance. That is, from (23) and (27),

$$P_{R_l,k}(t) = \sum_{r=k}^{N} \Pr\{R_l = r\} P_{r,k}(t | R_l = r), \; k = N-n,\ldots,N, \quad (29)$$

from (24) and (27),

$$G_{R_l,0}(t) = \sum_{r=N-n}^{N} \Pr\{R_l = r\} G_{r,0}(t | R_l = r), \; t > t_l, \quad (30)$$

from (25) and (27),

$$G_{R_l,k}(t) = \sum_{r=k}^{N} \Pr\{R_l = r\} G_{r,k}(t | R_l = r), \; k = N-n,\ldots,N, \quad (31)$$

and the reliability function from (26) and (27) is

$$\Pr\{Y(R_l, t_l) > x\} = \sum_{r=N-n}^{N} \Pr\{R_l = r\} \Pr\{Y(r, t_l) > x\}. \quad (32)$$

Note that the above expression for the reliability function corrects an error in equations (23), (24), and (25) of Goel and Okumoto (1979)[6] (see Shanthikumar,[17] page 71). Since evaluation of equation (27) is of combinatorial nature, a simple binomial approximation is proposed based on $E(R_l) = \hat{r}$. It is

$$\Pr\{R_l = N - n + r\} \cong \binom{n}{r} \bar{\beta}^{n-r} (1 - \bar{\beta})^r, \; r = 0,1,\ldots,n, \quad (33)$$

where $\bar{\beta} = \frac{1}{n} \sum_{i=1}^{n} \beta_i$ so that $E(R_l) = N - n\bar{\beta} = \hat{r}$ is preserved. Now equations (23)–(26), (29)–(32), and (33) can be used for performance prediction.

CONCLUSION

In this paper, assuming a state- and time-dependent software failure rate and imperfect debuggings, we developed a simple binomial model for software error occurrences. Maximum likelihood estimates for the required perameters of this model are also derived. For this we use $\phi(t) = \alpha b \exp(-bt)$, $t > 0$ for the time-dependent failure rate function. In the appendix it is established that the imperfect debugging model of Goel and Okumoto[6] is a special case of this model (specifically when $\phi(t) = \lambda$ is a constant. Then, obviously, when $\phi(t) = \lambda$ and the probability p of perfect debugging is equal to one, we will get the results for the Jelinski-Moranda model.[7] It can also be shown (see Shanthikumar[20]) that when $p = 1$, $N \to \infty$, $\alpha \to 0$, and $N\alpha \to \alpha < \infty$, the above model reduces to the non-homogeneous Poisson process model discussed by Schneidewind[16] and Goel and Okumoto.[5] Because of this generality of this model, it is expected that this model will prove to be versatile.

APPENDIX

In this appendix we will systematically transfer Goel and Okumoto's results[6] to match a special case of our results. The special case considered here is $\phi(t) = \lambda$, $t \geq 0$. Then $G(t) = \lambda t$, $t \geq 0$ and $\bar{F}(t) = \exp(-\lambda p t)$, $t > 0$.

Distribution of Time to a Completely Debugged Software System

The distribution $G_{N,0}(t)$ of time to a completely debugged system is given by equation (12) in Goel and Okumoto's "A Markovian Model for Reliability and Other Performance Measures of Software Systems.[6] It is

$$G_{N,0}(t) = \sum_{j=1}^{N} C_{N,j} (1 - e^{-jp\lambda t}), \; t \geq 0 \quad (A1)$$

where

$$C_{N,j} = \binom{N}{j} (-1)^{j-1} \quad (A2)$$

Substituting (A2) in (A1), we get

$$G_{N,0}(t) = \sum_{j=1}^{N} -\binom{N}{j}(-1)^j + \binom{N}{j}(-e^{-p\lambda t})^j = (1 - e^{-p\lambda t})^N \quad (A3)$$

is obtained using the combinatorial identity

$$\sum_{j=0}^{N} \binom{N}{j} f^j = (1 + f)^N$$

Note that (A3) agrees with (10) when $G(t)=\lambda t$.

Distribution of Time to a Specified Number of Remaining Errors

The distribution $G_{N,n_0}(t)$ of time to n_0 number of remaining errors is given by equation (14) of "A Markovian Model." It is

$$G_{N,n_0}(t)=\sum_{j=1}^{N-n_0} B_{N,j,n_0}\{1-e^{-(n_0+j)p\lambda t}\},\ t\geq 0 \qquad \text{(A4)}$$

where

$$B_{N,j,n_0}=\frac{N!}{n_0!j!(N-n_0-j)!}(-1)^{j-1}\frac{j}{n_0+j} \qquad \text{(A5)}$$

Rewriting (A5) we get

$$\begin{aligned} B_{N,j,n_0} &= \binom{N}{j+n_0}\binom{n_0+j-1}{j-1}(-1)^{j-1} \\ &= \binom{N}{j+n_0}(-1)^{j-1}\sum_{r=1}^{j}\binom{j+n_0}{j-r}(-1)^{r-1} \qquad (\dagger) \\ &= \sum_{r=1}^{j}\binom{N}{j+n_0}\binom{j+n_0}{j-r}(-1)^{j+r} \\ &= \sum_{r=1}^{j}\binom{N}{r+n_0}\binom{N-r-n_0}{j-r}(-1)^{j+r} \\ &= \sum_{n=n_0+1}^{j+n_0}\binom{N}{n}\binom{N-n}{j-n+n_0}(-1)^{j+n-n_0} \qquad \text{(A6)} \end{aligned}$$

Now substituting (A6) in (A4) and interchanging the order of summations, we get

$$\begin{aligned} G_{N,n_0}(t) &= \sum_{n=n_0+1}^{N}\sum_{l=0}^{N-n}\binom{N}{n}\binom{N-n}{l}\{(-1)^l+(-e^{-p\lambda t})^l e^{-np\lambda t}\} \\ &= 1-\sum_{n=n_0+1}^{N}\binom{N}{n}e^{-np\lambda t}(1-e^{-p\lambda t})^{N-n} \\ &= \sum_{n=0}^{N_0}\binom{N}{n}(e^{-p\lambda t})^n(1-e^{-p\lambda t})^{N-n} \qquad \text{(A7)} \end{aligned}$$

Note that (A7) agrees with (11), when $\bar{F}(t)=\exp(-\lambda pt)$.

Distribution of Number of Remaining Errors

The distribution of $P_{N,n_0}(t)$ of n_0 remaining errors at time t is given by equation (17) of "A Markovian Model." It is

$$P_{N,n_0}(t)=G_{N,n_0}(t)-G_{N,n_0-1}(t) \qquad \text{(A8)}$$

Substituting (A7) in (A8) we get

$$P_{N,n_0}(t)=\binom{N}{n_0}(e^{-p\lambda t})^{n_0}(1-e^{-p\lambda t})^{N-n_0} \qquad \text{(A9)}$$

a binomial distribution. Equation (A9) agrees with equation (8), $\bar{F}(t)=\exp(-\lambda pt)$.

†This is obtained using the combinatorial identity (1.5) in page 1 of Gould, H.W., *Combinatorial Identities*, Morgantown Printing and Binding Co., (1972).

REFERENCES

1. Angus, J.E., R.E. Schafer, and A. Sukert, "Software Reliability Model Validation," *Proc. of Annual Reliability and Maintainability Symposium* (1980), pp. 191-193.
2. Basin, S.L., *Estimation of Software Error Rate Via Capture-Recapture Sampling,* Science Applications, Inc., Palo Alto, California (1974).
3. Endres, A., "An Analysis of Errors and Their Causes in System Programs," *Proc. of the 1975 International Conference on Reliable Software* (1975), pp. 327-336.
4. Forman, E.H. and N.D. Singpurwalla, "An Empirical Stopping Rule for Debugging and Testing Computer Software," *J. of American Statistical Association,* Vol. 72 (1977), pp. 750-757.
5. Goel, A.L. and K. Okumoto, "Time-Dependent Error-Detection Rate Model for Software Reliability and Other Performance Measures," *IEEE Transactions on Reliability,* Vol. 28 (1979), pp. 206-211.
6. Goel, A.L. and K. Okumoto, "A Markovian Model for Reliability and Other Performance Measures of Software Systems," *Proc. of the National Computer Conference* (1979), pp. 769-774.
7. Jelinski, Z. and P. Moranda, "Software Reliability Research," *Statistical Computer Performance Evaluation,* W. Freiberger (Ed.), Academic Press (1972), pp. 465-484.
8. Littlewood, B. and J.L. Verrall, "A Bayesian Reliability Growth Model for Computer Software, *Applied Statistics,* Vol. 22, (1973), pp. 332-246.
9. Littlewood, B., "A Reliability Model for Systems With Markov Structure," *Applied Statistics,* Vol. 24 (1975), pp. 172-177.
10. Miyamoto, I., "Software Reliability in On-Line Real Time Environment," *Proc. of the 1975 International Conference on Reliable Software* (1975), pp. 194-203.
11. Moranda, P., "Prediction of Software Reliability During Debugging," *Proc. of the Annual Reliability and Maintainability Symposium* (1975), pp. 327-332.
12. Moranda, P., "Error Detection Models for Application During Program Development," *Proc. of the Nineteenth Annual Technical Symposium—Pathways of System Integrity* (1980), pp. 75-78.
13. Musa, J.D. "A Theory of Software Reliability and Its Application," *IEEE Transactions on Software Engineering* (1975), pp. 312-327.
14. Schick, G.J. and R.W. Wolverton, "Assessment of Software Reliability," *Proc. Operations Research,* Physica-Verlag, Wurzburg-Wien (1973), pp. 395-422.
15. Schick, G.J. and R.W. Wolverton, "An Analysis of Competing Software Reliability Models," *IEEE Transactions on Software Engineering* (1978), pp. 104-120.
16. Schneidewind, N.J., "Analysis of Error Process in Computer Software," *Proc. of the 1975 International Conference on Reliable Software* (1975), pp. 337-346.
17. Shanthikumar, J.G., "Software Performance Prediction Using a State-Department Error Occurrence-Rate Model," *Proc. of the Nineteenth Annual Technical Symposium—Pathways to System Integrity* (1980), pp. 67-72.
18. Shanthikumar, J.G., "A Binomial Model for Software Performance Prediction," *Proc. of the Eighteenth Annual Allerton Conference on Communication, Control, and Computing,* (1980), to appear.
19. Shanthikumar, J.G. and S. Tufekci, "Optimal Software Release Time Using Generalized Decision Trees," *Proc. of the Fourteenth Annual Hawaii International Conference on System Sciences* (1981), to appear.
20. Shanthikumar, J.G., "A General Software Reliability Model for Performance Prediction," Technical Report, Dept. of Ind. Eng. & Opns. Res., Syracuse University (1980), p. 18.
21. Shooman, M.L., "Software Reliability: Measurement and Models," *Proc. of the Annual Reliability and Maintainability Symposium* (1975), pp. 485-491.
22. Trivedi, A.K. and M.L. Shooman, "A Many-State Markov Model for the Estimation and Prediction of Computer Software Performance Parameters," *Proc. of the 1975 International Conference on Reliable Software* (1975), pp. 208-220.