Optimal Testing Strategies

Harald A. Stieber

University of Applied Sciences
Nuremberg, Germany
harald.stieber@fh-nuernberg.de

1) Introduction

Software systems are composed of components (subsystems, ...). Some of the components may be reused or developed by another company, some of them may be new. The reliability of the components may be quite different. Testing is an expensive and important part of the software life cycle. So the question is: how much time of testing should be spent for the different components to get the highest reliability in a given time or to get a given reliability in the shortest time (that is called optimal testing strategy).

According to the standard theory, the proportion of testing time of the different functionalities (components) has to be the same as given by the usage profile. That's the so-called 'usage testing' ([3],[4])

It can be shown, that even if we assume that all the functionalities (components) have the same reliability (failure intensity) at the beginning of the test, usage testing may not be the 'optimal testing strategy'. The 'optimal testing strategy' depends on the number of components and the reliability improvement during test.

2. The usage profile and the testing profile

A profile is a set of disjoint (only one can occur at a time) alternatives called elements, each with the probability that it will occur.

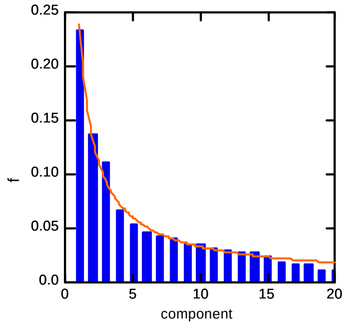
So a usage profile (operational profile) is a quantitative characterization of how a system will be used (see [3] and [4]). Let

$U := \{(u_1, \dots ,u_n) \in \mathbf{R}^n / u_1 + \dots + u_n = 1\} \subset \mathbf{R}^n_+$.

So if u_k is the relative frequency of usage of component k a usage profile is described by a vector

$\mathbf{u} = (u_1, \dots ,u_n) \in U \subset \mathbf{R}^n_+$. U is the space of profiles.

The following figure shows a typical usage profile. Here the commands (or components) are sorted according to their frequency of use: $u_1 \geq \dots \geq u_n$.



In a similar way a testing profile describes the proportion of time which is spent for the test of component n. So it is given by a vector $\mathbf{t} = (t_1, \dots ,t_n) \in U \subset \mathbf{R}^n_+$ where t_k is the relative frequency for testing of component k. It may look similar to figure 1. An important question is to find an optimal testing profile (see chapter 3)

3. Optimal testing strategies

Let's assume a software system C is composed of (components, subsystems, ...) $C_1, \dots, C_n$.

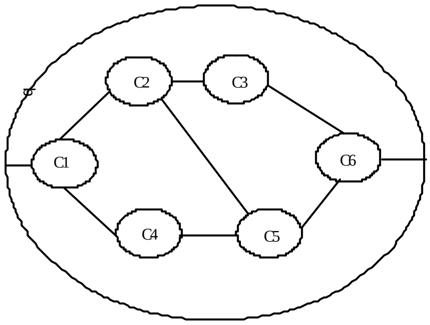


figure 2

Given a usage profile $\mathbf{u} = (u_1, \dots ,u_n)$ the mapping λ: $\mathbf{R}^n_+ \to R$ with

$$\lambda(t_1, \dots ,t_n) = \lambda_1(t_1) * u_1 + \dots + \lambda_n(t_n) * u_n$$

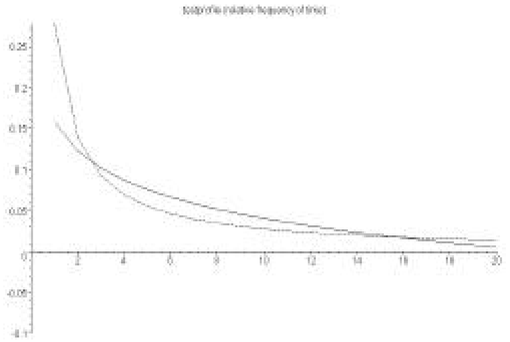
gives the failure intensity of the system under **u** if $\lambda_k(t)$ is the failure intensity function of component

C_k . Let λ_e be the desired failure intensity at the end of testing. Now a minimum of $t_e := t_1 + ... + t_n$ (total testing time) has to be found with the constraint $\lambda (t_1, ... ,t_n) = \lambda_e$. That can be a numerical problem if n is large (let's say 100 or more). This optimization was done with MAPLE or MATLAB. The resulting $(t_1 , ... , t_n)$ divided by t_e (to get sum =1) is the optimal testing profile. In the same way we can look for a minimum of $\lambda(t_1, ... ,t_n)$ with the constraint $t_e := t_1 + ... + t_n$.

The following pictures show a typical example. Testing according to the optimal test profile can be much better than usage testing. The factor of improvement depends on the reliability improvement, the usage profile, the number of components and the reliability of the individual components at the start of the test.

The higher the wanted reliability improvement during test, the more the optimal testing profile approaches a uniform testing profile. The higher the number of components, the higher the improvement by testing according to the 'optimal testing profile'.

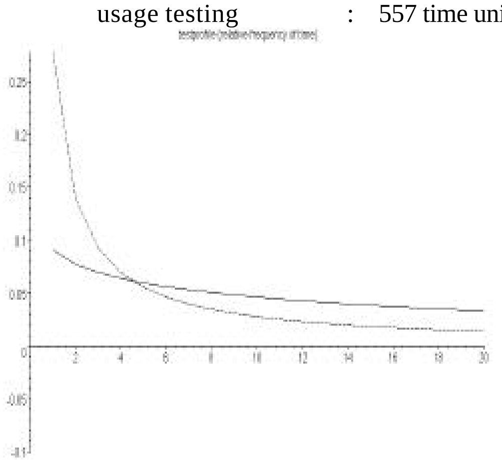
The pictures show the usage profile (dashed line) and the optimal testing profile (solid line) of a system with 20 components. In this example it was assumed that each of the components had the same failure intensity λ_0 at the beginning of the test phase. It was assumed that reliability growth of each component follows the logarithmic poisson model (with the same λ_0 and θ).



$\lambda = 1/4 * \lambda_0$

testing time:

optimal testing strategy:		492 time units
usage testing	:	557 time units



$\lambda = 1/20 * \lambda_0$

testing time:

optimal testing strategy:	1297 time units
usage testing	2066 time units

So after 1297 time units the failure intensity was reduced by a factor of 20 with testing according to the optimal testing profile whereas 2066 time units were necessary to get the same with usage testing.

References

[1] Jack, O, Stieber, H. : "The Optimal Testing Profile", to appear

[2] Musa, J. D.; A. Iannino, K. Okumoto: "Software Reliability: Measurement, Prediction, Application ", McGraw-Hill, New York, 1987

[3] Musa, J. D., G. Fuoco, N. Irving, D. Kropfl, B. Juhlin: The Operational Profil, in Lyu, M. (ed.) : "Handbook of Software Reliability", McGraw-Hill, New York, 1996, pp. 167-216

[4] Musa, J. D. : "Software Reliability Engineering", McGraw-Hill, New York, 1998

[5] Stieber, H. :Usage Profiles, Optimal Testing Strategies and Reuse, Proceedings of the 26 th Euromicro Conference, Maastricht, the Netherlands 2000