An optimal statistical testing policy for software reliability demonstration of safety-critical systems

O. Tal[a,*], A. Bendell[b] and C. McCollin[c]

[a] *Department of Mathematics, Statistics and Operational Research, Nottingham Trent University, Burton Street, Nottingham NG1 B4U, UK*

[b] *The Management Centre, University of Leicester, University Road, Leicester LE1 7RH, UK*

[c] *Department of Mathematics, Statistics and Operational Research, Nottingham Trent University, Burton Street, Nottingham NG1 B4U, UK*

* Corresponding author. Fax: +44 115-8482998

Abstract

When software reliability demonstration of safety-critical systems by statistical testing is treated as a TAAF (Test, Analyse and Fix) process, an optimal testing policy can be found, which maximises the probability of success of the whole process, over a pre-determined period of time. The optimisation problem is formulated, solved by stochastic dynamic programming, and demonstrated by two numerical examples.

Keywords: Reliability; Software reliability demonstration; Statistical testing; Safety-critical systems; Stochastic dynamic programming.

1. Introduction

Reliability demonstration by statistical testing is a standard procedure for hardware systems[1], which has also been suggested for software systems[2]. In both cases the testing is based on PRST- Probability Ratio Sequential Testing [3]. An alternative method for safety-critical systems, SRST (Single Risk Sequential Testing), has been suggested, based on the binomial formula [4]. Combining the SRST method with the TAAF (Test, Analyse and Fix) approach [5] leads to an optimisation problem, whose objective is to maximise the probability of success of the whole process [6].

2. Problem parameters

- The **required reliability** is **1-θ**. Hence, θ is the maximum allowed unreliability.
- The **required confidence** in the value of 1-θ is α.
- The **total time** allocated to reliability demonstration testing is $\mathbf{T_{max}}$ [hours].
- The **average mission duration** of the system is $\mathbf{T_{ave}}$ [hours].
- The **estimated failure rate** of the software is λ [1/hour].

Assumption 1: the failure rate of the software is small, and does not change much from version to version during the software reliability demonstration process. Hence it will conservatively assumed to be constant.

- The **average correction time** required for correcting the current software version and submitting a new version for reliability demonstration testing is $\mathbf{T_{corr}}$ [hours].

3. Problem Analysis and definitions

The required number of tests, **n**, for any number of failures found, **F**, is the largest numerical solution of the following inequality, based on the binomial formula [7]:

$$1-\alpha \geq \sum_{j=0}^{F} nCj\,(1-\theta)^{n-j}\,\theta^{j} \qquad F=0,1,2.... \qquad (1)$$

Let us define $\mathbf{n_F}$ as the required number of missions during reliability demonstration testing when F failures are found, and $\mathbf{T_F}$ as the expected required time:

$$T_F = n_F\, T_{ave} \qquad (2)$$

For instance, for F=0 equation (1) reduces to:

$$1-\alpha \geq (1-\theta)^{n} \qquad (3)$$

Therefore:

$\mathbf{n_0=\ln(1-\alpha)/\ln(1-\theta)}$ where $\mathbf{n_0}$ is the greatest **n** to satisfy the inequality.

(4)

and

$$T_0 = n_0\, T_{ave} \qquad (5)$$

For any number of failures, F, there is a corresponding n_F and hence a corresponding T_F. For example, for 1-θ=0.999, α=0.95 and T_{ave}=1 hour, T_0=2,995, T_1=4,742, T_2=6,294 hours, etc. The **maximum allowable number of failures during reliability demonstration**, **F_{max}**, is therefore the smallest F such that:

$$\mathbf{T_{F+1}=n_{F+1}\,T_{ave} \geq T_{max}} \qquad \textbf{(6)}$$

Using the TAAF approach the software reliability demonstration testing can be regarded as a process [6]: Test the current version to demonstrate its reliability. If no failures are found during T_0 hours, stop testing. Clearly, T_{max} must be larger than T_0, otherwise there is no way of demonstrating the required reliability. When the Fth failure is found, choose between the following options:

1. Correct the current version to produce a new version and start testing all over again. This option requires, of course, that for the new version $T_{max} \geq T_0+T_{corr}$.
2. Go on testing, aiming at T_F hours with only f failures, f=F,F+1,... F_{max}.

Because of the time constraint (T_{max}), this process can either succeed or fail.

4. Problem formulation and policy definition

What is the optimal testing policy, in order to maximise the probability of success in the software reliability demonstration process?

Any testing policy can be defined by a **decision rule P(F,t)** where F is the **number of the failure just found** in the current version, and t is the **time left** till the deadline (T_{max} hours). P(F,t) can only take three values:

1. GO-ON testing;
2. FIX and start testing;
3. STOP testing: the reliability demonstration has either succeeded or failed.

The probability of success of this policy can be defined is p[P(F,t)]. Therefore, the optimisation problem is:

$$\mathbf{\max\ p[P(F,t)] \quad for\ 0 \leq F \leq F_{max},\ 0<t \leq T_{max}} \qquad \textbf{(7)}$$

The maximum number of **new** versions when a failure is found t hours before the deadline can be calculated as follows:

Assuming that there is enough time for at least one more version, $t \geq T_{corr}+T_0$, and the testing of the first new version will start T_{corr} hours later. If the first failure in the new version is found immediately, then the testing of the second new version will start at least $2T_{corr}$ hours later. Similarly, the testing of the nth new version will start at least nT_{corr} hours later. The only constraint is that the time left, $t-nT_{corr}$, is still larger than $T_{corr}+T_0$. Therefore, the maximum number of new versions, **excluding the immediate one**, is the integer solution of:

$$\mathbf{t-vT_{corr} \geq T_{corr}+T_0} \qquad \mathbf{(8)}$$

And

$$\mathbf{v_{max}=v+1=1+int\{[T_{max}-(T_{corr}+T_0)]/T_{corr}\}=int[(T_{max}-T_0)/T_{corr}]} \qquad \mathbf{(9)}$$

The optimal policy depends on F_{max}, and therefore will be considered separately for several cases.

4.1 Case no. 0: $F_{max}=0$ or $T_0 \leq T_{max} \leq T_1$

When testing begins, the optimal policy is, clearly, GO-ON testing. The optimal policy does not change as long as no failures are found, till T_0 hours have passed, and the required reliability has been demonstrated. Therefore:

P(0,t)=	**GO-ON**	$\mathbf{T_{max}-T_0<t\leq T_{max}}$
	STOP (success)	$\mathbf{0<t\leq T_{max}-T_0}$

When the first failure is found there is no choice but to FIX the current version and start testing the new version, if there is enough time to do it. Therefore:

P(1,t)=	**FIX**	$\mathbf{T_0+T_{corr}<t\leq T_{max}}$
	STOP (failure)	$\mathbf{0<t\leq T_0+T_{corr}}$

The optimal policy is described graphically in figure 1. This figure is based on the fact that $T_0+T_{corr}>T_{max}-T_0$, which is always true. In order for this inequality to be false, one gets $2T_0+T_{corr}<T_{max}$. But since $T_{max}<T_1$, and since T_1 is always smaller than $2T_0$ [6], the figure

is valid. Whenever the testing of a new version is started, the same graph has to be plotted again, with the new value of T_{max}.

4.2 Case no. 1: $F_{max}=1$ or $T_1 \leq T_{max} \leq T_2$

4.2.1 P(0,t)

When testing begins, the optimal policy is, clearly, GO-ON testing. The optimal policy does not change as long as no failures are found, till T_0 hours have passed, and the required reliability has been demonstrated. Therefore:

P(0,t)=	**GO-ON**	$T_{max}-T_0<t\leq T_{max}$
	STOP (success)	$0\leq t\leq T_{max}-T_0$

4.2.2 P(2,t)

When the second failure is found there is no choice but to FIX the current version and start testing the new version, if there is enough time to do it. Therefore:

P(2,t)=	**FIX**	$T_0+T_{corr}<t\leq T_{max}$
	STOP (failure)	$0\leq t\leq T_0+T_{corr}$

4.2.3 P(1,t)

When the first failure is found there can be two situations:

- There is not enough time to FIX the software, i.e. $t<T_0+T_{corr}$. In this case the only choice is GO-ON testing, till T_1 hours have passed.

P(1,t)=	**GO-ON**	$T_{max}-T_1<t<T_0+T_{corr}$
	STOP (success)	$0\leq t\leq T_{max}-T_1$

- There is enough time to FIX the software. i.e. $T_0+T_{corr}\leq t\leq T_{max}$. In this case there is a real choice between GO-ON testing and FIXing the software. Since at $t=T_{max}$ the optimal decision is FIX and at $t< T_0+T_{corr}$ the optimal decision is GO-ON testing, there may be a “**break-even” point** in this region, when the best policy shifts from FIX to GO-ON. In this case, the break-even point can be found by equating the probabilities of success of both options.

Figures 2 and 3 describe the optimal policies in both cases.

5. Formulating the problem as a stochastic dynamic programming problem

Finite-stage stochastic dynamic programming deals with a process, whose **state** at the beginning of a given time period is known. After observing its state an **action** must be chosen. Based on the state and the action only, an expected **reward** is earned, and the **probability distribution** for the state in the next time period is determined. The optimisation problem is to find a policy, that maximises the expected value of the sum of rewards earned over a given number of time periods, or **stages**. The optimality equation, which enables a recursive solution of this problem, is given by [8]:

$$\mathbf{V_n(i)=\max_a [R(i,a)+\sum_j P_{ij}(a)V_{n-1}(j)]} \qquad \mathbf{(10)}$$

$$\mathbf{V_1(i)=\max_a R(i,a)} \qquad \mathbf{(11)}$$

where:

n- the total number of stages at the beginning of the process

i- the starting state

a- one of the possible actions

R(i,a)- the expected (immediate) reward of action a in state i.

P_{ij}(a)- the transition probability from state i to state j, as a result of choosing action a.

V_n(i)- the maximum expected return for an n-stage problem, that starts in state i.

This formulation assumes that R(i,a) and P_{ij}(a) do not depend on n. It also assumes that the process must go through all the stages, from n to 1, because each action causes a new stage.

In order to formulate the software reliability demonstration optimisation problem as a stochastic dynamic programming problem, the following modifications are needed [6]:

1. **Stage**: n is defined as the maximum number of new versions which can still be produced and demonstrated till the deadline, which is determined only upon **finding a new failure**. Therefore it is possible to **stay in the same stage** or to **skip stages** following an action. The minimum value of n is **0**.
2. **State**: the state of the problem is defined by i, the number of the failure just found, **and by t**, the time left till the deadline. Whenever a new version arrives, i=0. The maximum i is F_{max}- the maximum allowable number of failures.
3. **Actions: t**he possible actions are FIX and GO-ON. Each action determines not only the immediate reward and the probability distribution of the next state, but also the **time left** at the beginning of the next stage (whose number does not have to be the next smaller integer).
4. **Reward:** the expected reward of GO-ON is the probability of demonstrating the software reliability of the **current version without any more failures**. It depends on i **and t**. Therefore the expected reward of FIX is zero.
5. **Transition probability:** the transition probability from state i **in stage n** to state j **in stage k** depends on a, **t and n.**

Incorporating these changes to the optimality equation yields the following general equations:

$$\mathbf{V_n(i,t)=\max_{a} [R(i,a,t)+\sum_{j,k} P_{ijk}(n,a,t)V_k(j,t-T_a)]} \qquad \mathbf{(12)}$$

$$\mathbf{V_0(F_{max},t)=p(0,t-T_{max}+T_{Fmax})} \qquad \mathbf{(13)}$$

where:

t- the time left until the deadline upon finding a failure, $T_{max} \geq t > 0$

i- the number of the failure just found, $i=0,1,....F_{max}$, counted separately for each version.

n- the maximum number of new versions which can be tested till the deadline, $v_{max} \geq n \geq 0$

Note: t and n are not independent, because for $n \geq 1$, $T_0+(n+1)T_{corr} > t \geq T_0+nT_{corr}$. For n=0, $T_0+T_{corr} > t > 0$.

$V_n(i,t)$ - the maximum probability of success when the maximum number of new versions is n, the number of the failure just found is i, and the time left is t.

a- FIX or GO-ON

R(i,a,t) - the probability of demonstrating the software reliability of the **current version without any more failures** (0 for the FIX option).

$P_{ijk}(n,a,t)$ - the probability of transition from state i in stage n to state j in stage k following action a in time t, because of finding a failure.

Note:

In the FIX option, j is always 0 and k is always n-1.

In the GO-ON option:

For $i<F_{max}$ j=i+1, k=0,1,....n.

For $i=F_{max}$ j=0, k=0,1,..... n-1.

T_a - the time lost in the transition to a new stage following action a: T_{corr} in case of FIX, and the expected time till the next failure in case of GO-ON.

Note: the time to failure is denoted by z, and its pdf is assumed to be $f(z)=\lambda\exp(-\lambda z)$.

After using the above constraints these equations become:

$$\mathbf{V_n(0,t)=p(0,T_0)+}\sum_{k=0}^{n}\int f(z)\,\mathbf{V_k(1,t-z)dz} \qquad \mathbf{(14)}$$

$$\mathbf{V_n(i,t)=\max}\begin{cases}\mathbf{V_{n-1}(0,t-T_{corr})} \\ \mathbf{p(0,t-T_{max}+T_i)+}\sum_{k=0}^{n}\int f(z)\,\mathbf{V_k(i+1,t-z)dz}\end{cases} \qquad \mathbf{i=1,....F_{max}-1} \qquad \mathbf{(15)}$$

$$\mathbf{V_n(F_{max},t)=\ \max}\begin{cases}\mathbf{V_{n-1}(0,t-T_{corr})} \\ \mathbf{p(0,t-T_{max}+T_{Fmax})+}\sum_{k=0}^{n}\int f(z)\,\mathbf{V_k(0,t-z-T_{corr})dz}\end{cases} \qquad \mathbf{(16)}$$

The integration boundaries are as follows:

1. i=0

k=0	$z=t-T_0-T_{corr}$	to	T_0
k=1,2,...n-1:	$z=t-T_0-(k+1)\,T_{corr}$	to	$t-T_0-k\,T_{corr}$
k=n:	z=0	to	$t-T_0-n\,T_{corr}$.

2. $i=1,2,....F_{max}-1$

k=0:	$z=t-T_0-T_{corr}$	to	$t-T_{max}+T_i$
k=1,2,...n-1:	$z=t-T_0-(k+1)\,T_{corr}$	to	$t-T_0-k\,T_{corr}$
k=n:	$z=0$	to	$t-T_0-n\,T_{corr}$.

3. $i=F_{max}$

k=0:	$z=0$	to	$t-T_{max}+T_{Fmax}$
k=1,2,...n-2:	$z=t-T_0-(k+2)\,T_{corr}$	to	$t-T_0-(k+1)\,T_{corr}$
k=n-1:	$z=0$	to	$t-T_0-n\,T_{corr}$.

Therefore the optimality equations become:

$$\mathbf{V_n(F_{max},t)=max}\begin{array}{ll} \mathbf{V_{n-1}(0,t-T_{corr})} & \mathbf{(FIX)} \\ \mathbf{p(0,t-T_{max}+T_{Fmax})}+\int_0^{t-T0-nTcorr} f(z)\,\mathbf{V_{n-1}(0,t-z-T_{corr})dz}+ & \\ +\sum_{k=1}^{n-2}\int_{t-T0-(k+2)Tcorr}^{t-T0-(k+1)Tcorr} f(z)\,\mathbf{V_k(0,t-z-T_{corr})dz}+ & \\ +\int_0^{t-T\max+TF\max} f(z)\,\mathbf{V_0(0.t-z-T_{corr})dz} & \mathbf{(GO\text{-}ON)} \end{array} \qquad \mathbf{(17)}$$

$$\begin{array}{ll} & \mathbf{V_{n-1}(0,t-T_{corr})} \qquad\qquad \mathbf{(FIX)} \\ \mathbf{V_n(i,t)=max} & \mathbf{p(0,t-T_{max}+T_i)}+\int_0^{t-T0-nTcorr} f(z)\,\mathbf{V_n(i+1,t-z)dz}+ \\ \mathbf{(i=1,2,..F_{max}-1)} & +\sum_{k=1}^{n-1}\int_{t-T0-(k+1)Tcorr}^{t-T0-kTcorr} f(z)\,\mathbf{V_k(i+1,t-z)dz}+ \\ & +\int_{t-T0-Tcorr}^{t-T\max+Ti} f(z)\,\mathbf{V_0(i+1,t-z)dz} \qquad \mathbf{(GO\text{-}ON)} \end{array} \qquad \mathbf{(18)}$$

$$\mathbf{V_n(0,t)=p(0,T_0)}+\int_0^{t-T0-nTcorr} f(z)\,\mathbf{V_n(1,t-z)dz}+\sum_{j=1}^{n-1}\int_{t-T0-(j+1)Tcorr}^{t-T0-jTcorr} f(z)\,\mathbf{V_j\,(1,t-z)dz}+ \qquad \mathbf{(19)}$$
$$+\int_{t-T0-Tcorr}^{T0} f(z)\,\mathbf{V_0(1,t-z)dz}$$

6. The algorithm

The algorithm for finding the T_i* values ($i=1,2,..F_{max}$) is as follows:

1. n=0.

2. $i=F_{max}$. Use eq. (17) to calculate $V_n(F_{max},t)$.

3. $i=i-1$. If $i\geq 1$ Use eq. (18) to calculate $V_n(i,t)$ and check for a break-even point in the region $T_0+(n+1)T_{corr}>t\geq T_0+nT_{corr}$. If it exists, **this is T_i***.
4. If $i=0$ and $n=v_{max}$: **stop**. Else use eq. (19) to calculate $V_n(0,t)$.
5. $n=n+1$. Go to step 2. Do not look again for the break-even points which have already been found.

7. Numerical examples

The optimal policy may be expressed by a series of **break-even points**, T_i^* ($i=1,2,...F_{max}$) such that when the ith failure is found, the optimal policy is FIX as long as $t>T_i^*$ and GO-ON when $t<T_i^*$.

7.1 Numerical Example no. 1

Data

$1-\theta=0.999$, $\alpha=0.95$, $\lambda=1/3000$, $T_{max}=5{,}000$ hr, $T_{ave}=1$ hr, **$T_{corr}=600$ hr**

Preliminary calculations

Using eq. (1) and (2):

$T_0=2{,}995\approx 3{,}000$ hr, $T_1=4{,}742\approx 4{,}750$ hr, $T_2=6{,}294$ hr$>T_{max}$

$\therefore F_{max}=1$

Using eq. (9):

$v_{max}=\text{int}[(T_{max}-T_0)/T_{corr}]=\text{int}[(5000-3000)/600]=3$

Recursive calculations

According to eq. (17):

$\mathbf{V_0(1,t)}= p(0,t-T_{max}+T_1)=p(0,t-5000+4750)=\mathbf{exp[-(t-250)/3000]}$ $\qquad 3600>t\geq 2000$

Explanation:

t can not be smaller than 2,000 hours, because then there would be no failures for at least 3,000 hours (T_0). Upon finding the first failure, the only option is GO-ON, because there is no time for a new version.

According to eq. (19):

$\mathbf{V_0(0,t)}$= $p(0,T_0)=\exp(-\lambda T_0)=\exp(-1)=\mathbf{0.368}$ $\qquad 3600>t\geq 3000$

Explanation:

If the testing of a new version starts at this stage, then once the first failure is found, there is no time for a new version nor enough time to demonstrate T_1 hours with 1 failure.

$$\mathbf{V_1(1,t)}=\max\begin{cases} V_0(0,t-T_{corr}) & \text{(FIX)} \\ p(0,t-T_{max}+T_1)+\int_0^{t-T0-Tcorr} f(z)\, V_0(0,t-z-T_{corr})dz & \text{(GO-ON)}\end{cases} \qquad 4200>t\geq 3600$$

The value of the FIX option is:

$\mathbf{V_0(0,t-T_{corr})}=V_0(0,t-600)=\exp(-\lambda T_0)=\mathbf{0.368}$

Explanation:

for $4200>t\geq 3600$ t-600 is between 3,000 and 3,600, in which $V_0(0,t)=0.368$.

The value of the GO-ON option is the sum of two expressions. The first expression is:

$\mathbf{p(0,t-T_{max}+T_1)}=p(0,t-5000+4750)=p(0,t-250)=\mathbf{\exp[-\lambda(t-250)]}$

The second expression is:

$$\int_0^{t-T0-Tcorr} f(z)\,\mathbf{V_0(0,t-z-T_{corr})dz}=\int_0^{t-3600}\lambda\exp(-\lambda z)\,\mathbf{V_0(0,t-z-600)dz=\exp(-\lambda T_0)\{1-\exp[-\lambda(t-3600)]\}}$$

And therefore:

$$\mathbf{V_1(1,t)=\max\begin{cases} \exp(-\lambda T_0) & \text{(FIX)} \\ \exp[-\lambda(t-250)]+\exp(-\lambda T_0)\{1-\exp[-\lambda(t-3600)]\} & \text{(GO-ON)}\end{cases} \qquad 4200>t\geq 3600}$$

For t=3,600 the value of the GO-ON option is **0.327**, less than the FIX option (0.368).

For t=4,200 the value of the GO-ON option is **0.334**, less than the FIX option (0.368).

Therefore **there is no break-even point** in this region, and:

$\mathbf{V_1(1,t)= \exp(-\lambda T_0)}$ $\qquad \mathbf{4200>t\geq 3600}$

Since the optimal policy for t<3,600 is GO-ON (there is no time for a new version), **then t=3,600 is the break-even point for $V_1(1,t)$,** and there is no need to continue the recursive calculations. The optimal testing policy is graphically described in figure 4.

7.2 Numerical Example no. 2

Data

$1-\theta=0.999$, $\alpha=0.95$, $\lambda=1/3000$, $T_{max}=5{,}000$ hr, $T_{ave}=1$ hr, $\mathbf{T_{corr}=200}$ **hr**

Preliminary calculations

Using eq. (1) and (2):

$T_0=2{,}995\approx3{,}000$ hr, $T_1=4{,}742\approx4{,}750$ hr, $T_2=6{,}294$ hr$>T_{max}$

$\therefore\ F_{max}=1$

Using eq. (9):

$v_{max}=\text{int}[(T_{max}-T_0)/T_{corr}]=\text{int}[(5000-3000)/200]=10$

Recursive calculations

According to eq. (17):

$\mathbf{V_0(1,t)}=p(0,t-T_{max}+T_1)=p(0,t-5000+4750)=\mathbf{\exp[-(t-250)/3000]}$ $\qquad 3200>t\geq2000$

Explanation:

t can not be smaller than 2,000 hours, because then there would be no failures for at least 3,000 hours (T_0). Upon finding the first failure, the only option is GO-ON, because there is no time for a new version. According to eq. (19):

$\mathbf{V_0(0,t)}=p(0,T_0)=\exp(-\lambda T_0)=\exp(-1)=\mathbf{0.368}$ $\qquad 3200>t\geq3000$

Explanation:

If the testing of a new version starts at this stage, then once the first failure is found, there is no time for a new version nor enough time to demonstrate T_1 hours with 1 failure.

$$\mathbf{V_1(1,t)}=\max\begin{cases} V_0(0,t-T_{corr}) & \text{(FIX)} \\ p(0,t-T_{max}+T_1)+\int_0^{t-T0-Tcorr} f(z)\, V_0(0,t-z-T_{corr})dz & \text{(GO-ON)}\end{cases} \qquad 3400>t\geq3200$$

The value of the FIX option is:

$\mathbf{V_0(0,t-T_{corr})}=V_0(0,t-200)=\exp(-\lambda T_0)=\mathbf{0.368}$

Explanation:

For $3400>t\geq3200$ t-600 is between 3,000 and 3,200, in which $V_0(0,t)=0.368$.

The value of the GO-ON option is the sum of two expressions. The first expression is:

$\mathbf{p(0,t-T_{max}+T_1)}=p(0,t-5000+4750)=p(0,t-250)=\mathbf{exp[-\lambda(t-250)]}$

The second expression is:

$$\int_0^{t-T0-Tcorr} f(z)\, \mathbf{V_0(0,t-z-T_{corr})dz}=\int_0^{t-3200}\lambda\exp(-\lambda z)\, \mathbf{V_0(0,t-z-600)dz=exp(-\lambda T_0)\{1-exp[-\lambda(t-3200)]\}}$$

And therefore:

$$\mathbf{V_1(1,t)=max}\begin{cases}\mathbf{exp(-\lambda T_0)} & \mathbf{(FIX)} \\ \mathbf{exp[-\lambda(t-250)]+exp(-\lambda T_0)\{1-exp[-\lambda(t-3200)]\}} & \mathbf{(GO\text{-}ON)}\end{cases} \qquad \mathbf{3400>t\geq3200}$$

For t=3,200 the value of the GO-ON option is **0.374**, more than the FIX option (0.368).

For t=3,400 the value of the GO-ON option is **0.398**, more than the FIX option (0.368).

Therefore **there is no break-even point** in this region, and:

$$\mathbf{V_1(1,t)= exp[-\lambda(t-250)]+exp(-\lambda T_0)\{1-exp[-\lambda(t-3200)]\}} \qquad \mathbf{3400>t\geq3200}$$

According to eq. (19):

$$\mathbf{V_1(0,t)=p(0,T_0)+} \int_0^{t-T0-Tcorr} f(z)\, \mathbf{V_1(1,t-z)dz+}\int_{t-T0-Tcorr}^{t-T0} f(z)\, \mathbf{V_0(1,t-z)dz} \qquad \mathbf{3400>t\geq3200}$$

$\mathbf{V_1(0,t)}$ is the sum of three expressions. The first expression is:

$\mathbf{p(0,T_0)= exp(-\lambda T_0)}$

The second expression is:

$$\int_0^{t-3200} f(z)\, \mathbf{V_1(1,t-z)dz=}\int_0^{t-3200} f(z)\, \mathbf{V_0(,0,t-z-200)dz=}\int_0^{t-3200} f(z)\, \mathbf{exp(-\lambda T_0)=}$$

$$\mathbf{=exp(-\lambda T_0)\{1-exp[-\lambda(t-3200)]\}dz}$$

Explanation:

When the next failure is found in V_1 there is no time for demonstrating T_1 hours with only 1 failure, and the only option would be FIX.

The third expression is:

$$\int_{t-3200}^{t-3000} f(z)\, \mathbf{V_0(1,t-z)dz=0}$$

Explanation:

When the next failure is found in V_0, there is no time for a new version nor for demonstrating T_1 hours with only 1 failure, and the testing is over.

Hence:

$$\mathbf{V_1(0,t)= \exp(-\lambda T_0)+ \exp(-\lambda T_0)\{1-\exp[-\lambda(t-3200)]\}} \qquad \mathbf{3400>t\geq 3200}$$

According to eq. (17):

$$\mathbf{V_2(1,t)=\max}\begin{cases} \mathbf{V_1(0,t-200)} & \mathbf{(FIX)} \\ & \mathbf{3600>t\geq 3400} \\ \mathbf{p(0,t-250)+\int_0^{t-3400} f(z)\, V_1(0,t-z-200)dz+\int_{t-3400}^{t-3200} f(z)\, V_0(0,t-z-200)dz} & \mathbf{(GO\text{-}ON)} \end{cases}$$

The value of the FIX option is:

$$\mathbf{V_1(0,t-200)=\exp(-\lambda T_0)+\exp(-\lambda T_0)\{1-\exp[-\lambda(t-3400)]\}} \qquad \mathbf{3600>t\geq 3400}$$

The value of the GO-ON option includes three expressions. The first expression is:

$$\mathbf{p(0,t-250)=\exp[-\lambda(t-250)]}$$

The second expression is:

$$\int_0^{t-3400} f(z)\, \mathbf{V_1(0,t-z-200)dz}=\int_0^{t-3400} f(z)\, \{\exp(-\lambda T_0)+\exp(-\lambda T_0)\{1-\exp[-\lambda(t-z-3400)]\}\}dz=$$

$$= 2\exp(-\lambda T_0)\{1-\exp[-\lambda(t-3400)]\}- \exp(-\lambda T_0)\lambda(t-3400) \exp[-\lambda(t-3400)]=$$

$$\mathbf{=2\exp(-\lambda T_0)- \exp[-\lambda(t-3400)][2\exp(-\lambda T_0)+\lambda(t-3400)]}$$

The value of the third expression is:

$$\int_{t-3400}^{t-3200} f(z)\, \mathbf{V_0(0,t-z-200)dz}=\int_{t-3400}^{t-3200} f(z)\, \exp(-\lambda T_0)dz=$$

$$\mathbf{=\exp(-\lambda T_0)\{\exp[-\lambda(t-3400)]-\exp[-\lambda(t-3200)]\}}$$

Therefore:

$$\mathbf{V_2(1,t)=\max}\begin{cases} \mathbf{\exp(-\lambda T_0)+\exp(-\lambda T_0)\{1-\exp[-\lambda(t-3400)]\}} & \mathbf{(FIX)} \\ & \mathbf{3600>t\geq 3400} \\ \mathbf{\exp[-\lambda(t-250)]+ 2\exp(-\lambda T_0)- \exp[-\lambda(t-3400)][2\exp(-\lambda T_0)]+} & \\ \mathbf{+\lambda(t-3400)]+ \exp(-\lambda T_0)\{\exp[-\lambda(t-3400)-\exp[-\lambda(t-3200)]\}} & \mathbf{(GO\text{-}ON)} \end{cases}$$

For t=3,400 the value of the FIX option is 0.368, and the value of the GO-ON option is 0.374.

For t=3,600 the value of the FIX option is 0.391, and the value of the GO-ON option is 0.335.

Therefore **there is** a break-even point in this region, which can be found by equating the two options and solving for t:

$$\exp(-\lambda T_0)+\exp(-\lambda T_0)\{1-\exp[-\lambda(t-3400)]\}=\exp[-\lambda(t-250)]+2\exp(-\lambda T_0)-$$
$$\exp[-\lambda(t-3400)][2\exp(-\lambda T_0)+\lambda(t-3400)]+\exp(-\lambda T_0)\{\exp[-\lambda(t-3400)]-\exp[-\lambda(t-3200)]\}$$

After some algebra:

t=3400+3000{[exp(250/3000)-exp(200/3000)]/[exp(3400/3000)]}

The solution is at **t=3417.35 hr.**

This is the break-even point for $V_2(1,t)$, and there is no need to continue the recursive calculations any further. This break-even point occurs before t=3,200 hours, which is the "forced" transition point from FIX to GO-ON (T_0+T_{corr}). The optimal testing policy is graphically described in figure 5.

Clearly, $T_{Fmax}*$, the break-even point for the last allowable failure, can exist either at the **forced** or transition point, i.e. $t=T_0+T_{corr}$, or at a larger value of t. In the latter case, it is called a **real** break-even point.

8. A Lemma about the break-even point

Lemma: the existence of a real break-even point depends only on the specific values of T_{corr}, T_{max} and T_{Fmax}, and does not depend on the values of λ and F_{max}.

Proof:

According to eq. (17):

$$V_1(F_{max},t)=\max\begin{cases} V_0(0,t-T_{corr}) & \text{(FIX)} \\ p(0,t-T_{max}+T_{Fmax})+\int_0^{t-T0-Tcorr} f(z)\, V_0(0,t-z-T_{corr})dz & \text{(GO-ON)} \end{cases} \quad (20)$$

According to eq. (19):

$$V_0(0,t)=p(0,T_0)=\exp(-\lambda T_0) \quad (21)$$

Substituting eq. (21) in eq. (20) yields:

$$V_1(F_{max},t)=\max\begin{cases} \exp(-\lambda T_0) & \text{(FIX)} \\ \end{cases} \quad (22)$$

$$\mathbf{p(0,t-T_{max}+T_{Fmax})+exp(-\lambda T_0)\{1-exp[-\lambda(t-T_0-T_{corr})]\}} \quad \textbf{(GO-ON)}$$

In order to have a break-even point in this region:

$$\mathbf{exp(-\lambda T_0)= exp[-\lambda(t-T_{max}+T_{Fmax})]+exp(-\lambda T_0)\{1-exp[-\lambda(t-T_0-T_{corr})]\}} \qquad \textbf{(23)}$$

After some algebra:

$$\mathbf{T_{corr}=T_{max}-T_{Fmax}} \qquad \textbf{(24)}$$

Since equation (24) does not depend on t, its meaning is as follows:

- For $\mathbf{T_{corr}>T_{max}-T_{Fmax}}$ the FIX option is better throughout the V_1 region, including $t=T_0+T_{corr}$. Therefore, the break-even point T_{Fmax}* is at $\mathbf{t=T_0+T_{corr}}$, which is the **forced** transition point between FIX and GO-ON.
- For the singular case $\mathbf{T_{corr}=T_{max}-T_{Fmax}}$ the FIX and the GO-ON options are equal throughout the V_1 region, including $t=T_0+2T_{corr}$. In this case there is a **real** break-even point at $\mathbf{t=T_0+2T_{corr}}$**.**
- For $\mathbf{T_{corr}<T_{max}-T_{Fmax}}$ the GO-ON option is better throughout the V_1 region, including $t=T_0+2T_{corr}$. Therefore, the break-even point T_{Fmax}* is not located in this region. Since for $t=T_{max}$ FIX is the better option, in this case there has to be a **real** break-even point at some $\mathbf{t> T_0+2T_{corr}}$.

Hence the existence of a real break-even point depends only on the validity of the inequality $\mathbf{T_{corr}\leq T_{max}-T_{Fmax}}$, Q.E.D.

9. Conclusions

The optimal statistical testing policy of safety-critical systems, using the SRST method and the TAAF approach, is based on the notion of break-even points. For every failure found, the optimal policy is GO-ON testing if the time left is smaller than the break-even point value, and STOP testing if the time left is larger than this value. The values of the various break-even points can be found by stochastic dynamic programming.

References

[1] USA Department of Defence (1996), MIL-HDBK-781A: Reliability Test Methods, Plans and Environments.

[2] H. Sandoh, Reliability Demonstration Testing for Software, IEEE Transactions on Reliability 40 (1) (1991) 117-119.

[3] O. Tal, A. Bendell, C. McCollin, Reliability Demonstration for Safety-Critical Systems, Discussion Paper No. 99/4, The Management Centre, University of Leicester, 1999.

[4] O. Tal, Software Dependability Demonstration for Safety-Critical Military Avionics Systems by Statistical Testing, PhD Thesis, the Nottingham Trent University, 1999.

[5] T.A Thayer, M. Lipow, E.C. Nelson, Software Reliability, North Holland, Amsterdam, 1978.

[6] S.M. Ross, An Introduction to Stochastic Dynamic Programming, Academic Press, New York, 1982.

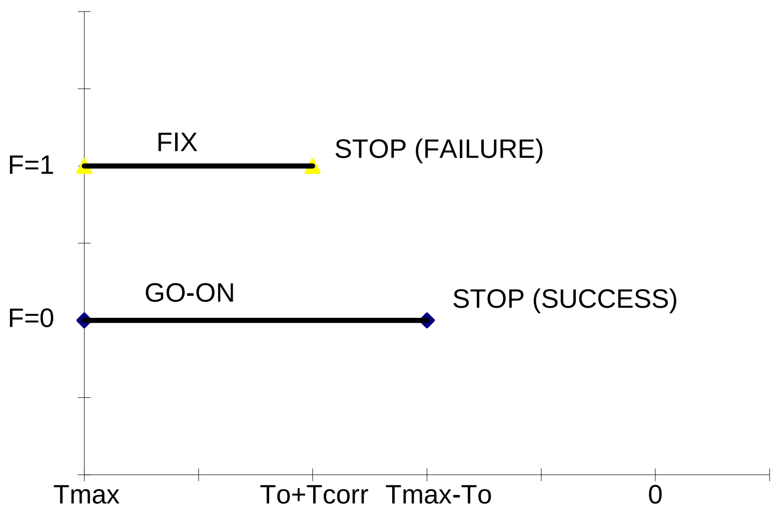

Figure 1: P(F,t) when F_{max}=0

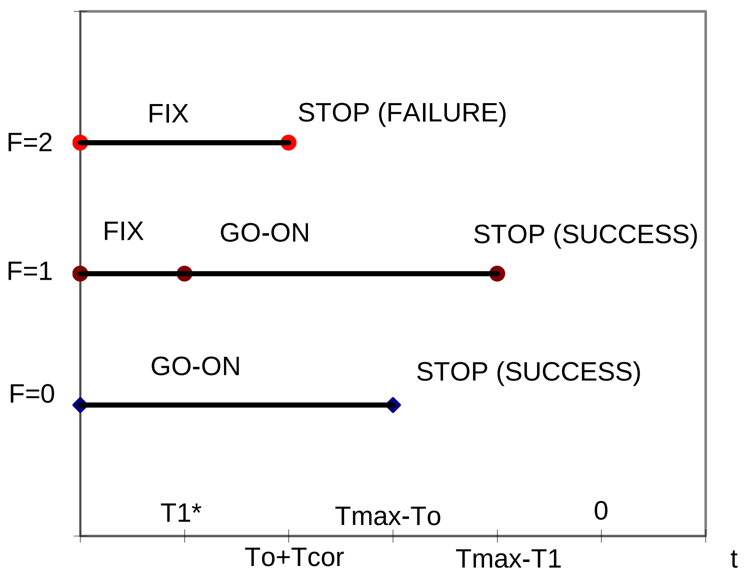

Figure 2: P(F,t) when F_{max}=1 and T_1* exists

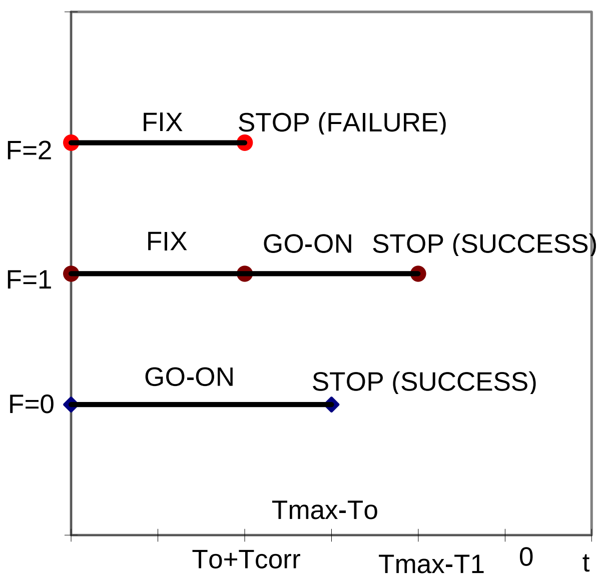

Figure 3: P(F,t) when F_{max}=1 and T_1* does not exist

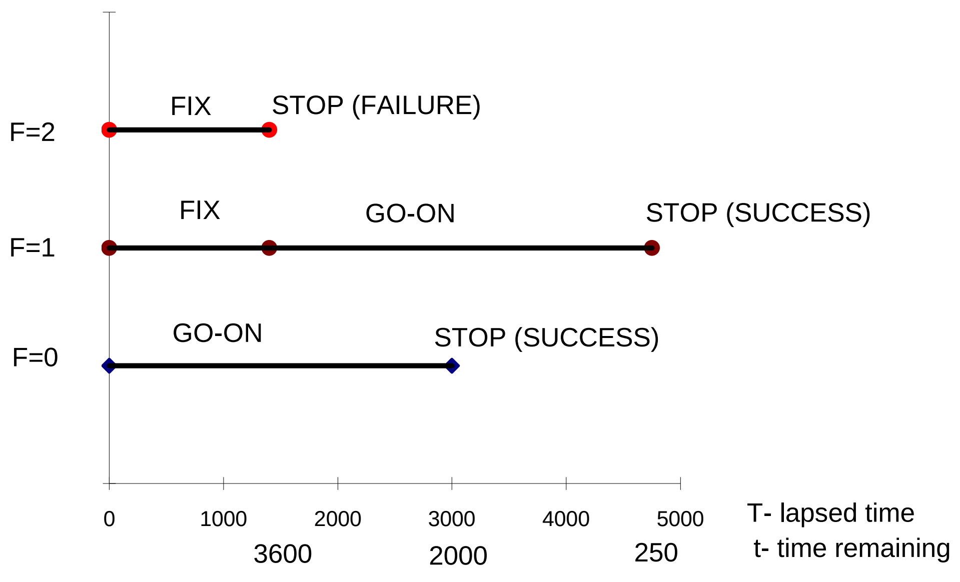

Figure 4: The optimal testing policy for the first numerical example

Note: the break-even point is at t=T_0+T_{corr}=3,600 hours, which is the forced transition point.

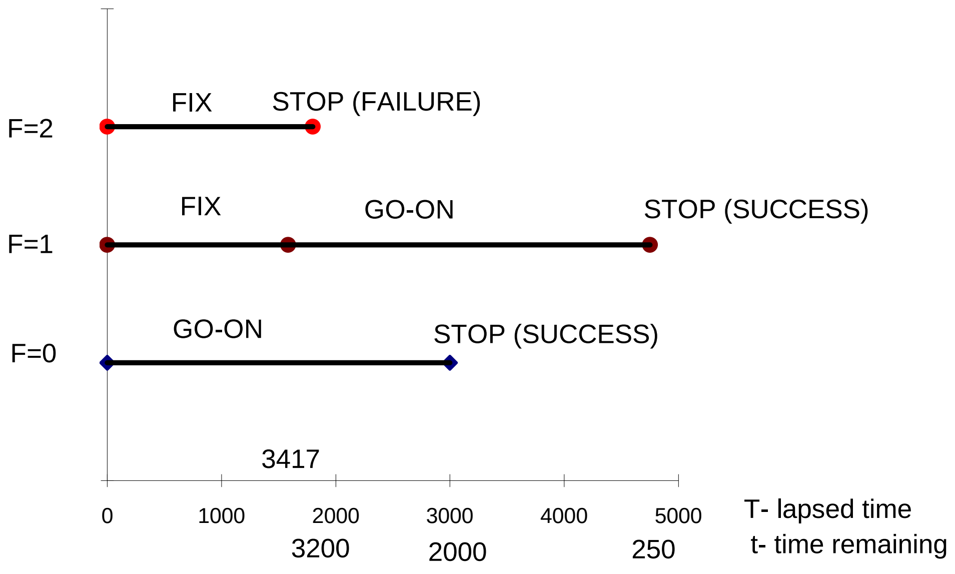

Figure 5: The optimal testing policy for the second numerical example

Note: the break-even point is at t=3,417 hr. which is **larger** than the forced transition point, 3,200 hours (T_0+T_{corr}).