A General NHPP Software Reliability Growth Model with Fault Removal Efficiency

Hong-Wei Liu, Xiao-Zong Yang, Feng Qu, and Yan-Jun Shu

Abstract—This paper presents a general software reliability growth model (SRGM) based on nonhomogeneous Poisson process (NHPP). Although many research have been devoted to unify some NHPP models, most of them either only consider imperfect debugging, learning phenomenon, or take fault removal efficiency as a constant. Consideration of the variation of fault removal efficiency during debugging period in the exiting models is limited. The general model in this paper is the first unified scheme of some NHPP models which take fault removal efficiency as a function of debugging time. Fault detection rate (FDR) is usually used to measure the effectiveness of fault detection of test techniques and test cases. Most researchers assume a constant FDR in deriving their SRGMs. Because of learning process of testers, some researchers take FDR as increasing functions over testing period. Some literature take FDR as decreasing functions because failures removed first have higher detected rate. A bell-shaped FDR function is proposed which integrates both learning phenomenon and inherent FDR. As a special case of the general SRGM, a NHPP SRGM called PBbell-SRGM is put forward which integrates the proposed FDR function and fault removal efficiency. PBbell-SRGM is evaluated using a set of software failure data. The results show that PBbell-SRGM fits the given failure data better than some selected NHPP models.

Index Terms—Imperfect debugging, learning phenomenon, nonhomogeneous Poisson process, software reliability growth model.

NOMENCLATURE

- $a(t)$: Time-dependent fault content function: total number of faults in the software including the initial and the introduced faults
- $b(t)$: Time-dependent fault detection rate function per fault per unit time
- $x(t)$: Expected number of faults removed by time $t$
- $p(t)$: Fault removal efficiency at time $t$
- $q(t)$: Fault introduction probability at time $t$
- $a_0$: Mean number of initial software faults
- $p_1(t)$: Fault detection rate at time $t$ if no learning process during software debugging
- $p_2(t)$: Fault detection ability factor at time $t$ caused by learning process

Abstract—This paper presents a general software reliability growth model (SRGM) based on nonhomogeneous Poisson process (NHPP). Although many research have been devoted to unify some NHPP models, most of them either only consider imperfect debugging, learning phenomenon, or take fault removal efficiency as a constant. Consideration of the variation of fault removal efficiency during debugging period in the exiting models is limited. The general model in this paper is the first unified scheme of some NHPP models which take fault removal efficiency as a function of debugging time. Fault detection rate (FDR) is usually used to measure the effectiveness of fault detection of test techniques and test cases. Most researchers assume a constant FDR in deriving their SRGMs. Because of learning process of testers, some researchers take FDR as increasing functions over testing period. Some literature take FDR as decreasing functions because failures removed first have higher detected rate. A bell-shaped FDR function is proposed which integrates both learning phenomenon and inherent FDR. As a special case of the general SRGM, a NHPP SRGM called PBbell-SRGM is put forward which integrates the proposed FDR function and fault removal efficiency. PBbell-SRGM is evaluated using a set of software failure data. The results show that PBbell-SRGM fits the given failure data better than some selected NHPP models.

Index Terms—Imperfect debugging, learning phenomenon, nonhomogeneous Poisson process, software reliability growth model.

NOMENCLATURE

- $a(t)$: Time-dependent fault content function: total number of faults in the software including the initial and the introduced faults
- $b(t)$: Time-dependent fault detection rate function per fault per unit time
- $x(t)$: Expected number of faults removed by time $t$
- $p(t)$: Fault removal efficiency at time $t$
- $q(t)$: Fault introduction probability at time $t$
- $a_0$: Mean number of initial software faults
- $p_1(t)$: Fault detection rate at time $t$ if no learning process during software debugging
- $p_2(t)$: Fault detection ability factor at time $t$ caused by learning process

Softw. fault debugging is a very complex process, and new faults can be introduced as part of the faults repair process. Imperfect debugging has been considered by some NHPP models [4], [10]-[13]. Fault removal efficiency is defined as the probability of perfectly removing a fault on the first repair attempt [14]. In practice, it is not unusual for the software development team to find that a software fault has been reported multiple times before it is finally removed [10]. FDR is the rate of discovering new faults in software during the testing phase. Typically, FDR depends on the ability of programmers/debuggers, the software structure, the maturity of software development procedure, and the correlation among program modules [6]. Therefore, fault introduction, fault removal efficiency, and FDR are important factors for software reliability estimation and software project management. Because of the complexity of software fault debugging, all these three factors ought to take time-varying forms.

The remainder of the paper is organized as follows: Section II describes several general NHPP SRGMs. In Section III, a general NHPP SRGM, which incorporates fault introduction function, fault removal efficiency function, and FDR, is proposed. Learning phenomenon and software inherent FDR are discussed, and a NHPP model with bell-shaped FDR function is proposed in Section IV. Section V discusses the goodness-of-fit and predictability measures of SRGMs used in this paper. Section VI evaluates the NHPP SRGM using a set of software failure.
data. Finally, the conclusions are given in Section VII.

II. NHPP SRGMs

NHPP SRGMs are time-domain models that assume software failures follow the behaviors of NHPPs. Let \( N(t) \) denotes the cumulative number of faults detected by time \( t \), and \( m(t) \) denotes its mean value function. Then \( m(t) = E[N(t)] \). The parameter of the stochastic process, \( \lambda(t) \), which denotes the failure intensity of the software at time \( t \), is time-dependent. \( m(t) \) and \( \lambda(t) \) are related as follows

\[
m(t) = \int_0^t \lambda(t) \, dt 
\]

\( N(t) \) is known to have a Poisson pmf with parameter \( m(t) \), that is

\[
P[N(t) = n] = \frac{m(t)^n}{n!} e^{-m(t)}. \tag{2}
\]

Software reliability \( R(x \mid t) \) is defined as the probability that no software failure is observed in the time interval \((t, t+x)\), given that the last failure occurred at a time point \( t (t \geq 0, x > 0) \). Based on NHPP, \( R(x \mid t) \) can be expressed as follows [15]

\[
R(x \mid t) = \exp\left[-\left(m(t+x) - m(t)\right)\right]. \tag{3}
\]

Many existing NHPP models assume that the failure rate at time \( t \) is proportional to the number of faults remaining in the software system [14]. Considering fault removal efficiency, a general NHPP SRGM can be obtained by solving the system of differential equations as follows [10]

\[
\frac{dm(t)}{dt} = b(t) \cdot [a(t) - p \cdot m(t)] \tag{4}
\]

\[
\frac{da(t)}{dt} = q(t) \cdot \frac{dm(t)}{dt}. \tag{5}
\]

The marginal conditions for (4) and (5) are \( m(0) = 0 \) and \( a(0) = a_0 \). The explicit expression of \( m(t) \) can be obtained as follows

\[
m(t) = a_0 \int_0^t b(u) \exp\left[-\int_u^t (p - q(v)) b(v) \, dv\right] \, du. \tag{6}
\]

The above general NHPP model integrates fault removal efficiency, as a constant, into its mean value function. To make the unified scheme of some NHPP models more common and applicable, fault removal efficiency ought to take a time-varying form.

III. A General NHPP Model

Fault removal efficiency function, FDR function, and fault introduction rate function are integrated into \( m(t) \) of an NHPP model. Only when faults are detected can fault removal and fault introduction process happen. The number of faults removed immediately at time \( t \) is proportional to the number of faults detected at that time. On the other hand, whether the fault is successfully removed or not, some new faults may be introduced into the software system with probability \( q(t) \). Analytically, the mean value function of \( m(t) \) can be obtained by solving the system of differential equations as follows

\[
\frac{dm(t)}{dt} = b(t) [a(t) - x(t)] \tag{7}
\]

\[
\frac{dx(t)}{dt} = p(t) \frac{dm(t)}{dt}.
\]

The marginal conditions for (7) are as follows

\[
m(0) = 0, \quad a(0) = a_0, \quad x(0) = 0. \tag{8}
\]

From (7) and (8), we can easily obtain

\[
x(t) = \int_0^t p(t) \frac{dm(t)}{dt} \, dt \tag{9}
\]

\[
a(t) = a_0 + \int_0^t q(t) \frac{dm(t)}{dt} \, dt \tag{10}
\]

Substituting (9) and (10) into (7), we can obtain

\[
\frac{dm(t)}{dt} = b(t) \left[a(t) + \left(\int_0^t q(t) - p(t) \frac{dm(t)}{dt} \right) \, dt\right]. \tag{11}
\]

Let

\[
L(t) = \int_0^t q(t) - p(t) \frac{dm(t)}{dt} \, dt. \tag{12}
\]

From (11) and (12), we can obtain

\[
\frac{dm(t)}{dt} = [q(t) - p(t)] b(t) [a(t) + L(t)]. \tag{13}
\]

The explicit expression of \( L(t) \) can be obtained as follows

\[
L(t) = a_0 \left\{\exp\left[\int_0^t (q(t) - p(t)) b(t) \, dt\right] - 1\right\}. \tag{14}
\]

From (11) and (14), the failure rate function can be expressed as follows

\[
\lambda(t) = \frac{dm(t)}{dt} = a_0 b(t) \exp\left[\int_0^t (q(v) - p(v)) b(v) \, dv\right] \cdot (\lambda(t) - \lambda(t)). \tag{15}
\]

Therefore, the explicit expression of the mean value function \( m(t) \) can be obtained as follows

\[
m(t) = a_0 \int_0^t b(t) \exp\left[\int_0^t (q(v) - p(v)) b(v) \, dv\right] \, dt. \tag{16}
\]

According to (15), the expected number of faults removed by time \( t \) given by (9) is

\[
x(t) = a_0 \int_0^t p(t) b(t) \exp\left[\int_0^t (q(v) - p(v)) b(v) \, dv\right] \, dt. \tag{17}
\]

The fault content function given by (10) is

\[
a(t) = a_0 \left[1 + \int_0^t q(t) b(t) \times \exp\left[\int_0^t (q(v) - p(v)) b(v) \, dv\right] \, dt\right]. \tag{18}
\]

The proposed general NHPP model considers fault removal efficiency as a time-varying form. When \( p(t) = 1 \), the proposed model can be transformed into another NHPP model [13]. In the simplest case, \( q(t) = 0 \), \( p(t) = 1 \), and \( b(t) = b \), the general model simplify to Goel-Okumoto.
model [15]. Notice that when \( p(t) = p \), the proposed model can be reduced to an existing general NHPP model [10].

IV. A NHPP MODEL WITH BELL-SHAPED FAULT DETECTION RATE FUNCTION

In this section, we derive a new NHPP model from the general model that we proposed in the previous section. In this new model, we will focus on discussing fault detection rate function. To simplify the form of the proposed general model, the following assumptions are made:

1) The fault removal efficiency is a constant, \( p(t) = p \).
2) The fault introduction rate is a constant, too, \( q(t) = q \).

For an effective software debugging process, the number of residual faults will decrease.

In general, \( q \leq p \) is true. Substituting \( p(t) = p \), \( q(t) = q \) into (16), we can obtain

\[
m(t) = \frac{a_0}{p-q} \cdot (1-\exp(q-p) \cdot B(t)),
\]

\[
B(t) = \int_0^t b(\tau) \cdot d\tau .
\]

The FDR is used to measure the effectiveness of fault detection of test techniques and test cases. Many models assume a constant FDR per fault [1], [15], [16]. That is, they assume that all faults have equal probability of being detected during the software debugging process. Some models [17] assume faults have different FDR and those with highest detected rates are removed first, for example, the occurrence of all exceptional situations may rarely be detected by the test cases. The FDR functions in these models are assumed to be decreasing functions. On the other hand, the learning phenomenon of the software testers has been studied [4], [10]. The learning is closely related to the changes in the efficiency of testing during a testing phase. These models assume that the FDRs are increasing over testing period. The FDR can be described as follows

\[
b(t) = p_0(t) \cdot p_1(t) .
\]

If the faults remaining in software are not detected for a specified period under specified conditions, the inherent FDR at this moment can be considered to trend to zero. The initial inherent FDR is \( b, \ 0 < b < 1 \). The following equation is used to capture the trend of the inherent FDR:

\[
p_0(t) = b \cdot \exp(-\beta_1 \cdot t)
\]

\[
p_1(t) = \frac{1 + \beta}{1 + \beta \cdot \exp(-b \cdot (1 + \beta) \cdot t)} .
\]

The initial learning factor is 1 and the maximum learning factor is \( 1 + \beta \).

Substituting (22) and (23) into (21), we obtain

\[
b(t) = \frac{b \cdot (1 + \beta) \cdot \exp(-\beta_1 \cdot t)}{1 + \beta \cdot \exp(-b \cdot (1 + \beta) \cdot t)} .
\]  

To simplify the form of (24), let \( \beta_0 \equiv b \cdot (1 + \beta) \). Then,

\[
b(t) = \frac{\beta_0 \cdot \exp(-\beta_1 \cdot t)}{1 + \beta \cdot \exp(-\beta_0 \cdot t)} .
\]

For the above FDR function:

1) The fault introduction rate is a constant, \( \beta_0 > \beta_1 \) and \( \beta \cdot (\beta_0 - \beta_1) - \beta_1 \leq 0 \) for \( t \in (0, \infty) \), \( b(t) \) is a decreasing function of \( t \). Hence, \( t_0 = 0 \) maximizes \( b(t) \), and \( b(0) = b \).

2) The fault introduction rate is a constant, \( \beta_0 > \beta_1 \) and \( \beta (\beta_0 - \beta_1) - \beta_1 > 0 \) then

\[
t_0 = \frac{1}{\beta_0} \ln \frac{\beta_1}{\beta_1 + k \cdot \beta_0}
\]

maximizes \( b(t) \), and the maximum FDR is

\[
b(t_0) = \left( \frac{\beta_0 - \beta_1}{\beta_1 + k \cdot \beta_0} \right)^{\beta_1} .
\]

In this case, the FDR is an increasing function for \( t \in (0, t_0) \), and a decreasing function for \( t \in (t_0, \infty) \). In fact, Kuo et al. pointed out that the FDR functions of some software failure data sets are increasing initially and then decreasing [6]. The combination of learning and inherent FDR can well explain that phenomenon.

By taking integrals on both sides of (27), we can obtain

\[
B(t) = \int_0^t \frac{\beta_0 \cdot \exp(-\beta_1 \cdot \tau)}{1 + \beta \cdot \exp(-\beta_0 \cdot \tau)} \cdot d\tau
\]

\[
= \sum_{k=0}^{\infty} \frac{\beta_0 \cdot (-\beta)^k \cdot (1 - \exp(-\beta_1 + k \cdot \beta_0) \cdot t))}{\beta_1 + k \cdot \beta_0} .
\]

Substituting (28) into (19), we can obtain

\[
m(t) = a_0 \cdot p - q \cdot (1 - \exp((q-p) \cdot x(t)),
\]

\[
\left[ \sum_{k=0}^{\infty} \frac{\beta_0 \cdot (-\beta)^k \cdot (1 - \exp(-\beta_1 + k \cdot \beta_0) \cdot t))}{\beta_1 + k \cdot \beta_0} \right].
\]

The expected number of faults removed by \( t \) is given by

\[
x(t) = \frac{a_0 \cdot p}{p-q} \cdot (1 - \exp((q-p) \cdot x(t)),
\]

\[
\left[ \sum_{k=0}^{\infty} \frac{\beta_0 \cdot (-\beta)^k \cdot (1 - \exp(-\beta_1 + k \cdot \beta_0) \cdot t))}{\beta_1 + k \cdot \beta_0} \right].
\]

The total number of faults in the software including the initial and the introduced faults by time \( t \) is given by

\[
a(t) = \frac{a_0}{p-q} \cdot (p-q \cdot \exp((q-p) \cdot x(t)),
\]

\[
\left[ \sum_{k=0}^{\infty} \frac{\beta_0 \cdot (-\beta)^k \cdot (1 - \exp(-\beta_1 + k \cdot \beta_0) \cdot t))}{\beta_1 + k \cdot \beta_0} \right].
\]
\[
\lambda(t) = a_0 \cdot \frac{\beta_0 \cdot \exp(-\beta_1 \cdot t)}{1 + \beta_0 \cdot \exp(-\beta_1 \cdot t)} \cdot \exp((q - p) \cdot \sum_{k=0}^{n} \beta_k \cdot (-\beta)^k \cdot (1 - \exp(-\beta_1 + k \cdot \beta_0) \cdot t))
\]

(32)

The fault detection rate function in this model, \( b(t) \), is a bell-shaped function, which captures the learning process of the software testers and the inherent FDR. The model also addresses the fault introduction rate and the fault removal efficiency. We call the model PBbell-SRGM.

V. GOODNESS-OF-FIT AND PREDICTABILITY MEASURES

Four criteria are used for model comparison. They are sums of squared error (SSE), R-square, average error (AE), and u-plot. Both the descriptive and predictive power of the models are considered. The SSE shows us the goodness-of-fit of the model. The R-square can take on any value between 0 and 1, with a value closer to 1 indicating a better goodness-of-fit. The R-square is the square of the correlation between the response values and the data that is explained by a regression. The average error (AE) is a measure of how well a model predicts the response values. The AE is obtained from the fitted mean value function. The lower the AE and the Kolmogorov-Smirnov distance, the better the model performs.

The R-squared value is the fraction of the variance in the data that is explained by a regression, and is the square of the correlation between the response values and the predicted response values. The R-square can take on any value between 0 and 1, with a value closer to 1 indicating a better fit, and it can be formulated as

\[
R - square = \frac{\sum_{i=1}^{n} (\hat{m}(t_i) - \bar{y})^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2},
\]

(34)

\[
\bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i.
\]

(35)

The AE is a measure of how well a model predicts throughout the test phase [18], [19]. The most common approach of software reliability evaluation is to use a grouped data. The testing duration is divided into a number of periods, for example, \( t_0 < t_1 < t_2 < \ldots < t_n \). For each period, one item of the data set \( \{t_i, y_i\} \) is obtained where \( y_i \) denotes the cumulative number of fault detected by time \( t_i \). Then the AE can be given by

\[
AE = \frac{1}{m} \sum_{i=1}^{m} \left| \frac{D_i - \hat{D}_i}{D_i} \right|
\]

(36)

where \( t_1, t_n \) are the lower and upper bound of test time used to evaluate the predictive power of a model; \( D \) is the cumulative number of faults detected by the end of test period; \( \hat{D}_i \) is the estimated cumulative number of faults detected by the end of test period by using a model with data \( \{t_0, y_0\}, \{t_1, y_1\}, \ldots, \{t_n, y_n\} \), and \( m \) is the total number of periods between \( t_1 \) and \( t_n \).

Another criterion used for model comparison is u-plot and Kolmogorov-Smirnov test (KS-test) [1]. The purpose of the u-plot is to determine whether the predictions, \( \hat{F}(t) \), are on average close to the true distributions, \( F(t) \). If the u-plot is a straight line, then the model is good. The Kolmogorov-Smirnov distance between u-plot and the line of unit slope describes the significance of the deviation between the sequence of predictions, \( \hat{F}(t) \), and the true \( F(t) \). The lower the AE and the Kolmogorov-Smirnov distance, the better the model performs.

VI. MODEL EVALUATION AND COMPARISON

In this section, the goodness-of-fit and the predictive power of PBbell-SRGM model are examined. The applicability of PBbell-SRGM is compared with some exiting NHPP models including Goel-Okumoto model, Logarithmic Poisson model, Goel generalized NHPP model, Comperetz growth curve model, Logistic growth curve model, and Yamada Delayed S-shaped model. The failure data used here are collected from the test of System T at AT&T [20]. 22 faults hided in System T were detected during 680.02 CPU time. This paper estimates the parameters of NHPP SRGMs by the least squares estimate (LSE) method. The estimates of the parameters and their implications of PBbell-SRGM model on the given failure data set are summarized as follows: the difference between fault removal efficiency and fault introduction probability \( p - q = 0.2271 \), which means that the software debugging process is not effective. The number of initial faults \( a_0 \) is estimated to be 5, and then the expected number of total detected faults is 22. Therefore the software has been tested enough at the end of test phase. The initial inherent FDR \( b \) is 0.01046, \( \beta = 1.088 \), \( \beta_0 = 0.02185 \), and \( \beta_1 = 0.00076 \). The failure data of System T and the fitted curves of the proposed model and some other famous NHPP SRGMs are shown in Fig. 1. Fig. 2 shows the cumulative faults, fitted faults, and the 95% global confidence bound of the proposed model.
Fig. 2. The actual cumulative number of detected faults, the fitted curve of PBbell-SRGM, and fitted curve's 95% global confidence interval are shown together.

### TABLE I

<table>
<thead>
<tr>
<th>Model Name</th>
<th>SSE</th>
<th>R-square</th>
<th>AE</th>
<th>Kolmogorov-Smirnov distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>PBbell-SRGMs</td>
<td>20.27</td>
<td>0.9800</td>
<td>0.1178</td>
<td>0.0995</td>
</tr>
<tr>
<td>Goel-Okumoto</td>
<td>27.37</td>
<td>0.9730</td>
<td>0.1489</td>
<td>0.1037</td>
</tr>
<tr>
<td>Logarithmic Poisson model</td>
<td>34.01</td>
<td>0.9616</td>
<td>0.1865</td>
<td>0.1434</td>
</tr>
<tr>
<td>Goel generalized NHPP model</td>
<td>26.67</td>
<td>0.9718</td>
<td>0.1771</td>
<td>0.1315</td>
</tr>
<tr>
<td>Compertz growth curve model</td>
<td>31.97</td>
<td>0.9684</td>
<td>0.1354</td>
<td>0.1430</td>
</tr>
<tr>
<td>Logistic growth curve mode</td>
<td>57.08</td>
<td>0.9436</td>
<td>0.1743</td>
<td>0.1992</td>
</tr>
<tr>
<td>Yamada Delayed S-shaped model</td>
<td>44.62</td>
<td>0.9559</td>
<td>0.1269</td>
<td>0.1779</td>
</tr>
</tbody>
</table>

Average error (AE) and u-plot are used as criteria to evaluate the predictive power of the proposed model. To calculate AE, the lower and upper bound of test time are restricted between the time points when the 15th and 21st faults are detected. Fig. 3 and Table I show that the u-plot of the proposed model is close to the line of unit slope; the proposed model has a smallest Kolmogorov-Smirnov distance, a smallest SSE, a smallest AE, and a biggest R-square among all the selected models.

A sensitivity analysis of the proposed model is conducted to study the effect of the parameters, so that attention can be paid to those parameters deemed most critical. We investigate the possible change of the expected number of detected faults by the end of the test phase when one of the parameters is modified. First, we define Relative Change (RC) as follows

\[
RC = \frac{NPF - NDF}{NDF}, \quad (37)
\]

where NDF is the number of detected faults by the end of test phase and NPF is the number of predictive faults if one of parameters is modified.

The results of the sensitivity analysis are shown in Table II. We can see that the variation of \(a_0\) and \(p - q\) have significant influence on the estimated number of faults. If \(a_0\) (\(p - q\)) is increased or decreased, the estimated number of faults will increase (decrease) or with almost the same proportion. The other three decrease (increase) parameters affect the estimated result slightly.

### VII. CONCLUSIONS

Fault content, FDR, and fault removal efficiency are three important factors of NHPP SRGMs. This paper presents a general NHPP SRGM which incorporates these three factors. To make the general model more universal, the fault removal efficiency function is proposed to be a time-varying form. Learning phenomenon and inherent FDR are studied and a bell-shape FDR function is proposed. A NHPP SRGM is proposed which incorporates...
imperfect debugging, learning phenomenon, and inherent FDR. The goodness-of-fit of the new model is examined using a software failure data set. Comparing with some selected MHPP SRGMs, the proposed model can provide a better goodness-of-fit and better predictability for this software failure data set.

ACKNOWLEDGMENT
The authors gratefully acknowledge stimulating discussions with Ph. D. candidate J. Zhao. She assisted in preparing some of the plots. All the plots were prepared using MATLAB®.

REFERENCES
[8]Xiang-Zong Yang was born in Liaoning Province, China, in 1939. He graduated in Computer Science and technology from Harbin Institute of Technology, China, in 1964. He is now a Professor and a Ph.D. supervisor of School of Computer Science and Technology of Harbin Institute of Technology. His research interests include dependable computing and ubiquitous computing.