Testing Coverage Models

7.1 Introduction

Software reliability models based on an NHPP have been used to estimate and predict the quality of software products such as reliability, number of remaining errors, and failure intensity. Imperfect debugging, learning phenomenon of software developers, and other realistic issues have been studied during the last three decades (see Chapter 6). However, software development is a very complex process and there are important issues that have not been addressed. Testing coverage is one of these issues. Testing coverage is important information for both software developers and customers of software products. This chapter discusses various software reliability models incorporating testing coverage and fault removal.

7.2 Testing Coverage Models

Among all SRGMs, a large family of stochastic reliability models based on a nonhomogeneous Poisson process, which are known as NHPP reliability models, has been widely used to track reliability improvement during software testing. These models enable software developers to evaluate software reliability in a quantitative manner. They have also been successfully used to provide guidance in making decisions such as when to terminate testing the software or how to allocate available resources. However, software development is a very complex process and there are still issues that have not yet been addressed. Testing coverage is one of these issues. Testing coverage information is an important measure for both software developers and customers of software products.

Testing coverage (Pham 2003d) is a measure that enables software developers to evaluate the quality of the tested software and determine how much additional effort is needed to improve the reliability of the software. Testing coverage, on the other hand, can provide customers with a quantitative confidence criterion when they plan to buy or use the software products. To our knowledge, testing coverage
has not been addressed in the existing software reliability models. Testing coverage is an important measure for both software developers and users.

In this section, we discuss models incorporating testing coverage in the software development process and relate it to the error detection rate function. We examine the goodness-of-fit of the testing coverage model and other existing NHPP models based on several sets of software testing data.

**Notation**

- \( a(t) \): Total errors content at time \( t \)
- \( b(t) \): Error detection rate at time \( t \)
- \( c(t) \): Testing coverage as a function of time \( t \)
- \( \lambda(t) \): Intensity function or fault detection rate per unit time
- \( m(t) \): Mean value function or the expected number of errors detected by time \( t \)
- \( R(x/t) \): Reliability function of software by time \( t \) for a mission time \( x \)
- \( N(t) \): Counting process representing the cumulative number of failures at time \( t \)
- \( \sum_k \): Sum over \( k \) from 1 to \( n \)
- \( SSE \): Sum of squared errors of a model fitting the actual data
- \( AIC \): Akaike’s Information Criterion
- \( PRR \): Predictive-risk ratio

**A Generalized Testing Coverage Model**

Pham and Zhang (2003d) introduce a generalized model which incorporates testing coverage measure into software reliability assessment. Let \( c(t) \) denote the percentage of the code coverage as a time dependent function which has been examined during software testing. Obviously, \( 1-c(t) \) is the percentage of the software code which has not yet been covered by test cases by time \( t \). The derivative of the testing coverage function, \( c'(t) \), represents the coverage rate. Therefore, the error detection rate function can be expressed as \( \frac{c'(t)}{1-c(t)} \).

**Theorem 7.1 (Pham and Zhang 2003d):** The generalized NHPP model incorporating testing coverage can be formulated as follows:

\[
\frac{dm(t)}{dt} = \frac{c'(t)}{1-c(t)} [(a(t) - m(t))]
\]

(7.1)

where \( a(t) \) is the total fault content function. The explicit solution of the mean value function is given by:

\[
m(t) = e^{-B(t)} \left( m_0 + \int_{t_0}^t a(\tau) e^{B(\tau)} \frac{c'(\tau)}{1-c(\tau)} d\tau \right)
\]

(7.2)

where \( B(t) = \int_{t_0}^t \frac{c'(\tau)}{1-c(\tau)} d\tau \) and \( m(t_0) = m_0 \) is the marginal condition of equation (7.2) with \( t_0 \) representing the starting time of the debugging process.

This model indicates that the failure intensity depends on both the rate at which the remaining faults are covered and the number of remaining faults at current time \( t \) divided by the current fractional population of uncovered faults.