Proceedings of the National Conference; INDIACom-2007.
Computing for Nation Development, February 23 – 24, 2007
Bharati Vidyapeeth's Institute of Computer Applications and Management, New Delhi

On Fuzzy Approximations to Release Time Problems

P. C. Jha **P. K. Kapur**
Department of Operational Research, Faculty of Mathematical Sciences, University of Delhi
Deepali Gupta
Department of Mathematics, Jaypee Institute of Information Technology, (Deemed University)

ABSTRACT:
The importance of testing phase in Software Development Life Cycle (SDLC) cannot be undermined. A timely release of the software is also necessary due to many reasons ranging from marketing considerations to cost. The optimal testing time is a function of several factors viz. size, level of reliability desired, skill and efficiency of testing personal, market environment, penalty cost due to delay in delivery and\or loss due to delay in release and finally penalties/ warranty cost in process failures. In this paper, the optimal release policy for software system maximizing expected gain and simultaneously achieving a given level of failure intensity incorporating the effect of testing effort expenditure is formulated. While fixing budgetary requirement and failure intensity constraint, management may not be in a position to provide precise targets or goals. This leads to fuzziness in objectives as well as constraints and hence fuzzy optimization approach has been proposed to overcome such difficulties.

1. INTRODUCTION

Several software release time problems have been discussed and solved in different ways in literature by many authors. One of these is to find release time so that the total cost incurred during remaining phases (i.e. testing and operational) of the SDLC is minimized [5,8]. Some of the release time problems are based upon reliability criterion alone. Models that minimize the number of remaining faults in the software or the failure intensity also fall under this category [1,5]. Release time problems have also been formulated for minimizing cost with minimum reliability requirement or maximizing reliability subject to budgetary constraint [5]. Bi-criterion release policy [3] maximizes reliability and minimizes cost subject to reliability requirement and testing resource availability constraints. Mathematical programming methods have been used to find solutions to such problems. In this paper, the optimal release policies for software system maximizing expected gain and simultaneously achieving a given level of failure intensity incorporating the effect of testing effort expenditure is formulated. While fixing intensity and gain requirements, management may not be in a position to provide precise targets or goals. They are rather ready to accept a compromise solution. Such imprecise statements regarding reliability requirement and testing resource availability lead to fuzziness in objectives as well as constraints and hence fuzzy optimization approach is used to overcome such difficulties.

2. SOFTWARE RELIABILITY GROWTH MODEL

All these release time problems discussed above require a functional relationship between the fault exposure and time (preferably calendar time). Software Reliability Growth Models (SRGMs) play an important role due to their ability to predict the fault detection/removal phenomenon during testing. Several classes of SRGMs have been proposed and validated on real life test data in the literature. One group of models that has been widely used is the Non homogeneous Poisson Process (NHPP) models. Models under this group are distinguished from each other by the form of the mean value functions of NHPP that describes the failure phenomenon [G-O, 1979]. The functional forms are either exponential or S-shaped. There exist other exponential and S-shaped models for counting the number of failures or removals without underlying NHPP [Musa, 1987].

Notations

a : Expected number of faults in the software
b : Proportionality constant
r : Fraction of independent faults
$w(t)$:Current testing effort expenditure at testing time t,

$$W(t) = \int_0^t w(x)dx$$

$m(t)$: Number of faults removed in $(0,t]$
T : Total testing time
C_b : Total allocated budget
T_w : warranty period

3. SRGM WITH TESTING EFFORT

A Software Reliability Growth Model (SRGM) explains the time dependent behavior of fault detection\removal phenomenon. Several SRGMs have been proposed in software reliability literature under different set of assumptions and testing environment, yet more are being proposed. The proposed SRGM in this paper takes into account the time dependent variation in testing effort. The testing effort (resources) that govern the pace of testing for almost all the software projects are [17]:

(a) Manpower, which includes
  (i) Failure identification personnel
  (ii) Failure correction personnel.

(b) Computer time.

The key function of manpower engaged in software testing is to run test cases and compare the test results with desired specifications. Any departure from the specifications is termed as a failure. On a failure the fault causing it is identified and then removed by failure correction personnel. The computer facilities represent the computer time, which is necessary for failure identification and correction. The influence of testing effort has also been included in some SRGMs [9,10,13,19,22]. The SRGM with testing effort developed in this paper is based upon the following primary assumptions. Most of the Non Homogeneous Poisson Process (NHPP) models discussed in the literature presume that the fault detection process is based upon the following basic assumptions.

1. Software is subject to failures at random time caused by faults present in it.
2. On a failure, the fault causing that failure is immediately removed and no new faults are introduced.
3. The expected number of faults removed in $(t, t+\Delta t)$ is proportional to the expected number of faults remaining.

4. MODEL FORMULATION

The parameters of an SRGM should be interpretable in terms of software testing phenomenon and one such popular model is due to Ohba [18]. Other similar models are due to Bittanti et al. [4] and Kapur and Garg [12]. Kapur, Jha and Bardhan[ ] proposed that the fault removal rate increases with time and assumed the presence of two types of faults in the software. The distinctive assumptions of the model can be summarized as follows:

5. The fault detection rate with respect to testing effort intensity is proportional to the current fault content in the software and the proportionality increases linearly with each additional fault removal.
6. Faults present in the software are of two types: mutually independent and mutually dependent. The mutually independent faults lie on different execution paths and mutually dependent faults lie on the same execution path. The second type of faults is detectable if and only if faults of first type are already detected.

Under the above assumptions following differential equation describe the model

$$\frac{\frac{d}{dt}m(t)}{w(t)} = \phi(t)(a - m(t)) \qquad \ldots(1)$$

Where $\phi(t) = b(r + (1-r)\frac{m(t)}{a}$

Solving equation (1) with the initial condition that, at $t = 0$, $X(t) = 0$, $m(t) = 0$ we get

$$m(t) = \frac{a(1-e^{-bW(t)})}{1+\frac{1-r}{r}e^{-bW(t)}} \qquad \ldots(2)$$

Depending upon the value of *r*, the SRGM (2) can describe both exponential and S-shaped growth curves. To describe the behavior of testing effort, either Exponential or Rayleigh function has been used [13,20,22]. Both can be derived from the assumption that, "the testing effort rate is proportional to the testing resource available".

$$\frac{dW(t)}{dt} = c(t)[\alpha - W(t)] \qquad \ldots(3)$$

Where $c(t)$ is the time dependent rate at which testing resources are consumed, with respect to the remaining available resources. Solving equation (3) under the initial condition $X(0) = 0$ we get

$$W(t) = \alpha\left[1 - \exp\int_0^t c(k)dk\right] \qquad \ldots(4)$$

When $c(t) = \beta$ a constant

$$W(t) = \alpha(1 - e^{-\beta t}) \qquad \ldots(5)$$

If $c(t) = \beta.t$, (1) gives a Rayleigh type curve

$$W(t) = \alpha(1 - e^{-\beta\frac{t^2}{2}}) \qquad \ldots(6)$$

Huang et al. [9] developed an SRGM, based upon NHPP with logistic testing effort function. SRGM with logistic testing effort function provides better result on some failure data sets. The cumulative testing effort consumed in the interval (0,t] has the following form

$$W(t) = \frac{\alpha}{1 + \beta e^{-ct}} \qquad \ldots(7)$$

Where α, β and c are constants.

More flexible and general testing effort function can be obtained using Weibull function and the cumulative testing effort consumed in the interval (0,t] has the following form

$$X(t) = \alpha(1 - e^{-(\frac{t}{c})^{\beta}}) \qquad \ldots(8)$$

Where α, β and c are constants.

5. RELEASE TIME PROBLEM

One of the most important decision that the management must take about testing is to determine the time when to stop testing and release the software known as release time problem. Here we consider maximization of expected gain and simultaneously achieving a given level of failure intensity incorporating the effect of testing effort expenditure. Since the gain function is derived from the cost function, we first discuss the cost function.

Cost Function

Cost function includes cost of testing, removing faults during testing and cost of failure and removal of faults during operational phase. Testing is performed under controlled environment. Cost involved in testing and removing of faults and documentation during testing can be estimated but real difficulties arise in quantifying the cost of a failure at the user end. A reasonably realistic approach of warranty cost is considered. The cost of failure and removal of a fault during a fixed warranty period after the release of the software is included. All these costs lead to the following form of the cost functions:

$C[W(T)]=C_1W(T)+C_2m[W(T)]+C_3(m[W(T+T_w)]-m[W(T)])$ …(9)

Hence release time problem now can be stated as:

Minimize C[W(T)]

Subject to $\lambda[W(t)] \le \lambda_0$ …(P1)

$T \ge 0$

Or

Maximize $g[W(T)] =-(C_3- C_2)\, m[W(T)] - C_1\, W(T)$

Subject to $\lambda[W(T)] \le \lambda_0$ …(P2)

$T \ge 0$

Where $m(T) = \dfrac{a\left(1-e^{-bW(T)}\right)}{1+\beta e^{-bW(T)}}$,

and $\lambda(T) = \dfrac{ab(1+\beta)w(T)e^{-bW(T)}}{\left(1+\beta e^{-bW(T)}\right)^2}$

C_1 : cost per unit testing effort expenditure and

$C_2(C_3)$: cost of removing an error before(after) releasing the software system for use.

Now if management wants to fix maximum levels (goal) for the failure intensity as well as the minimum gain, the problem can be given as:

Maximize $g[W(T)] =- (C_3- C_2)\, m[W(T)] - C_1\, W(T)$

Subject to $\lambda[W(T)] \le \lambda_0$

$g[W(T)] \ge g_0$ … (P3)

$T \ge 0$

If maximum desired levels of the failure intensity as well as the minimum desired level of gain are pre-fixed by the management, which may result conflicting in nature. The problem can be solved using Goal programming approach. However a solution using goal programming is fairly sensitive to weighting vectors (relative importance) and aspiration levels (goals), multi criterion crisp optimization offer no mechanism to handle the uncertainty quantitatively. Arriving at precise values for minimum desired level of gain to achieve desired level of failure intensity during early stages of testing is not possible. Such imprecise statements lead to fuzziness in the objective as well as constraint functions. To overcome them, fuzzy optimization approach can be used. In fuzzy optimization, the membership function that measures degree of satisfaction for each of the fuzzy objective and constraint function are defined. The major difficulties in fuzzy optimization are to define an appropriate membership function for each objective as well as constraint function. Several ways to define a membership function has been proposed in literature but we are adopting Zimmermann's approach [10] for defining the membership function ($\mu_i(T)$ corresponding i^{th} objective function i =1,2) in the above problem as follows:

$$\begin{Bmatrix} 1 & \text{if} \quad g[W(T)] \ge g^0 \\ \mu_1(T) = \begin{cases} \dfrac{\left[g[W(T)] - g^*\right]}{g^0 - g^*} & \text{if} \quad g^* \le g[W(T)] < g^0 \end{cases} \\ 0 & \text{if} \quad g[W(T)] < g^* \end{Bmatrix} \quad \ldots(10)$$

Where and g^0 and g^* are upper and lower tolerance levels for the gain objective function.

Similarly, since the constraint function is of type $\lambda[W(T)] \le \lambda_0$ we have following membership function $\mu_i(T)$ assigning a tolerance levels for the intensity constraint $\lambda_0 > \lambda_0'$ as follows:

$$\begin{Bmatrix} 1 & \text{if} \quad \lambda[W(T)] \le \lambda_0 \\ \mu_2(T) = \begin{cases} \dfrac{\left[\lambda_0' - \lambda[W(T)]\right]}{\lambda_0' - \lambda_0} & \text{if} \quad \lambda_0 < \lambda[W(T)] \le \lambda_0' \end{cases} \\ 0 & \text{if} \quad \lambda[W(T)] > \lambda_b' \end{Bmatrix} \quad \ldots(11)$$

Now the fuzzy optimization formulation of the above problem is given as:

Maximize α

Subject to $(\mu_i T) \ge \alpha$, $i=1,2$ …(P4)

$\alpha \ge 0$, $T \ge 0$

The problem (P4) after incorporating parameter values can be solved by mathematical programming approach using LINGO or any other standard software package. The details of solution technique through numerical illustration would be explained in full-length paper.

6. NUMERICAL EXAMPLE

It is assumed that parameters α, β, *c*, *a*, *b* and *r* of SRGM (2) have already been estimated from the testing data sets. Further it is assumed that values of C_1, C_2 , C_3 , and T_W are known. The release time problem based on the following data could be analyzed.

A = 455, b =0.508267, C_1 = 2, β = 3.1358,C_2 = 5,

C_3 = 15, x = 5, T_W = 10.. Further the cost g_0 = 4600, λ_0 = .01 and g'_0 = 4500, λ'_0 = .0115. Using above values of various

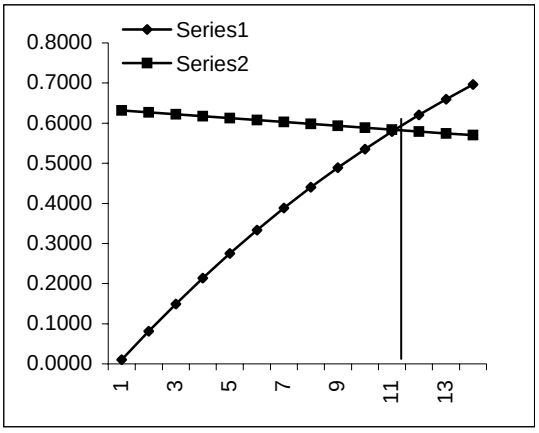
parameters and constants, solution of the problem is obtained using LINGO as follows:

T^*=137, λ_0^* = .01112, g(T*) = 4574 R(x / T^*) = 0.900716,

C (T^*) = 8556.511.

In figure-1, series-1 and series-2 show the fuzzy measures of Intensity and gain functions respectively.

Figure-1:



Reference:

1. Goel AL, Okumoto K. "Time dependent error detection rate model for software reliability and other performance measures" *IEEE Transactions on Reliability* 1979; R-28(3): 206-211.
2. Kapur PK, Aggarwal S, Garg RB. "Bicriterion release policy for exponential software reliability growth model" *Recherche Operationnelle – Operations Research* 1994; 28: 165-180.
3. Kapur PK, Garg RB. "Optimal release policies for software systems with testing effort" *Int. Journal System Science*, 1990; 22(9), 1563-1571.
4. Kapur PK, Garg RB, Kumar S. "Contributions to hardware and software reliability" 1999; ingapore: World Scientific Publishing Co. Ltd.
5. Kapur PK, Jha PC, Bardhan AK. "Optimal allocation of testing resource for a modular software" Asia-pacific journal of operational research, 2004; 24 (2), 1-22.
6. Ohba M. "Software reliability analysis models" IBM Journal of Research and Development1984;28: 428-443.
7. Okumoto K, Goel AL. "Optimal release time for computer software" *IEEE Transactions On Software Engineering*. 1983; SE-9 (3): 323-327.
8. Yamada S, Ohba M., Osaki S. "S-shaped software reliability growth modelling for software error detection" *IEEE Trans. On Reliability*, 1983; R-32(5): 475-484.
9. Zimmermann, H.J., "Fuzzy set theory and its applications", 1991; Academic Publisher.