Communications of the Korean Statistical Society
2010, Vol. 17, No. 6, 909–916

DOI: 10.5351/CKSS.2010.17.6.909

Selection of a Predictive Coverage Growth Function

Joong-Yang Park[a], Gyemin Lee[1,a]

[a]Department of Information and Statistics, RINS and RICI, Gyeongsang National University

Abstract

A trend in software reliability engineering is to take into account the coverage growth behavior during testing. A coverage growth function that represents the coverage growth behavior is an essential factor in software reliability models. When multiple competitive coverage growth functions are available, there is a need for a criterion to select the best coverage growth functions. This paper proposes a selection criterion based on the prediction error. The conditional coverage growth function is introduced for predicting future coverage growth. Then the sum of the squares of the prediction error is defined and used for selecting the best coverage growth function.

Keywords: Construct, coverage, coverage growth function, prediction error, software testing, software reliability.

1. Introduction

Software systems are becoming critical components of computer systems. Failures in a software system can cause severe consequences. Both software developers and users are concerned about the quality of software systems, especially reliability. The reliability of a software system is improved only by detecting and removing faults resident in the software system. Software systems are tested for fault detection before they are released. Software testing is a key activity to improve software reliability.

Theoretically, it is impossible to execute all the possible inputs of a software system under testing. Consequently, it is nearly impossible to detect and remove all the faults in a software system. Developed software should be released at an appropriate time and demands software testers perform testing activity in a reasonable amount of time. Software developers usually determine when to stop testing and release the software system based on the estimates of reliability measures.

Many software reliability growth models(SRGMs) have been proposed and applied in practice for the estimation of software reliability measures Musa *et al.* (1987), Lyu (1996) and Musa (1999). Most SRGMs describe a relationship between a reliability measure and the testing time. Such a relationship is obtained by modeling the fault detection and removal process during testing. However, it was recognized that the testing time was insufficient to express the fault detection and removal process. Attempts to integrate coverage information into SRGMs have been made by Gokhale *et al.* (1996), Malaiya *et al.* (2002), Pham and Zhang (2003), Park and Fujiwara (2006), Crespo *et al.* (2008, 2009) and Park *et al.* (2008b). Each coverage-based SRGM involves a coverage growth function(CGF) that describes the coverage growth process during testing. The performance of such SRGMs depends on how closely its CGF represents the actual coverage growth phenomenon. Recently, a class of CGFs for software reliability modeling was proposed by Park *et al.* (2007, 2008a). Three specific CGFs of the class have been empirically validated.

[1] Corresponding author: Professor, Department of Information and Statistics, RINS and RICI, Gyeongsang National University. 900, Gazwa-dong, Jinju 660-701, Korea. E-mail: gyemin@gnu.ac.kr

When multiple competitive CGFs are available, it is necessary to select the best one. This paper proposes a selection criterion based on predictability. Section 2 first describes the testing process and then reviews the CGFs of Park *et al.* (2007, 2008a). The conditional CGF for predicting the future coverage growth is introduced in Section 3. Then the sum of squares of the prediction error is defined and used as a selection criterion. The proposed selection criterion is applied to a real data set in Section 4.

2. Testing Process and CGF

The reliability of a software system is defined as the probability that the software system operates without failures for a specified time under a given usage environment. The set of all the possible inputs of the software system under testing is called the input domain. The usage environment is represented by a testing profile, which is a probability distribution over the input domain. Since testing requires the execution of the software system, test cases are selected randomly from the input domain according to the testing profile and then executed. Whenever a failure occurs, the failure time is recorded and the fault that caused the failure is removed. Alternatively, the number of faults detected up to some testing time points can be collected. Usually the cumulative execution time or the number of executed test cases is recorded as the testing time. These two types of testing data are respectively referred to as the failure time data and the fault count data. The traditional SRGMs are statistical models for evaluating software reliability based on the two types of data.

A recent trend is to measure coverage values and augment them to the data set. Let us first define the coverage and CGF. A software system can be considered as a collection of constructs, where a construct is a basic building element of a software system. Some usual constructs are statements, blocks, branches, c-uses and p-uses. Let M be the set of constructs of the software system under testing. Then M is the software system itself. The set of constructs executed up to t testing time is denoted by $M(t)$. One metric for measuring the thoroughness and/or the progress of the testing is the coverage defined as $C(t) = |M(t)|/|M|$, where $|\cdot|$ is the cardinality of a set. Since test cases are selected randomly from the inputs domain according to the given testing profile, $M(t)$ and $C(t)$ are stochastic processes. CGF $c(t)$ is defined as the expected value of $C(t)$, *i.e.*, the expected proportion of constructs executed by t.

Park *et al.* (2007) considered the case where the testing time is discrete and modeled the coverage growth process under the following assumptions:

(i) Constructs in M are executed independently,

(ii) The execution probability p of a construct follows a distribution with *cdf* $F(p)$ and *pdf* $f(p)$.

Assumption (ii) reflects that constructs in M may have different execution probabilities. Let T denote the time to execution of a construct. Due to Assumption (i), the time to execution of a construct with execution probability p follows a geometric distribution. The probability that a construct is executed up to t testing time is obtained as

$$\pi(t) = \Pr(T \le t) = \int_0^1 \Pr(T \le t \,|\, p)\ f(p)\,dp = \int_0^1 \left[1 - (1-p)^t\right] f(p)\,dp. \tag{2.1}$$

If $F(p)$ is a beta distribution with parameters α and β,

$$\pi_{\text{beta}}(t) = 1 - \frac{B(\alpha, \beta + t)}{B(\alpha, \beta)}, \tag{2.2}$$

where $B(\alpha,\beta)$ is the beta function. $\pi(t)$ for the continuous testing time obtained in Park *et al.* (2008a) by replacing Assumption (ii) with

(ii)′ The execution rate λ of a construct follows a distribution with *cdf* $G(\lambda)$ and *pdf* $g(\lambda)$.

Then the time to execution of a construct with execution rate λ is exponentially distributed. Similar to the discrete testing time case, the probability that a construct is executed up to t testing time is computed as

$$\pi(t) = \Pr(T \le t) = \int_0^\infty \left[1 - e^{-\lambda t}\right] g(p)\, dp. \tag{2.3}$$

If $G(\lambda)$ follows a gamma distribution with parameters α and β, then

$$\pi_{\text{gamma}}(t) = 1 - (1 + \beta t)^{-\alpha}. \tag{2.4}$$

If $G(\lambda)$ follows a lognormal distribution with parameters μ and σ, then

$$\pi_{\text{lognormal}}(t) = 1 - \int_0^\infty e^{-\lambda t}\, \frac{e^{-(\ln\lambda-\mu)^2/2\sigma^2}}{\lambda\sigma\sqrt{2\pi}}\, d\lambda. \tag{2.5}$$

Since $\pi(t)$ can be interpreted as the proportion of executed constructs up to t, $\pi(t)$ was proposed as a plausible CGF. However, a slight modification is required for a practical reason. Generally 100% coverage can rarely be achieved because of the presence of infeasible constructs and constructs with a negligible execution probability or execution rate. The upper bound is imposed on $\pi(t)$,

$$\widetilde{\pi}(t) = c_{\max}\, \pi(t), \tag{2.6}$$

where $c_{\max}$ is the maximum achievable coverage. It was also shown that $\widetilde{\pi}(t)$'s could be used as CGF irrespective of the continuity of the testing time and worked well for various real data sets.

3. Conditional CGF and Prediction Error

Suppose that a coverage growth process was observed at t_i for $i = 1, 2, \ldots, n$. Let c_{t_i} be values of $C(t)$ measured at t_i. Execution of the software system without the coverage increase does not help testers detect the remaining faults. This results in the overestimation of reliability. In order to decide whether we stop testing, we must evaluate the expected coverage increase from additional testing. Thus we consider the problem of predicting the coverage at some $t' > t_n$ when the corresponding CGF is given by $\widetilde{\pi}(t)$.

Coverage growth occurs only when some constructs in $M - M(t_n)$ are executed. It should be noted that the distribution of the execution probability over $M - M(t_n)$ is not $F(p)$ any more. In order to describe the coverage growth process after t_n, we need to derive the distribution of the execution probability of the constructs in $M - M(t_n)$. A similar argument can be made for the execution rate.

Let us first consider the case where the testing time is discrete. Since the distribution of the execution probability over $M - M(t_n)$ is obtained as

$$f(p \mid T > t_n) = \frac{\left[1 - \Pr(T \le t_n \mid p)\right] f(p)}{1 - \pi(t_n)}, \tag{3.1}$$

the probability that a construct in $M - M(t_n)$ is executed by t additional testing time after t_n is obtained as

$$\pi(t \mid t_n) = \int_0^1 \Pr(T \le t \mid p)\ f(p \mid T > t_n)\ dp = \frac{\pi(t_n + t) - \pi(t_n)}{1 - \pi(t_n)}. \tag{3.2}$$

It can be verified that

$$\pi(t \mid t_n) = \frac{\pi(t_n + t) - \pi(t_n)}{1 - \pi(t_n)} \tag{3.3}$$

also holds for the continuous testing time.

Now suppose that we want to predict the future coverage at t_n. The coverage at t_n is observed as c_{t_n} and $\pi(t \mid t_n)$ works on $M - M(t_n)$, not M. The coverage at $t_n + t$ given that $C(t_n) = c_{t_n}$ is therefore expected to be

$$c(t_n + t \mid c_{t_n}) = c_{t_n} + (c_{\max} - c_{t_n})\,\pi(t \mid t_n), \tag{3.4}$$

where $c(t_n + t \mid c_{t_n})$ is called the conditional CGF. Note that $c(t) = c(t_0 + t \mid t_0) = \widetilde{\pi}(t)$. Given that $C(t_{i-1}) = c_{t_{i-1}}$, the coverage at t_i can thus be predicted as $c(t_i \mid c_{t_{i-1}})$. The sum of squares of the prediction error(SSPE) of CGF $\widetilde{\pi}(t)$ is defined as

$$\sum_{i=1}^{n} \left[c_{t_i} - c(t_i \mid c_{t_{i-1}})\right]^2 = \sum_{i=1}^{n} \left[c_{t_i} - c_{t_{i-1}} - (c_{\max} - c_{t_{i-1}})\,\pi(t_i - t_{i-1} \mid t_{i-1})\right]^2. \tag{3.5}$$

When multiple CGFs are available, it is reasonable to select the CGF giving the minimum SSPE as the best CGF.

4. Application to a Real Data Set

It was empirically shown in Park *et al.* (2007, 2008a) that $\widetilde{\pi}_{\text{beta}}(t)$, $\widetilde{\pi}_{\text{gamma}}(t)$ and $\widetilde{\pi}_{\text{lognormal}}(t)$ performed well for various data sets and coverage metrics. In this section we illustrate the application of the proposed selection method to a real data set recently reported by Crespo *et al.* (2008). The data set consists of five coverage metrics, which are respectively all-nodes(NODES), all-arcs(ARCS), all-potential-uses(PU), all-potential-uses/du(PUDU) and all-potential-du-paths(PDU). The testing time is the number of executed test cases.

$\widetilde{\pi}_{\text{beta}}(t)$, $\widetilde{\pi}_{\text{gamma}}(t)$ and $\widetilde{\pi}_{\text{lognormal}}(t)$ were fitted to each coverage metric by the maximum Likelihood(ML) method. $\widetilde{\pi}_{\text{beta}}(t)$, $\widetilde{\pi}_{\text{gamma}}(t)$ and $\widetilde{\pi}_{\text{lognormal}}(t)$ fitted to 4 coverage metrics were displayed in Figure 1∼5 for the sake of brevity. Figure 1∼5 suggests that $\widetilde{\pi}_{\text{beta}}(t)$, $\widetilde{\pi}_{\text{gamma}}(t)$ and $\widetilde{\pi}_{\text{lognormal}}(t)$ work fairly well for this data set. Besides the visual inspection of the fitted CGFs, we need an objective numerical criterion to select the best CGF. The parameter estimates and the SSPE were computed and shown in Table 1. Judging from the SSPE, we can conclude that $\widetilde{\pi}_{\text{lognormal}}(t)$ is the best CGF for all the 5 coverage metrics.

Parameter $c_{\max}$ plays an important role in evaluating the testing progress. Unlikely to the data sets analyzed in Park *et al.* (2007, 2008a), estimates of $c_{\max}$ in Table 1 are quite different among the fitted competitive CGFs. It is desirable to perform additional testing before the most predictive CGF is chosen.

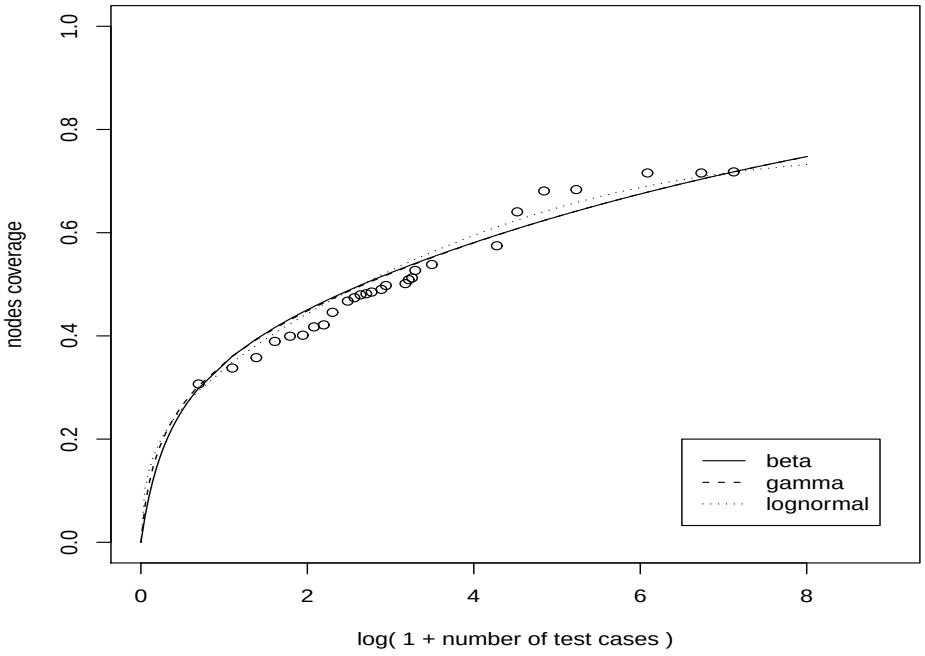

Figure 1: $\widetilde{\pi}_{beta}(t)$, $\widetilde{\pi}_{gamma}(t)$ and $\widetilde{\pi}_{lognormal}(t)$ *fitted to the data reported by Crespo et al. (2008): NODES coverage*

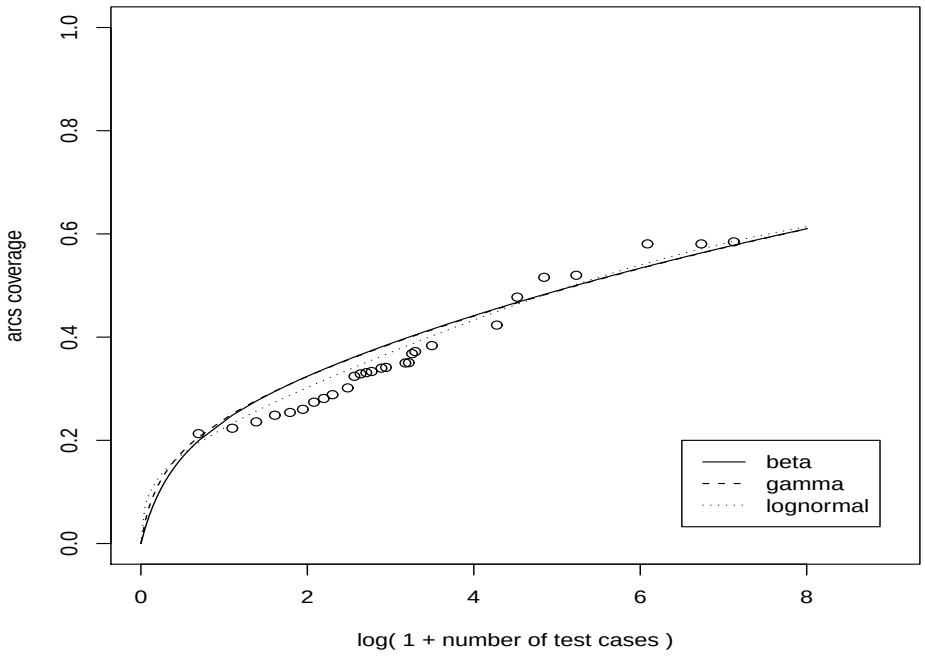

Figure 2: $\widetilde{\pi}_{beta}(t)$, $\widetilde{\pi}_{gamma}(t)$ and $\widetilde{\pi}_{lognormal}(t)$ *fitted to the data reported by Crespo et al. (2008): ARCS coverage*

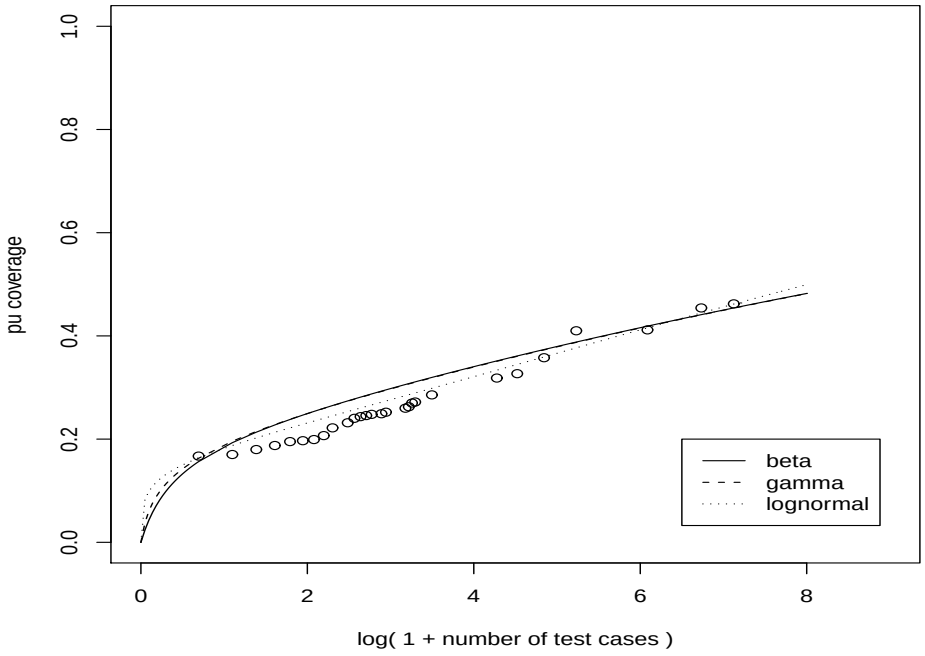

Figure 3: $\widetilde{\pi}_{beta}(t)$, $\widetilde{\pi}_{gamma}(t)$ and $\widetilde{\pi}_{lognormal}(t)$ fitted to the data reported by Crespo et al. (2008): PU coverage

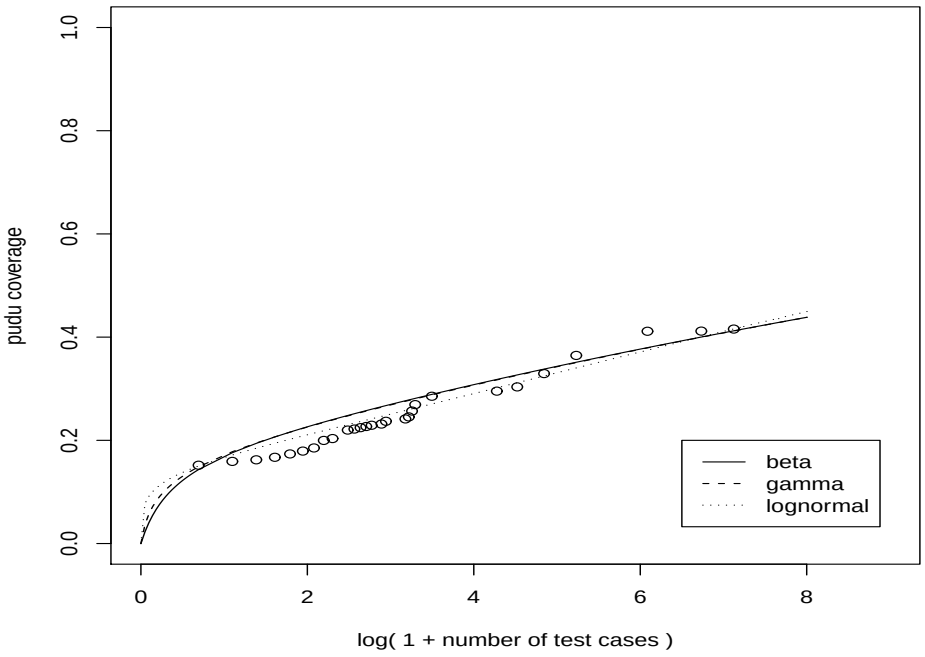

Figure 4: $\widetilde{\pi}_{beta}(t)$, $\widetilde{\pi}_{gamma}(t)$ and $\widetilde{\pi}_{lognormal}(t)$ fitted to the data reported by Crespo et al. (2008): PUDU coverage

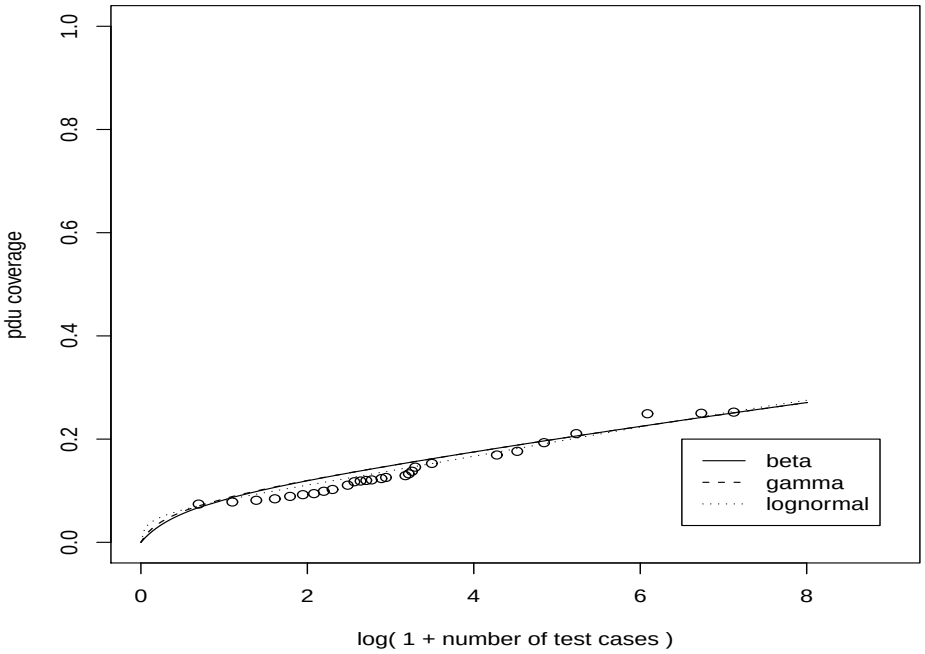

Figure 5: $\widetilde{\pi}_{beta}(t)$, $\widetilde{\pi}_{gamma}(t)$ and $\widetilde{\pi}_{lognormal}(t)$ *fitted to the data reported by Crespo et al. (2008): PDU coverage.*

Table 1: Maximum likelihood estimates and SSPE of the fitted CGFs.

CGF	Parameter	Coverage metric				
		NODES	ARCS	PU	PUDU	PDU
$\widetilde{\pi}_{\text{beta}}(t)$	$c_{\max}$	0.9994	1.0000	1.0000	1.0000	1.0000
	α	0.1263	0.0893	0.0602	0.0520	0.0307
	β	0.2965	0.3586	0.3248	0.3105	0.4247
	SSPE	0.0067	0.0065	0.0047	0.0034	0.0012
$\widetilde{\pi}_{\text{gamma}}(t)$	$c_{\max}$	0.9803	1.0000	1.0000	1.0000	1.0000
	α	0.1344	0.0895	0.0604	0.0523	0.0307
	β	14.6888	12.2138	17.9629	20.6918	9.8433
	SSPE	0.0063	0.0060	0.0044	0.0031	0.0011
$\widetilde{\pi}_{\text{lognormal}}(t)$	$c_{\max}$	0.7528	0.6861	0.7656	0.6930	0.3802
	μ	−1.5827	−3.0686	−5.9269	−5.9500	−5.3736
	σ	3.4601	4.2167	6.6349	6.7491	5.2210
	SSPE	0.0060	0.0055	0.0038	0.0028	0.0010

5. Conclusion

CGF is an essential component for modeling software reliability. When multiple competitive CGFs are available, we need to select the best CGF. CGF is used for predicting the additional amount of testing as well as evaluating the progress of testing. Predictive ability is considered a key aspect of CGF. This paper defined SSPE as a measure for the predictive ability of CGF and applied to CGFs that belong to the class of CGFs proposed by Park *el al.* (2007, 2008a). It is therefore required to investigate whether the proposed SSPE criterion was applicable to other CGFs. In this paper CGFs were first fitted by the method of maximum likelihood and then the predictive ability of the fitted CGFs was evaluated and compared in terms of SSPE. The method of maximum likelihood is an estimation

method optimal in the sense of goodness-of-fit. The primary concern was the predictive ability of CGFs and it would be better to fit CGFs by a method other than maximum likelihood.

References

Crespo, A. N., Pasquini, A., Jino, M. and Maldonado, J. C. (2008). A binomial software reliability model based on coverage of structural testing criteria, *Empirical Software Engineering,* **13**, 185–209.

Crespo, A. N., Pasquini, A., Jino, M. and Maldonado, J. C. (2009). Applying code coverage approach to an infinite failure software reliability model, In *Proceedings of 23rd Brazilian Symposium on Software Reliability Engineering,* 216–226.

Gokhale, S. S., Philip, T., Marinos, P. N. and Trivedi, K. S. (1996). Unification of finite failure non-homogeneous Poisson process models through test coverage, In *Proceedings of 7th IEEE International Symposium on Software Reliability Engineering,* 299–307.

Lyu, M. R. (1996). *Handbook of Software Reliability Engineering,* McGraw-Hill, New York.

Malaiya, Y. K., Li, M. N., Bieman, J. M. and Karcich, R. (2002). Software reliability growth and test coverage, *IEEE Transactions on Reliability,* **51**, 420–426.

Musa, J. D. (1999). *Software Reliability Engineering: More Reliable Faster Development and Testing,* McGraw-Hill, New York.

Musa, J. D., Iannino, A. and Okumoto, K. (1987). *Software Reliability: Measurement, Prediction, Application,* McGraw-Hill, New York.

Park, J. Y. and Fujiwara, T. (2006). Coverage growth functions for software reliability modeling, *Proceedings of 2nd Asian International Workshop on Advanced Reliability Modeling*, 435–442.

Park, J. Y., Lee, G. and Park, J. H. (2007). A class of discrete time coverage growth functions for software reliability engineering, *Communications of the Korean Statistical Society*, **14**, 497–506.

Park, J. Y., Lee, G. and Park, J. H. (2008a). A class of coverage growth functions and its practical application, *Journal of the Korean Statistical Society*, **37**, 241–247.

Park, J. Y., Lee, G. and Park, J. H. (2008b). A general coverage-based NHPP SRGM framework, *Communications of the Korean Statistical Society*, **15**, 875–881.

Pham, H. and Zhang, X. (2003). NHPP software reliability and cost models with testing coverage, *European Journal of Operational Research*, **145**, 443–454.

Received July 2010; Accepted October 2010