RAIRO. RECHERCHE OPÉRATIONNELLE

P. K. KAPUR
SANJAY AGARWALA
R. B. GARG

# Bicriterion release policy for exponential software reliability growth model

*RAIRO. Recherche opérationnelle*, tome 28, n° 2 (1994), p. 165-180

<http://www.numdam.org/item?id=RO_1994__28_2_165_0>

Recherche opérationnelle/Operations Research
(vol. 28, n° 2, 1994, p. 165 à 180)

# BICRITERION RELEASE POLICY FOR EXPONENTIAL SOFTWARE RELIABILITY GROWTH MODEL (*)

by P. K. KAPUR ([1]), Sanjay AGARWALA ([1]) and R. B. GARG ([2])

Communicated by Shunji OSAKI

Abstract. – *In this paper we propose a new software release policy called the Bicriterion Release policy. This policy optimizes two conflicting objectives, namely, software reliability and total expected cost subject to budget and reliability constraints. Various existing policies based on reliability or cost criteria are particular cases of the proposed policy. Post optimality analysis is also performed to help a software developer in fixing the software life cycle length and the reliability goal. The results are illustrated by numerical examples.*

Keywords: Bicriterion release policy, life cycle, software reliability.

Résumé. – *Nous proposons dans cet article une nouvelle politique de mise à disposition d'un logiciel appelée politique bicritère de mise à disposition. Cette politique optimise les deux objectifs antagonistes suivants : la fiabilité du logiciel et le coût total espéré soumis aux contraintes de budget et de fiabilité. La politique proposée ici contient comme cas particuliers diverses politiques existantes fondées sur des critères de fiabilité ou de coût. Nous effectuons en outre une analyse post-optimale pour aider le développeur de logiciels à établir la longueur du cycle de vie du logiciel et son objectif de fiabilité. Des exemples numériques illustrent les résultats.*

Mots clés : Politique bicriterion de mise à disposition, cycle de vie, fiabilité du logiciel.

## 1. INTRODUCTION

Several software reliability growth models (SRGMs) based on non-homogeneous Poisson process (NHPP) have been developed over the last decade to estimate the remaining error content in a software [1, 5, 7, 11]. These SRGMs have also been used to decide upon the potential release time of the software. Several stopping rules exist in this regard in the literature [2, 3, 10]. Moreover, release policies based on cost and reliability/intensity criteria have also been discussed [4, 6, 7, 9, 12]. The emphasis in these release policies has been on minimizing software cost over the software life

(*) Received June 1992.
([1]) Department of Operational Research, University of Delhi, Delhi, India.
([2]) Delhi University Computer Centre, University of Delhi, Delhi, India.

Recherche opérationnelle/Operations Research, 0399-0559/94/02/$ 4.00

cycle subject to achieving a given (desired) level of reliability/intensity at the time of release of the software to the user. At times it may happen that the minimum cost which a software developer will have to incur (based on the release policy) may exceed the allocated budget for the software. Moreover, once reliability/intensity objective is fixed, the main emphasis is on minimizing the cost, whereas maximizing the reliability (minimizing intensity) may be equally or even more important. Thus depending on the type of the project being undertaken emphasis could be on maximizing reliability subject to the budgetary constraint or on minimizing cost subject to the reliability constraint.

The proposed policy maximizes reliability and minimizes cost simultaneously subject to reliability and cost (budget) constraints. Such a policy is termed as the *Bicriterion Release Policy.* This release policy gives a flexibility to the software developer to find out the optimal release time based on his priority (relative importance) in respect of reliability and cost components. If reliability is more important irrespective of the cost then higher weights may be attached to reliability as in the case of safety critical projects. Similarly, for business application software packages, more weights may be attached to cost (*see*, numerical example for illustration). It may be further noted that the twin problems of minimizing cost subject to reliability constraint and maximizing reliability subject to budgetary constraint are particular cases of the proposed policy. In this paper, we have discussed this policy for exponential SRGM and have illustrated various cases with numerical examples. We have further carried out the post optimality analysis to show how changes in software life cycle length affects the optimal release time. The analysis presented may help a software developer in fixing the software life cycle length as well as the reliability objective.

## 2. EXPONENTIAL SRGM

### Assumptions

1. The error detection phenomenon in the software is modelled by NHPP.
2. Software is subject to failures at random times caused by errors remaining in the software.
3. Corresponding to the failure phenomenon at the user/manufacturer's end, there is an equivalent failure phenomenon at the manufacturer/user's end.

4. An error which caused a failure will not cause any further failure until removed.
5. All errors in the software are mutually independent.
6. The expected number of software failures in time interval $(t, t + \Delta t)$ is proportional to the expected number of software errors remaining at time $t$.
7. All the errors are perfectly removed.

## Notations

$a$: initial error content

$b$: error detection rate per error, $0 < b < 1$

$m(t)$: mean value function in the NHPP model, $m(0) = 0$ (expected number of software errors detected in time $t$)

$C_1$ ($C_2$): Cost of fixing an error during testing (operation) ($C_2 > C_1$)

$C_3$: testing cost per unit time

$C_B$: total budget allocated for the software

$\bar{C}_1$ ($\bar{C}_2$) $= C_1/C_B$ ($C_2/C_B$)

$\bar{C}_3 = C_3/C_B$

$T^*$: optimal release time

$T_L$: software life cycle length

$C(T)$: total expected software cost incurred during software life cycle when the software is released at time $T$

$R(x/t)$: software reliability, *i.e.*, probability that a software failure does not occur in $(t, t + x)$, given that most recent failure occurred at time $\leq t$

## 3. RELEASE TIME PROBLEM FORMULATION

Assume that a decision maker (software developer) is interested not only in minimizing the total expected software cost but also in maximizing the reliability of the software. Further, the optimization is carried out under both budget and reliability constraints. The budget constraint takes care that the total cost of testing does not exceed the total funds available whereas the reliability constraint does not let the software reliability fall below a prescribed level.

$$\left.\begin{array}{l}\text{Mathematically, the problem may be formulated as} \\ \qquad\qquad \text{Maximize } R(x/T) \\ \qquad\qquad \text{Minimize } C(T) \\ \text{subject to} \\ \qquad\qquad C(T) \leq C_B \\ \qquad\qquad R(x/T) \geq R_0 \\ \qquad T \geq 0, \qquad 0 < R_0 < 1\end{array}\right\} \tag{1}$$

where $R_0$ is the desired level of reliability.

$$\left.\begin{array}{l}\text{The problem may alternatively be stated as} \\ \qquad\qquad \text{Maximize } \log R(x/T) \\ \qquad\qquad \text{Minimize } \bar{C}(T) \\ \text{subject to} \\ \qquad\qquad \bar{C}(T) \leq 1 \\ \qquad\qquad R(x/T) \geq R_0 \\ \qquad T \geq 0, \qquad 0 < R_0 < 1\end{array}\right\} \tag{2}$$

where $\bar{C}(T) = C(T)/C_B$.

The above bicriterion problem may be reduced to a single objective optimization problem by introducing $\lambda = \begin{bmatrix}\lambda_1 \\ \lambda_2\end{bmatrix} \in R^2$, $\lambda_1 \geq 0$, $\lambda_2 \geq 0$, $\sum_{i=1}^{2} \lambda_i = 1$ where $\lambda_i$ $(i = 1, 2)$ is the decision maker's priority for the $i^{th}$ objective function. Using $\lambda_1$, $\lambda_2$ the problem in (2) can be reformulated as

$$\left.\begin{array}{l}\text{Maximize } F(T) = \lambda_1 \log R(x/T) - \lambda_2 \bar{C}(T) \\ \text{subject to} \\ \qquad\qquad \bar{C}(T) \leq 1 \\ \qquad\qquad R(x/T) \geq R_0 \\ \qquad T \geq 0, \qquad 0 < R_0 < 1\end{array}\right\} \tag{3}$$

This form introduces a degree of flexibility over the traditional single objective software release policy, where, only the cost or the reliability could be optimized. Here, $\lambda_1$, $\lambda_2$ can be fixed according to decision maker's requirements and hence a solution satisfying both the requirements is obtained.

## 4. OPTIMAL RELEASE POLICY

For an exponential SRGM [4]

$$m(T) = a(1 - e^{-bT})$$
$$R(x/T) = e^{-[m(T+x)-m(T)]}$$

The software cost during software life cycle is given by

$$C(T) = C_1 m(T) + C_2\{m(T_L) - m(T)\} + C_3 T$$

Therefore,

$$\bar{C}(T) = C(T)/C_B = \bar{C}_1 m(T) + \bar{C}_2 \{m(T_L) - m(T)\} + \bar{C}_3 T$$

Further, the objective function $F(T)$ in (3) is

$$\begin{aligned} F(T) &= \lambda_1 \log R(x/T) - \lambda_2 \bar{C}(T) \\ &= a(1 - e^{-bT})(\lambda_1 + \lambda_2(\bar{C}_2 - \bar{C}_1)) \\ &\quad - \lambda_1 a(1 - e^{-b(T+x)}) - \lambda_2 \bar{C}_3 T - \lambda_2 \bar{C}_2 a(1 - e^{-bT_L}) \end{aligned} \quad (4)$$

Differentiating $F(T)$ with respect to $T$, we have

$$\frac{dF(T)}{dT} = abe^{-bT}[\lambda_1 + \lambda_2(\bar{C}_2 - \bar{C}_1) - \lambda_1 e^{-bx}] - \lambda_2 \bar{C}_3 \quad (5)$$

From (5)

$$\left.\frac{dF(T)}{dT}\right|_{T=0} = ab\,S - \lambda_2 \bar{C}_3$$

and

$$\left.\frac{dF(T)}{dT}\right|_{T\to\infty} = -\lambda_2 \bar{C}_3 < 0 \quad (6)$$

where

$$S = \lambda_1(1 - e^{-bx}) + (\bar{C}_2 - \bar{C}_1)\lambda_2 > 0 \quad (7)$$

From (4) and (6), we have the following cases

**Case I** (A) $ab > \lambda_2 \bar{C}_3/S$ $(\lambda_2 \neq 0)$

Then there exists a point $T = T_0$ $(0 < T_0 < \infty)$ satisfying $dF(T)/dT = 0$ such that $d^2F(T)/dT^2|_{T=T_0} < 0$ and hence $F(T)$ achieves its maxima at $T_0$.

(B) $ab > \lambda_2 \bar{C}_3/S$ $(\lambda_2 = 0)$

Then there exists a point $T = T_0$ $(0 < T_0 < \infty)$ satisfying $F(T) = 0$ for all $T \geq T_0$ and hence $F(T)$ has the maximum value zero for all $T \geq T_0$.

**Case II** $ab \leq \lambda_2 \bar{C}_3/S$

Here for all $T$, $F(T)$ decreases and hence $dF(T)/dT = 0$ has no positive solution.

Further, differentiating $\bar{C}(T)$ with respect to $T$, we have

$$\frac{d\bar{C}(T)}{dT} = \bar{C}_3 - abe^{-bT}(\bar{C}_2 - \bar{C}_1) \tag{8}$$

From (8)

$$\left.\frac{d\bar{C}(T)}{dT}\right|_{T=0} = \bar{C}_3 - ab(\bar{C}_2 - \bar{C}_1)$$

and

$$\left.\frac{d\bar{C}(T)}{dT}\right|_{T\to\infty} = \bar{C}_3$$

and hence we have the following cases for $\bar{C}(T)$:

**Case I** $\bar{C}_3 < ab(\bar{C}_2 - \bar{C}_1)$

Then there exists a point $T = T_C$ $(0 < T_C < \infty)$ satisfying $d\bar{C}(T)/dT = 0$ such that $d^2\bar{C}(T)/dT^2|_{T=T_C} > 0$ and hence $\bar{C}(T)$ achieves its minima at $T_C$.

Also for $\bar{C}(T) = 1$, following sub cases arise:

(i) $\bar{C}(T_C) > 1$

Then the budget constraint is violated for all $T$ and hence provision has to be made for more budget.

(ii) $\bar{C}(T_C) = 1$

Then there exists a unique point $T = T_1 (= T_C) > 0$ such that $\bar{C}(T_1) = 1$.

(iii) $\bar{C}(T_C) < 1 \leq \bar{C}(0)$

Then there exist two points $T = T_1$ and $T_2$ $(0 \leq T_1 < T_C < T_2)$ such that $\bar{C}(T_1) = \bar{C}(T_2) = 1$.

(iv) $\bar{C}(0) < 1$

Then there exists a unique point $T = T_1 (0 < T_C < T_1)$ such that $\bar{C}(T_1) = 1$.

**Case II** $\bar{C}_3 \geq ab(\bar{C}_2 - \bar{C}_1)$

Then for all $T$, $\bar{C}(T)$ increases and hence $d\bar{C}(T)/dT = 0$ has no positive solution. Also for $\bar{C}(0) \leq 1$ there exists only one point $T = T_1 (\geq 0)$ such that $\bar{C}(T_1) = 1$, and for $\bar{C}(0) > 1$ budget constraint is violated.

To find the nature of reliability function, we differentiate $R(x/T)$ with respect to $T$, to get

$$\frac{\partial R(x/T)}{\partial T} = abe^{-bT}(1 - e^{-bx})\, R(x/T) \tag{9}$$

which is always $\geq 0$. Hence $R(x/T)$ is a monotonically increasing function of $T$.

Further, for a specified operational time $x$, and reliability $R_0$, if $R(x/0) < R_0 < 1$, then there exists a positive and unique point $T = T_R$ such that $R(x/T_R) = R_0$, where

$$R(x/0) = \text{Exp}[-m(x)], \qquad R(x/\infty) = 1.$$

Combining cost-reliability objective function, budgetary limitation and reliability requirement, we may state the following theorem for optimal release policy.

THEOREM: Suppose that $\bar{C}_2 > \bar{C}_1 > 0$, $\bar{C}_3 > 0$, $C_B > 0$, $x \geq 0$, $0 < R_0 < 1$, $\lambda_1 \geq 0$, $\lambda_2 \geq 0$, $\lambda_1 + \lambda_2 = 1$.

**(A) Assume that** $ab > \lambda_2 \bar{C}_3/S$ **and** $ab > \bar{C}_3/(\bar{C}_2 - \bar{C}_1)$

(1) If $\bar{C}(T_C) > 1$ then more budget is needed to carry out the testing (the problem has no solution).

(2) If $\bar{C}(T_C) = 1$, $R(x/0) < R_0 < 1$ and $T_R \leq T_C$ then $T^* = T_C$.

(3) If $\bar{C}(T_C) = 1$, $R(x/0) < R_0 < 1$ and $T_C < T_R$ then more budget is needed to carry out the testing.

(4) If $\bar{C}(T_C) = 1$ and $R_0 \leq R(x/0)$, then $T^* = T_C$.

(5) If $\bar{C}(T_C) < 1 < \bar{C}(0)$, $R(x/0) < R_0 < 1$ and $T_R < T_1$ then
(5.1) $T^* = T_1$ for $T \leq T_1$
(5.2) $T^* = \text{Min}\{T_0, T_2\}$ for $T_1 < T_0$.

(6) If $\bar{C}(T_C) < 1 \leq \bar{C}(0)$, $R(x/0) < R_0 < 1$ and $T_1 \leq T_R \leq T_2$ then
(6.1) $T^* = \text{Max}\{T_R, T_0\}$ for $T_0 \leq T_2$
(6.2) $T^* = T_2$ for $T_2 < T_0$.

(7) If $\bar{C}(T_C) < 1 \leq \bar{C}(0)$, $R(x/0) < R_0 < 1$ and $T_2 < T_R$ then more budget is needed to carry out the testing.

(8) If $\bar{C}(T_C) < 1 < \bar{C}(0)$ and $R_0 \leqslant R(x/0)$ then
(8.1) $T^* = T_1$ for $T_0 \leq T_1$
(8.2) $T^* = \text{Min}\{T_0, T_2\}$ for $T_1 < T_0$

(9) If $\bar{C}(0) = 1$ and $R_0 \leq R(x/0)$ then $T^* = \text{Min}\{T_0, T_2\}$.

(10) If $\bar{C}(0) < 1$, $R(x/0) < R_0 < 1$ and $T_R \leq T_1$ then
(10.1) $T^* = T_R$ for $T_0 \leq T_R$
(10.2) $T^* = \text{Min}\{T_0, T_1\}$ for $T_R < T_0$.

(11) If $\bar{C}(0) < 1$, $R(x/0) < R_0 < 1$ and $T_1 < T_R$ then more budget is needed to carry out the testing.

(12) If $\bar{C}(0) < 1$ and $R_0 \leq R(x/0)$ then $T^* = \text{Min}\{T_0, T_1\}$.

**(B) Assume that** $ab > \lambda_2 \bar{C}_3/S$ **and** $ab \leq \bar{C}_3/(\bar{C}_2 - \bar{C}_1)$

(1) If $\bar{C}(0) < 1$, $R(x/0) < R_0 < 1$ and $T_R \leq T_1$ then
(1.1) $T^* = T_R$ for $T_0 \leq T_R$
(1.2) $T^* = \text{Min}\{T_0, T_1\}$ for $T_R < T_0$.

(2) If $\bar{C}(0) < 1$, $R(x/0) < R_0 < 1$ and $T_1 < T_R$, then more budget is needed to carry out the testing.

(3) If $\bar{C}(0) < 1$ and $R_0 \leq R(x/0)$ then $T^* = \text{Min}\{T_0, T_1\}$.

(4) If $\bar{C}(0) = 1$ and $R(x/0) < R_0 < 1$ then more budget is needed to carry out the testing.

(5) If $\bar{C}(0) = 1$ and $R_0 \leq R(x/0)$ then $T^* = 0$.

(6) If $\bar{C}(0) > 1$ then more budget is needed to carry out the testing.

**(C) Assume that** $ab \leq \lambda_2 \bar{C}_3/S$ **and** $ab < \bar{C}_3/(\bar{C}_2 - \bar{C}_1)$

(1) If $\bar{C}(0) < 1$, $R(x/0) < R_0 < 1$ and $T_R \leq T_1$ then $T^* = T_R$.

(2) If $\bar{C}(0) < 1$, $R(x/0) < R_0 < 1$ and $T_R > T_1$ then more budget is needed to carry out the testing.

(3) If $\bar{C}(0) = 1$ and $R(x/0) < R_0 < 1$ then more budget is needed to carry out the testing.

(4) If $\bar{C}(0) \leq 1$ and $R_0 \leq R(x/0)$ then $T^* = 0$.

(5) If $\bar{C}(0) > 1$ then more budget is needed to carry out the testing.

## Particular Cases

If $\lambda_1 = 0$, $\lambda_2 = 1$ and $C_B$ is sufficiently large (so that $C_B > \min \cdot C(T)$), (3) reduces to the classical cost optimization [12] release policy. On the other hand if $\lambda_1 = 1$ and $\lambda_2 = 0$, then (3) reduces to reliability optimization release policy, a policy which is particularly suitable for high reliability projects like nuclear reactors, space shuttles and heart monitors etc. The reliable

operation of these projects depends critically on the reliable operation of their software components. Because of the total dependence of these systems on the underlying software, they may demand a very high reliability. Values of $\lambda_1$, $\lambda_2$ other than 0 and 1 give other possible solutions, based on the specific requirements of a software project.

## 5. NUMERICAL EXAMPLE

Taking maximum likelihood estimates of the model parameters $a$ and $b$ as $\hat{a} = 33.99$, $\hat{b} = 0.0059$ (Goel *et al.* [4] where data is given in the form $\{S_k, k = 1, 2, \ldots, 26\}$ where $S_k$ is the time (in days) to the $k$-th failure), we discuss below the bicriterion release policy for an exponential SRGM. We further assume $C_1 = 5$, $C_2 = 15$, $C_3 = 1.5$, $C_B = 475$, $T_L = 250$, $R_0 = .75$ and $x = 2.00$ (we have assumed these values for illustrating our policy since it is expected that software developers have reliable estimates of various model and cost parameters from past experience and pretest (*see* also the discussion under post optimality analysis). Using the assumed values of various parameters, we have $T_C = 46.89$, $T_1 = 219.90$, $T_R = 53.19$, $C(0) = 389.95$, $\bar{C}(0) = 0.821$, $C(T_C) = 379.47$, $\bar{C}(T_C) = .798$, $C(T_R) = 379.64$, $\bar{C}(T_R) = .799$, $R(x/0) = 0.676$, $R(x/T_C) = 0.742$, $R(x/T_1) = 0.896$. For different values of $\lambda_1$ and $\lambda_2$ (as decided by the decision maker according to the relative significance of reliability and cost respectively), we find $T_0$ and consequently $T^*$ (figs. 1-3). These values are summarized in table (1).

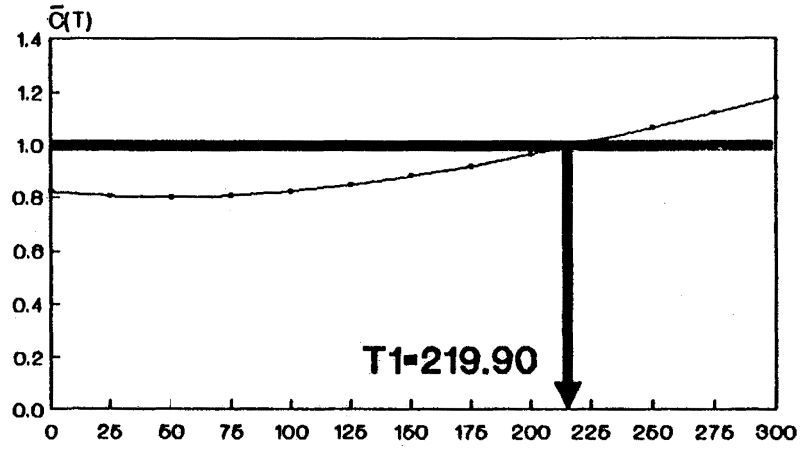


**Figure 1**

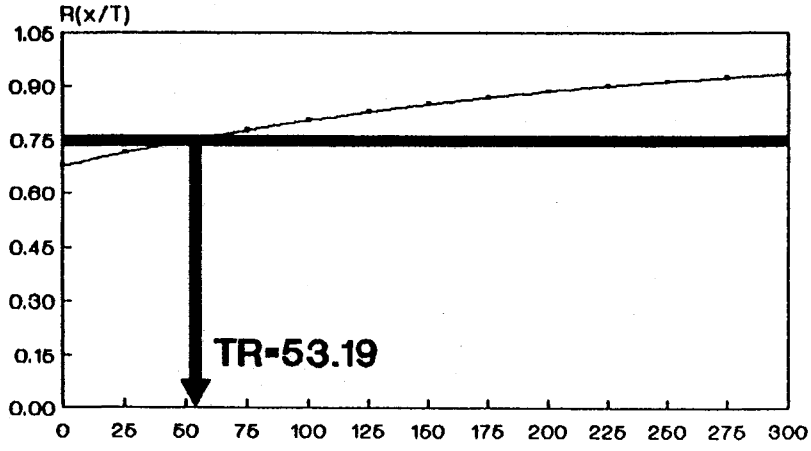


Figure 2

From table (1) we see that different values of $\lambda_1$ and $\lambda_2$ give rise to different $T^*$ and hence different $R(x/T^*)$ and $C(T^*)$. More weightage to reliability (*i.e.* higher $\lambda_1$) gives an optimal solution with higher value of $R(x/T^*)$. If the emphasis is on maximizing the reliability only (*i.e.*, $\lambda_1 = 1$, $\lambda_2 = 0$) then the highest possible reliability achieved by exhausting the total budget is 0.896. We also note that the cost weight ($\lambda_2$) remains ineffective until it is more than 0.2, *i.e.*, for $\lambda_2 = 0.0$, 0.1 and 0.2 ($\lambda_1 = 1.0$,

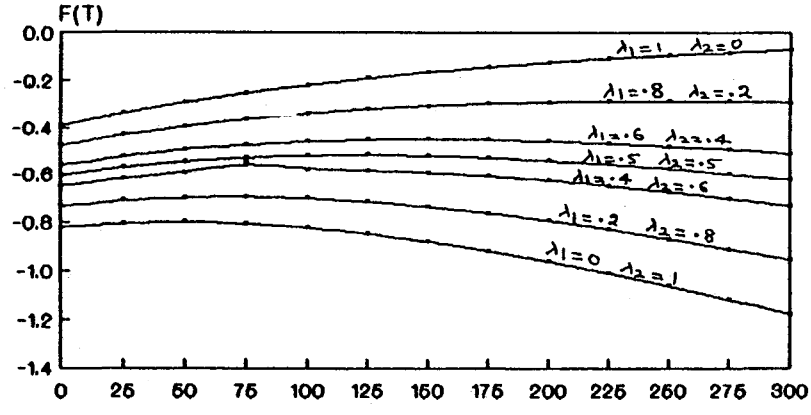


Figure 3

TABLE 1

| $ab = .196$ | | | | | $\bar{C}_3/(\bar{C}_2 - \bar{C}_1) = .150$ | | |
|---|---|---|---|---|---|---|---|
| $T_R = 53.19$ | | | | | $T_1 = 219.90$ | | |
| $\bar{C}(0) = .821$ | | | $\bar{C}(T_C) = .798$ | | $R(x/0) = .676$ | | |
| Weights | | $\frac{\lambda_2 \bar{C}_3}{S}$ | $T_0$ | Theorem | $T^*$ (days) | $R(x/T^*)$ | $C(T^*)$ |
| $\lambda_1$ | $\lambda_2$ | | | | | | |
| 1 | 0 | .013 | 2008.2 | A(10.2) | 219.90 | .896 | 475.00 |
| .9 | .1 | .025 | 338.91 | A(10.2) | 219.90 | .896 | 475.00 |
| .8 | .2 | .047 | 243.30 | A(10.2) | 219.90 | .896 | 475.00 |
| .7 | .3 | .065 | 184.80 | A(10.2) | 184.80 | .874 | 443.85 |
| .6 | .4 | .082 | 148.59 | A(10.2) | 148.59 | .847 | 416.73 |
| .5 | .5 | .096 | 120.39 | A(10.2) | 120.39 | .822 | 399.93 |
| .4 | .6 | .109 | 100.09 | A(10.2) | 100.09 | .803 | 390.59 |
| .3 | .7 | .121 | 82.79 | A(10.2) | 82.79 | .785 | 384.70 |
| .2 | .8 | .131 | 68.79 | A(10.2) | 68.79 | .768 | 381.47 |
| .1 | .9 | .141 | 57.09 | A(10.2) | 57.09 | .755 | 379.92 |
| 0 | 1 | .150 | 46.89 | A(10.1) | 53.19 | .750 | 379.65 |

0.9 and 0.8 respectively), same reliability (0.896) is obtained by spending the entire budget (475). When $\lambda_2$ is more than 0.2, cost weight becomes effective and there is a decrease in the reliability achieved as well as the cost incurred. This continues till $\lambda_1 = 0.0$, $\lambda_2 = 1$, when the cost incurred falls down to 379.65, which is the minimum amount needed to obtain the desired reliability $R_0$ (.75). Hence introduction of $\lambda_1$ and $\lambda_2$ gives more freedom to the decision maker in setting his objectives and thus he may have a trade off between cost and reliability depending upon the importance of each.

## 6. POST OPTIMALITY ANALYSIS

Software life cycle length ($T_L$) is expected to be fixed well in advance. However, the following post optimality analysis may help in fixing $T_L$ and the reliability objective $R_0$ as other parameters namely cost are expected to be known because of the past experience of the software developer. In the above example, we vary $T_L$ between 100 and 550 (step size-50) and observe its effect on $T^*$ and consequently on $R(x/T^*)$ and $C(T^*)$. This effect will depend on the weights $\lambda_1$ and $\lambda_2$. For illustration we choose $\lambda_1 = 0.7$ and $\lambda_2 = 0.3$. The results obtained for different values of $T_L$ are summarized in table (2).

From the table (2) we see that when $T_L$ is smaller than both $T_g$ and $T_0$ (where $T_g$ is the greatest $T$ such that $\bar{C}(T) = 1$ and hence $T_g = T_1$ when $\bar{C}(0) < 1$ and $T_g = T_2$ when $\bar{C}(0) \geq 1 \geq \bar{C}(T_C)$), *i.e.*, when $T_L = 100$ and 150, $T^*$ is same as $T_L$ which implies that $T_L - T^* = 0$. This brings forth the fact that if the software life cycle is not suitably fixed, no maintenance can be provided after the release of the software. Also in these cases a large part of the budget remains unutilized. Such a software life cycle length will not be justifiable since the software developer will not be able to provide any maintenance after the release, which may not be acceptable to the user. Now as $T_L$ increases beyond 150, the maintenance period $(T_L - T^*)$ increases. For $T_L = 200, 250$ and 300, the time of release as well as the reliability achieved is same but the cost incurred is different (it increases with the life cycle). We also observe that for these value of $T_L$, $F(T^*)$ decreases. Thus from view point of maximizing the objective function $F(T)$, $T_L = 200$ may be the best choice as it gives the greatest value of $F(T^*)$. However, in this case the maintenance period is small as compared to the cases when $T_L = 250$ and 300. In the case when $T_L = 300$, almost the entire budget is utilized and the reliability achieved is 0.874, which is same as in the cases when $T_L = 200$ and 250. Thus, we see that at $T_L = 300$, cost weight $(\lambda_2)$ starts becoming ineffective. When $T_L$ is more than 300, $T^*$ and $R(x/T^*)$ decrease and the

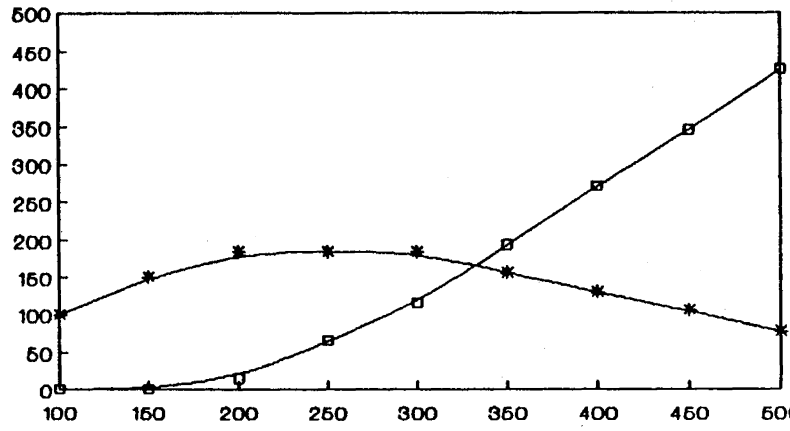

Figure 4

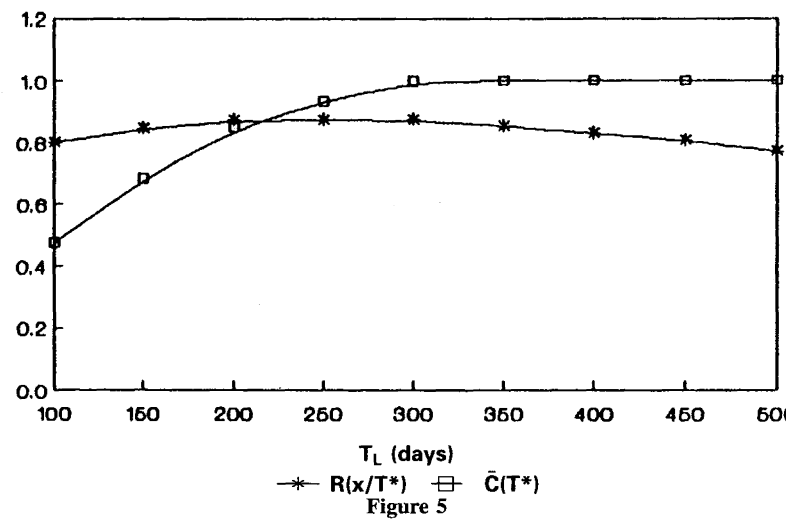

Figure 5

entire budget is spent in the testing. Thus for $T_L > 300$, we achieve lesser reliability by spending more (figure 5). From the view point of objective function $F(T)$, these life cycle lengths wouldn't be chosen. However, if the software developer is interested in providing more maintenance period (which is increasing with $T_L$ and is around 423 for $T_L = 500$) at the cost of expenditure and reliability, he may choose any $T_L$ between 350 and 500. At $T_L = 550$ the minimum cost incurred exceeds the budget and hence more budget is needed to have this much length of the cycle.

TABLE 2

| $ab = .196$ | | | $\lambda_2 \bar{C}_3/S = .065$ | | | | $\bar{C}_3/(\bar{C}_2 - \bar{C}_1) = .150$ | | | | $R(x/0) = .676$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $T_L$ | $T_C$ | $\bar{C}(0)$ | $\bar{C}(T_C)$ | $C(T_C)$ | $T_g$ | $T_0$ | $T_R$ | Theorem | $T^*$ (days) | $T_L - T^*$ (days) | $F(T^*)$ | $R(x/T^*)$ | $C(T^*)$ | $\bar{C}(T^*)$ |
| 100 | 46.89 | .471 | .449 | 213.62 | 366.8 | 184.80 | 53.2 | # | 100 | 0 | −.439 | .800 | 224.70 | .473 |
| 150 | 46.89 | .623 | .600 | 285.44 | 307.9 | 184.80 | 53.2 | # | 150 | 0 | −.371 | .847 | 323.64 | .681 |
| 200 | 46.89 | .736 | .714 | 339.21 | 259.8 | 184.80 | 53.2 | A(10.2) | 184.80 | 15.20 | −.348 | .874 | 403.59 | .849 |
| 250 | 46.89 | .821 | .798 | 379.47 | 219.90 | 184.80 | 53.2 | A(10.2) | 184.80 | 65.20 | −.374 | .874 | 443.85 | .934 |
| 300 | 46.89 | .884 | .862 | 409.61 | 186.10 | 184.80 | 53.2 | A(10.2) | 184.80 | 115.20 | −.393 | .874 | 473.98 | .998 |
| 350 | 46.89 | .930 | .909 | 432.17 | 156.70 | 184.80 | 53.2 | A(10.2) | 156.70 | 193.30 | −.411 | .853 | 475.00 | 1.00 |
| 400 | 46.89 | .967 | .945 | 449.07 | 130.40 | 184.80 | 53.2 | A(10.2) | 130.40 | 269.60 | −.429 | .830 | 475.00 | 1.00 |
| 450 | 46.89 | .994 | .972 | 461.71 | 105.40 | 184.80 | 53.2 | A(10.2) | 105.40 | 344.60 | −.449 | .808 | 475.00 | 1.00 |
| 500 | 46.89 | 1.014 | .991 | 471.18 | 77.49 | 184.80 | 53.2 | A(10.2) | 077.49 | 422.51 | −.475 | .779 | 475.00 | 1.00 |
| 550 | 46.89 | 1.028 | 1.006 | 478.26* | – | – | – | – | – | – | – | – | – | – |

# $T_g > T_L$, $T_0 > T_L \Rightarrow T^* = T_L$

* For $T_L = 550$, $C(T_C) > C_B$ therefore the budget of 475 wouldn't be able to support a life cycle of 550 days.

Thus it is for the software developer to decide the software life cycle length, given the above analysis. The above discussion is further sketched in figures 4 and 5. Figure 4 illustrates the effect of $T_L$ on $T^*$ and $T_L - T^*$ (maintenance period) whereas figure 5 illustrates the effect of $T_L$ on $R(x|T^*)$ and $\bar{C}(T^*)$.

Now, once the developer has fixed the software life cycle length $T_L$, the reliability goal $R_0$ can be decided between $R(x/T_1)$ and $R(x/T_2)$ or $R(x/0)$ and $R(x/T_1)$, depending on whether $\bar{C}(0) \geq 1 > \bar{C}(T_C)$ or $\bar{C}(0) < 1$. For illustration, $R_0$, for the numerical example in table (1), can be fixed any where between .676 and .896.

## CONCLUSION

In this paper we have discussed a flexible release policy, optimizing two conflicting objectives, namely software cost and software reliability simultaneously. Such a policy allows the software developer flexibility depending on the project being undertaken. Earlier policies discussed in the literature are only suitable for a particular project being undertaken which may be in terms of minimizing only cost or maximizing only reliability under suitable constraints. In that sense the proposed policy brings forth a trade off between cost and reliability depending upon the importance of each. Besides, we have also discussed the effect of software life cycle length on the optimal release time, reliability and cost, which can help the decision maker to fix suitably the software life cycle length and the reliability goal.

## ACKNOWLEDGMENT

Second author wishes to thank the Council of Scientific and Industrial Research (CSIR), New Delhi, India, for providing the financial grant for carrying out this research work. The authors also acknowledge with gratitude the suggestions of the referees which helped in revising the paper.

## REFERENCES

1. S. Bittanti, P. Bolzern, E. Pedrotti, M. Pozzi and R. Scattolini, A flexible modelling approach for software reliability growth, In *Software Reliability Modelling and identification*, Ed. S. Bittanti, Springer-Verlag, Berlin, 1988.
2. P. A. Caspi and E. F. Kouka, Stopping rules for a debugging process based on different software reliability models. *Proc. Int. Conf. on Fault-Tolerant Computing*, 1984, pp. 114-119.
3. E. H. Forman and N. D. Singpurwalla, An empirical stopping rule for debugging and testing computer software, *Jour. Amer. Stat. Asso.*, 72, 1977, pp. 750-757.

4. A. L. GOEL and K. OKUMOTO, Time dependent error detection rate model for software reliability and performance measures, *IEEE Trans. Reliab. 28(3),* 1979, pp. 206-211.
5. A. L. GOEL, Software reliability models: assumptions, limitations and applicability, *IEEE Trans. software Eng. SE-11,* 1985, pp. 1411-1423.
6. P. K. KAPUR and R. B. GARG, Optimal software release policies for software reliability growth models under imperfect debugging, *R.A.I.R.O. 24(3),* 1990, pp. 295-305.
7. P. K. KAPUR and R. B. GARG, Cost-Reliability optimum release policies for a software system with testing effort. *OPSEARCH, 27(2),* 1990, pp. 109-114.
8. P. K. KAPUR and R. B. GARG, Optimum release policy for inflection S-shaped software reliability growth model, *Micro. electron. & Reliab.*, 1991, pp. 39-42.
9. P. K. KAPUR and V. K. BHALLA, Optimal release policies for flexible software reliability growth model, *Rel. Eng. & Sys. Safety*, 35(1), 1992, pp. 49-54.
10. S. M. ROSS, Software reliability: the stopping rule problem, *IEEE Trans. Soft. Eng., SE-11,* 1985, pp. 1472-1476.
11. S. YAMADA and S. OSAKI, Software reliability growth modelling: models and applications, *IEEE Trans. Soft. Eng., SE-11*, 1985, pp. 1431-1437.
12. S. YAMADA and S. OSAKI, Optimal Software release policies with simultaneous cost and reliability requirements. *EJOR, 31*, 1987, pp. 46-51.