MARKOVIAN SOFTWARE RELIABILITY MODEL FOR TWO TYPES OF FAILURES WITH IMPERFECT DEBUGGING RATE AND GENERATION OF ERRORS

M. Jain
Department of Mathematics, IIT, Roorkee- 247667 (India)
madhufma@iitr.ernet.in, drmadhujain@yahoo.co.in

S. C. Agrawal, P. Agarwal*
School of Basic and Applied Sciences, Shobhit University, Meerut-250001, (India)
sagrawal7@rediffmail.com, priyanka1354@gmail.com

*Corresponding Author
(Received: November 25, 2010 – Accepted in Revised Form: January 19, 2012)

Abstract

This investigation deals with a software reliability model based on Markov process. For formulating the model, we define a random variable representing the cumulative number of faults successfully corrected up to a specified point of time. This model is based on the assumption that there are two types of software failures. Further the concepts of imperfect debugging environment and error generation phenomenon are taken into consideration. Transient analysis based on Laplace transform and matrix approach has been done to find the solution of the system of differential difference equations. Several performance indices for software reliability assessment are derived for this model. Numerical results with the help of Runge-Kutta Method show that the proposed framework incorporating both concepts of imperfect debugging phenomenon and error generation for two types of faults has a fairly accurate prediction capability.

Keywords
Software reliability; Imperfect debugging; Error generation; Markov process; Software reliability growth.

1. INTRODUCTION

Software engineers generally need a period of time to read, and analyze the collected software failure data. Software reliability models based on stochastic process have gained wide acceptance in the software industry because they are useful engineering tools to analyze and to correct the faults. Software reliability estimates are generally made by building probability models of data collected during testing. The extent to which software reliability is influenced by factors that can be controlled by the software developer is not obvious. We seek a more fundamental approach that directly relates influence factors to the stochastic model’s parameters. Software reliability growth models have high validity and usefulness for software development or operation phase. Markov models have been proved to be very useful in many practical applications and there are several

In the case of software reliability, two major issues confound to estimate software reliability, i.e. imperfect debugging and error generation. This concept in a software reliability modeling is a controversial issue. No real world software company possesses infinite resources to test and correct every software fault in the real world. However, a Markov model approach in this regard may be worthwhile and useful to deal with such a situation.

There are several reasons for using NHPP models or hazard rate models in a specific situation. Generally, models are developed for the analysis of failure data collected during the testing stage then we develop our model analytically with the help of NHPP models or hazard rate models. These group of models provides an analytical framework for describing the software failure phenomenon during testing phase and with the help of these group of models we can estimate the future behavior of the software system.

One common assumption of conventional software reliability modeling is that the detected faults are immediately removed; but this assumption may not be realistic in actual software development. In reality, most latent faults may remain uncorrected for a very long time, even after they are detected by professional testers, which increase their impact. Kapur et al. [12] obtained the transient solution of a software reliability model with imperfect debugging and error generation. Yamada et al. [28] did the software reliability measurement in imperfect debugging environment and also discussed its application. Gokhale et al. [6] developed a non-homogeneous markov software reliability model with imperfect repair. Sridharan and Jayashree [21] considered a transient solution of a software reliability model with imperfect debugging and generation of errors by two servers. Tokuno and Yamada [23] studied a markovian software reliability model with a decreasing perfect debugging rate. Tokuno and Yamada [24] established the imperfect debugging model with two types of hazard rates for software reliability measurement and assessment. Tokuno and Yamada [25] suggested the markovian software reliability measurement with geometrically decreasing perfect debugging rate. Tokuno and Yamada [26] proposed the relationship between software availability measurement and the number of restorations with imperfect debugging. Gupta and Singh [7] estimated software reliability by sequential testing with simulated annealing of mean field approximation. Mathematical modeling of software reliability testing with imperfect debugging was discussed by Cai et al. [1]. Meedeniya et al. [16] derived reliability deployment optimization technique for embedded systems.

Most specifically, we develop a broader class of SRGMs for two types of faults at each discrete time point with imperfect debugging and error generation behavior. This paper is organized as follows: Section 2 provides the description of Markov model along with assumptions and notations. In section 2.1 and 2.2, we construct the governing equations of the proposed model and specifically for an illustration, respectively. Solution approach including Laplace transform and matrix method is described in section 3. Section 4 is devoted to the performance measures. Numerical
results are obtained in section 5 with the help of R-K and matrix method. To explore the effects of different parameter, the sensitivity analysis is also carried out. Finally conclusions are drawn in section 6.

2. MODEL DESCRIPTION

A stochastic model is sought that represents the injection (due to the occurrence of development and debugging errors) and removal (due to successful repairs) of software. The stochastic behavior of the fault correction phenomenon with imperfect debugging is described by a Markov process. In this investigation, we assume that there exist two types of faults in the software such as
(i) Due to originally latent in the system before the testing.
(ii) Due to generated during the testing phase and the error generation phenomenon never leads the software to having infinite errors.

We develop a software model by making the following assumptions:

- The software has a finite number of two types of faults; there are m faults of type I whereas n faults of type II.
- The probability that two or more software failures occur simultaneously is negligible.
- The failure rate is proportional to the number of fault remaining in the software.
- The debugging process is performed as soon as the software failure occurs.
- When the debugging process is performed, at most one fault is corrected and the fault correction time is considered negligible.
- The maximum number of faults in the software never exceeds a finite limit, i.e., m<M, n<N.
- When a failure occurs, an instantaneous repair effort starts and the following cases arise:
  (i). The fault content function is reduced by one for type I faults with probability p0.
  (ii). The fault content function remains unchanged for type I faults with probability p1.
  (iii). The fault content function is increased by one for type I faults with probability p2.
  (iv). The fault content function is reduced by one for type II faults with probability q0.

(v). The fault content function remains unchanged for type II of faults with probability q1.
(vi). The fault content function is increased by one for type II faults with probability q2.

The debugging activity is performed without distinguishing between both types of faults.

Notations

\( \alpha \) : Failure rate per remaining software fault for type I faults.
\( \beta \) : Failure rate per remaining software fault for type II faults.
\( m \) : Initial fault content for I type of faults.
\( M \) : Maximum fault content for I type of faults.
\( n \) : Initial fault content for the II type of faults.
\( N \) : Maximum fault content for II type of faults.
\( F_{ij}(t) \): Probability that there are \( i \) faults of type I and \( j \) faults of type II in the software at time \( t \).

where \( 0 \leq i \leq M \) and \( 0 \leq j \leq N \)

2.1. Governing Equations

The transient equations governing the model are constructed by considering the transition flow rates. The differential difference equations associated with the various states are as follows (for balance equations are based on transition diagram in Fig 1):

\[
F_{00}’(t) = \alpha p_0 F_{10}(t) + \beta q_0 F_{01}(t)
\]

(1)

\[
F_{0i}’(t) = -(ia - i\alpha p_0) F_{i0}(t) + (i-1) \alpha p_0 F_{i-1,0}(t) + (i+1) \alpha p_0 F_{i+1,0}(t) + \beta q_i F_{i,0}(t), \quad 2 \leq i \leq M - 1
\]

(2)

\[
F_{i0}’(t) = -\alpha p_i F_{i0} - \alpha p_2 F_{i0} + 2\alpha p_i F_{i-1,0}(t) + \beta q_i F_{i,0}(t), \quad i = 1
\]

(3)

\[
F’_{M0}(t) = -M \alpha p_0 F_{M0}(t) + (M-1) \alpha p_0 F_{M-1,0}(t)
\]

(4)

\[
F’_{ij}(t) = -\beta q_i F_{ij}(t) - \beta q_j F_{ij}(t) + \alpha p_i F_{i-1,j}(t) + 2\beta q_i F_{i,j+1}(t), \quad j = 1
\]

(5)

\[
F’_{ij}(t) = -(j\beta - j\beta q_i) F_{ij}(t) + (j-1) \beta q_j F_{i,j-1}(t) + \alpha p_j F_{i,j+1}(t), \quad 2 \leq j \leq N - 1
\]

(6)

\[
F’_{00}(t) = -N \beta q_0 F_{00}(t) + (N-1) \beta q_0 F_{00,N-1}(t)
\]

(7)

\[
F’_{i0}(t) = -[(\beta - \beta q_i) + (ia - i\alpha p_0)] F_{i0}(t) + \alpha p_2 F_{i0}(t) + (i+1) \alpha p_i F_{i+1,0}(t) + i \beta q_i F_{i0}(t), \quad 2 \leq i = M - 2
\]

(8)
In this section, for illustration purpose we present a markov model for two types of errors where maximum number of faults of each type are five (i.e. M=N=5). The differential equations governing the system are as follows:

\[ F_i'(t) = -(\alpha p_i + \beta q_i + \alpha p_i + \beta q_i) F_i(t) + 2\alpha p_i F_{i+1}(t) + 2\beta q_i F_{i-1}(t), \quad i = 1 \]  
\[ F_i'(t) = -(\beta q_i + (M-1) \alpha p_i) F_i(t) + (M-2) \alpha p_i F_{i+2}(t), \quad i = 0 \]  
\[ F_i'(t) = -[(\alpha - \alpha p_i) + (j \beta - j \beta q_i)] F_i(t) + (j-1) \beta q_i F_{i+1}(t) + 2\alpha p_i F_{i+2}(t) + (j + 1) \beta q_i F_{i+3}(t), \quad 2 \leq j \leq N-2 \]  
\[ F_i'(t) = -(\alpha p_i + (N-1) \beta q_i) F_i(N-i,t) + (N-2) \beta q_i F_i(N-i-2,t) \]  
\[ F_i'(t) = -[(ia - i\alpha p_i) + (j\beta - j\beta q_i)] F_i(t) + (i-1) \alpha p_i F_{i+1}(t) + (j-1) \beta q_i F_{i+1}(t) + (i+1) \alpha p_i F_{i+1}(t) + (j + 1) \beta q_i F_{i+1}(t), \quad i = M-3, j = N-2 \]  
\[ F_i'(t) = -i \alpha p_i F_i(t) - (i+1) \beta q_i F_i(t) + i \beta q_i F_{i+1}(t) + (i-1) \alpha p_i F_{i+1}(t), \quad i = M-2, j = N-3 \]

2.2. Illustration

In this section, for illustration purpose we present a Markov model for two types of errors where maximum number of faults of each type are five (i.e., M = N = 5). The differential equations related to this particular case are as follows:

\[ F_{00}'(t) = \alpha p_0 F_{10}(t) + \beta q_0 F_{01}(t) \]  
\[ F_{01}'(t) = -\alpha p_0 F_{00}(t) + 2\alpha p_0 F_{20}(t) + \beta q_0 F_{11}(t) \]  
\[ F_{00}'(t) = -(2\alpha - 2\alpha p_1) F_{00}(t) + \alpha p_2 F_{10}(t) + 3\alpha p_0 F_{00}(t) + \beta q_2 F_{21}(t) \]  
\[ F_{00}'(t) = -(3\alpha - 3\alpha p_1) F_{10}(t) + 2\alpha p_2 F_{20}(t) + 4\alpha p_0 F_{00}(t) + \beta q_3 F_{31}(t) \]  
\[ F_{00}'(t) = -(4\alpha - 4\alpha p_1) F_{20}(t) + 3\alpha p_2 F_{30}(t) + 5\alpha p_0 F_{00}(t) + \beta q_4 F_{41}(t) \]  
\[ F_{00}'(t) = -(\beta - \beta q_1) F_{00}(t) + 2\beta q_0 F_{02}(t) + \alpha p_0 F_{01}(t) \]  

3. THE SOLUTION APPROACH

We denote the Laplace transform of \( f_n(t) \) by \( \tilde{f}_n(s) \). For solving the set of equations governing the model, we take Laplace transforms of equations (16)-(36) and solve using matrix method with initial conditions \( f_0(0) = 1, f_n(0) = 0 \) for \( n \neq k \).
Fig 1. State transition diagram
\(-\alpha p_1 \tilde{F}_{10}(s) - \beta q_1 \tilde{F}_{10}(s) = 1 \quad (37)\)

\(-2\alpha p_1 \tilde{F}_{20}(s) - \beta q_1 \tilde{F}_{10}(s) + \left[ s + (\alpha - \alpha p_1) \right] \tilde{F}_{10}(s) = 0 \quad (38)\)

\(-\alpha p_1 \tilde{F}_{10}(s) - 3\alpha p_1 \tilde{F}_{10}(s) - \beta q_1 \tilde{F}_{12}(s) + \left[ s + (2\alpha - 2\alpha p_1) \right] \tilde{F}_{12}(s) = 0 \quad (39)\)

\(-2\alpha p_2 \tilde{F}_{20}(s) - 4\alpha p_2 \tilde{F}_{40}(s) - \beta q_1 \tilde{F}_{31}(s) + \left[ s + (3\alpha - 3\alpha p_1) \right] \tilde{F}_{31}(s) = 0 \quad (40)\)

\(-3\alpha p_2 \tilde{F}_{50}(s) - 5\alpha p_2 \tilde{F}_{50}(s) - \beta q_0 \tilde{F}_{41}(s) + \left[ s + (4\alpha - 4\alpha p_1) \right] \tilde{F}_{41}(s) = 0 \quad (41)\)

\(-4\alpha p_2 \tilde{F}_{40}(s) + \left[ s + 5\alpha p_0 \right] \tilde{F}_{50}(s) = 0 \quad (42)\)

\(-2\beta q_1 \tilde{F}_{20}(s) - \alpha p_0 \tilde{F}_{10}(s) + \left[ s + (\beta - \beta q_1) \right] \tilde{F}_{20}(s) = 0 \quad (43)\)

\(-2\beta q_1 \tilde{F}_{12}(s) - 2\alpha p_1 \tilde{F}_{12}(s) + \left[ s + (\beta - \beta q_1) + (\alpha - \alpha p_1) \right] \tilde{F}_{12}(s) = 0 \quad (44)\)

\(-\alpha p_1 \tilde{F}_{11}(s) - 2\beta q_0 \tilde{F}_{22}(s) - 3\alpha p_1 \tilde{F}_{11}(s) + \left[ s + (\beta - \beta q_1) + (2\alpha - 2\alpha p_1) \right] \tilde{F}_{22}(s) = 0 \quad (45)\)

\(-2\alpha p_1 \tilde{F}_{21}(s) - 3\beta q_1 \tilde{F}_{12}(s) - 4\alpha p_1 \tilde{F}_{21}(s) + \left[ s + (\beta - \beta q_1) + (3\alpha - 3\alpha p_1) \right] \tilde{F}_{21}(s) = 0 \quad (46)\)

\(-3\alpha p_2 \tilde{F}_{31}(s) + \left[ s + (\beta q_0 + 4\alpha p_0) \right] \tilde{F}_{41}(s) = 0 \quad (47)\)

\(-\beta q_1 \tilde{F}_{60}(s) - 3\beta q_0 \tilde{F}_{60}(s) - \alpha p_1 \tilde{F}_{12}(s) + \left[ s + (2\beta - 2\beta q_1) \right] \tilde{F}_{60}(s) = 0 \quad (48)\)

\(-\beta q_2 \tilde{F}_{13}(s) - 3\beta q_0 \tilde{F}_{13}(s) - 2\alpha p_1 \tilde{F}_{22}(s) + \left[ s + (2\beta - 2\beta q_1) + (\alpha - \alpha p_1) \right] \tilde{F}_{13}(s) = 0 \quad (49)\)

\(-\beta q_2 \tilde{F}_{23}(s) - 3\beta q_0 \tilde{F}_{23}(s) - 3\alpha p_1 \tilde{F}_{12}(s) - \alpha p_2 \tilde{F}_{12}(s) + \left[ (2\beta - 2\beta q_1) + (2\alpha - 2\alpha p_1) \right] \tilde{F}_{22}(s) = 0 \quad (50)\)

\(-\beta q_2 \tilde{F}_{33}(s) - 2\alpha p_2 \tilde{F}_{22}(s) + \left[ s + (3\beta q_0 + 3\alpha p_0) \right] \tilde{F}_{22}(s) = 0 \quad (51)\)

\(-2\beta q_1 \tilde{F}_{02}(s) - 4\beta q_0 \tilde{F}_{04}(s) - \alpha p_0 \tilde{F}_{13}(s) + \left[ s + (3\beta - 3\beta q_1) \right] \tilde{F}_{04}(s) = 0 \quad (52)\)

For brevity, we denote the probabilities \(F_{i,j}\) and Laplace transform of probabilities \(\tilde{F}_{i,j}(s)\) with single suffix i.e. by \(F_i\) as defined and \(\tilde{F}_i(s)\), respectively below:

\(F_{i,0} = F_{i,1}, \tilde{F}_{i,0}(s) = \tilde{F}_{i,1}(s), \quad 0 \leq i \leq 5;\)

\(F_{i,1} = F_{i,2}, \tilde{F}_{i,1}(s) = \tilde{F}_{i,2}(s), \quad 0 \leq i \leq 4;\)

\(F_{i,2} = F_{i+1,1}, \tilde{F}_{i,2}(s) = \tilde{F}_{i+1,1}(s), \quad 0 \leq i \leq 3;\)

\(F_{i,3} = F_{i+1,2}, \tilde{F}_{i,3}(s) = \tilde{F}_{i+1,2}(s), \quad 0 \leq i \leq 2;\)

\(F_{i,4} = F_{i+1,3}, \tilde{F}_{i,4}(s) = \tilde{F}_{i+1,3}(s), \quad 0 \leq i \leq 1;\)

\(F_{i,5} = F_{i+1,4}, \tilde{F}_{i,5}(s) = \tilde{F}_{i+1,4}(s)\)

The system of Eqs (37)-(57) reduces to matrix form as

\[Q(s)\tilde{F}(s) = F(0)\]  

where, \(\tilde{F}(s)\) and \(F(0)\) are the column vector i.e.,

\[\tilde{F}(s) = [\tilde{F}_1(s), \tilde{F}_2(s), ..., \tilde{F}_{21}(s)]^T, \quad \text{and} \quad F(0) = [1, 0, 0, ..., 0]^T\]

Also \(Q(s)\) is matrix given by

\[Q(s) = \begin{bmatrix} A_1 & A_2 & 0 \\ A_1 & A_4 & A_5 \\ 0 & A_6 & A_{12i+1} \end{bmatrix} \]

Here submatrices \(A_i\) (i=1,2,......9) are constructed for particular case as follows:

182 - Vol. 25, No. 2, April 2012 IJE Transactions A: Basics
\[
A = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & -\beta q_0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

\[
A_4 = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & -2\beta q_0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

\[
A_5 = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

\[
A_6 = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

\[
A_7 = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

\[
A_8 = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

IJE Transactions A: Basics
Vol. 25, No. 2, April 2012 - 183
For breviety of notations, in sub matrices, we have used
\[ \theta_i = \left[ (\beta - \beta q_i) + i\alpha (1 - p_i) \right], \quad i=1, 2, 3 \]
\[ \theta_i = \left[ i\beta (1 - q_i) + \alpha (1 - p_i) \right], \quad i=4, 5, 6 \]

Using Cramer’s rule, the probabilities \( \tilde{F}_k(s) \), can be obtained as
\[ \tilde{F}_k(s) = \frac{|Q_{k,k}(s)|}{|Q(s)|}, \quad 0 \leq k \leq L \tag{59} \]

For calculating the characteristic roots of the matrix \( Q(s) \), we note that
\( s = 0 \) is one of the roots. Let \( s = -d \), so that we get
\[ Q(-d) = (Q - dI) \tag{60} \]

Now Eq (58) becomes
\[ Q(-d).\tilde{F}(s) = (Q - dI) \tilde{F}(s) = F(0) \tag{61} \]

It may be observed that the eigen values of \( Q \) are real and distinct and \( Q \) is positive definite. So, all eigen values of \( Q \) are positive. Let \( v_j \) \( (1 \leq k \leq L) \) denotes the eigen values of \( Q \), then we get
\[ |Q(s)| = s^L \prod_{j=1}^{L} (s + v_j) \tag{62} \]
so that
\[ \tilde{F}_k(s) = \frac{|Q_{k,k}(s)|}{s^L \prod_{j=1}^{L} (s + v_j)}, \quad 1 \leq k \leq L \tag{63} \]

We may expand \( \tilde{F}_k(s) \) by partial fractions, i.e., in the form
\[ \tilde{F}_k(s) = \frac{a_0}{s} + \sum_{j=1}^{L} \frac{a_{k,j}}{s + v_j} \tag{64} \]
\[ \tilde{F}_k(s) = \sum_{j=1}^{L} \frac{a_{k,j}}{s + v_j}, \quad k = 2, 3, \ldots, L \tag{65} \]
where \( a_0 \) and \( a_{k,j} \) \( (n=1,2,\ldots, L) \) are real numbers calculated as
\[ a_0 = \frac{|Q(0)|}{\prod_{j=1}^{L} v_j} \tag{66} \]
and
\[ a_{k,j} = -\frac{|Q_{k,j}(-v_j)|}{v_j \prod_{j=1}^{L} (v_j - v_k)}, \quad 1 \leq l \leq L, \quad 2 \leq k \leq L \tag{67} \]

On taking inverse Laplace transform of Eqs (59) and (60), we get
\[ F_l(t) = \frac{|Q(0)|}{\prod_{j=1}^{L} v_j} \sum_{j=1}^{L} \frac{|Q_{k,j}(-v_j)| \exp(-v_j t)}{\prod_{j=1}^{L} (v_j - v_k)} \tag{68} \]
On inverting Eq (63), we have
\[ F_l(t) = \sum_{j=1}^{L} \frac{|Q_{k,j}(-v_j)| \exp(-v_j t)}{v_j \prod_{j=1}^{L} (v_j - v_k)}, \quad \text{where} \quad 2 \leq l \leq L \tag{69} \]

### 4. PERFORMANCE INDICES

Now we give some performance measures for the quantification of software reliability indices as follows:

- The probability of a perfect program at time \( t \) is given by \( F_1(t) \).
- The mean number of faults remaining in the software at time \( t \) is given as
  \[ E[D(t)] = \sum_{i=1}^{L} iF_i(t) \tag{70} \]
- The software reliability is defined as
  \[ R(x/t) = \sum_{i=1}^{L} F_i(t) \times \left\{ \exp\left[ -(\alpha + \beta)x \right] \right\}^{i} \tag{71} \]

### 5. SENSITIVITY ANALYSIS

Since no live data is available so instead of estimating the parameters we have used the secondary data for validity and practical utility of our model (Ref. Kapur et al. 1992).

In this section, we perform computational experiment for the transient analysis by employing Runge-Kutta technique (RKT) of fourth order and matrix method to solve the system of differential equations. R-K method is implemented by exploiting MATLAB’s ‘ode45’ function. A time span is considered with equal intervals. Eigen
values are evaluated by the using MATLAB’6.5 software. For illustration purpose, we choose default parameters as $\alpha = 0.49$, $\beta = 0.02$, $p_0=0.6$, $p_1=0.3$, $p_2=0.1$, $q_0=0.5$, $q_1=0.3$, and $q_2=0.2$.

From Tables 1 and 2, we notice the patterns of various performance indices namely $R(t)$ and $E\{D(t)\}$ by varying the probabilities $p_0$, $p_1$, $p_2$ and $q_0$, $q_1$, $q_2$, respectively. It is observed that there is an increasing trend in the values of $R(t)$ and decreasing trend in $E\{D(t)\}$ with the increasing values of $p_0$ and $q_0$.

Table 1. Software reliability and $E\{D(t)\}$ for different values of $p_0$, $p_1$ and $p_2$.

<table>
<thead>
<tr>
<th>$p_0$</th>
<th>$p_1$</th>
<th>$p_2$</th>
<th>$t$</th>
<th>$R(t)$</th>
<th>$E{D(t)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.6</td>
<td>0.25</td>
<td>0.15</td>
<td>0</td>
<td>0.787</td>
<td>12.000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.852</td>
<td>8.122</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>0.897</td>
<td>5.591</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0.926</td>
<td>3.950</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>0.946</td>
<td>2.893</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>0.958</td>
<td>2.213</td>
</tr>
</tbody>
</table>

Table 2. Software reliability and $E\{D(t)\}$ for different values of $q_0$, $q_1$ and $q_2$.

<table>
<thead>
<tr>
<th>$q_0$</th>
<th>$q_1$</th>
<th>$q_2$</th>
<th>$t$</th>
<th>$R(t)$</th>
<th>$E{D(t)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.6</td>
<td>0.3</td>
<td>0.1</td>
<td>0</td>
<td>0.787</td>
<td>12.000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.852</td>
<td>8.122</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>0.897</td>
<td>5.591</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0.926</td>
<td>3.950</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>0.946</td>
<td>2.893</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>0.958</td>
<td>2.213</td>
</tr>
</tbody>
</table>

Table 3. Performance indices for different values of $\alpha$.

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>$t$</th>
<th>$R(t)$</th>
<th>$E{D(t)}$</th>
<th>$R(t)$</th>
<th>$E{D(t)}$</th>
<th>$R(t)$</th>
<th>$E{D(t)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>0</td>
<td>0.698</td>
<td>12.000</td>
<td>0.549</td>
<td>12.000</td>
<td>0.432</td>
<td>12.000</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.796</td>
<td>7.848</td>
<td>0.691</td>
<td>7.848</td>
<td>0.603</td>
<td>7.848</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.862</td>
<td>5.227</td>
<td>0.787</td>
<td>5.227</td>
<td>0.724</td>
<td>5.227</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.903</td>
<td>3.596</td>
<td>0.850</td>
<td>3.596</td>
<td>0.803</td>
<td>3.596</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.929</td>
<td>2.589</td>
<td>0.889</td>
<td>2.589</td>
<td>0.852</td>
<td>2.589</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.944</td>
<td>1.971</td>
<td>0.913</td>
<td>1.971</td>
<td>0.883</td>
<td>1.971</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0.955</td>
<td>1.593</td>
<td>0.928</td>
<td>1.593</td>
<td>0.902</td>
<td>1.593</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.961</td>
<td>1.362</td>
<td>0.937</td>
<td>1.362</td>
<td>0.914</td>
<td>1.362</td>
</tr>
</tbody>
</table>

Table 4. Performance indices for different values of $\beta$.

<table>
<thead>
<tr>
<th>$\beta$</th>
<th>$t$</th>
<th>$R(t)$</th>
<th>$E{D(t)}$</th>
<th>$R(t)$</th>
<th>$E{D(t)}$</th>
<th>$R(t)$</th>
<th>$E{D(t)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0</td>
<td>0.000</td>
<td>12.000</td>
<td>0.000</td>
<td>12.000</td>
<td>0.000</td>
<td>12.000</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.071</td>
<td>7.848</td>
<td>0.073</td>
<td>7.524</td>
<td>0.074</td>
<td>7.215</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.167</td>
<td>5.227</td>
<td>0.166</td>
<td>4.832</td>
<td>0.164</td>
<td>4.473</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.240</td>
<td>3.596</td>
<td>0.233</td>
<td>3.238</td>
<td>0.223</td>
<td>2.929</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.289</td>
<td>2.589</td>
<td>0.275</td>
<td>2.303</td>
<td>0.258</td>
<td>2.067</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.321</td>
<td>1.971</td>
<td>0.300</td>
<td>1.757</td>
<td>0.279</td>
<td>1.590</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0.340</td>
<td>1.593</td>
<td>0.315</td>
<td>1.439</td>
<td>0.290</td>
<td>1.326</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.352</td>
<td>1.362</td>
<td>0.324</td>
<td>1.255</td>
<td>0.296</td>
<td>1.180</td>
</tr>
</tbody>
</table>

Figs 2a and 2b depict the trend of probability of perfect program for different parameters $\alpha$ and $\beta$. It is noticed that the accuracy of the software increases as testing time increases. We also notice that there is no significant change with the increasing values of $\alpha$ but as $\beta$ increases, the probability of perfect program increases for some time, and finally becomes constant.
Figs 3a and 3b are plotted for the reliability by varying $\alpha$ and $\beta$ for I and II type of faults, respectively. From Fig 3a, we notice that the reliability decreases as $\alpha$ increases. But in Fig 3b, initially reliability increases with the testing time and remains almost same with the increasing the value of $\beta$.

From Figs 4a and 4b, mean number of remaining faults $E\{D(t)\}$ has been examined by varying the parameters $\alpha$ and $\beta$. It is seen that $E\{D(t)\}$ decreases as time increases but remains same for all values of $\alpha$ and $\beta$.

Overall, with the help of numerical results we observe that the optimal release time of the software can be determined successfully. For
example, if the initial error content function is assumed to be 12, we can notice from the Figs 4a and 4b that \( E[D(t)] \leq 12 \). This means that the software developer can decide the time to a specific software quality level with the condition that the reliability may reach at a maximum level. Similar conclusions for other performance measures are also evident from the other figures and tables.

Based on the sensitivity analysis which has been given in this paper, one can estimate the idea about the release time of the software. For example, from Tables 9.3 and 9.2, we see that as remaining faults in the software are becoming less then reliability of the software is increasing and finally it became constant that shows the real situation of testing the software. In that sense, the markovian software reliability model with imperfect debugging and generation of errors proposed in this paper are intuitively understandable and can provide to a software developer a more tractable framework for developing a real time situation assessment tools in spite of their simple structure.

6. CONCLUSION

In this paper, we have developed the markovian software reliability model by including the concept of imperfect debugging and error generation phenomenon. The suggested approach is suitable for practical application in reliability engineering. Our stochastic model provides a theoretical framework during the software development for understanding the factors that affect the software reliability. The suggested model may be helpful in measuring and assessing the software reliability, during operational phase.

7. REFERENCES

Engineering, SE-1, 312-327.