Reliability is one of the most important quality attributes of any commercial software. The purpose of software developer is to increase the reliability by removing the software defects. It depends on quantifying software failure during the development process and our ability to detect them with different techniques. This study deals with the mean value function to establish corresponding reliability characteristics of the simple, hard and complex system. We categorize the faults according to the amount of the testing-efforts needed and time-lag to remove the faults. Generally, software development efforts are described by traditional exponential, Rayleigh, Weibull, Logistic, Generalized logistic testing effort functions. In this chapter, we discuss a log-logistic testing-effort function (TEF) that is more suitable to the natural flow of the software development. Estimation of reliability is a well known problem for developing a software product. For obtaining more accurate and realistic reliability indices, we employ fuzzy set theoretic approach to discuss the analytical results with reference to fuzzified system parameters. A numerical example is given to illustrate the validity of the proposed model. The sensitivity analysis has also taken into consideration.

4.1 Introduction

Systems are becoming more critical and computerized swiftly with increasing latest technologies. The revolution of computer technology has made our lives much better and comfortable than before by providing various useful systems such as computer, communication system, etc.. In view of this it is very essential for computer manufacturer that the system should be of desired quality and more reliable. When manufacturer runs various tests on any computer system, to ensure its reliability, the deficiencies detected are removed by using proper techniques and he gets more confidence for the declaration of the reliability. This is more correct in case of a software release than hardware systems. Software reliability growth models are being successfully used for estimating mean time to failure (MTTF), number of remaining faults, failure intensity function, defect levels and reliability of the software
system. These factors can be used to determine the software reliability and enhancing the software development status. To achieve a highly reliable software system, the developers must keep in mind, the efficiency of testing team members, design of test cases, software architecture, testing environment, types of faults for detection and cost effective release time, optimal testing time for removing the faults etc., through proper techniques. If software reliability growth process is studied with respect to the amount of expended testing effort, the manufacturer may get more realistic results.

Numerous software reliability growth models which relate the number of faults (detected, isolated, removed) have been developed by many researchers. Zeephongsekul et al. (1994) studied primary and secondary faults under imperfect debugging environment. Huang (2005a) discussed various aspects of optimization framework for software reliability. The most critical problem in software industry is optimum release time decision for the safety of critical system under uncertainty. Release time decision policies for the safety of critical system were established by Jha et al. (2008). Sebakhy (2009) discussed comparison between software reliability identification using functional networks.

The impact of testing efforts and efficiency on the modeling of software reliability has been discussed by many researchers. Yamada et al. (1986) introduced the testing effort function (TEF) in the SRGM. Yamada and Othera (1990) and Kapur and Bardhan (2002) discussed the problem of testing effort control of SRGM. Huang and Kuo (2002) analyzed the logistic testing effort function into software reliability modeling. In 2006, Bokhari and Ahmad suggested log-logistic testing effort function for reliability growth model. Software reliability models with testing effort functions were developed by Fiondella and Gokhale (2008).

The fuzzy theory has been used by several authors to study reliability problem. The basic and fundamental work for this approach can be found in Bellman and Zadeh (1970), Zadeh (1978) and Dubois and Prade (1980). Zeephongsekul and Xia (1996) estimated software programs on fuzzy debugging. The recent collection of papers by Aliev and Kara (2004), Yao et al. (2008) gave different approaches for finding reliability through fuzzy set theory. Frentiu and Pop (2009) have employed the fuzzy methods for efforts estimation by analogy.
In this chapter we investigate log-logistic testing effort to predict the reliability of the software having different types of faults. The rest of the chapter is structured as follows. Section 4.2, briefly reviews the conventional TEF in the existing literature. The log-logistic testing effort function is also described in this section. In section 4.3, we present a detailed account of a new SRGM which incorporates the log-logistic TEF for simple, hard and complex types of faults. In section 4.4, we discuss about fuzzy membership assignment. The mathematical modeling of reliability with fuzzy logic is explained in section 4.5. Section 4.6 is devoted to describe the proposed model with fuzzy parameters and conventional approaches. Finally, conclusion has been drawn in section 4.7 by giving future research prospects. The applications of the model proposed are also given in this section.

4.2 Testing-Effort for SRGM

In literature, several researchers have developed software reliability growth models (SRGMs) to measure the growth of reliability based on non-homogeneous Poisson process. Generally, SRGMs may include the physical properties of the code, for example, the number of faults remaining in the software system, optimal testing time, etc. The testing effort functions (TEF) play a key role in the software reliability modeling (SRM). These functions describe how an effort is distributed over the exposure period and how much it is effective. The testing efforts expended in testing can be characterized as number of test-runs, scheduling, generating, or some other suitable measures.

In past studies, different types of testing efforts are used to develop SRGM; the commonly used functions are exponential, Rayleigh and Weibull distributions etc. Some notable works based on different testing efforts are given in table 4.1. These functions can be derived by the assumption that the testing effort rate is proportional to the testing resources available. Recently, Bokhari and Ahmad (2006) proposed the log-logistic function to describe the time-dependent behavior of the testing effort consumptions during testing.
Chapter-4: Fuzzy Reliability Model For Log-Logistic Testing Efforts...

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Testing Effort</th>
<th>Function</th>
<th>Introduced by</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Exponential</td>
<td>$W(t) = \alpha \left(1 - e^{-\beta t}\right)$</td>
<td>Yamada et al. (1986)</td>
</tr>
<tr>
<td>2.</td>
<td>Rayleigh</td>
<td>$W(t) = \alpha \left(1 - e^{-\left(\frac{\beta}{2}\right) t^2}\right)$</td>
<td>Yamada and Othera (1990)</td>
</tr>
<tr>
<td>3.</td>
<td>Weibull</td>
<td>$W(t) = \alpha \left(1 - e^{-\beta t^b}\right)$</td>
<td>Yamada (1991)</td>
</tr>
<tr>
<td>4.</td>
<td>Logistic</td>
<td>$W(t) = \frac{N}{1 + Ae^{-at}}$</td>
<td>Huang and Kuo (2002)</td>
</tr>
<tr>
<td>5.</td>
<td>Generalized logistic</td>
<td>$w(t) = \frac{k}{\sqrt{1 + Ae^{-at}}}$</td>
<td>Huang and Lyu (2005)</td>
</tr>
</tbody>
</table>

Table 4.1: Some testing effort functions

Cumulative log-logistic TE consumption over the time period $(0, t]$ can be expressed as

$$W(t) = \alpha \left[\left(\beta t\right)^{\delta} \right] \left[1 + (\beta t)^{\delta}\right]^{-\delta}$$

...\(4.1\)

The current TE expenditure rate at testing time $t$ is given by

$$w(t) = \frac{d}{dt} W(t) = \frac{\alpha \delta \beta \delta t^{\delta-1}}{1 + (\beta t)^{\delta}}$$

\(4.2\)

The current testing effort $w(t)$ reaches its maximum value at time

$$t_{\text{max}} = \left[\frac{\delta - 1}{\beta^\delta (1 + \delta)}\right]^{\frac{1}{\delta}}$$

\(4.3\)

4.3 SRGM with Log-Logistic TEF

In this model, we consider the complexity of faults with respect to time in detecting, isolating and removing them. Software faults are assumed of different severity such as simple, hard and complex. In reality, a fault is removed in three stages; first of all it is detected then isolated and finally removed. Due to complexity of faults or skill of the testing team, there is a time delay between the detection,
isolation and removal/correction of the underlying faults. The proposed SRGM takes account of the time dependent variation in testing effort.

Now, we develop NHPP based SRGM with a log-logistic TEF; for modeling the software growth the following assumptions are made:

- The software is subject to failure at random time during execution because of remaining faults in the software.
- The fault removal process follows a non-homogeneous poisson process (NHPP).
- The time dependent behavior of testing effort is modeled by a log-logistic testing effort function.
- Each time a failure is detected/isolated, an immediate effort takes place to correct or remove it i.e. correction of faults takes only negligible time. The mean number of faults isolated/detected with respect to testing effort in the time interval \((t, t+\Delta t)\) is proportional to the mean number of remaining faults in the system.
- Each time a failure occurs, immediately it is corrected or removed and no new faults are introduced into the system.

The notations used for the mathematical formulation of the model are as follows:

- \(a\) : Expected number of total faults in the software.
- \(q_j\) : Proportion of \(j\)th type faults in the software.
- \(b_j\) : Fault detection rate for \(j\)th type fault.
- \(m_{Dj}(t)\) : Mean number of faults detected by time \(t\) for \(j\)th type fault.
- \(m_{Ij}(t)\) : Mean number of faults isolated by time \(t\) for \(j\)th type fault.
- \(m_{Rj}(t)\) : Mean number of faults removed by time \(t\) for \(j\)th type fault.
- \(\alpha\) : Total eventual effort.
- \(\beta, \delta\) : Scale and shape parameters of log-logistic testing-effort.
4.3.1 Mean Value Function

4.3.1.1 Simple Faults, where $1 \leq j \leq p_1$.

Simple faults can be removed in a single stage process. These faults can be removed instantly as soon as they are observed. The differential equation for mean value function in this case is:

$$\frac{dm_{Rj}(t)}{dt} \times \frac{1}{w(t)} = b_j [a - m_{Rj}(t)], \quad 1 \leq j \leq p_1 \quad \ldots(4.4)$$

So that

$$m_{Rj}(t) = a \left[ 1 - e^{-b_j w(t)} \right], \quad 1 \leq j \leq p_1 \quad \ldots(4.5)$$

4.3.1.2 Hard Faults, where $p_1 + 1 \leq j \leq p_1 + p_2$.

This is a two stage process. In this stage, testing team has to spend more time than the first stage for analyzing the cause of the failure. The differential equations for mean value function $m_{Dj}(t)$ (for detecting the error), and $m_{Rj}(t)$ (for removing the detected error) are given by

$$\frac{dm_{Dj}(t)}{dt} \times \frac{1}{w(t)} = b_j [a - m_{Dj}(t)] \quad \ldots(4.6)$$

and

$$\frac{dm_{Rj}(t)}{dt} \times \frac{1}{w(t)} = b_j [m_{Dj}(t) - m_{Rj}(t)] \quad \ldots(4.7)$$

Thus we have,

$$m_{Rj}(t) = a \left[ 1 - (1 + b_j W(t)) e^{-b_j w(t)} \right] \quad \ldots(4.8)$$

4.3.1.3 Complex Faults, where $p_1 + p_2 + 1 \leq j \leq p_1 + p_2 + \ldots + p_n$.

This is a multi stage process. Testing team requires more time and efforts for removing the faults after detection and isolation, because of the complexity of the fault, these (fault) are removed in many steps. The differential equations for mean
value function $m_{Dj}(t)$ (for detecting the error), $m_{Ij}(t)$ (for isolating the detected error) and $m_{Rij}(t)$ (for removing the isolated error) are given as under.

\[
\frac{dm_{Dj}(t)}{dt} \times \frac{1}{w(t)} = b_j \left[ a - m_{Dj}(t) \right] \quad \text{...(4.9)}
\]

\[
\frac{dm_{Ij}(t)}{dt} \times \frac{1}{w(t)} = b_j \left[ m_{Dj}(t) - m_{Ij}(t) \right] \quad \text{...(4.10)}
\]

\[
\frac{dm_{Rij}(t)}{dt} \times \frac{1}{w(t)} = b_j \left[ m_{Ij}(t) - m_{Rij}(t) \right] \quad \text{...(4.11)}
\]

\[
\frac{dm_{Rij}(t)}{dt} \times \frac{1}{w(t)} = b_j \left[ m_{R_{i-1j}}(t) - m_{Rij}(t) \right], \quad k = 2, 3, ..., n \quad \text{...(4.12)}
\]

Thus we have,

\[
m_{Rij}(t) = a \left[ 1 - \sum_{k=0}^{n+1} \frac{b_j W(t)^k}{k!} \exp\left( -b_j W(t) \right) \right], \quad 1 \leq k \leq n. \quad \text{...(4.13)}
\]

### 4.3.2 Total Fault Removal

The model proposed in this chapter is the superposition of the simple, hard and complex components with mean value functions given by equations (4.5), (4.8), and (4.13) respectively. Thus the mean value function of the superposed NHPP is

\[
m(t) = \sum_{j=1}^{p_1} q_i m_{Rj}(t) + \sum_{j=p_1+1}^{p_1+p_2} q_i m_{Rj}(t) + \sum_{i=1}^{n} \sum_{j=p_1+1}^{p_1+p_2} q_j m_{Rij}(t)
\]

\[
= \sum_{j=1}^{p_1} aq_j \left[ 1 - \exp\left( -b_j W(t) \right) \right] + \sum_{j=p_1+1}^{p_1+p_2} aq_j \left[ 1 - (1 + b_j W(t)) \exp\left( -b_j W(t) \right) \right]
\]

\[
+ \sum_{i=1}^{n} \sum_{j=p_1+1}^{p_1+p_2} aq_i \left[ 1 - \left( 1 + \sum_{k=1}^{n} \frac{(b_j W(t))^k}{k!} \right) \exp\left( -b_j W(t) \right) \right] \quad \text{...(4.14)}
\]
where

\[
\sum_{j=1}^{p_1} q_j + \sum_{j=p_1+1}^{p_1+p_2} q_j + \sum_{j=p_1+p_2+1}^{p_1+p_2+p_3} q_j = 1
\]

\[\ldots(4.15)\]

### 4.4 Fuzzy Membership Assignment

The concept of fuzzy number is the most commonly used for defining the fuzzy set application. For the sake of simplicity in performing software reliability evaluation, triangular fuzzy number is used in the proposed model. A triangular membership function is defined by three parameters such as \([a_l, a, a_u]\) and its graph has a pointed top and just like a triangular curve as shown in fig. 4.1. This figure defines the maximum and minimum levels of presumption in the interval \([0, 1]\).

![Fig 4.1: Fuzzy triangular Number](image)

A fuzzy number ‘A’ is called a triangular fuzzy number (TFN) if its membership function \(\mu_A(x)\) is given by

\[
\mu_A(x) = \begin{cases} 
0, & x \leq a_l, x > a_u, \\
\frac{x-a_l}{a-a_l}, & a_l \leq x \leq a, \\
\frac{a_u-x}{a_u-a}, & a \leq x \leq a_u.
\end{cases}
\]

\[\ldots(4.16)\]

For developing the confidence level of the fuzzy number, the triangular fuzzy number (TFN) ‘A’ is denoted by the triplet \(A = (a_l,a,a_u)\) and having the shape of a
triangle. The interval confidence \( (\alpha) \) of a triangular fuzzy number can be characterized as

\[
\begin{align*}
A_\alpha &= \left[ a^L_\alpha, a^R_\alpha \right] \\
&= \left[ (a-a_l)\alpha + a_l, (a-a_u)\alpha + a_u \right] \quad \forall \alpha \in [0,1] \quad \cdots (4.17)
\end{align*}
\]

4.5 Fuzzy Software Reliability Modeling

Fuzzy software reliability is obtained based on the fuzzy set theory. We consider the fault detection rate as triangular fuzzy number; as such the reliability is also determined as triangular fuzzy number. Of course, the calculation of the reliability must be restricted between 0 and 1. We obtain the reliability as a fuzzy number using the formula

\[
\tilde{R}(x/t) = \exp\left[-\{\tilde{m}(t+x) - \tilde{m}(t)\}\right] \quad \cdots (4.18)
\]

Let fault detection rates for simple \( (b_1) \), hard \( (b_2) \) and complex \( (b_3) \) faults are represented by a triangular fuzzy numbers \( \tilde{b}_1, \tilde{b}_2 \) and \( \tilde{b}_3 \), respectively. Let \( A_{\tilde{b}_1}(x_1), A_{\tilde{b}_2}(x_2) \) and \( A_{\tilde{b}_3}(x_3) \) denote the membership functions of \( \tilde{b}_1, \tilde{b}_2 \) and \( \tilde{b}_3 \), respectively. Then we have the following fuzzy sets

\[
\begin{align*}
\tilde{b}_1 &= \left\{ \left( x_1, A_{\tilde{b}_1}(x) \right) : x_1 \in X_1 \right\} ; \tilde{b}_2 &= \left\{ \left( x_2, A_{\tilde{b}_2}(x) \right) : x_2 \in X_2 \right\} ; \tilde{b}_3 &= \left\{ \left( x_3, A_{\tilde{b}_3}(x) \right) : x_3 \in X_3 \right\}
\end{align*}
\]

\cdots (4.19)

where \( x_1, x_2 \) and \( x_3 \) are the crisp universal sets of the fault detection rates for simple, hard and complex types of faults.

4.6 Numerical Results

Below we provide two types of numerical experiments.

4.6.1 Fuzzy Approach

The proposed model facilitates the results for multi stage process. Now we consider that there are three types of faults i.e. simple, hard and complex and each type fault has only one stage. Thus
\[
m(t) = aq_1\left[1 - \exp(-b_1 W(t))\right] + aq_2\left[1 - (1 + b_2 W(t))\exp(-b_2 W(t))\right] \\
\quad + aq_3\left[1 + b_3 W(t) + \left(\frac{b_3 W(t)^2}{2!}\right)\exp(-b_3 W(t))\right] \tag{4.20}
\]

Suppose expected number of initial faults \(a=500\) and total eventual testing efforts \(\alpha = 30\), \(\beta = .2\), \(\delta = .07\). Fault detection rates are represented by the triangular fuzzy number as

\[
\tilde{b}_1 = [2 .3 .4], \quad \tilde{b}_2 = [3 .4 .5], \quad \tilde{b}_3 = [.4 .5 .6]
\]

On fuzzyfying these parameters in the equation (4.21), we obtain the \(m(t)\) as a fuzzy number given below:

\[
\tilde{m}(t) = aq_1\left[1 - \exp(-\tilde{b}_1 W(t))\right] + aq_2\left[1 - (1 + \tilde{b}_2 W(t))\exp(-\tilde{b}_2 W(t))\right] \\
\quad + aq_3\left[1 + \tilde{b}_3 W(t) + \left(\frac{\tilde{b}_3 W(t)^2}{2!}\right)\exp(-\tilde{b}_3 W(t))\right] \tag{4.21}
\]

The membership function graph of \(\tilde{R}(t)\) is shown in fig. 4.2. In this graph, the reliability is represented by three values (low, nominal and higher). Since \(\tilde{R} = \text{TFN} (0.8534 0.9418 0.98)\), the range of reliability is 0.8534 to 0.98 with 0.9418 being the nominal value.

### 4.6.2 Conventional Approach

This approach is applied for the same values of parameters as used in fuzzy approach. The membership function graph of software reliability is shown in fig. 4.2. Figs (4.3)-(4.5) depict the pattern of conventional reliability Vs time for different values of \(b_1, b_2, b_3\), respectively. We notice that the reliability initially increases sharply with time and after some time (\(t = 70\)) it becomes almost constant for different values of error detection rates \(b_1, b_2, b_3\), respectively. From these figures, we also note that as error detection rate increases, the reliability also increases.
4.7 Conclusion

In this chapter, we have studied reliability in critical systems for software applications by using both fuzzy and conventional approaches. In the conventional method, input parameters are a certain value and the uncertainties are not reflected in the concerned graphs. By using fuzzy approach, the fault detection rate can be accounted easily with fuzzy sets. The uncertainty of the parameters can be easily incorporated as shown in the concerned software reliability model. It is to be mentioned that the fuzzy approach used gives more adequate, and flexible results as compared to conventional approach. By using results of our model, the developer can achieve a reasonable high level reliability in the complex software system.

Because of the great scope for the generalization and extension of the present work, our work has potential applications in the software industry. In addition to the fuzziness of the fault detection rate, we can also fuzzified the number of initial faults and testing efforts. Moreover, optimization issues to obtain the optimal parameters can be further resolved by extending the present work by constructing the appropriate cost function.
Fig. 4.2: The membership function of reliability

Fig. 4.3: Reliability Vs t for different values of $b_1$

Fig. 4.4: Reliability Vs t for different values of $b_2$

Fig. 4.5: Reliability Vs t for different values of $b_3$