Incorporating Burr Type XII Testing-efforts into Software Reliability Growth Modeling and Actual Data Analysis with Applications

Mohammad Ubaidullah Bokhari
Department of Computer Science, Aligarh Muslim University, Aligarh, India
Email: mubokhari@gmail.com

Nesar Ahmad *
University Department of Statistics and Computer Applications
T. M. Bhagalpur University, Bhagalpur, India
Email: nesar_bgp@yahoo.co.in

***Abstract*—Software reliability is the probability that the given software functions correctly under a given environment, during the specified period of time. During the software-testing phase, software reliability is highly related to the amount of development resources spent on detecting and correcting latent software errors, i.e. the amount of testing effort expenditures. This paper develops software reliability growth models (SRGM) based on non homogeneous Poisson process (NHPP) which incorporates the Burr Type XII testing-effort functions (TEF). Numerous testing-effort functions for modeling software reliability growth based on NHPP have been proposed in the past decade. This paper shows that the Burr Type XII testing-effort function can be expressed as the actual testing-effort consumption during software development process. Its fault-prediction capability is evaluated through the numerical experiments. SRGM parameters are estimated by least square estimation (LSE) and maximum likelihood estimation (MLE) methods and computational experiments performed on actual software failure data set from various software projects. The results show that the proposed testing-efforts functions predicts fault better than the other existing models. Thus, the proposed models evaluate software reliability more realistically. In addition, the optimal release policy based on reliability and cost criteria for software system are proposed.**

***Index Terms*—SRGM, NHPP, Burr Type XII TEF, LSE, MLE, Testing effort consumptions**

I. Introduction

In modern society, computer-controlled and computer-embedded systems are heavily dependent on the correct performance of software. So, it is quite natural to produce reliable software systems efficiently since the breakdown of the computer systems, which is caused by software errors, results in a tremendous loss and damage for social life. In the past years, several software reliability growth models (SRGM) based on NHPP which incorporates the testing–efforts have been proposed by many authors [2], [3], [6], [11], [13]-[17], [20]-[22], [37], [39], [40]. The testing-effort can be measured by the man power spent during the testing phase, the number of CPU hours, the number of executed test cases, and so on. Software reliability growth models proposed in the literature incorporating the effect of testing-effort expenditures described by the traditional Weibull type and Logistic type. However, it is difficult to represent the consumption curve only by these testing-effort consumption curves in various software development environments.

This paper describes the time dependent behavior of testing-effort expenditure by Burr Type XII model [9] as its curve is flexible having a wide variety of possible expenditure patterns in real software projects. This family includes exponential, Weibull and log-logistic as special cases. It also covers the curve shape characteristics of normal, log-normal, gamma, logistic and Pearson type X distributions as well as a significant portion of the curve shape characteristic for Pearson Type I (Beta), II, V, VII, IX and XII families [4], [29], [30], [33], [34]. Another advantage is that Burr XII has simple algebraic forms for reliability and hazard rate functions [4]. Thus Burr Type XII Provides a wide variety of density shapes along with functional simplicity. Currently there are few studies for the use of the Burr Type XII failure model in reliability and survival analysis [4], so this paper is to promote its use in software reliability analysis. The Burr Type XII failure model can be widely and effectively used in software reliability analysis, because it has a wide variety of shapes in its model and failure rate curves [4], making it useful for fitting many types of actual software failure data from various software projects.

Reference [1] has used of Burr Type XII distribution on software reliability growth modeling. This paper develops a realistic software reliability growth models based on NHHP which incorporates the Burr Type XII

Manuscript received March 1, 2013 revised August 10, 2013 accepted August 12, 2013
*Corresponding author

doi:10.4304/jsw.9.6.1389-1400

testing–effort function [5]. It is assuming that the error detection rate in software testing is proportional to the current error content and the proportionality is the instantaneous software testing-effort expenditures at an arbitrary testing time. Its parameters are estimated by Least Square Estimation and Maximum Likelihood Estimation methods. Computational experiments are performed for three real software data and the results are compared with other existing model. It is shown that the proposed SRGM with Burr Type XII testing-effort function is wide and effective models for software reliability analysis. It can estimate the number of initial faults better as compare to other existing models. In addition, the optimal release policy of this model based on cost-reliability criterion is discussed.

II. Burr Type XII Testing Effort Function

From the previous studies in [12], [13], and [20], we know that actual test effort data expressed various consumption pattern, sometimes the test effort consumption are difficult to describe only by Exponential, Rayleigh, Weibull or Logistic curve. Therefore, we try to incorporate a Burr Type XII test-effort function instead of above consumption function as the test effort function during the software development process in [7] and [8]. So, we proposed Burr Type XII curve as the test-effort function into SRGM.

The current testing effort consumption curve at testing time t is given as

$$w(t)=\frac{\alpha\ \beta\, m\delta\left(\beta\cdot t\right)^{\delta-1}}{\left[1+\left(\beta\cdot t\right)^{\delta}\right]^{m+1}},$$

$$\alpha>0,\beta>0,m>0,\delta>0,t>0 \tag{1}$$

where α, β, m and δ are constant parameters, α is the total amount of test-effort expenditure required by software testing, β is the scale parameter, and m, δ are shape parameters.

The integral form of (1) is called the cumulative test-effort consumption of Burr Type XII in the time [0, t] and is given by:

$$W\left(t\right)=\int_0^t w\left(x\right)dx=\alpha\left[(1-\left(1+(\beta\cdot t\right)^{\delta})^{-m}\right]$$

$$\alpha,\beta,m,\delta>0,\quad t\geq 0 \tag{2}$$

The testing-effort function $w(t)$ reaches its maximum value at the time t

$$t_{\max}=\frac{1}{\beta}\left[\frac{\delta-1}{\delta m+1}\right]^{1/\delta}$$

III. Software Reliability Growth Model

A. Model Description

A number of SRGMs have been proposed on the subject of software reliability. Among these models, Goel and Okumoto used an NHPP as the stochastic process to describe the fault process [11] and [23] modify the G-O model and incorporate the concept of testing-effort in an NHPP model to get a better description of the software fault detection phenomenon. We also propose a new SRGM with the Burr Type XII testing-effort function to predict the behavior of failure occurrences and the fault content of a software product

B. Assumptions

- The fault removal process is modeled by an NHPP.
- The software application is subject to failures at random times caused by the remaining faults in the system.
- The mean number of faults detected in the time interval $\left(t,\,t+\Delta t\right)$ by the current testing-effort is proportional to the mean number of remaining faults in the system at time t, and the proportionality is a constant over time.
- Testing effort expenditures are described by the Burr Type XII testing-effort function.
- Each time a failure occurs, the corresponding fault is immediately removed and no new faults are introduced.
- The hazard rate for software occurring initially after the testing is proportional to the elapsed time r and the remaining faults.

An implemented software system is tested in the software development process. During the testing phase software errors remaining in the system cause software failures and the errors are detected and corrected by test personnel. A software failure is defined as an unacceptable departure of program operation. Following the usual assumptions in the area of software reliability growth modeling [10], we assume that the number of detected errors to the current test-effort expenditures is proportional to the current error content. Let $m\left(t\right)$ represent the expected mean number of errors detected by testing calendar time t which is assumed to be a bounded non-decreasing function of t with $m(0)=0$. Then, using the Burr Type XII test-effort function in (1), we have the following differential equation [37]:

$$\frac{dm\left(t\right)}{dt}/\,w\left(t\right)=r\left[a-m\left(t\right)\right],\,a>0,\ \ 0<r<1 \tag{3}$$

where $m\left(t\right)$ is the expected mean number of faults detected in time (0, t), $w_k(t)$ is the current testing-effort consumption at time t, a is the expected number of initial faults, and r is the fault detection rate per unit testing-effort at testing time t and $r>0$.

Solving the differential equation (3) under the boundary condition $m(0)=0$ (*i.e.*, the mean value function $m\left(t\right)$ is equal to zero at time 0), we have:

$$m\left(t\right)=a\left(1-e^{-rW\left(t\right)}\right) \tag{4}$$

Substituting (2) for $W(t)$ in (4) we get:

$$m(t)=a\left[1-e^{-r\alpha\left(1-(1+(\beta t)^{\delta})^{-m}\right)}\right] \tag{5}$$

From (4), we have the following important relationship between $m(t)$ and $W(t)$:

$$W(t)=\frac{1}{r}\ln\left(\frac{a}{a-m(t)}\right). \tag{6}$$

For stochastic modeling of a software error detection phenomenon, let $\{N(t),\ t>0\}$ be a counting process representing the cumulative number of errors detected by testing time t. Defining the expected value of $N(t)$ by $m(t)$ in (5), we can describe a software reliability growth model incorporating the Burr Type XII test-effort function [13] and [36] by an NHPP as :

$$\Pr\{N(k)=n\}=\frac{[m(t)]^{n}\cdot e^{-m(t)}}{n!},\quad n=0,1,2,\ \dots$$
$$=\ Poim\ m(n;m(t)) \tag{7}$$

where $m(t)$ is called mean value function of the NHPP [10], [38], [39] and $Poim\ m(n;m(t))$ is a Poisson pmf with parameter $m(t)$. The intensity function of the NHPP is given by:

$$\lambda(t)=\frac{dm(t)}{dt}=\alpha.r.w(t)e^{-rW\cdot(t)} \tag{8}$$

which means the instantaneous error detection rate. From (7) we can show that the limit distribution of $N(t)$ is a Poisson distribution with the following mean :

$$m(\infty)=a\left(1-e^{-r\alpha}\right) \tag{9}$$

Equation (9) implies that even if a software system is tested during an infinitely long duration, all errors remaining in the system cannot be detected [39], [40]. Thus, the mean number of undetected errors $d(t)$ if a test is applied for an infinite amount of time is :

$$a-m(\infty)=a-a\left(1-e^{-r\alpha}\right)$$
$$\Rightarrow d(t)=ae^{-r\alpha}$$

C. Software Reliability Measures

Let $N(t)$ represent the number of errors remaining in the system of testing time t. Based on the NHPP model with $m(t)$, given by in equation (4), two quantitative measures for software reliability assessment can be derived [10], [36]. The expectation of $\bar{N}(t)$ and its variance are given by:

$$r(t)=E\left[\bar{N}(t)\right]=E\left[N(\infty)-N(t)\right]$$
$$=m(\infty)-m(t)=a\left[e^{-rW(t)}-e^{-rW(\infty)}\right]$$
$$=Var\left\{\bar{N}(t)\right\}, \tag{10}$$

The software reliability representing the probability that a software failure does not occur in the time interval *(t, t + x)* is given by:

$$R=R(x\mid t)=e^{-[m(t+x)-m(t)]}$$
$$=e^{-a\left[e^{-rW(t)}-e^{-rW(t+x)}\right]} \tag{11}$$

It can be easily seen that $R(x\mid t)$ is a monotonic increasing function of t. Taking log both side in (11)

$$\ln R=-[m(t+x)-m(t)]$$

Solving the above equation with $m(t)$, one can estimate derived reliability R. The instantaneous mean time between failures (MTBF) at arbitrary testing can be defined as a reciprocal of error detection rate in equation (8). Then the instantaneous MTBF is given by:

$$MTBF(t)=\frac{1}{\lambda\cdot(t)}=\frac{1}{a\cdot r\cdot w(t)e^{-rW(t)}}$$
$$=\frac{e^{r\alpha\left[(1-(1+(\beta\cdot t)^{\delta})^{-m}\right]}}{a.r.\dfrac{\alpha\ \beta\ m\delta(\beta\cdot t)^{\delta-1}}{\left[1+(\beta\cdot t)^{\delta}\right]^{m+1}}} \tag{12}$$

IV. Estimation Methods of Parameters

A. Estimation of Testing-Effort Parameters

Two most popular estimation techniques are Maximum Likelihood Estimation (MLE) and Least Squares Estimation (LSE) [23], [26]. The parameters α, β, m and δ in the Burr Type XII testing-effort functions defined by the equation (1) can be estimated by least squares. The estimators for α, β, m and δ are investigated for testing-effort w_k spent during $(0,t_k)$ (k = 1, 2 ,…, n). Then, based on the usual procedures, the least-squares estimators $\hat{\alpha}$, $\hat{\beta}$, $\hat{m}$ and $\hat{\delta}$ can be obtained by minimizing

$$\text{Minimize}\quad S(\alpha,\beta,m,\delta)=\sum_{k=1}^{n}(W_k-W(t_k))^2$$

Taking log in Equation (1), we get

$$\ln w_k=\ln\alpha+\ln\beta+\ln m+\ln\delta+ \\ (\delta-1)\ln(\beta t_k)-(m+1)\ln[1+(\beta t_k)^{\delta}] \tag{13}$$

Then, the least–squares estimates $\hat{\alpha},\hat{\beta},\hat{m}$ and $\hat{\delta}$ of parameters α, β, m and δ can be obtained by minimizing the following sum of squares

$$S(\alpha,\beta,m,\delta)=\sum_{k=1}^{n}\begin{bmatrix}\ln w_k-\ln\alpha-\ln\beta-\ln\delta-\ln m\\ -(\delta-1)\ln(\beta t_k)+(m+1)\ln(1+(\beta t_k)^{\delta})\end{bmatrix}^2 \tag{14}$$

Differentiate the above equation with respect to the α, β, m and δ and set the partial derivatives to zero, we get the following non-linear equations,

$$\frac{\partial S}{\partial \alpha}=\sum_{k=1}^{n} 2\begin{bmatrix}\ln w_k-\ln\alpha-\ln\beta-\ln m-\ln\delta-(\delta-1)\ln(\beta t_k)\\ +(m+1)\ln(1+(\beta t_k)^{\delta})\end{bmatrix}\times\frac{-1}{\alpha}=0$$

$$\Rightarrow \sum_{k=1}^{n}\ln w_k-n\ln\alpha-n\ln\beta-n\ln m-n\ln\delta-(\delta-1)\sum_{k=1}^{n}\ln(\beta t_k)+ \quad (15)$$

$$(m+1)\sum_{k=1}^{n}\ln(1+(\beta t_k)^{\delta})=0$$

$$\frac{\partial S}{\partial \beta}=\sum_{k=1}^{n} 2\left[\ln w_k-\ln\alpha-\ln\beta-\ln m-\ln\delta-(\delta-1)\ln(\beta t_k)+(m+1)\ln(1+(\beta t_k)^{\delta})\right]$$

$$\times\left[\frac{-1}{\beta}-(\delta-1)\times(\frac{t_k}{\beta t_k}+\frac{(m+1)}{(1+(\beta t_k)^{\delta})}\times\delta(\beta t_k)^{\delta-1}.t_k\right]$$

$$\Rightarrow \sum_{k=1}^{n}\left[\ln wk-\ln\alpha-\ln\beta-\ln m-\ln\delta-(\delta-1)\ln(\beta t_k)+(m+1)\ln(1+(\beta t_k)^{\delta}\right]$$

$$\times\left[-\frac{1}{\beta}+\frac{(m+1)\times(\beta t_k)^{\delta-1}t_k}{(1+(\beta t_k)^{\delta})}\right]=0 \quad (16)$$

$$\frac{\partial S}{\partial m}=\sum_{k=1}^{n} 2\left[\ln w_k-\ln\alpha-\ln\beta-\ln m-\ln\delta-(\delta-1)\ln(\beta t_k)+(m+1)\ln(1+(\beta t_k)^{\delta})\right]$$

$$\times\left[\frac{-1}{m}+\ln(1+(\beta t_k)^{\delta})\right]=0 \quad (17)$$

$$\frac{\partial S}{\partial \delta}=\sum_{k=1}^{n} 2\left[\ln w_k-\ln\alpha-\ln\beta-\ln m-\ln\delta-(\delta-1)\ln(\beta t_k)+(m+1)\ln(1+(\beta t_k)^{\delta})\right]$$

$$\times\left[\frac{-1}{\delta}-\ln(\beta t_k)+(m+1)\frac{1}{(1+(\beta t_k)^{\delta})}\times(\beta t_k)^{\delta}.\ln(\beta t_k)\right]=0 \quad (18)$$

These non-linear equations can be solved numerically to get the estimate of α, β, m and δ.

C. Estimation of Reliability Growth Parameters

The reliability growth parameters a and r in the NHPP model with $m(t)$ in (4) can be estimated by the method of maximum-likelihood [10]. Let the estimated parameters $\hat{\alpha},\hat{\beta},\hat{m}$ and $\hat{\delta}$ in the Burr type XII test-effort function in (1) have been obtained by the method of least-squares. The $\hat{a}$ and $\hat{r}$ are determined for the n observed data pairs (t_k, y_k) $(k=1,2,....,n)$. Then, the joint p.m.f, the log-likelihood function, for the unknown parameters a and r in the NHPP model with $m(t)$ in (4), is :

$$\ln L=\sum_{k=1}^{n}(y_k-y_{k-1})\cdot\ln a+\sum_{k=1}^{n}(y_k-y_{k-1})\cdot\ln\left(\exp\left[-rW(t_{k-1})\right]-\exp\left[-rW(t_k)\right]\right)$$

$$-a\left(1-\exp\left[-rW(t_n)\right]\right)-\sum_{k=1}^{n}\ln\left[(y_k-y_{k-1})!\right], \quad (19)$$

$$t_0\equiv 0 \text{ and } y_0\equiv 0.$$

The usual calculus methods for an interior maximum result in

$$y_n=a\cdot f_n, \qquad \Rightarrow\hat{a}=\frac{y_n}{f_n} \quad (20)$$

and

$$a\cdot g_n=\sum_{k=1}^{n}\frac{(y_k-y_{k-1})\ \ (g_k-g_{k-1})}{(f_k-f_{k-1})}, \quad (21)$$

where,

$$\begin{aligned}&f_k\equiv 1-\exp\left[-r\cdot W(t_k)\right],\\ &g_k\equiv W(t_k)\cdot\exp\left[-r\cdot W(t_k)\right],\ (k=1,2,\cdots,n),\end{aligned} \quad (22)$$

which can be solved numerically.

If the sample size n of the observed data is sufficient large, the maximum-likelihood estimates $\hat{a}$ and $\hat{r}$ asymptotically follow a bivariate s-normal distribution [28],

$$\begin{pmatrix}\hat{a}\\ \hat{r}\end{pmatrix}\sim BVN\left(\begin{pmatrix}a\\ r\end{pmatrix},\Sigma\right), \quad (n\to\infty), \quad (23)$$

The Σ in the asymptotic properties of (23) is useful in quantifying the variability of the estimated parameters $\hat{a}$ and $\hat{r}$, and is the inverse of F

$$F=\begin{bmatrix}E\left\{\frac{-\partial^2\ln L}{\partial a^2}\right\} & E\left\{\frac{-\partial^2\ln L}{\partial a\partial r}\right\}\\ E\left\{\frac{-\partial^2\ln L}{\partial a\partial r}\right\} & E\left\{\frac{-\partial^2\ln L}{\partial r^2}\right\}\end{bmatrix}$$

$$=\begin{bmatrix}\frac{f_n}{a} & g_n\\ g_n & \frac{a\sum_{k=1}^{n}(g_k-g_{k-1})^2}{(f_k-f_{k-1})}\end{bmatrix} \quad (24)$$

where, $g_k=W(t)\cdot\exp\left[-rW(t_k)\right]$

and $f_k=1-\exp\left[-rW(t_k)\right]$ where $[k=1,\ldots,n]$

Substituting the value of a and r in (4.2.6) and calculate F^{-1}. The estimated asymptotic variance-covariance matrix is:

$$\hat{\Sigma}=F^{-1}=\begin{pmatrix}Var(a) & Cov(a,r)\\ Cov(a,r) & Var(r)\end{pmatrix}$$

V. Software Failure Data Analysis

The two performance comparison criteria are given here to check the performance of the proposed software reliability growth model and to make affair comparison with the other existing SRGM

- *The Mean square of fitting error (MSE):*

$$MSE=\sum_{i=1}^{k}\frac{(\hat{m}(t_i)-y_i)^2}{k}$$

where k is the number of observation. A smaller MSE indicates a smaller fitting error and better performance [21], [24].

- *AE (Accuracy of Estimation) is defined as:*

$$A.E=\left|\frac{M_a-a}{M_a}\right|$$

M_a is the actual cumulative number of detected faults after the test, and a is estimated number of initial faults [10], [22], [26].

A. Performance Analysis

First Data Set: The first set of real data in this paper is the System T1 data of the Rome Air Development Center

(RADC) projects and cited from [25] and [26]. The number of object instructions for the system T1 which is used for a real-time command and control application. In this case, the size of the software is approximately 21,700 object instructions. The software was tested for 21 weeks with 9 programmers. During the test phase, about 25.3 CPU Hours were used and 136 faults were detected. Similarly the MLE and LSE are used to estimate the parameters for the equation (1) and equation (4)

In order to estimate the parameters α, β, m and δ of the log-logistic test-effort function, the actual testing-effort data into equations (1) has been fitted and solve it by using the method of least squares. The estimated values of parameters of the Burr Type XII testing-effort function are:

$\hat{\alpha}$ =35.242, $\hat{\beta} = 0.063$, $\hat{m}$ =0.326 and $\hat{\delta}$=11.259

Fig. 1 and Fig. 2 shows the fitting of the estimated testing-effort by using Equation (1) and (2).The fitted curves are shown as a dotted line and solid line for actual software data in the graphs. Using the estimated parameters α, β, m and δ the other parameters a, r in (4) can be solved by MLE method. The cumulative numbers of estimated failures by equation (4) are: a = 133.7025, r = 0.1553

For these estimates, the optimality was checked numerically. Table I summarizes the experimental results of estimated parameters with their standard errors and 95 % confidence bound.

Following the same procedure, we plotted a fitted curve of the estimated mean value function with the actual software data in Fig. 3. Also a comparison table of the estimates of this model along with other SRGMs with initial faults a and MSE is given in Table II. From Figs. 1, 2, and 3 and the comparison criteria in Table II, it is conceivable that the proposed SRGM has a better goodness of fit. Kolmogorov Smirnov goodness-of-fit test shows that this proposed SRGM described by an NHPP with $\hat{m}(t)$ fits pretty well at the 5 % level of significance. Fig. 4 shows that the estimated intensity functions $\hat{\lambda}(t)$ from equation (8).

Substituting the estimated parameters α, β, m and δ in equation of t_{max}, the testing effort function reaches the maximum at time t = 16.9143 debug days which corresponds to $w(t)$ = 3.5986 CPU hours and $W(t)$ = 11.1148 CPU hours. Besides, the number of errors removed up to this time t_{max} is 109.9073 and when t goes to infinity, the numbers of errors removed is 133.14116.

TABLE1
SUMMARY OF ESTIMATE OF NHPP MODEL PARAMETERS

Parameter	Estimate	Standard Error	95% Confidence Lower	95% Confidence Upper
a	133.279	6.166	120.794	146.609
r	0.1553	0.021	0.109	0.201

TABLE II
COMPARISON RESULTS FOR THE FIRST DATA SET

Model	a	r	MSE
Burr Type XII Model	133.70	0.155	77.909
G-O Model [27]	142.32	0.125	2438.3
Exponential Model [11]	137.2	0.156	3019.66
Rayleigh Function [22]	866.94	0.0096	89.241
Delayed s-shaped Model [13]	237.19	0.0963	245.246

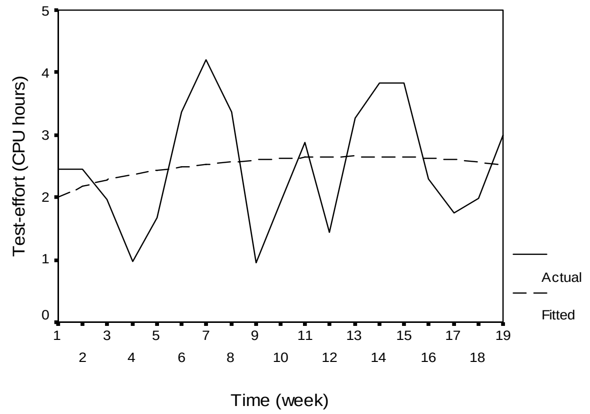

Figure 1. Observed/estimated current test-effort function vs. time

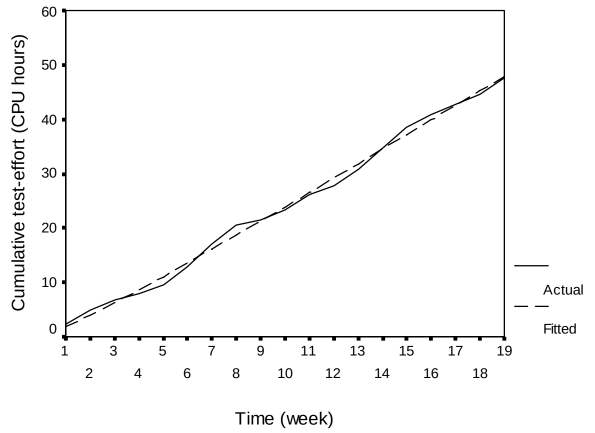

Figure 2. Observed/estimated cumulative test-effort function vs. time

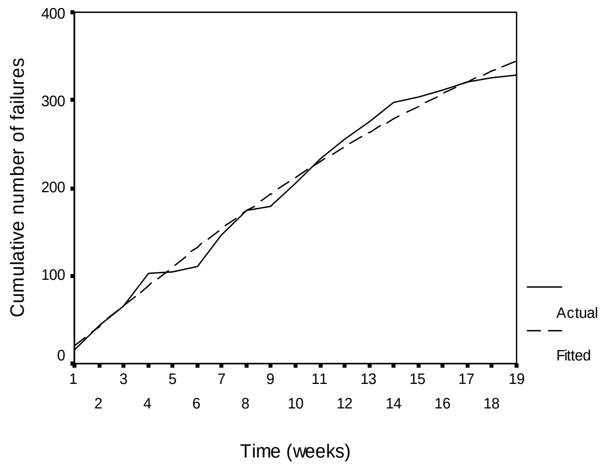

Figure 3. Observed/estimated cumulative number of failures vs. time

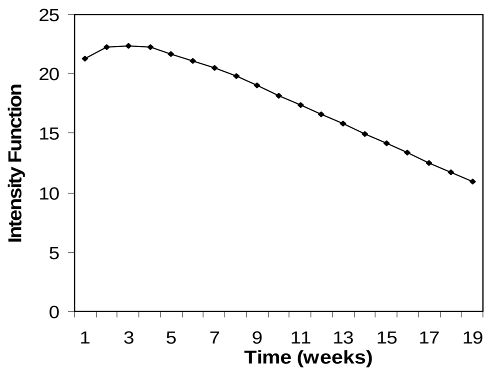

Figure 4. Estimated intensity function for actual data

Second Data Set: The second set of real data is the pattern of discovery of faults by [32]. The debugging time and the number of detected faults per day are reported. The cumulative number of discovered faults up to twenty two days is 86 and the total consumed debugging times is 93 CPU hours. All debugging data are used in this experiment. The testing-effort data are applied to estimate the parameters α, β, m and δ of the Burr Type XII distributed function described in equations (1) by using the method of least squares. Hence, we can find the estimates only through numerical procedures. We can estimate each parameters by the Maximum Likelihood Estimation and Least Square Estimation in the Burr Type XII Distribution Function (proposed SRGM).The estimated values of parameters are:

$$\hat{\alpha} = 121.4621,\ \hat{\beta} = 0.005657,\ \hat{\delta} = 1.908,$$
$$\hat{m} = 78.914,\ a = 94.435,\ r = 0.0255$$

Fig. 5 and Fig. 6 depict the fitting of the current estimated testing-effort by using Burr Type XII testing-effort function.

For these estimates, the optimality was checked numerically. Table III summarizes the experimental results of estimated parameters with their standard errors and 95 % confidence bound. Similarly, we plotted a fitted curve of the estimated mean value function with the actual software data in Fig. 7. Table IV shows the estimated values of parameters by using different SRGMs and comparison criteria. Similarly, smaller AE and MSE indicate least fitting errors and better performance. From Figures 5, 6, and 7 and the comparison criteria in Table IV, we conclude that this proposed model is good enough a give more accurate description of resource consumption during the source development phase and gives better fit in this experiment. Kolmogorov Smirnov goodness-of-fit test shows that our proposed SRGM described by an NHPP with $\hat{m}(t)$ fits pretty well at the 5% level of significance. Figure 8 shows that the estimated intensity functions $\hat{\lambda}(t)$ from equation (8).

In addition, substituting the estimated parameters α, β, m and δ in equation of $t_{\max}$, the testing effort function reaches the maximum at time t = 12.1664 debug days which corresponds to $w(t)$ = 5.5867 CPU hours and $W(t)$ = 45.6458 CPU hours. Besides, the number of errors removed up to this time $t_{\max}$ is 65.0016 and when t goes to infinity, the numbers of errors removed is 90.1894.

TABLE III
SUMMARY OF ESTIMATE OF NHPP MODEL PARAMETERS

Parameter	Estimate	Standard Error	95% Confidence Lower	95% Confidence Upper
a	94.4345	2.556930	89.10086	99.7682
r	0.02554	0.0016003	0.022030	0.02888

TABLE IV
COMPARISON RESULTS FOR THE SECOND DATA SET

Model	a	r	MSE
Burr Type XII Model	94.4345	0.02554	6.726
G-O Model [27]	137.072	0.0515445	25.33
Weibull Function [12]	87.0318	0.0345417	7.772
Delayed s-shaped Model [13]	88.6533	0.228148	6.3127
Logistic Function [22]	88.8931	0.0390591	25.228

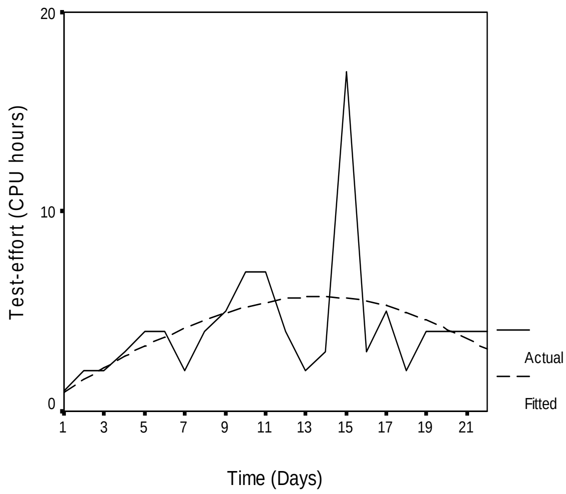

Figure 5. Observed/estimated current test-effort function vs. time

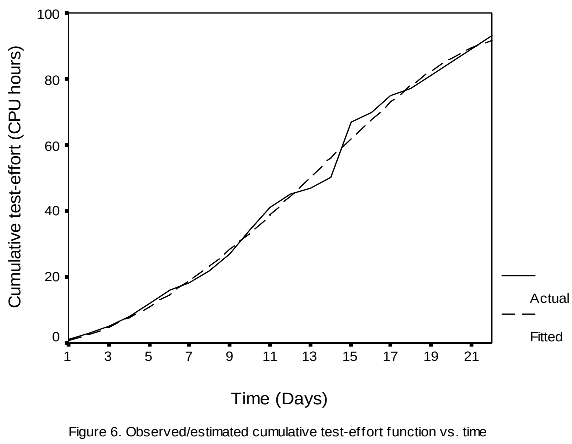

Figure 6. Observed/estimated cumulative test-effort function vs. time

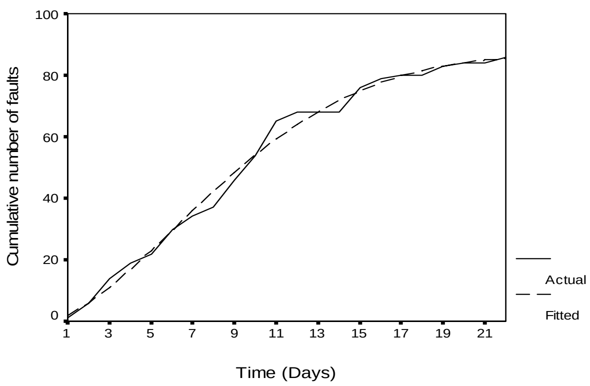

Figure 7. Observed/estimated cumulative number of faults vs. time

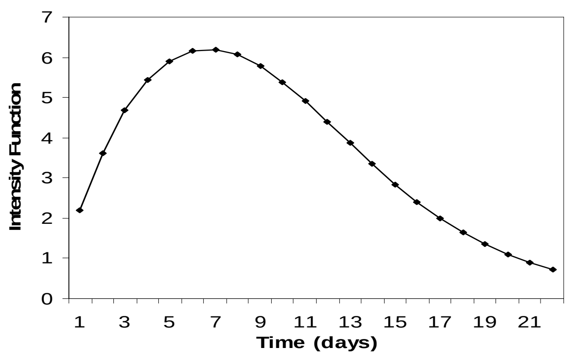

Figure 8. Estimated Intensity Function for Actual Data

Third Data Set: The third set of real data is from the study by [27]. The system is PL/1 data base application software, consisting of approximately 1,317, 000 lines of code. During the nineteen weeks experiments, 47.65 CPU times were consumed and about 328 software errors were removed. The original data report gives that the total cumulative number of detected faults after a long period of testing is 358 faults [27]. In order to estimate the parameters α, β, m and δ of the Burr Type XII distributed function; we fit the actual testing-effort data into equations (1) and (2) and solve it by using the method of least squares. Hence, we can find the estimates only through numerical procedures. These estimated parameters are:

$$\hat{\alpha} = 675.20762,\ \hat{\beta} = 0.000251,$$

$$\hat{\delta} = 1.11883,\ \ \hat{m} = 29.1946$$

Fig. 9 and Fig. 10 show the fitting of the estimated testing-effort. Here, the fitted curves are shown as a dotted line and solid line is actual software data. Using the estimated parameters α, β, m and δ , the other parameters a, r in (4) can be solved by MLE method for these failure data:

$$a = 565.6973, \quad r = 0.01964$$

For these estimates, the optimality was checked numerically. Table V summarizes the experimental results of estimated parameters with their standard errors and 95 % confidence bound.

Similarly, fitted curve of the estimated mean value function with the actual software data in Fig. 11 has been plotted. Also a comparison table of the estimates of this model along with other models with initial faults a and MSE is given in Table VI. From Figures 9, 10, and 11 and the comparison criteria shows that this SRGM is better fit than the other models for PL/1 application program. Kolmogorov Smirnov goodness-of-fit test shows that our proposed SRGM described by an NHPP with $\hat{m}(t)$ fits pretty well at the 5 % level of significance. Figure 12 shows that the estimated intensity functions $\hat{\lambda}(t)$ from equation (8).

TABLE V
SUMMARY OF ESTIMATE OF NHPP MODEL PARAMETERS

Parameter	Estimate	Standard Error	95% Confidence	
			Lower	Upper
a	565.6973	565.69734	444.9546	686.44005
r	0.01964	0.002826	0.013677	0.0255998

TABLE VI
COMPARISON RESULTS FOR THE THIRD DATA SET

Model	a	r	AE%	MSE
Burr Type XII Model	565.697	0.01964	58.02	116.40
Inflection s-shaped Model [27]	389.1	0.0935493	8.69	133.53
Exponential Model [27]	455.37	0.0267368	27.09	206.93
Weibull Function [22]	565.35	0.0196597	57.91	122.09
Rayleigh Function [22]	459.08	0.0273367	28.23	268.42
Exponential Function [12]	828.252	0.0117836	131.35	140.66
Delayed s-shaped Model [13]	374.05	0.197651	4.48	168.67
Delayed s-shaped Model with Rayleigh Function [12]	333.136	0.100415	6.93	798.49
S-Shaped Model with Logistic Function [22]	338.136	0.10004	5.54	242.79

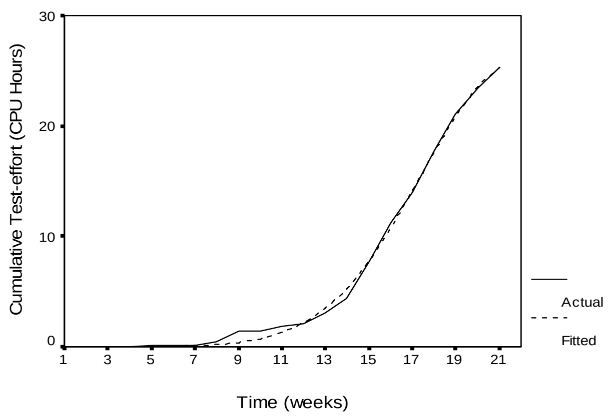

Figure 10. Observed/estimated Cumulative Test-effort Function vs. Time

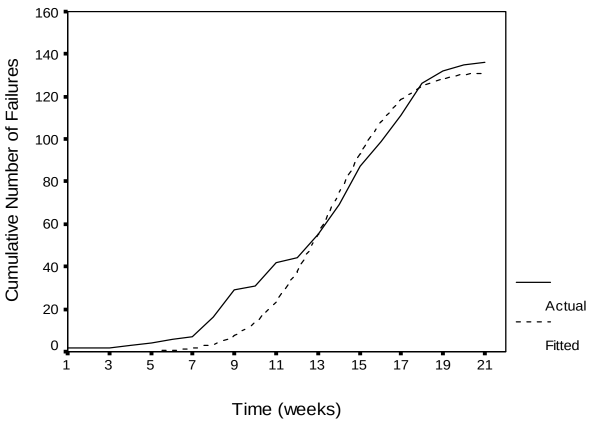

Figure 11. Observed/estimated Cumulative Number of Failures vs. Time

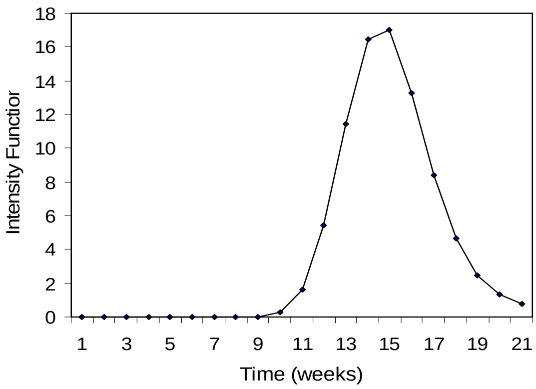

Figure12. Estimated Intensity Function for Actual Data

In addition, substituting the estimated parameters α, β, m and δ in equation of $t_{\max}$, the testing effort function reaches the maximum at time t = 10.8094 weeks which corresponds to *w(t)* = 6.5948 CPU hours and *W(t)* = 67.2605 CPU hours. Besides, the expected number of errors removed up to this time $t_{\max}$ is 414.7098 and when t goes to infinity, the expected numbers of errors removed is 565.6993.

VI. Optimal Release Policy for Software

Besides, developing software reliability growth models, it is also of great interest to know when to stop testing and the software for use. If the release of the software is unduly delayed, the manufacturer (Software developer) may suffer in terms of revenue loss, while a premature release may cost heavily in terms of fixes (removals) to be done after release and may even harm the manufacturer's reputation. Software release time problems have been classified in different way. One is, when to release software so that the cost incurred during the life cycle (consisting of the development and operational phases) of the software is minimized or the reliability is maximized [28].

A. Reliability Criteria

In general, the software-release time problem is associated with the reliability of a software system.. If the reliability of a software system is known to have reached an acceptable level, then we can obtain the right time to release this software. References [28] and [35] discussed the release problem by considering the software cost-benefit. The conditional reliability function after the last failure occurs at time t is:

$$R = R(x\,|\,t) = e^{-[m(t+x)-m(t)]} = e^{-a[e^{-rW(t)}-e^{-rW(t+x)}]} \qquad (25)$$

Differentiate $R(x\,|\,t)$ with respect to t, then $\frac{dR}{dt} \geq 0$. Hence R is a monotonic increasing function of t. Taking the logarithm on both side of the above equation, we obtain.

$$\log R = -\left[m(t+x) - m(t)\right] \qquad (26)$$

Solving (26) and (4) determines the testing time needed to reach a desired R. $R(t)$ is increasing in t ($0 < t < T_{LC}$). Using (26), one can get the required testing time needed to reach the reliability objective R or decide whether R is reached or not in a specified time interval.

- *Reliability Analysis For Real Data Sets*

First Data Set: From the previous estimated parameters: we know that $\hat{\alpha}$ =35.2418, $\hat{\beta}$ =0.0634, $\hat{m}$ =0.3261, $\hat{\delta}$ =11.2592, a = 133.7025, r = 0 .1553

Suppose this software system is desired that this testing would be continued till the operational reliability is equal to 0.85 (at Δt = 0.1), from equation (26) and equation (4), we get t = 20.3456 weeks. If the desired reliability is 0.90, then t = 21.1729 weeks. If the desired reliability is 0.95, then t = 22.7449 weeks. If the desired reliability is 0.99, then t = 27.2316 weeks.

Second Data Set: In second data set, from equation (26)and equation (4), for $\hat{\alpha}$ = 121.4621, $\hat{\beta}$ = 0.005657, $\hat{\delta}$ = 1.908, $\hat{m}$ =0.3261 = 78.9143, a = 94.4345, r = 0.02554 The testing time t = 17.8319 days is obtained, if we assume that the testing of this software system is desired to be continued till the operational reliability is equal to 0.85 (at Δt = 0.1). If the desired reliability is 0.90, then t = 20.8985 days. If the desired reliability is 0.95 (0.99), then t = 24.3609 (33.7320) weeks

Third Data Set: From the previous estimated parameters: $\hat{\alpha}$ = 675.20762, $\hat{\beta}$ = 0.000251, $\hat{\delta}$ = 1.11883, $\hat{m}$ =0.3261 = 29.1946, a = 565.6973, r = 0.0196, suppose this software system is desired that the testing would be continued till the operational reliability is equal to 0.8 (at Δt = 0.1), from equation (26) and equation (4), we get testing time t = 10.3198 weeks. If the desired reliability is 0.85, then t = 10.9415 days. If the desired reliability is 0.92 (0.98), then t = 12.2399 (14.9614) weeks

B. Cost-Reliability Criteria

This section discusses the cost model and release policy based on the cost-reliability criterion we can evaluate the total software cost by using cost criterion, the cost of testing-effort expenditures during software testing and development phase, and the cost of correcting errors before and after release as follows [18], [19], [38], [39]:

$$C(T)=C_1m(T)+C_2\left[m(T_{LC})-m(T)\right]+C_3\int_0^T w(x)dx \tag{27}$$

Where C_1 is the cost of correcting an error during testing, C_2 is the cost of *correcting* an error in operational use ($C_2>C_1$), C_3 is the cost testing per unit testing-effort expenditures and T_{CL} is the software life-cycle length.

Differentiating the above equation w. r. t. T and setting it to zero, we obtain

$$\frac{dC(T)}{dT}=C_1\frac{dm(T)}{dT}-C_2\frac{dm(T)}{dT}+C_3w(T)=0$$

$$\text{Or,}\quad \frac{dC(T)}{dT}=w(T)\left[-(C_2-C_1).a.r.e^{-rW(T)}+C_3\right] \tag{28}$$

Now

$$\frac{w(T)C_3}{C_2-C_1}=w(T).a.r.e^{-rW(T)}$$

$$\text{0r,}\quad \frac{C_3}{C_2-C_1}=a.r.e^{-rW(T)}\text{s}$$

$$=\frac{\lambda(T)}{w(T)}=r(a-m(T))$$

$$\therefore\frac{\lambda(T)}{w(T)}=\frac{C_3}{C_2-C_1}=r(a-m(T)) \tag{29}$$

Case 1: If $T=0$, then $m(0)=0$, and

$$\frac{\lambda(T)}{w(T)}=ar$$

Case 2: If $T\to\infty$, then $W(\infty)=\infty$, $m(\infty)=a(1-e^{-r\alpha})$

$$\frac{\lambda(T)}{w(T)}=a.r.e^{-r\alpha}$$

Therefore, $\frac{\lambda(T)}{w(T)}$ is monotonically decreasing in T.

If $\frac{\lambda(0)}{w(0)}=a.r.\le\frac{C_3}{C_2-C_1}$

Then,

$$\frac{\lambda(T)}{w(T)}\le\frac{C_3}{C_2-C_1}\text{ for }\quad 0<T<T_{LC}$$

Hence for this case, the optimal software release time $T^*=0$,

since $\frac{dC(T)}{dT}>0$ for $0<T<T_{LC}$.

If $\frac{\lambda(0)}{w(0)}=a.r.>\frac{C_3}{C_2-C_1}>\frac{\lambda(T)}{w(T)}=a.r.e^{-r\alpha}$,

Then, there exist a finite and unique solution.

To satisfying equation (29) that is,

$$\frac{\lambda(t)}{w(t)}=\frac{C_3}{C_2-C_1}=r(a-m(T))$$

$$=a.r.e^{-rW(T)}$$

$$=a.r.e^{-r\alpha[1-(1+(\beta T)^{\delta})^{-m}]}$$

Rearranging this equation gives,

$$e^{r\alpha[1-(1+(\beta T)^{\delta})^{-m}]}=\frac{a.r(C_2-C_1)}{C_3}$$

$$or\ \ r\alpha[1-(1+(\beta T)^{\delta})^{-m}]=\ln\left[\frac{a.r(C_2-C_1)}{C_3}\right]$$

$$or\ (\beta T)^{\delta}=\left[\frac{r.\alpha}{r.\alpha-\ln\left[\frac{a.r(C_2-C_1)}{C_3}\right]}\right]^{1/m}-1$$

$$or\ T=\frac{1}{\beta}\left[\left[\frac{r.\alpha}{r.\alpha-\ln\left[\frac{a.r(C_2-C_1)}{C_3}\right]}\right]^{1/m}-1\right]^{1/\delta} \tag{30}$$

Minimizes $C(T)$

Because $\frac{dC(T)}{dT}<0$ for $0<T<T_0$ and

$$\frac{dC(T)}{dT}>0\ \text{ for }\ T_0<T<T_{LC},$$

The minimum of $C(T)$ is at $T=T_0$ for $T_0<T$, because $\frac{d^2C(T)}{dT^2}>0$, then $C(T)$ is a convex function.

Here our goal is to minimize the total software cost under the consideration of desired software reliability, and the optimal software release time is obtained. That is, the optimal software release problem can be formulated as follows.

Minimize $C(T)$ (31)
Subject to $R(x|t)\ge R_0$,
$T\ge 0$ for $C_2>C_1>0$, $C_3>0$, $x\ge 0$,
$0<R_0<1$.

Then, we can obtain the solutions for the cost reliability optimum software release time:

$$T^*=\max\,[T_0, T_1]$$

Where T_0 is finite and the unique solution T of (31), T_1 is finite and unique T Satisfying $R(x|t)=R_0$, $0<R_0<1$.

Theorem: We assume that;
$C_1>0$, $C_2>0$, $C_3>0$, $C_2>C_1$, $x>0$, $0<R_0<1$, then

- If $\frac{\lambda(0)}{w(0)}>\frac{C_3}{C_2-C_1}$ and $\frac{\lambda(T)}{w(T)}=\alpha.r..e^{-r\alpha}<\frac{C_3}{C_2-C_1}$
  then $T^*=\max\,[T_0, T_1]$ for $R(x|t)<R_0<1$ or
  $T^*=T_0$ for $0<R<R(x|t=0)$.

- If $\dfrac{\lambda(0)}{w(0)} \le \dfrac{C_3}{C_2 - C_1}$ then,

  $T^* = T_1$ for $R(x|0) < R_0 < 1$ or
  $T^* = 0$ for $0 < R_0 < R(x|0)$

- If $\dfrac{\lambda(0)}{w(0)} \ge \dfrac{C_3}{C_2 - C_1}$ then $T^* \ge T_1$

for $R(x|0) < R_0 < 1$ or $T^* \ge 0$ for $0 < R_0 \le R(x|0)$.

To illustrate the above item, we use again the first real data set for numerical example on optimal software release problem.

Numerical Example: From the previously estimated parameters it is known that

$\hat{\alpha}$ =35.2418, $\hat{\beta}$ = 0.0634, $\hat{m}$ =0.3261, $\hat{\delta}$ =11.2592, a = 133.7025, r = 0 .1553

Also assume $C_1 = 10$, $C_2 = 50$, $C_3 = 100$, $T_{LC} = 100$, $R_0 = 0.90$ $x = 0.1$. Then we get the optimal release time T_0 estimated as 17.62038 based on minimizing $C(T)$ of equation (27), and T_1 is estimated as 21.1729 based on satisfying the reliability criterion of $R(t+x|\ t) = R_0$. Moreover, since

$$\frac{\lambda(0)}{w(0)} > \frac{C_3}{C_2 - C_1} \text{ and } \frac{\lambda(T)}{w(T)} = \alpha . r .. e^{-r\alpha} < \frac{C_3}{C_2 - C_1}$$

and $R(x|0) < R_0$,

The T^* is estimated as max {17.6204, 21.1729} = 21.1729 weeks. The optimal total software cost $C(T^*)$ =4853.35 and the achieved software reliability $R(21.1729+x\ (= 0.1)/\ 21.1729)$ is 0.90.

VII. Conclusion

Software reliability measurement during testing phase is essential for examining the degree of quality or reliability of developed software systems.

In this paper, we have discussed a software reliability growth model (SRGM) based on NHPP, which incorporates Burr Type XII testing-effort expenditure. We have also discussed the optimal release-time determination based on cost and reliability criteria within our framework. We conclude that the Burr Type XII testing-effort function can be used to represent a software reliability growth model. Computation results show that the testing-effort function proposed here, gives a good fault predictive capability and better performance for three actual software failures data set. We also conclude that the proposed model has a better goodness of fit as compared to the other existing models. Burr Type XII testing-effort curve gives better estimates than Exponential, Rayleigh, and Weibull type consumption curves.

Acknowledgment

The authors would like to thank the referee for his valuable comments and suggestions that helped us to improve the presentation of the paper.

References

[1] A. A. Abdel-Ghaly, G. R. Al-Dayian, and F. H. Al-Kashkari, "The Use of Burr Type XII Distribution on Software Reliability Growth Modelling," *Microelectronics and Reliability*, Vol. 37 (2), pp. 305-313, 1997.

[2] N. Ahmad, M. U. Bokhari, S. M. K. Quadri, and M. G. M. Khan, "The Exponentiated Weibull software reliability growth model with various testing-efforts and optimal release policy: a performance analysis," *International Journal of Quality and Reliability Management*, Vol. 25 (2), pp. 211-235, 2008.

[3] N. Ahmad, M. G. M. Khan, S. M. K. Quadri, and M. Kumar, "Modelling and Analysis of Software Reliability with Burr Type X Testing-Effort and Release-Time Determination," *Journal of Modelling in Management*, Vol. 4 (1), 28 – 54, 2009.

[4] N. Ahmad and A. Islam, "Optimal Accelerated Life Test Designs for Burr Type XII Distributions under Periodic Inspection and Type Censoring," *Naval Research Logistics*, Vol. 43, pp. 1049-1077, 1996.

[5] H. Ascher and H. Feingold, *Repairable Systems Reliability: Modeling, Inference, Misconceptions and their Causes*, Marcel Dekkes, New York, 1984.

[6] M. U. Bokhari and N. Ahmad, "Analysis of a software reliability growth models: the case of log-logistic test-effort function," *In: Proceedings of the 17th IASTED International Conference on Modeling and Simulation* (*MS'2006*), Montreal, Canada, pp. 540-545, 2006.

[7] M. U. Bokhari, S. M. K. Quadri, and N. Ahmad, "Analysis of Non-homogeneous Poisson Process Software Reliability Growth Model with Burr Type XII Testing-effort Function," presented in *the International Conference on Modelling and Optimization of Structures, Processes and Systems (ICMOSPS'07)*, Jan, 22 – 24, Durban, South Africa, 2007.

[8] M. U. Bokhari, M. I. Ahmad, and N. Ahmad, "Software Reliability Growth Modeling for Burr Type XII Function: Performance Analysis", presented in *the International Conference on Modelling and Optimization of Structures, Processes and Systems (ICMOSPS'07)*, Jan, 22 – 24, Durban, South Africa, 2007a.

[9] I. W. Burr, "Cumulative Frequency Function," Annals of Mathematical Statistics, Vol. 13, pp .215-232, 1942.

[10] A. L. Goel, and K. Okumoto, "Time Dependent Error-Detection Rate Model for Software Reliability and Other Performance Measures," *IEEE Transactions on Reliability*, Vol. R- 28, No. 3, pp. 206-211, 1979.

[11] C. Y. Huang and S.Y. Kuo, "Analysis of Incorporating Logistic Testing Effort Function Into Software Reliability Modeling," *IEEE Transactions on Reliability*, Vol. 51, No. 3, pp. 261-270, 2002.

[12] C. Y. Huang, J. H. Lo, and S. Y. Kuo, "A Pragmatic Study of Parametric Decomposition Model for Estimating Software Reliability Growth," in proceeding 9th *International Symposium Software Reliability Engineering (ISSRE' 98)*, PP. 111-123, 1998.

[13] C. Y. Huang, S. Y. Kuo, and I. Y. Chen, "Analysis of Software Reliability Growth Model with Logistic Testing-Effort Function," Proceeding of the 8th *International Symposium on Software Reliability Engineering (ISSRE'97)*, Albuquerque, New Maxico, pp. 378-388, 1997.

[14] C. Y. Huang, "Performance analysis of software reliability growth models with testing-effort and change-point", *Journal of Systems and Software*, Vol. 76, pp. 181-194, 2005.

[15] C. Y. Huang, S. Y. Kuo, and M. R. Lyu, "An assessment of testing-effort dependent software reliability growth

models," *IEEE Transactions on Reliability*, Vol. 56, no.2, pp. 198-211, 2007.
[16] C. Y. Huang, S. Y. Kuo, and M. R. Lyu, "Effort-Index-Based Software Reliability Growth Modeling and Performance Assessment," Proceeding *24th Annual International Computer Software and Application Conf. (COMPSAC2000)*, pp. 454-459, 2000.
[17] C. Y. Huang, J. H. Lo, S. Y. Kuo, and M. R. Lyu, "Software Reliability Growth Modeling and Cost Estimation Incorporating Testing-Effort and Efficiency," Proceeding *10th International Symposium Software Reliability Engineering (ISSRE'1999)*, pp. 62-72, 1999.
[18] C. Y. Huang, J. H. Lo, S. Y. Kuo, and M. R. Lyu, "Optimal Allocation of Testing Resource Considering Cost, Reliability, and Testing Effort," Proceeding. *10th IEEE Pacific Rim International Symposium as Dependable Computing (PRDC'04)*, pp. 1-9, 2004.
[19] P. K. Kapur and R. B. Garg, "Cost Reliability Optimum Release Policies for a Software System with Testing Effort," *OPSEARCH*, Vol. 27, No. 2, pp. 109-116, 1990.
[20] P. K. Kapur and S. Younes, "Modeling an Imperfect Debugging Phenomenon With Testing Effort," In Proceeding *5th International Software Reliability Engineering (ISSRE)*, Pp. 178-183, 1994.
[21] P. K. Kapur and R. B. Garg, "Modeling an Imperfect Debugging Phenomenon in Software Reliability," *Microelectronics and Reliability*, Vol. 36, pp. 645-650, 1996.
[22] S. Y. Kuo, C. Y. Huang, and M. R. Lyu, "Framework for Modeling Software Reliability, Using Various Testing-Efforts and Fault Detection Rates," *IEEE Transactions on Reliability*, Vol. 50, No. 3, pp 310-320, 2001.
[23] M. R. Lyu (ed.), *Handbook of Software Reliability Engineering*, IEEE Computer Society Press, Los Alamitos, California, McGraw-Hill, 1996.
[24] M. R. Lyu and A. Nikora, "Applying Software Reliability Model More Effectively," *IEEE Software*, pp. 43-52, 1992.
[25] J. D. Musa, *Software Reliability More Reliable Software Faster Development and Testing*, McGraw-Hill, 1999.
[26] J. D. Musa, A. Iannino, and K. Okumoto, *Software Reliability Measurement, Prediction, Application*, McGraw-Hill, 1987.
[27] M. Ohba, "Software Reliability Analysis Model," *IBM Journal Research Develop.*, Vol. 28, No. 4, pp. 428-443, 1984.
[28] K. Okumoto and A. L. Goel, "Optimum Release Time for Software System Based on Reliability and Cost Criteria," *Journal System. Software*, Vol. 1, pp. 315-318, 1980.
[29] R. N. Rodriguez, "A Guide to the Burr Type XII Distributions," *Biometrika*, Vol. 64, pp.129-134, 1973.
[30] R. N. Rodriguez, "Burr Distribution," *Encyclopedia of Statistical Science*, Vol.1, pp.335-340, 1982.
[31] J. Tian, P. Lu, and J. Palma, "Test-Execution-Based Reliability Measurement and Modeling for Large Commercial Software," *IEEE Transactions on Software Engineering*, Vol. 21, No. 5, pp. 405-414, 1995.
[32] Y. Tohma, R. Jacoby, Y. Murata, and M. Yamamoto, "Hyper-Geometric Distribution Model to Estimate the Number of Residual Software Fault," Proceeding of *COMPSAC-89, IEEE CS Press*, Orlando, pp. 610-617, 1989.
[33] D. R. Wingo, "Maximum Likelihood methods for fitting the Burr Type XII Distribution to Life Test Data," *Biometrical Journal (Zeitschrift fuer Math. Meth in de Biowissenschaften)*, Vol. 25, PP. 77-88, 1983.
[34] D. R. Wingo, "Maximum Likelihood Methods for Fitting the Burr Type XII Distribution to Multiply (Progressively) Censored Life Test Data," *Metrika*, Vol. 40, pp. 203-210, 1993.
[35] M. Xie, *Software Reliability Modeling*, World Scientific Publication, Singapore, 1991.
[36] S. Yamada, "Software Quality/Reliability Measurement and Assessment: Software Reliability Growth Models and Data Analysis," *Journal Information Processing*, Vol.14, pp. 254-266, 1991.
[37] S. Yamada and H. Ohtera, "Software Reliability Growth Model for Testing Effort Control," *European Journal Operation Research*, 46, pp. 343-349, 1990.
[38] S. Yamada and S. Osaki, "Cost-Reliability Optimal Release Policies for Software Systems," *IEEE Transactions on Reliability*, Vol.R-34, No. 5, pp. 422-424, 1985.
[39] S. Yamada, J. Hishitani, and S. Osaki, "Software Reliability Growth Model with Weibull Testing –Effort: A model and Application," *IEEE Transactions on Reliability*, Vol. R-42, pp. 100-105, 1993.
[40] S. Yamada, H. Ohtera, and H. Narihisa, "Software Reliability Growth Models with Testing-Effort," *IEEE Transactions on Reliability*, Vol. R-35, No. 1, pp. 19-23, 1986.

Mohammad Ubaidullah Bokhari is presently working as an Associate Professor as well as Chairman in the Department of Computer Science, AMU, Aligarh, (INDIA) and Principal Investigator (PI) of the ambitious project, NMEICT ERP Mission Project (Govt. of India).

He graduated from the Magadh University, Bodh Gaya, India in 1985. He obtained his M.C.A. and Ph. D. degree in computer science from the Aligarh Muslim University in 1989 and 2006, respectively. He received the CWC merit scholarship (1986-1988) and University Grant Commission Scholarship (1988-1989). He had also worked as Associate Professor and Director of Studies in Australian Institute of Engineering & Technology, Victoria, Melbourne (Australia).

Dr. Bokhari has a vast teaching experience of more than twenty three years. His research interests include Software Reliability, Security and Cryptography, Software Engineering, Database Management System and E-Learning. He has published more than 70 research papers in reputed National and International Journals and Conferences and authored 5 books on different areas of computer science. He is lifetime member of Computer Society of India (C.S.I.) and member of IEEE.

Nesar Ahmad is an Associate Professor in University Department of Statistics and Computer Applications at Tilka Manjhi Bhagalpur University, Bhagalpur, India. He received the B. Sc. degree in Mathematics from Bihar niversity, India, in 1984 and the M. Sc., M. Phil., and Ph. D. degree in Statistics from Aligarh Muslim University, Aligarh, India, in 1987, 1990, and 1993, respectively.

From 1995 to 1996 he was a research associate of UGC/CSIR at Aligarh Muslim University, Aligarh. He has been a Lecturer in Statistics from January 2006 to December 2009 at the University of the South Pacific, Suva, Fiji Islands. After working five months as a lecturer at Poona College, Pune, he joined the University Department of Statistics and Computer Applications at Tilka Manjhi Bhagalpur University, Bhagalpur, India in December 1996. He has about 16 years of experience in teaching and research. His research interests include life testing, statistical modeling, reliability analysis, software reliability, software engineering and optimization. He is an author and coauthor of more than 40 journal papers and conference papers in these areas.

Dr. Ahmad is life member of Indian Science Congress and Bihar Journal of Mathematical Society, and regular member of Aligarh Statistical Association.