HAL
open science

Optimal times for software release when repair is imperfect

Philip J. Boland, Nóra Ní Chuív

To cite this version:

Philip J. Boland, Nóra Ní Chuív. Optimal times for software release when repair is imperfect. Statistics and Probability Letters, 2009, 77 (12), pp.1176. 10.1016/j.spl.2007.03.004 . hal-00565450

HAL Id: hal-00565450

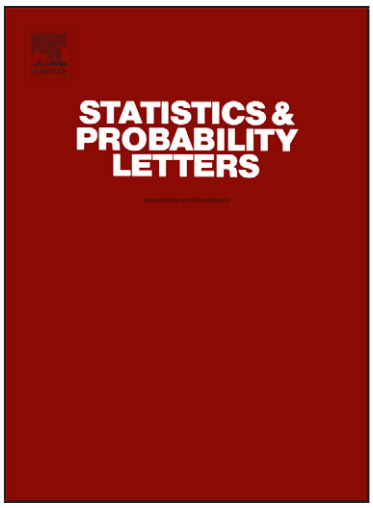
https://hal.science/hal-00565450v1

Submitted on 13 Feb 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Author's Accepted Manuscript



Optimal times for software release when repair is imperfect

Philip J. Boland, Nóra Ní Chuív

PII: S0167-7152(07)00082-X
DOI: doi:10.1016/j.spl.2007.03.004
Reference: STAPRO 4603

To appear in: *Statistics & Probability Letters*

www.elsevier.com/locate/stapro

Cite this article as: Philip J. Boland and Nóra Ní Chuív, Optimal times for software release when repair is imperfect, *Statistics & Probability Letters* (2007), doi:10.1016/j.spl.2007.03.004

This is a PDF file of an unedited manuscript that has been accepted for publication. As a service to our customers we are providing this early version of the manuscript. The manuscript will undergo copyediting, typesetting, and review of the resulting galley proof before it is published in its final citable form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

ACCEPTED MANUSCRIPT

Optimal Times for Software Release when Repair is Imperfect

Philip J. Boland *

Statistics and Actuarial Science National University of Ireland - Dublin,

Belfield, Dublin 4, Ireland

Nóra Ní Chuív

Department of Mathematics and Statistics, University of New Brunswick,

Fredericton, New Brunswick, E3B 5A3, Canada

December 2006

MRC: primary 90B25, secondary 68M20; 68N30

KEY WORDS: Software Testing, Software Reliability, Optimal Release Time, Birth Processes, Cost Models, Imperfect Repair.

Accepted manuscript

Abstract

The determination of the optimal release time for a new piece of software is of primary importance in the process of software development. We study a model where initially there are N faults in the software each with failure detection rate $\lambda(t)$, but where the probability of a perfect repair of a fault when found is p (in general repair is not perfect). We investigate various cost models for this situation and give some insight into how the optimal release times and costs for the software vary with p and $\lambda(t)$.

1 Introduction

Software reliability is a crucial concern in our rapidly developing information technology world. It is often defined as **the probability of failure free operation of a computer programme in a specified environment for a specified period of time**. For many years statisticians and software engineers have been developing models for failures and breakdowns in computer software (Farr, 1999; Musa, Iannino and Okumoto, 1987, Singpurwalla and Wilson, 1999; Xie, 2000; and Boland and Singh, 2002).

*corresponding author.

ACCEPTED MANUSCRIPT

One may (broadly speaking) break the software life-cycle into the four stages of: requirements and specification, design, coding, and testing. Although the time spent in the different stages clearly varies with products and customers, it is not unreasonable to expect testing to take in the region of 30% of the development time. Failures in software are not physical in nature and usually result as a consequence of errors in logic. They can lead to results inconsistent with performance specifications and expectations, and hence knowledge about them often influences important economic decisions concerning software release. Software is tested in order to detect faults and consequently reduce the chances of failures occurring during its post release operation. If software is tested for too little time it will involve high costs of customer dissatisfaction, while if it is tested for too long it will involve large costs of testing and opportunity lost as a result of releasing the software at a later stage.

Software reliability models may be broken into two broad categories (Singpurwalla and Wilson, 1999). The so called **Type I models** are developed by defining times between successive software failures. Classic examples include the Jelinski Moranda model (1972) and the Moranda de-Eutrophication model (Moranda, 1975; Boland and Singh, 2003). **Type II models** on the other hand are concerned with modelling the cumulative number of failures up to any time t. Two classic examples are the Goel Okumoto (1978) time dependent (NHPP) error detection model, and the Musa Okumoto (Musa, Iannino, Okumoto, 1987) logarithmic Poisson execution time model.

In this paper we will restrict our attention to some generalizations of the Jelinski Moranda model. In this classic model one assumes there are a finite number of faults N in the software and that each fault has an equal and independent failure (discovery) rate of λ. Here one assumes that once a fault is discovered it is perfectly repaired and the new failure rate for the software is reduced by λ, and also that the time between the ith and $(i+1)^{st}$ observed failures are exponentially distributed with parameter $(N-i)\lambda$. The total number of faults $M(t)$ detected by time t is a birth process and letting $P_n(t) = P(M(t) = n)$, it follows that $P_n(t) = \binom{N}{n} e^{-\lambda(N-n)t}(1-e^{-\lambda t})^n$ and $E(M(t)) = N(1 - e^{-\lambda t})$.

Imperfect Repair may occur in attempting to remove a fault from software. The Irish software testing company Lionbridge Technologies suggests that typically in the region of 15% of faults are not correctly repaired. We consider a modification of the Jelinski Moranda model to take account of imperfect repair and let p represent the probability that a fault which is discovered or detected is perfectly repaired. In extending earlier work of Boland and Ní Chuív (2001), we also assume that the detection (failure) rate of a fault is a function of time $\lambda(t)$ which is not

Accepted manuscript

ACCEPTED MANUSCRIPT

necessarily constant. A study is made of the resulting optimal release times for software under a cost structure for development.

2 Numbers of Faults Detected and Removed in Software Testing

We shall assume that initially there are N faults in the software each with failure rate $\lambda(t)$, resulting in an initial failure rate for the system of $N\,\lambda(0)$. We let $\Lambda(t) = \int_0^t \lambda(s)\,ds$ for $t > 0$, and in particular we will consider the fault detection rate functions $\lambda(t) = be^{-bt}$ and $\lambda(t) = b/(1+bt)$, in addition to the standard $\lambda(t) = b$. When a fault is detected in the software at time t, we assume it is perfectly repaired with probability p, in which case the number of faults in the software is reduced by 1 (and consequently the failure rate is reduced by $\lambda(t)$). Otherwise the number of faults in the software (and the failure rate for the software) remains the same.

The probability density function for the time x_{k+1} that the $(k+1)^{th}$ fault is successfully removed, given the time x_k of removal of the kth fault, is given by

$$f(x_{k+1}|x_k) = p(N-k)\,\lambda(x_{k+1})\,e^{-(N-k)p\,[\Lambda(x_{k+1})-\Lambda(x_k)]} \tag{1}$$

Moreover the probability density function for the time x_k of the kth fault to be removed is given by

$$\begin{aligned} f(x_k) &= \frac{N!}{(N-k)!(k-1)!}\,p\,\lambda(x_k)\,e^{-p\Lambda(x_k)(N-k+1)}\left[1-e^{-p\,\Lambda(x_k)}\right]^{k-1} \\ &= \frac{N!}{(N-k)!(k-1)!}\,e^{-p\,\Lambda(x_k)(N-k)}\left[1-e^{-p\,\Lambda(x_k)}\right]^{k-1} d\left(-e^{-p\,\Lambda(x_k)}\right) \end{aligned}$$

Note that this is the density of the kth order statistic, where the failures of each of the faults are independently identically distributed with failure rate $\lambda(t)$.

For any interval of time $(0,t]$, we let $M(t)$ be the total number of faults detected or met by time t, and $R(t)$ be the number of faults which have been removed (that is detected and perfectly repaired) by time t. The number of *unsuccessful* repairs in the interval $(0,t]$ will be denoted by $U(t) = M(t) - R(t)$. In our model the times between removals of faults are still independent (in the classic Jelinski Moranda model where $\lambda(t) = b$ they are exponentially distributed with different parameters). For any integers u and r (where $0 \le r \le N$) we let $P_r(t) = P(R(t) = r)$ and $P_{u,r}(t) = P(U(t) = u, R(t) = r)$. We derive useful expressions for these quantities in terms of t, $\Lambda(t)$, and the parameters N and p. Using a standard differential equations approach, it may be established that for any $1 \le r \le N$

$$P_r'(t) = -(N-r)\,\lambda(t)\,p\,P_r(t) + (N-r+1)\,\lambda(t)\,p\,P_{r-1}(t). \tag{2}$$

Accepted manuscript

ACCEPTED MANUSCRIPT

With the initial conditions that $P_r(0) = 0$ for $1 \le r \le N$ and $P_0(t) = e^{-Np\Lambda(t)}$ for $t \ge 0$, one may establish (by induction or otherwise) that

$$P_r(t) = \binom{N}{r} e^{-p(N-r)\,\Lambda(t)}\,(1 - e^{-p\Lambda(t)})^r. \tag{3}$$

Note that $R(t)$ is a *binomial* birth process ($R(t)$ is $Bin(N, 1-e^{-pt\Lambda(t)})$ for any $t > 0$), and when $p = 1$ (perfect repair) and $\lambda(t) = b$ we obtain the standard Jelinski Moranda model.

We now derive an expression for $P_{u,r}(t)$ using an integral approach. We let $x_1 \le x_2 \le \cdots \le x_r$ be the occurrence times of the r faults which are detected and successfully removed in the time interval $(0, t]$, and $u_0, u_1, \ldots, u_r$ be the number of faults which are encountered but *not* removed in the time intervals $(0, x_1], (x_1, x_2], \ldots, (x_r, t]$. For convenience we will furthermore set $x_0 = 0$ and $x_{r+1} = t$. Letting $\mathbf{x} = (x_0, x_1, x_2, \ldots, x_r)$, $\mathbf{u} = (u_0, u_1, \ldots, u_r)$, and $u = \Sigma_{i=0}^r u_i$, we have that the joint density function for $\mathbf{x}$ and $\mathbf{u}$ is given by

$$\begin{aligned}
f(\mathbf{x}, \mathbf{u}) &= p^r \prod_{i=0}^{r} \left\{ \frac{[(N-i)(\Lambda(x_{i+1}) - \Lambda(x_i))]^{u_i}}{u_i!} e^{-(N-i)(\Lambda(x_{i+1})-\Lambda(x_i))} (1-p)^{u_i} \right\} \prod_{i=0}^{r-1} (N-i)\lambda(x_i) \\
&= p^r (1-p)^{\sum_{i=0}^r u_i} \frac{N!}{(N-r)!} (e^{-\sum_{i=0}^r (N-i)(\Lambda(x_{i+1})-\Lambda(x_i))}) \prod_{i=1}^{r} \lambda(x_i) \\
&\quad \cdot \left\{ \prod_{i=0}^{r} \frac{[(N-i)(\Lambda(x_{i+1}) - \Lambda(x_i)]^{u_i}}{u_i!} \right\} \\
&= p^r (1-p)^u \binom{N}{r} r!\ (e^{[-(N-r)\Lambda(t) - \Lambda(x_r) - \cdots - \Lambda(x_1)]}) \\
&\quad \cdot \prod_{i=1}^{r} \lambda(x_i) \prod_{i=0}^{r} \frac{[(N-i)(\Lambda(x_{i+1}) - \Lambda(x_i))]^{u_i}}{u_i!}.
\end{aligned}$$

On integration it follows that

$$\begin{aligned}
P_{u,r}(t) &= \sum_{u_0+\cdots+u_r=u} \int_0^t \int_0^{x_r} \cdots \int_0^{x_2} f(\mathbf{x}, \mathbf{u})\ dx_1 \ldots dx_r \\
&= \binom{N}{r} r! p^r (1-p)^u \int_0^t \int_0^{x_r} \cdots \int_0^{x_2} e^{-[(N-r)\Lambda(t) + \Lambda(x_r) + \cdots + \Lambda(x_1)]} \prod_{i=1}^{r} \lambda(x_i) \\
&\quad \cdot \left\{ \frac{1}{u!} \sum_{u_0+\cdots+u_r=u} \prod_{i=0}^{r} \frac{u![(N-i)(\Lambda(x_{i+1} - \Lambda(x_i))]^{u_i}}{u_i!} \right\} dx_1 \ldots dx_r.
\end{aligned}$$

Using the multinomial expansion

$$\sum_{u_0+\cdots+u_r=u} \prod_{i=0}^{r} \frac{u![(N-i)(\Lambda(x_{i+1}) - \Lambda(x_i))]^{u_i}}{u_i!} = \left[\sum_{i=0}^{r} (N-i)(\Lambda(x_{i+1}) - \Lambda(x_i)) \right]^u$$

the expression for $P_{u,r}(t)$ simplifies to

$$\begin{aligned}
P_{u,r}(t) &= \binom{N}{r} p^r (1-p)^u r! \int_0^t \int_0^{x_r} \cdots \int_0^{x_2} \frac{[(N-r)\Lambda(t) + \Lambda(x_r) + \cdots + \Lambda(x_1)]^u}{u!} \\
&\quad \cdot e^{-[(N-r)\Lambda(t) + \Lambda(x_r) + \cdots + \Lambda(x_1)]} \prod_{i=1}^{r} \lambda(x_i)\ dx_1 \cdots dx_r.
\end{aligned}$$

Accepted manuscript

ACCEPTED MANUSCRIPT

Hence

$$
\begin{aligned}
P_r(t) &= \sum_{u=0}^{\infty} P_{u,r}(t) \\
&= \sum_{u=0}^{\infty} \binom{N}{r} p^r (1-p)^u r! \cdot \int_0^t \int_0^{x_r} \cdots \int_0^{x_2} \frac{[(N-r)\Lambda(t) + \Lambda(x_r) + \cdots + \Lambda(x_1)]^u}{u!} \\
&\quad e^{-[(N-r)\Lambda(t)+\Lambda(x_r)+\cdots+\Lambda(x_1)]} \prod_{i=1}^{r} \lambda(x_i)\ dx_1 \cdots dx_r. \\
&= \binom{N}{r} p^r r! \int_0^t \int_0^{x_r} \cdots \int_0^{x_2} \sum_{u=0}^{\infty} \frac{[(1-p)[(N-r)\Lambda(t) + \Lambda(x_r) + \cdots + \Lambda(x_1)]^u}{u!} \\
&\quad e^{-[(N-r)\Lambda(t)+\Lambda(x_r)+\cdots+\Lambda(x_1)]} \prod_{i=1}^{r} \lambda(x_i)\ dx_1 \cdots dx_r. \\
&= \binom{N}{r} p^r r! \int_0^t \int_0^{x_r} \cdots \int_0^{x_2} e^{(1-p)[(N-r)\Lambda(t)+\Lambda(x_r)+\cdots+\Lambda(x_1)]} \\
&\quad e^{-[(N-r)\Lambda(t)+\Lambda(x_r)+\cdots+\Lambda(x_1)]} \prod_{i=1}^{r} \lambda(x_i)\ dx_1 \cdots dx_r. \\
&= \binom{N}{r} p^r r! \int_0^t \int_0^{x_r} \cdots \int_0^{x_2} e^{-p[(N-r)\Lambda(t)+\Lambda(x_r)+\cdots+\Lambda(x_1)]} \prod_{i=1}^{r} \lambda(x_i)\ dx_1 \cdots dx_r. \\
&= \binom{N}{r} p^r r! e^{-p(N-r)\Lambda(t)} \int_0^t \int_0^{x_r} \cdots \int_0^{x_2} e^{-p\Lambda(x_r)} \lambda(x_r) \cdots e^{-p\Lambda(x_1)} \lambda(x_1)\ dx_1 \cdots dx_r \\
&= \binom{N}{r} p^r r!\ e^{-p(N-r)\Lambda(t)} (1 - e^{-p\Lambda(t)})^r \frac{1}{p^r r!} = \binom{N}{r} e^{-p\Lambda(t)(N-r)} (1 - e^{-p\Lambda(t)})^r
\end{aligned}
$$

This is the same expression as (3). It follows immediately that $E[R(t)] = N(1 - e^{-p\Lambda(t)})$. Similarly

$$
\begin{aligned}
E[U(t)] &= \sum_{r=0}^{N} \sum_{u=0}^{\infty} u P_{u,r}(t) \\
&= \sum_{r=0}^{N} \binom{N}{r} p^r r! \int_0^t \int_0^{x_r} \cdots \int_0^{x_2} \sum_{u=0}^{\infty} u \frac{(1-p)^u [(N-r)\Lambda(t) + \Lambda(x_r) + \cdots + \Lambda(x_1)]^u}{u!} \\
&\quad \cdot e^{[-(N-r)\Lambda(t)-\Lambda(x_r)-\cdots-\Lambda(x_1)]} \prod_{i=1}^{r} \lambda(x_i)\ dx_1 \cdots dx_r \\
&= \sum_{r=0}^{N} \binom{N}{r} p^r r! \int_0^t \int_0^{x_r} \cdots \int_0^{x_2} (1-p)[(N-r)\Lambda(t) + \Lambda(x_r) + \cdots + \Lambda(x_1)] \\
&\quad e^{-p[(N-r)\Lambda(t)+\Lambda(x_r)\cdots+\Lambda(x_1)]} \prod_{i=1}^{r} \lambda(x_i)\ dx_1 \cdots dx_r \\
&= \sum_{r=0}^{N} \binom{N}{r} p^r r! \\
&\quad \int_0^t \int_0^{x_r} \cdots \int_0^{x_2} (1-p) \left[\frac{-\partial}{\partial p} e^{-p[(N-r)\Lambda(t)+\Lambda(x_r)+\cdots+\Lambda(x_1)]} \right] \prod_{i=1}^{r} \lambda(x_i)\ dx_1 \cdots dx_r
\end{aligned}
$$

Accepted manuscript

ACCEPTED MANUSCRIPT

$$
\begin{aligned}
&= -\sum_{r=0}^{N}\binom{N}{r}p^r r! \\
&\quad (1-p)\frac{\partial}{\partial p}\int_0^t\int_0^{x_r}\cdots\int_0^{x_2} e^{-p[(N-r)\Lambda(t)+\Lambda(x_r)+\cdots+\Lambda(x_1)]}\prod_{i=1}^{r}\lambda(x_i)\ dx_1\cdots dx_r \\
&= -(1-p)\sum_{r=0}^{N}\binom{N}{r}p^r r!\ \frac{\partial}{\partial p}\{e^{-p(N-r)\Lambda(t)}(1-e^{-p\Lambda(t)})^r\frac{1}{p^r r!}\} \\
&= -(1-p)\sum_{r=0}^{N}\binom{N}{r}p^r r! \\
&\quad \left[\frac{1}{p^r r!}\frac{\partial}{\partial p}\{e^{-p(N-r)\Lambda(t)}(1-e^{-p\Lambda(t)})^r\} - \frac{r}{p^{r+1}r!}\{e^{-p(N-r)\Lambda(t)}(1-e^{-p\Lambda(t)})^r\}\right] \\
&= -(1-p) \qquad (4) \\
&\quad \left[\frac{\partial}{\partial p}\{\sum_{r=0}^{N}\binom{N}{r}e^{-p(N-r)\Lambda(t)}(1-e^{-p\Lambda(t)})^r\} - \sum_{r=0}^{N}\binom{N}{r}\frac{r}{p}\{e^{-p(N-r)\Lambda(t)}(1-e^{-p\Lambda(t)})^r\}\right] \\
&= \frac{N(1-p)}{p}(1-e^{-p\Lambda(t)})\sum_{r-1=0}^{N-1}\binom{N-1}{r-1}e^{-p\Lambda(t)(N-1-(r-1))}(1-e^{-p\Lambda(t)})^{r-1} \\
&= \frac{N(1-p)}{p}(1-e^{-p\Lambda(t)}). \qquad (5)
\end{aligned}
$$

Given that $M(t) = R(t) + U(t)$, it is clear therefore that $E[M(t)] = (N/p)(1 - e^{-p\Lambda(t)})$.

3 Optimal Release Time for Testing Software.

In our cost models we will consider two distinct cases. In the first case we consider a fixed software *life-cycle* time t_0 where it is desired that if the software is released at time T, it functions well in the period $(T, t_0]$. In the second case we will want the software to function for some fixed *mission* time τ after release (that is to say, in the period $(T, T+\tau]$). Let us use c_1 to be the cost associated with encountering a fault (whether successfully repaired or not) during the testing period $(0, T]$, c_2 to be the cost of dealing with a fault encountered after the release time T, and c_3 to be the cost of testing per unit time during the testing period. A common cost function model used for releasing the software at time T (see Okumoto and Goel (1980), McDaid and Wilson (2001), and Boland and Singh (2002)) takes the form

$$C(T) = c_1 M(T) + c_2[M(g(T)) - M(T)] + c_3 T \qquad (6)$$

where $g(T) = g_1(T) = t_0$ in case 1 (where we are considering a fixed life cycle time t_0), and $g(T) = g_2(T) = T + \tau$ in case 2 (when we consider a fixed mission time τ). The total expected cost for releasing the software at time T therefore takes the form

$$E[C(T)] \quad = \quad c_1 E[M(T)] + c_2(\,E[M(g(T))] - E[M(T)]\,) + c_3 T$$

Accepted manuscript

ACCEPTED MANUSCRIPT

$$= \quad c_2 \frac{N}{p}(1 - e^{-p\Lambda(g(T))}) - (c_2 - c_1)\frac{N}{p}(1 - e^{-p\Lambda(T)}) + c_3 T, \tag{7}$$

and the objective is to find a T^* which minimizes expected costs. Differentiating (7) with respect to T one finds critical points, and the T^* minimizing expected costs is a solution of

$$\lambda(T)\, e^{-p\Lambda(T)} \quad = \quad \frac{c_3}{N(c_2 - c_1)}, \tag{8}$$

for case 1, while in case 2 it is a solution of

$$(c_2 - c_1)\, \lambda(T)\, e^{-p\Lambda(T)} - c_2\, \lambda(T + \tau)\, e^{-p\Lambda(T+\tau)} \quad = \quad \frac{c_3}{N}. \tag{9}$$

If $\lambda(t)$ (failure rate or discovery rate of faults) is non increasing and $c_2 > c_1$, then any solution T_1 to equation (8) is unique. This gives a minimum cost for release if $\lambda(0)N[c_2 - c_1] > c_3$, and otherwise the optimal value is $T^* = 0$. Remember that in case 1 we must have $T^* \leq g(T) = t_0$, thus, denoting the optimal choice of T by T^*,

$$T^* = min\,(t_0, max(0, T_1))\,, \tag{10}$$

where T_1 is the solution of (8).

In the case of a fixed mission time τ, the critical point is given by the solution of

$$c_2[\lambda(T)\, e^{-p\Lambda(T)} - \lambda(T + \tau)\, e^{-p\Lambda(T+\tau)}] - c_1\lambda(T)\, e^{-p\Lambda(T)} = \frac{c_3}{N} \tag{11}$$

In the case of a non increasing failure rate $\lambda(t)$ the solution gives a unique minimum if

$$1 - \frac{c_1}{c_2} > \frac{p[\lambda(T + \tau)]^2 - \lambda'(T + \tau)}{p[\lambda(T)]^2 - \lambda'(T)}\; e^{-p[\Lambda(T+\tau)-\Lambda(T)]} \tag{12}$$

4 Numerical Examples

We now apply our general results to specific failure rate models. As our first example we take $\lambda(t) = be^{-bt}$. Then $\Lambda(t) = 1 - e^{-bt}$. We illustrate the example with values of N and b suggested by the failure data set DS1 discussed by Goel (1980). This data gives rise to estimates of $N = 1348$ and $b = 0.124$. We also use the same cost parameters of $c_1 = 1, c_2 = 5, c_3 = 100$ suggested by Okumoto and Goel (1980) (and used by Boland and Singh (2002) and (2003)) in order to make comparisons with other studies.

Table 1 gives the optimal testing time T^* for different values of p (the probability of perfect fault repair), for the cases where the fixed life cycle time is $t_0 = 100$ and for the situation when we are interested in the software functioning well during a mission time of $\tau = 2, 5, 10, 20, 50, 100$. Figure 1 illustrates the behavior of T^* as a function of p plotted for a number of values of τ.

Accepted manuscript

ACCEPTED MANUSCRIPT

p	$t_0 = 100$	$\tau = 2$	$\tau = 5$	$\tau = 10$	$\tau = 20$	$\tau = 50$	$\tau = 100$
0.2	13.99	0	6.13	10.69	13.17	13.98	13.99
0.4	12.76	0	5.94	9.80	12.01	12.74	12.76
0.6	11.63	0	5.73	9.02	10.96	11.61	11.63
0.8	10.60	0.61	5.53	8.34	10.02	10.59	10.60
1.0	9.68	1.10	5.33	7.74	9.18	9.67	9.68

Table 1: Optimal Testing Time T^* when $\lambda(t) = 0.124\, e^{-0.124\, t}$

The corresponding minimum expected costs are given in the following table:

p	$t_0 = 100$	$\tau = 2$	$\tau = 5$	$\tau = 10$	$\tau = 20$	$\tau = 50$	$\tau = 100$
0.2	3414	1448	2575	3068	3328	3412	3414
0.4	3161	1417	2387	2835	3080	3159	3161
0.6	2928	1387	2218	2624	2852	2927	2928
0.8	2714	1351	2067	2433	2643	2712	2714
1.0	2517	1306	1932	2261	2452	2516	2517

Table 2: Minimum Expected Cost when $\lambda(t) = 0.124\, e^{-0.124\, t}$

For our second example we take $\lambda(t) = b/(1 + bt)$. Then $\Lambda(t) = \log(1 + b\, t)$. For the case of a fixed life cycle time t_0, the optimal testing time T^* is given by $min(t_0, max(0, T_1))$, where

$$T_1 = \frac{1}{b}\left[\left(\frac{Nb(c_1 - c_2)}{c_3}\right)^{\frac{1}{1+p}} - 1\right]$$

For the case of a fixed mission time τ, $T^* = max(0, T_2)$ where T_2 is given as the solution of

$$\frac{(c_2 - c_1)b}{(1 + bT)^{1+p}} - \frac{c_2 b}{(1 + b(T + \tau))^{1+p}} = \frac{c_3}{N}. \tag{13}$$

The optimal testing times T^* and the corresponding minimal expected costs are given in Tables 3 and 4.

p	$t_0 = 100$	$\tau = 2$	$\tau = 5$	$\tau = 10$	$\tau = 20$	$\tau = 50$	$\tau = 100$
0.2	31.22	0	2.65	7.10	12.19	19.12	23.65
0.4	23.27	0	3.44	7.52	11.80	16.98	19.87
0.6	18.38	0	3.96	7.60	11.16	14.98	16.79
0.8	15.11	0.61	4.28	7.50	10.43	13.22	14.34
1.0	12.79	0.63	4.47	7.28	9.68	11.71	12.41

Table 3: Optimal Testing Time T^* when $\lambda(t) = 0.124/(1 + 0.124\, t)$

Accepted manuscript

ACCEPTED MANUSCRIPT

p	$t_0 = 100$	$\tau = 2$	$\tau = 5$	$\tau = 10$	$\tau = 20$	$\tau = 50$	$\tau = 100$
0.2	9450	1461	2985	4371	5951	8345	10330
0.4	7563	1429	2755	3861	5057	6731	7981
0.6	6124	1398	2539	3432	4351	5537	6329
0.8	5036	1368	2342	3071	3785	4634	5139
1.0	4209	1330	2164	2765	3326	3939	4262

Table 4: Minimum Expected Cost when $\lambda(t) = 0.124/(1 + 0.124\,t)$

We compare these two examples with the results obtained in Boland & Ni Chuiv (2001). In that case $\lambda(t) = b$. The results are tabulated in Tables 5 and 6.

p	$t_0 = 100$	$\tau = 2$	$\tau = 5$	$\tau = 10$	$\tau = 20$	$\tau = 50$	$\tau = 100$
0.2	76.61	0	0	0	18.87	58.51	72.16
0.4	38.31	0	0	9.43	25.75	36.08	38.13
0.6	25.54	0	0	13.42	21.08	25.12	25.53
0.8	19.15	0	4.72	12.88	17.25	19.06	19.15
1.0	15.32	0	6.32	11.70	14.43	15.30	15.32

Table 5: Optimal Testing Time T^* **when** $\lambda(t) = 0.124$

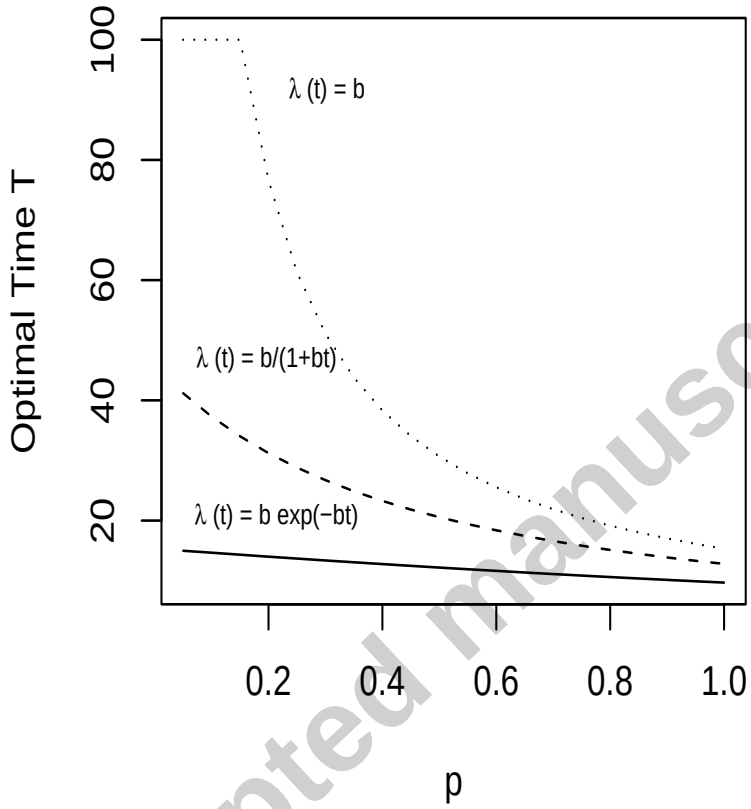
p	$t_0 = 100$	$\tau = 2$	$\tau = 5$	$\tau = 10$	$\tau = 20$	$\tau = 50$	$\tau = 100$
0.2	15612	1631	3930	7402	12659	16623	17988
0.4	9099	1591	3701	6329	7961	8994	9199
0.6	6138	1553	3490	4933	5699	6103	6144
0.8	4608	1516	3165	3981	4418	4600	4608
1.0	3687	1480	2787	3325	3598	3685	3687

Table 6: Minimum Expected Cost when $\lambda(t) = 0.124$

Accepted manuscript

One would naturally expect to observe some monotonic trends or relationships between T^*, p, either t_0 or τ, and the function $\lambda(t)$ for a given cost structure. Some interesting observations may be made from our results. Note that each of our 3 detection (failure) rate functions start at b, are decreasing, and moreover $b\,e^{-b\,t} \leq b/(1 + b\,t) \leq b$ for all $t > 0$. For case 1 (fixed life cycle time $t_0 = 100$), it appears that T^* is a decreasing function of p for any of these detection

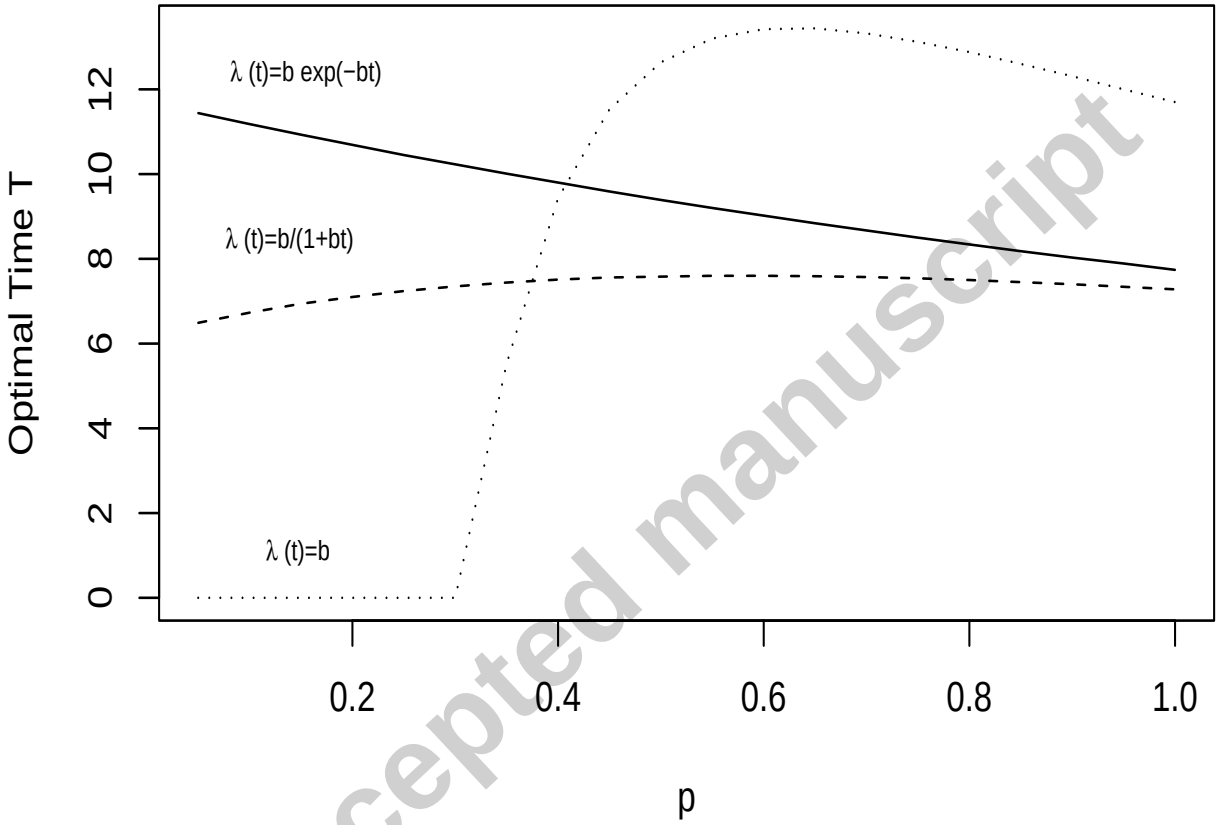
ACCEPTED MANUSCRIPT



Accepted manuscript

Figure 1: Optimal Testing Times for 3 Models with $t_0 = 100$ and $b = 0.124$.

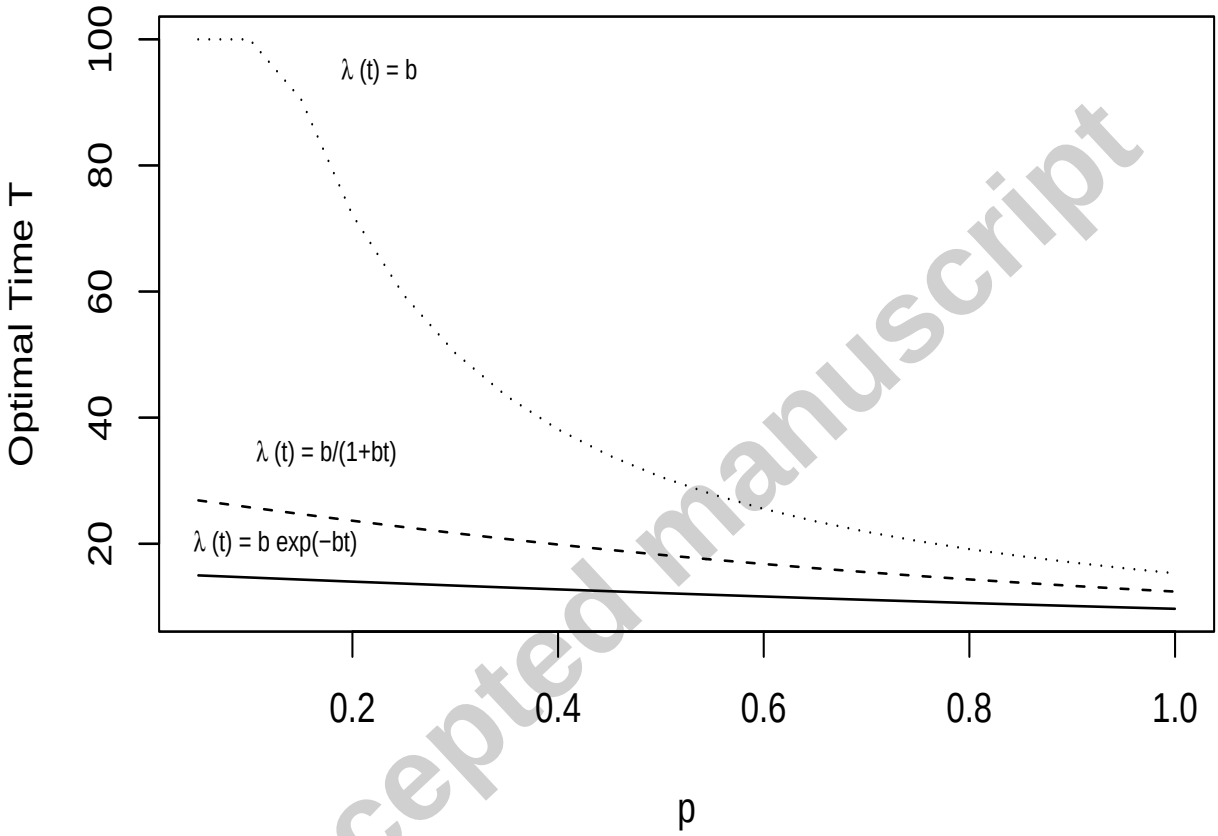
ACCEPTED MANUSCRIPT



Accepted manuscript

Figure 2: Optimal Testing Times for 3 Models with $\tau = 10$ and $b = 0.124$.

ACCEPTED MANUSCRIPT



Accepted manuscript

Figure 3: Optimal Testing Times for 3 Models with $\tau = 100$ and $b = 0.124$.

ACCEPTED MANUSCRIPT

rate functions, and moreover for fixed p it increases with the failure rate function. For case 2 (mission time of τ), T^* appears to be an increasing function of the mission time τ for fixed p and fixed detection rate function. However, somewhat surprisingly, it is not necessarily a decreasing function of p for any fixed τ and given detection rate function.

References

[1] Boland, P.J. and Ni Chuiv N. (2001). "Cost Implications of Imperfect Repair in Software Reliability", International Journal of Reliability and Applications, **2**, 3, 147-160.

[2] Boland, P.J. and Singh, H. (2002). "Determining the Optimal Release Time for Software in the Geometric Poisson Reliability Model", International Journal of Reliability, Quality and Safety Engineering, **9**, 3, 201-213.

[3] Boland, P.J. and Singh, H. (2003). "A Birth Process approach to Moranda's Geometric software Reliability Model", IEEE Transactions in Reliability, volume 52, 2, 168-174.

[4] Dalal, S. R. and Mallows, C. L. (1988). "When to stop testing software". Journal of American Statistical Association, **83**, 872-879.

[5] Farr, W. (1996). Software Reliability Modeling Survey, Chapter 3 in **Handbook of Software Reliability Engineering**, edited by M. R. Lyu, McGraw Hill.

[6] Finkelstein M.S. (1997). "Imperfect repair models for systems subject to shocks". Applied Stochastic Models and Data Analysis. **13**, 385-390.

[7] Goel, A.L. (1980). "Software Error Detection model with applications". The Journal of Systems and Software, **1**, 243-249.

[8] Goel, A.L. and Okumoto, K. (1978). "An Analysis of Recurrent Software Failures on a Real-Time Control System". In Proceedings of the ACM Annual Technical Conference, 496-500.

[9] Jelinski, Z. and Moranda, P. (1972). "Software Reliability Research". **Statistical Computer Performance Evaluation**, Ed. W. Freiberger, 465-84, New York, Academic.

[10] McDaid, K. and Wilson, S. (2001). "Deciding how long to test software." Journal of the Royal Statistical Society, Series D, R-50. 117-134.

Accepted manuscript

ACCEPTED MANUSCRIPT

[11] Moranda, P.B. (1975). "Prediction of Software Reliability Software During Debugging". Proceedings on the 1975 Annual Reliability and Maintainability Symposium, 327-32.

[12] Musa, J. D., Iannino, A. and Okumoto, K. (1987). **Software Reliability: Measurement, Prediction, Application**, New York Wiley.

[13] Okumoto, K. and Goel, A. L. (1980). "Optimum release time for software systems based on reliability and cost criteria". The Journal of Systems and Software, **1**, 315-318.

[14] Singpurwalla, N. D. and Wilson, S. P. (1999). **Statistical Methods in Software Engineering: Reliability and Risk**. Springer.

[15] Singpurwalla, N. D. (1991). "Determining an optimal time interval for testing and debugging software". IEEE Transactions on Software Engineering, **17**, 313-319.

[16] Xie, M. (2000) "Software Reliability Models - Past, Present and Future". in **Recent Advances in Reliability Theory - Methodology, Practive and Inference.** Eds Limnios, N and Nikulin M., Birkhauser, 325-340.

Accepted manuscript