A Non-parametric Software Reliability Modeling Approach by Using Gene Expression Programming

HAI-FENG LI, MIN-YAN LU, MIN ZENG AND BAI-QIAO HUANG
School of Reliability and System Engineering
BeiHang University
Beijing, 100191 China
E-mail :{ lihaifeng;happybuaa}@dse.buaa.edu.cn; lmy@buaa.edu.cn; studyzm@163.com;

Software reliability growth models (SRGMs) are very important for estimating and predicting software reliability. However, because the assumptions of traditional parametric SRGMs (PSRMs) are usually not consistent with the real conditions, the applicability and prediction accuracy of PSRMs are hence not very satisfying in most cases. In contrast to PSRMs, the non-parametric SRGMs (NPSRMs) which use machine learning (ML) techniques, such as artificial neural networks (ANN), support vector machine (SVM) and genetic programming (GP), for reliability modeling by mining failure data without any assumptions, can provide better prediction results across various projects. Gene Expression Programming (GEP) which is a new evolutionary algorithm based on Genetic algorithm (GA) and GP, has been acknowledged as a powerful ML and widely used in the field of data mining. Thus, we apply GEP into non-parametric software reliability modeling in this paper due to its unique and pretty characters, such as genetic encoding method, translation process of chromosomes. This new GEP-based modeling approach considers some important characters of reliability modeling in several main components of GEP, i.e. function set, terminal criteria, fitness function, and then obtains the final NPSRM (GEP-NPSRM) by training on failure data. Finally, on several real failure data-sets based on time or coverage, five case studies are proposed by respectively comparing GEP-NPSRM with several representative PSRMs, NPSRMs based on ANN, SVM and GP, and fault detection models based on coverage (Coverage-FDM) in the form of fitting and prediction power which show that compared with the comparison models, the GEP-NPSRM has well applicability and provides a significantly better power of reliability fitting and prediction. In other words, the GEP is promising and effective for reliability modeling. So far as we know, it is the first time that GEP is applied into constructing NPSRM and Coverage-FDM.

Keywords: software reliability modeling, gene express programming, non-parametric model, machine learning, software reliability

1. INTRODUCTION

Software reliability is a very important customer oriented character of software quality and can be defined as the probability of failure-free software operation for a special period of time in a special usage environment [1]. As the main means for reliability estimation and prediction during software development and operation process, many software reliability growth models (SRGMs) have been proposed over the past 30 years and successfully applied into the development process of various types of safety-critical software. SRGMs use the failure data collected in testing process to estimate and predict the failure occurrences in the operational phase as well as to provide useful decision-making

information to project manager for effectively improving the reliability level and development process of software products [2].

According to the difference between modeling theory, most SRGMs can be classified into two categories [3]:

1) Parametric SRGM (PSRM). PSRMs are generally based on several assumptions on the nature of software faults and the stochastic behavior of testing process [5], and use some statistical theory (such as NHPP) to obtain the corresponding analytical models. The PSRMs have explicit expressions form and physical interpretation, and thus can be easily understood and used [4]. The well-known Jelinski-Moranda (JM) model and Goel-Okumoto (GO) model are two representative PSRMs. However, because the assumptions of PSRMs are usually not consistent with the real conditions, the applicability and prediction accuracy of PSRMs can't keep satisfactory all the time across various projects.

2) Non-parametric data-driven SRGM (NPSRM). NPSRMs utilize some machine learning (ML) techniques for modeling the inherent patterns of failure process, and then obtain estimation and prediction results of software reliability. Because NPSRMs don't require any prior assumptions and directly discover functions only by mining the failure data-set, they usually have pretty adaptive and self-learning performance and thus improve the applicability and prediction accuracy compared with PSRMs [5, 6]. Therefore, the researchers paid much attention to NPSRMs and have proposed many diverse modeling approaches for NPSRMs in recent years based on several ML techniques, such as artificial neural networks (ANN) [3, 6~15], support vector machine (SVM) [5, 16~21] and genetic programming (GP) [4, 22~23]. Obviously, the applicability and prediction accuracy of NPSRMs are heavily determined by the study power and application strategy of the used ML techniques. Thus, applying new and powerful ML techniques to obtain or improve NPSRMs is promising and meaningful for the research on reliability modeling.

Gene Express Programming (GEP) proposed by Ferreira [24] is a new evolutionary algorithm as the extension of genetic algorithm (GA) and GP in order to combine their advantageous features and overcome some limitations of them. Compared with GA and GP, GEP has the following unique characters [24~27]: 1) the chromosomes (candidate solution) are encoded as linear strings of fixed length which are afterwards directly translated into an expression tree (ET, the actual candidate solution) with no ambiguity; 2) It separates genotype (linear chromosomes) from phenotype (ET) which was one of the greatest limitations of GA and GP; 3) In GEP, genetic operators are applied on the chromosomes, not directly on ET. This reproduction method together with the encoding method and translation process of chromosomes, allows the unconstrained genetic modifications which always result in producing valid expression trees. On account of these characters, GEP outperforms GP by two to four orders of magnitude in terms of convergence speed [26] for solving complex modeling and optimization problems (such as, symbolic regression and function discovering), and thus has been applied into various engineering fields [27~29].

Thus, we suggest that GEP should be very suitable for the non-parametric software reliability modeling due to its unique and powerful characters for function discovering without any prior knowledge or assumptions. In this paper, we propose a new non-parametric reliability modeling approach based on GEP. This GEP-based modeling approach considers some important characters of software reliability modeling in several

main components of GEP such as the function set, fitness function and terminal criteria, to obtain the final NPSRM (GEP-NPSRM). Finally, on several real failure data-sets, we compare the GEP-NPSRM with several representative PSRMs and NPSRMs based on ANN, SVM and GP to validate its efficiency and applicability. Furthermore, we also apply GEP into modeling the fault prediction models based on test coverage (Coverage-FDMs) which are also very important because they are independent of the operation profile [23], and also compare the GEP-based Coverage-FDM with several existing Coverage-FDMs. So far as we know, it is the first time that GEP is applied into modeling NPSRMs as well as Coverage-FDMs.

The rest of this paper is organized as follows: Section 2 introduces some related works of NPSRMs. Section 3 introduces the GEP algorithm and proposes the non-parametric software reliability modeling approach based on GEP. Section 4 presents five case studies and discusses the results. Section 5 concludes this paper.

2. RELATED WORKS

Several ML techniques such as ANN, SVM and GP, have been successfully applied for modeling NPSRMs.

1. ANN-NPSRMs

Karunanithi [7] first applied ANN to predict software reliability with different configurations (such as, feed forward network, recurrent network), various training regimes and data representation methods. Then, Sitte [8] compared ANN-NPSRM with parametric recalibration on several data-sets to validate its effectiveness. Aljahdali [13] used the feed forward network in which the number of neurons in the input layer represents the number of delay in the input data. Cai [9] proposed a new ANN-NPSRM based on the neural back-propagation network and examined the performance of ANN architectures with various numbers of input nodes and hidden nodes. Ho [10] used a modified Elman recurrent network for reliability modeling and studied the effects of different feedback weights in the proposed model. Tian [11] proposed an evolutionary ANN-NPSRM based on the multiple-delayed-input single-output architecture and used GA to optimize the numbers of input nodes and hidden nodes. Zheng [12] used the ensemble of neural networks to modeling NPSRMs and Su [14] used the neural network approach to combine various SRGMs into a dynamic weighted combinational model. Emad [15] presented the functional networks as a new framework for non-parametric modeling. The above researches all show that ANN can model NPSRMs with varying complexity and adaptability for various failure data-sets.

2. SVM-NPSRMs

Besides ANN, many researches have applied SVM into reliability modeling and shown that SVM-NPSRMs also have well generalization capability of reliability prediction due to the structural risk minimization principle of SVM. Tian [21] proposed a SVM-NPSRM for reliability prediction and compared the new model with some ANN-NPSRMs. Pai [16] used the simulated annealing to optimize the parameters of the proposed SVM-NPSRMs (SVMSA). Xing [18] applied SVM for early software quality prediction. Literatures [19~20] applied SVM for system reliability modeling. Yang [17] proposed a SVM-NPSRM (DDSVM) and discussed the issues about failure data selection

and parameter optimization. Based on [17], Yang [5] proposed a generic SVM-NPSRM (SVMGA) by relaxing some unrealistic assumptions and using GA to optimize model parameters.

3. GP-NPSRMs

Costa suggested that ANN-NPSRMs are not easily interpreted [4] and thus proposed to apply GP into reliability modeling due to its powerful search efficiency. Costa [22] first applied GP into reliability modeling (GP-model) and compared this model with ANN-NPSRM. In [23], Costa introduced AdaBoosting technique into GP-model and the modified model (GPB-model) significantly improves the prediction power of GP-model. Furthermore, Costa [4] proposed a new GP-NPSRM (($\mu + \lambda$) GP-model) based on a new GP based approach. Compared with GPB-model, this new model has the same prediction performance with lower computational cost. The obtained results of [4, 22~23] shown that compared with the traditional PSRMs and ANN-NPSRMs, GP-NPSRMs adapt better to the reliability curve and can be used in projects with different characteristics.

3. The Non-parametric software reliability modeling approach by GEP

3.1 An overview of GEP algorithm

A complete GEP algorithm can be defined as the following 9-tuple:

$$GEP = \{C, E, P_0, M, \Phi, \Gamma, \Psi, \Pi, T\} \quad (1)$$

where C is encoding method of chromosomes; E is fitness evaluate function; P_0 is initial population; M is population size; Φ is selection and replication operator; Γ is recombination operator; Ψ is mutation operator; Π is transposition operator; T is termination criterion.

The flowchart of GEP is shown in Fig.1. According to Fig.1, we summarize the main steps of GEP here [26~27]:

Input: The control parameter settings for GEP (such as population size, recombination rate, transposition rate and so forth) and the training data-set.

Step1: Creating initial population P_0 which contains several individuals representing different candidate solutions. The individual, i.e. chromosome, is composed of one or more genes by the linking function with a fixed length. Each gene can be divided into a head composed of the function set (composed of some functions, i.e. +, -, *, /) and the terminal set (composed of some variables or constants), and a tail composed only of the terminal set.

Step2: Encoding chromosomes. In GEP, the chromosome can be represented by a fixed-length of linear character strings and afterwards translated into expression trees (ET) following a width-first fashion with different sizes and shapes when evaluating their fitness. The translation process starts from the first position in the string which corresponds to the root of the ET and reads through the string one-by-one from left to right for encoding the symbols in string into the nodes of ET. This tree expanding process continues layer-by-layer until all leaf nodes in ET are composed of elements

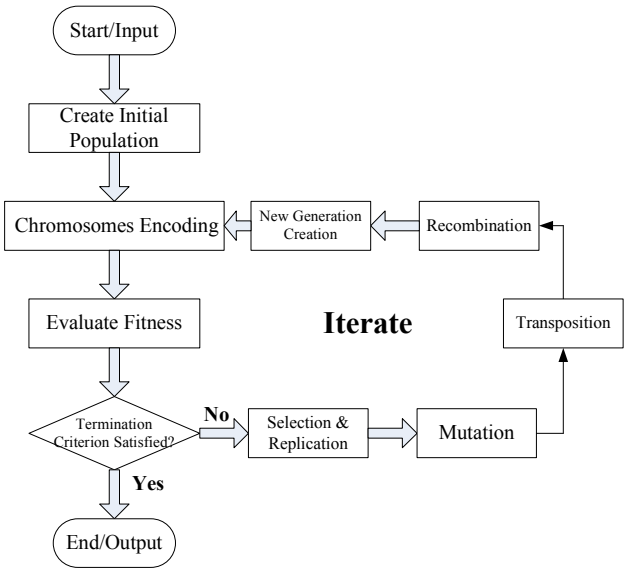

Fig. 1. The flowchart of GEP

from the terminal set. The reverse process that encoding the ET into a mathematical expression, implies reading the ET from left to right and from top to bottom. The details can be seen in [26~27].

Step3: Fitness evaluation. The fitness of each individual can be calculated by fitness evaluation function E (i.e. the fitness results of the mathematical expression corresponding to this individual on training data). If the termination criterion T (achieving the desired fitness or producing a given number of generations) is not satisfied, turning to Step 4. Otherwise stopping iteration and turning to the Output.

Step4: Creating new generation by selection and genetic operators. The chromosome is selected according to its fitness by the roulette-wheel method coupled with elitism. The roulette-wheel method allows the fitter chromosome to have a higher probability of producing offspring. Elitism allows the best chromosome is replicated unchanged into the next generation, preserving the best material from one generation to another. Then the selected chromosomes are modified with three classes of genetic operators for creating new generation, i.e., mutation, (IS, RIS, Gene) transposition, and (one-point, two-point, gene) recombination. Especially transposition operators are only used in GEP in contrast with GA and GP. Turning to Step 2 for a new iterative process.

Output: Encoding the fittest chromosome to produce the optimal solution $g(x)$ in the form required by the problem as it was developed by the GEP algorithm.

3.2 Software reliability modeling based on GEP

In this study, we introduce how to use GEP to extract the required non-parametric SRGM (i.e. GEP-NPSRM) from training failure data-set. There are five important components (i.e. the function set, terminal set, fitness function, control parameters and termination criterion) must be determined before using GEP. Thus, based on the general GEP algorithm introduced in 3.1, the GEP-based non-parametric software reliability modeling

approach is given here by considering some characters of reliability modeling into the above five components.

Input:

1. Control parameters of GEP. Because the data number of failure data-set is usually smaller than 1000 and the varying trend of failures with time is not very complex, we suggest that reliability modeling belongs to an ordinary problem of data mining. Thus, in this paper, we set the control parameters of GEP shown in Tab.1 according to the recommendation of [25, 30] and don't give any additional comments to this settings in Tab.1.

Table 1 The settings of control parameters of GEP

Population size	30
Head length	6
Number of Genes	3
One-point recombination rate	0.3
Two-point recombination rate	0.3
Gene recombination rate	0.1
Gene transposition rate	0.1
Mutation rate	0.044
Inversion rate	0.1
IS Transposition	0.1
RIS Transposition	0.1
Linking function	+

2. The training failure data-set D_0 which can be generally shown as two input forms $(t_1, m_1)...(t_j, m_j)...(t_n, m_n)$ or $(m_1, t_1)... (m_j, t_j)...(m_n, t_n)$, where n is data number of D_0, m_j is cumulated faults, t is failure time (interval time or cumulated time). If the form of NPSRM is shown as the cumulated fault with time $M(t)$, the former input form is used. If the form of NPSRM is shown as the time with cumulated faults $T(m)$, the latter input form is used.

Data Pre-process

Because of the complexity and uncertainty of testing process, the initial failure data-set unavoidably contains much noise which may affect the prediction accuracy of SRGM. Thus before reliability modeling, the initial failure data-set should be pre-processed first.

If the time data t in D_0 is recorded as the interval time, it should be converted to the cumulated time which may present a smoother curve shown in Fig.2 (the failure data-set is SYS1 [1]) and eliminate the noise more effectively compared with the interval time. Furthermore, we also recommend using several denoising methods, such as K-order moving average (recommended in [4]) or exponential smoothing, for data pre-processing.

Modeling Process

Step1: The initial population P_0 can be created by some initialization strategy (such as random initialization). If P_0 has the dominant characters, i.e. the genes are more diversified and suitable for the modeling object, the evolutional efficiency and the modeling quality can be effectively improved. Thus, for creating P_0 with dominant characters, in this paper, we recommended several elementary functions as the elements

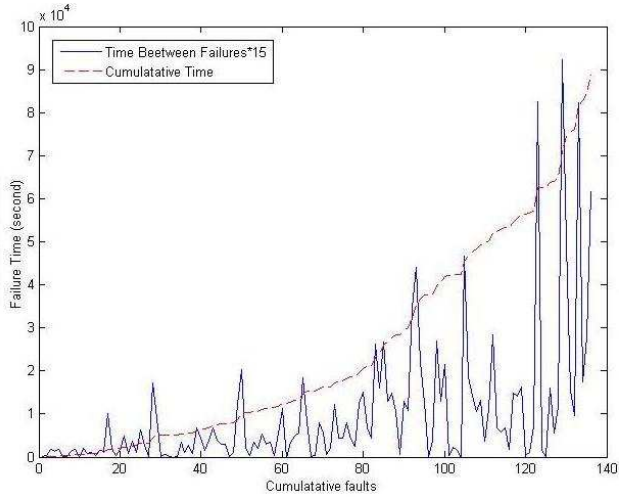

Fig. 2. The interval and cumulated curves of SYS1

of function set *Fs*, which are frequently used for software reliability modeling and shown as follows:

$$Fs=\{+,—,/,*,\exp(x),\mathrm{Sqrt},\mathrm{Log}\} \tag{2}$$

For further validating that the function set *Fs* shown in Eq.2 is indeed suitable for non-parametric reliability modeling, in section 4.1, we also compare the *Fs* with the following function set *Fs'* shown in Eq.3 which is also composed of several general elementary functions.

$$Fs'=\{+,—,/,*,10^{x},\sin,\cos\} \tag{3}$$

In the same way, because the GEP-NPSRM is used for reliability estimation and prediction, we recommend that the terminal set is compound by the failure time or the number of cumulated faults [4] in the training data-set and the random constant between 0 and 9.

Step2: Encoding chromosomes.

Step3: Fitness evaluation. The fitness function used to evaluate the fitness of chromosomes can guide the evolution process for finding the optimal solutions. The form of fitness function heavily depends on the type of problem. Thus, we recommend the following two fitness functions which are usually used as the comparison criteria of fitting or prediction power of SRGMs.

1. Mean Squared Error (MSE):

$$MSE=\frac{1}{n}\sum_{i=1}^{n}\left(y_i-y_i^{'}\right)^2 \tag{4}$$

2. R-Square:

$$R=1-\frac{\sum_{i=1}^{n}\left(y_i^{'}-y_i\right)^2}{\sum_{i=1}^{n}\left(y_i-y_{ave}\right)^2} \tag{5}$$

where y_i is the observed data, y_i' is the fitting data, y_{ave} is the average value of y_i. Ob-

viously, the value of MSE is the smaller or R-Square is closer to 1, the fitness of chromosome is the better.

Step4: If the current fitness of chromosome doesn't satisfy the terminal criterion T, turning to Step 5. Otherwise stopping iteration and turning to the Output. We recommend the following three forms of the terminal criterion T: 1) the fitness of chromosome achieves the required value; 2) the evolution process achieves a required number of generations; 3) the value of fitness has no change during the give number of generations.

Step5: Creating new generation by selection and a series of genetic operators.

Step6: Turning to Step 2 for a new iterative process.

Output: The required GEP-NPSRM satisfying the terminal criterion T: $M(t)$ or $T(m)$.

4. Case Study

For validating the effectiveness and applicability of GEP-NPSRM in the form of reliability estimation and prediction, we compare it with several existing SRGMs which have been widely cited by many researchers or are the representative and recent researches in the field of software reliability modeling, such as PSRMs, ANN-, SVM-, GP-NPSRMs, and Coverage-FDMs, on several real failure data-sets based on time and coverage which are popular and frequently used as the benchmark for the comparison of SRGMs. For the convenience of comparing GEP-NPSRM with the comparison models, the failure data-sets, comparison criteria, and the size of the training data in data-sets, which are used in the study 2~4 are all the same with the ones in the corresponding literatures. Moreover, in the study 2~4, the fitting and prediction results of ANN-, SVM-, and GP-NPSRMs are all directly obtained from the corresponding literatures.

The GeneXproTools 4.0 [30] which was developed by Ferreira is used for implementing GEP in this paper. The control parameters used to configure GeneXproTools are presented in Tab.1. The selected function set is shown as Eq.2 and the selected fitness function is MSE (shown in Eq.4). If there is no especial explanation, the interval time data of failure data-sets in this study is converted to the cumulated time data first for further modeling.

On each failure data-set, we apply the GEP algorithm 20 times for modeling and obtaining 20 GEP-NPSRMs. Then the GEP-NPSRM which has the best fitting power is selected for comparison on this data-set. The terminal criterion T in each study is determined as: If the fitness (MSE) of chromosome achieves the required value, the evolution process is stopped. Else if the fitness value has no change during the given number (50000) of generations, the evolution process is stopped. Else if the total number of generations achieves the given value (200000), the evolution process is stopped.

4.1 Study 1: GEP-NPSRM vs. PSRM

1. Description of Study 1

1) Thirteen NHPP PSRMs are selected for comparison, namely Goel-Okumoto (GO) [2], Delayed S-shaped (DS) [2], Inflection S-shaped (IS) [2], Yamada Weibull (YW) [32],

Yamada Rayleigh (YR) [32], Generalized GO (GGO) [32], Yamada Imperfect Debugging 1 & 2 (YID 1 & YID 2) [31], Ohba Imperfect Debugging (OID) [32], P-Z Imperfect Debugging Coverage (PNCZ) [33], P-Z Imperfect Debugging (PNZ) [33], Log-Logistic Coverage (LL) [33], Logistic test coverage (LTCS) [34].

2) Five real failure data-sets are selected, namely 'ATT [16]', 'Ohba [35]', 'Wood [35]', 'SYS1 [1]' and 'S5 [36]'. The whole of each data-set is used as the training set and for calculating the fitting result.

3) Three criteria are selected for comparing the fitting performance of SRGMs respectively, namely MSE, R-Square, and the average error [14] (AE, shown in Eq.6):

$$AE = \frac{1}{n}\sum_{j=1}^{n}\left|\frac{y_j - y_j^{'}}{y_j}\right| \times 100 \tag{6}$$

The value of MSE or AE is the smaller, and the value of R-Square is closer to 1, the fitting power is the better.

4) This study uses two forms of GEP-NPSRM for comparison, i.e. GEP(1) modeled by the function set *Fs* (shown in Eq.2) and GEP(2) modeled by the function set *Fs'* (shown in Eq.3).

5) Because the forms of these thirteen NHPP PSRMs are all $M(t)$, the output form of GEP-NPSRM in this study is also $M(t)$. Correspondingly, the input form of these five failure data-sets is $(t_1, m_1)...(t_j, m_j)...(t_n, m_n)$.

2. Comparison of fitting performance

1) The fitting results (i.e. the values of MSE, R-Square and AE) of the two GEP-NPSRMs and thirteen NHPP PSRMs on five failure data-sets are shown in Tab.2 respectively. The bold number indicates the best fitting result in this column.

2) From Tab.2, on each data-sets, the fitting results of GEP(1) are nearly all better (i.e. the values of AE and MSE are both smaller and the value of R-square is more close to 1) than GEP(2). Only on the data-set 'S5', the AE value of GEP(1) is a little larger than GEP(2). Therefore, it shows that GEP(1) indeed has better fitting power and applicability than GEP(2). In other words, the function set *Fs* is more suitable for non-parametric reliability modeling than the function set *Fs'* in this paper. The underlying reason may be that some elements of function set *Fs'*, i.e. $\{10^x,\sin,\cos\}$ are not suitable for reliability modeling, Thus, the function set *Fs* will be used for obtaining the GEP-NPSRM in this paper.

3) From Tab.2, on the five failure data-sets, the fitting results of GEP-NPSRM are all nearly better than the ones of the thirteen PSRMs. Especially, several fitting results are significantly better than the other PSRMs, such as the MSE value (44.85) on 'Ohba', and the AE value (10.3) on 'ATT'. Only on 'SYS1', the MSE value of GEP-NPSRM is a little larger than the GGO, but still smaller than the other twelve PSRMs.

To sum up, according to the above analysis, since the GEP-NPSRM nearly yields the best fitting results across each failure data-set, we conclude that compared with the traditional PSRMs, the fitting power and the applicability of GEP-NPSRM are significantly better.

4.2 Study 2: GEP-NPSRM vs. ANN-NPSRM

Table 2 The fitting results of GEP-NPSRM and PSRMs

Model	ATT (22)			Ohba (19)			Wood (20)			SYS1 (136)			S5 (34)		
	MSE	R	AE	MSE	R	AE	MSE	R	AE	MSE	R	AE	MSE	R	AE
GO	1.4	0.954	40.7	139.8	0.986	7.28	11.6	0.913	19.6	46.5	0.971	84.2	16.8	0.995	6.64
DS	1.15	0.968	354	168.7	0.984	19.8	25.3	0.969	31.8	249.8	0.842	588	19.5	0.997	29
IS	1.4	0.970	69.9	127.3	0.992	6.24	9.0	0.989	8.42	46.5	0.972	84.2	5.82	**0.998**	3.79
YW	1.18	0.970	214	260	---	32.5	16.6	0.987	8.73	218.1	0.867	24.1	7.0	**0.998**	5.68
YR	1.58	0.403	30.5	268.4	0.733	28.2	39.7	0.951	54.6	766.2	0.506	---	41.4	0.987	49.7
GGO	2.1	0.967	140	102.1	0.990	6.0	10.9	0.987	8.73	**6.27**	0.991	6.05	6.92	**0.998**	5.77
YID1	1.63	---	34.7	154.8	0.986	7.28	12.1	0.986	7.6	---	---	---	16.8	0.995	6.59
YID2	1.6	0.954	35.6	565.5	0.986	7.28	36.9	0.986	7.6	46.5	0.971	80.1	16.8	0.995	182
OID	1.4	0.954	35.4	139.8	0.986	7.25	11.6	0.986	7.6	46.5	0.973	80.1	16.9	0.995	6.77
PNCZ	1.12	0.964	261	138.7	0.987	11.5	19.6	0.976	21.7	171.7	0.895	---	11.7	0.996	19.1
PNZ]	1.34	0.970	60	223.9	0.992	6.24	9.2	0.988	8.42	46.5	0.970	84.2	5.82	**0.998**	3.79
LL	1.18	**0.971**	---	194.1	0.989	6.01	15.4	0.984	9.0	12.0	0.993	7.71	7.33	**0.998**	6.03
LTCS	1.08	**0.971**	65.7	86.1	0.992	6.30	9.4	0.987	8.53	---	---	---	5.88	**0.998**	3.82
GEP(2)	1.95	0.953	19.8	45.8	0.995	6.01	11.78	0.987	6.11	14.19	0.991	10.9	6.68	**0.998**	**3.25**
GEP(1)	**0.89**	**0.971**	**10.3**	**44.85**	**0.996**	**5.23**	**8.11**	**0.991**	**5.12**	7.76	**0.995**	**5.09**	**5.67**	**0.998**	3.79

Notes: 1) The number in bracket in the first row s the size of this data-set; 2) The bold number is the best result in this column; 3) '---' means this result is unreasonable or significantly worse than the other results in this column.

1. Description of Study 2

1) The FunNets model proposed in [15] is selected as the major comparison ANN-NPSRM in this study. Meanwhile, the multiple regression (MR), feed forward neural networks (FFN) [13], and SVM [16] are also selected for the incidental comparison models.

2) Two real failure data-sets are selected, i.e. 'ATT' and 'SYS1'. For the convenience of comparing with the results of [15], we use 70% of each data-set for training, while the remaining 30% is used for predicting. The input form of 'ATT' or 'SYS1' is $(m_1,t_1)...\ (m_j,t_j)...(m_n,\ t_n)$ and the output form of GEP-NPSRM is $T(m)$.

3) The following two comparison criteria are selected, namely root mean squares error (RMSE, shown in Eq.7) and correlation coefficient (CC, shown in Eq.8):

$$\text{RMSE}=\left[\frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i-y_i^{'}}{y_i}*100\right)^2\right]^{\frac{1}{2}} \tag{7}$$

$$\text{CC}=\sqrt{1-\frac{\sum_{i=1}^{n}\left(y_i^{'}-y_i\right)^2}{\sum_{i=1}^{n}\left(y_i-y_{ave}\right)^2}} \tag{8}$$

The value of RMSE is the smaller, and the value of CC is closer to 1, the fitting or prediction power is the better.

2. Comparison of fitting and prediction performance

1) The fitting and prediction results (CC and RMSE) of GEP-NPSRM and four incidental models on 'ATT' and 'SYS1' are shown in Tab.3 and Tab.4 respectively.

2) From Tab.3, on 'ATT', the fitting results (CC is 0.9986, RMSE is 11.82) and the prediction results (CC is 0.9981, RMSE is 4.68) of the proposed GEP-NPSRM are all significantly smaller than the two representative ANN-NPSRMs (i.e. FunNets and FFN) as well as MR and SVM-NPSRM. Furthermore, from Tab.4, on 'SYS1', the fitting results (CC is 0.9982, RMSE is 576.99) and the prediction results (CC is 0.9852, RMSE is 780.14) of the GEP-NPSRM are also nearly all significantly smaller than the four comparison models. Only the prediction result in the form of CC is a little larger than Fun-

Nets, but still smaller than the other three comparison models.

To sum up, the comparison results shown in Tab.3 and 4 both show that the GEP-NPSRM has the evidently better fitting and prediction power and the better applicability compared with the ANN-NPSRMs.

Table 3 The fitting and prediction results on ATT

Model	Fitting		Prediction	
	CC	RMSE	CC	RMSE
MR	0.932	51.83	0.988	132.29
FFN	0.984	29.92	0.996	50.47
SVM	0.982	28.79	0.996	19.56
FunNets	0.998	24	0.998	11.2
GEP	**0.9986**	**11.82**	**0.9981**	**4.68**

Note: The bold number is the best result in this column

Table 4 The fitting and prediction results on SYS1

Model	Fitting		Prediction	
	CC	RMSE	CC	RMSE
MR	0.935	7838.41	0.9611	8132.59
FFN	0.9963	2112.73	0.9973	1697.91
SVM	0.9948	3136.8	0.9978	3336.2
FunNets	0.9963	1859.35	**0.998**	1669.65
GEP	**0.9982**	**576.99**	0.9852	**780.14**

Note: The bold number is the best result in this column

4.3 Study 3: GEP-NPSRM vs. SVM-NPSRM

1. Description of Study 3

1) Two representative SVM-NPSRMs are selected as the major comparison models, namely SVMSA [16] and SVMGA [5].

2) The failure data-sets used for comparing with SVMSA are 'ATT' and 'Musa' [16]. The time data of 'ATT' is shown in the form of interval time. The former 18 data of 'ATT' are used as the training data-set for fitting and predicting the whole 22 data of 'ATT'. The former 33 data of 'Musa' (the size is 101) are used for training and the latter 60 data are used for predicting (the middle 8 data are not used for calculation in [16]). The input form of 'ATT' or 'Musa' is $(m_1,t_1)... (m_j,t_j)...(m_n, t_n)$ and the output form of GEP-NPSRM is $T(m)$.

3) The failure data-sets used for comparing with SVMGA are 'ATT' and 'Wood2' [5]. The time data of 'ATT' is also interval time. The former 18 data of 'ATT' are used for training and the latter 4 data are used for predicting. The former 15 data of 'Wood2' (the size is 19) are used for training and the latter 4 data are used for predicting. The input form of 'ATT' is $(m_1,t_1)... (m_j,t_j)...(m_n, t_n)$ and the output form of GEP-NPSRM is $T(m)$. The input form of 'Wood2' is $(t_1, m_1)...(t_j, m_j)...(t_n, m_n)$ and the output form of GEP-NPSRM is $M(t)$.

4) Two comparison criteria are selected, i.e. MSE and the sum of square errors (SSE) shown in Eq.9:

$$SSE = \sum_{i=1}^{n} \left(y_i - y_i^{'} \right)^2 \tag{9}$$

The value of SSE or MSE is the smaller, the fitting or prediction power is the better

2. Comparison of prediction performance

1) The prediction value of each data in 'ATT' and the prediction results (SSE) obtained by GEP-NPSRM, SVMSA and four Weibull Bayes prediction models [37~38] (Weibull Bayes I-IV) on 'ATT' are shown in Tab.5 respectively. The prediction results (MSE) of GEP-NPSRM, SVMSA and four autoregressive prediction models [39] on 'Musa' are shown in Tab.6. The prediction results (MSE) of GEP-NPSRM, SVMGA and DDSVM [17] on 'ATT' and 'Wood2' are shown in Tab.7.

Table 5 The prediction results of GEP and SVMSA on 'ATT'

Actual values	GEP	SVMSA	Weibull Bayes I	Weibull Bayes II	Weibull Bayes III	Weibull Bayes IV
5.5	1.94537	7.16150	5.48073	5.48073	NA	NA
1.83	1.87316	0.16848	2.74316	2.74315	NA	NA
2.75	1.89084	4.41150	2.74347	2.74345	NA	NA
70.89	74.24861	69.2280	2.80002	2.80000	71.19535	69.92294
3.94	3.98981	5.60150	14.36394	14.06833	NA	NA
14.98	4.41486	13.3180	11.31019	10.68980	NA	NA
3.47	5.22478	5.13150	15.41370	14.65534	NA	NA
9.96	6.33338	8.29850	12.09344	11.00586	NA	NA
11.39	7.77988	13.0520	12.47982	11.19407	NA	NA
19.88	9.63910	18.2180	12.93261	11.47537	NA	NA
7.81	12.01062	8.13590	20.13083	18.82182	NA	NA
14.59	15.01560	16.2520	14.14916	12.30426	NA	NA
11.42	18.78692	9.7585	14.97879	13.30455	NA	NA
18.94	23.42496	20.6020	14.90016	12.94462	NA	NA
65.3	28.77998	63.6380	19.19392	19.22512	59.18903	53.49376
0.04	32.67066	1.70150	24.22551	23.92506	NA	NA
125.67	128.04522	124.010	71.28477	69.31352	NA	NA
82.69	74.72239	84.3520	31.38095	26.66547	NA	NA
0.45	47.93228	16.0420	32.52159	27.31916	NA	NA
31.61	42.61851	25.4320	31.6149	26.44283	NA	NA
129.31	121.22631	208.580	63.87793	62.67870	156.8901	150.5327
47.6	64.53164	44.2530	39.68720	34.45885	NA	NA
SSE	**5558.0301**	6623.26	18810.48	19490.2	9910.025	9611.389

Notes: 1) The bold number is the best result in this column; 2) NA means this prediction value is not given in the literature [16].

2) From Tab.5, on 'ATT', the prediction result of GEP-NPSRM is the best (the SSE value is 5558.0311 which is the smallest) compared with SVMSA and four Bayes prediction models. It should be noted that, the prediction result of each model on 'ATT' in Tab.5 or 7 seems not very good (i.e., in Tab.5, the value of SSE achieves more than 10^3 or even 10^4, and in Tab.7, the value of MSE achieves more than 10^2). The underlying reason may be that because the time data of 'ATT' is shown in the form of interval time in this study, the difference between the magnitude of time data in 'ATT' may be a little larger, such as from 10^2 (129.31) to 10^{-2} (0.04), which makes the changing trend of time data (with the growth of total faults) not very obvious. Thus, the quantitative relationship between the interval time and cumulated faults is difficult to be described very accurately by the comparison models in Tab.5 and 7, namely, the fitting or prediction value of each data can't keep very close to the corresponding real value all the time. The above analysis again shows that the modeling performance of the interval time data is generally worse than the cumulated time data. So we'd better transform the interval time data into the cumulated time for obtaining better modeling power which has also been mentioned in section 3.2.

3) It should be noted that, from Tab.6, the prediction result (MSE) of GEP-NPSRM

Table 6 The prediction results on 'Musa'

Model	Prediction (MSE)
SVMSA	**3.1012**
Model I (normal distribution)	5.5812
Model II (Kalman filter I)	15.2369
Model III (Kalman filter II)	10.5903
Model IV (adaptive Kalman filter)	20.8915
GEP	3.6020

Note: The bold number is the best result in this column

Table 7 The prediction results on 'ATT' and 'Wood2'

Model	ATT (MSE)	Wood2 (MSE)
DDSVM	2343.2	1.42
SVMGA	**670.56**	0.0487
GEP	681.94	**0.0158**

Note: The bold number is the best result in this column

on 'Musa' is a little worse than SVMSA, but the difference is trivial (3.6020 vs. 3.1012). And from Tab.7, the prediction result (MSE) of GEP-NPSRM on 'Wood2' is significantly better than the ones of SVMGA and DDSVM, namely, the MSE value (0.0158) of GEP-NPSRM is the smallest among the tree comparison models. Moreover, on 'ATT', the prediction result (MSE) of GEP-NPSRM is a little worse than or nearly the same with the one of SVMGA (681.94 vs. 670.56), but significantly better than the one of DDSVM.

To sum up, according to the above analysis, we suggest that compared with the SVM-NPSRM, the GEP-NPSRM yields a comparable at least or even a little better prediction power as well as the applicability on the whole.

4.4 Study 4: GEP-NPSRM vs. GP-NPSRM

1. Description of Study 4

1) Three representative GP-NPSRMs, i.e. GP-model [24], GPB-model [25] and ($\mu + \lambda$) GP-model [4], are selected as the comparison models in this study.

2) Nine real failure data-sets shown in [40] are selected for comparison, i.e. '3', '27', '4', '2', '6', 'Musa', 'SYS1', 'SS4', and 'SS3'. The size of each data is shown in Tab.8. We use the first 2/3 of each data-set for training and use the remaining 1/3 of each data-set is used for predicting. The input form of each data-set is $(m_1,t_1)...\ (m_j,t_j)...(m_n,\ t_n)$ and the output form of GEP-NPSRM is $T(m)$. Each data-set is processed by the moving average method before modeling [4].

3) The comparison criteria are AE (shown in Eq.6) and CC (shown in Eq.8).

2. Comparison of prediction performance

1) The prediction results of GEP-NPSRM, GP-, GPB- and ($\mu + \lambda$) GP-model on the nine data-sets are shown in Tab.8 respectively in the form of AE and CC. The bold number means the best result in this column.

2) According to Tab.8, on six data-sets (i.e. '3', '6', 'Musa', 'SYS1', 'SS4' and 'SS3'), the prediction results (AE and CC) of GEP-NPSRM are all better than the other three GP-NPSRMs respectively, namely, the AE value is the smallest and the CC value is the closest to 1. On '27' and '4', the prediction results of GEP-NPSRM are a little

Table 8 The prediction results of GEP-NPSRM and GP-NPSRMs

Data Set	Prediction Results (AE)	Prediction Results (CC)

	GP	GPB	($\mu+\lambda$)GP	GEP	GP	GPB	($\mu+\lambda$)GP	GEP
3(38)	17.45	10.20	5.36	**4.99**	0.6632	0.9796	0.9909	**0.9912**
27(41)	17.66	10.20	**5.25**	5.92	0.8522	0.9421	**0.9938**	0.9910
4(53)	15.86	16.40	**7.78**	12.46	0.895	0.876	**0.9859**	0.9798
2(54)	4.08	3.40	**3.32**	4.92	0.996	0.9969	**0.9973**	0.9954
6(73)	9.46	9.60	8.94	**6.38**	0.981	0.9812	0.894	**0.9843**
Musa(101)	38.82	10.20	8.18	**1.15**	0.9044	0.9573	0.9933	**0.9990**
SYS1(136)	6.75	5.20	4.35	**4.30**	0.9875	0.9958	0.9981	**0.9983**
SS4(196)	9.174	9.30	14	**2.00**	0.9855	0.9753	0.9964	**0.9981**
SS3(278)	14.77	8.60	15.71	**3.67**	0.9657	0.9892	0.9876	**0.9951**

Notes: 1) The number in bracket is the size of this data-set; 2) The bold number is the best result in this column.

worse than the ($\mu+\lambda$) GP-model, but still are better than the GP-, and GPB-model. Only on the data-set '2', the prediction results of GEP-NPSRM are the worst. Thus, the above analysis shows that compared with the three GP-NPSRMs, the GEP-NPSRM provides the best prediction power and applicability on the whole (best on six data-sets and better on two data-sets).

3) Costa [4] suggested that although the ($\mu+\lambda$) GP-model is very suitable for the data-sets with small size (i.e. the data number of data-set is smaller than 100), there is no significant difference in the performance compared with the GPB-model on the data-set with large size (i.e. the data number is larger than 100) (such as the prediction results on 'SS4' and 'SS3'). However, from Tab.8, compared with the three GP-NPSRMs, the proposed GEP-NPSRM provides better prediction results on both the small data-sets (such as '3', '27', ''4' and '6') and the large data-sets (such as 'Musa', 'SYS1', 'SS4' and 'SS3'). Thus, it can be conclude that the applicability of GEP-NPSRM is better than the ($\mu+\lambda$) GP-model

To sum up, according to the above analysis, we suggest that compared with GP-NPSRMs, the GEP-NPSRM has better prediction power and applicability.

4.5 Study 5: GEP-FDM vs. Coverage-FDM

1. Description of Study 5

1) In this study, we apply GEP into modeling non-parametric Coverage-FDM and select four existing parametric Coverage-FDMs for comparison, i.e. Beta & Hyper-Exp [42], Log-Exp [41] and Logistic FDMs [34].

2) We select four coverage-based (block, branch, P-use and C-use coverage) failure data-sets from the DS3 data-set as the comparison data, which were collected from a NASA supported project and shown in [41]. The whole of each data-set is used as the training set and for calculating the fitting result. The input form of each data-set is (c_1, m_1)...(c_j, m_j)...(c_n, m_n), where c_j means the test coverage corresponding to the cumulated faults m_j, and thus the output form of GEP-NPSRM is $M(c)$.

3) The comparison criteria are MSE, R-Square, and AE.

2. Comparison of fitting performance

1) The fitting results (the values of MSE, R-Square and AE) of the GEP-based non-parametric Coverage-FDM and four Coverage-FDMs on four coverage-based data-sets are shown in Tab.9 respectively. The bold number means the best fitting result in

this column.

2) From Tab.9, on the four coverage-based failure data-sets, the fitting results of GEP-based FDM are all nearly obviously better than the ones of the four Coverage-FDMs. Only on the 'P-use Coverage', the AE value of GEP-based FDM is a little larger than L-G, but still smaller than the other comparison Coverage-FDMs.

To sum up, we conclude that on the four coverage-based failure data-sets, compared with the parametric coverage-FDMs, the GEP-based non-parametric Coverage-FDM provides more accurate fitting results and applicability as a whole.

Table 9 The fitting results of GEP-NPSRM and coverage-based fault detection models

Coverage-FDM	Block Coverage			Branch Coverage			P-use Coverage			C-use Coverage		
	MSE	R	AE	MSE	R	AE	MSE	R	AE	MSE	R	AE
Beta	0.40	0.9513	12.11	0.45	0.9451	13.36	0.54	0.9342	15.07	1.89	0.7714	23.33
Hyper-Exp	3.45	0.5821	30.01	2.94	0.6437	28.14	1.96	0.7625	24.60	5.32	0.3547	35.86
Log-Exp	0.24	0.9709	12.38	0.16	0.9803	9.85	0.10	0.9877	7.27	2.10	0.746	24.75
Logistic	0.16	0.9808	10.05	0.095	0.9885	8.04	0.05	0.9940	**3.70**	0.291	0.9647	12.2
GEP	**0.08**	**0.9903**	**6.06**	**0.057**	**0.9934**	**5.98**	**0.044**	**0.9947**	3.86	**0.255**	**0.9694**	**10.40**

Note: The bold number is the best result in this column

5. Conclusion

This paper applies the GEP algorithm into reliability modeling and proposes a novel GEP-based non-parametric software reliability modeling approach. This modeling approach considers some important characters of reliability modeling in several main components of GEP algorithm for resulting in the GEP-NPSRMs by using GEP to mine the failure data-set to discover the relationship between the observed failure time (or test coverage) and faults directly without any assumptions. On several real failure data-sets, five comparative studies are presented respectively for comparing the fitting and prediction power of GEP-NPSRMs with several representative PSRMs, ANN-, SVM-, GP-NPSRMs and Coverage-FDMs. The experimental results show that compared with these comparison models, the proposed GEP-NPSRM provides significantly better fitting and prediction results on most data-sets without any assumptions due to the unique and powerful encoding method as well as genetic operators of GEP algorithm. In other words, it shows that the application of GEP algorithm to non-parametric reliability modeling is an effective and novel attempt which may be very significant for further researches and applications.

There are some discussions as follows: 1) Because of the randomness existing in the process of creating initial population and genetic operations, the fitting and prediction results of the final GEP-NPSRM can't always keep better than the comparison models all the time. Even if we apply GEP on the training data-set many times, it may still be not able to obtain the satisfactory results, such as on several data-sets, the GEP-NPSRM can't provide better results than some comparison model all the time. This is an unavoidable problem existing in many machine learning techniques. However, we have shown that in most cases, the proposed GEP-NPSRM indeed obtains better and satisfying results. Therefore, we suggest that when using GEP for modeling, the experiments should be made as many as possible for obtaining the best fitting and prediction results to ensure the effectiveness of GEP-NPSRM; 2) The GEP algorithm is more suitable for mining the failure data-sets with the obvious tendency (such as the cumulated failure time data with

the obvious increasing trend). Thus, we suggest that before modeling, the training data-set should be translated into the form of cumulated failure time.

The following issues will be further discussed in our future work: 1) Combining some other ML techniques with the GEP algorithm, such as AdaBoosting, simulated annealing; 2) Selecting several fault defection model (i.e. $M(t)$) as the elements of the function set for improving the dominant characters of initial population; 3) The overfitting problem existing in the non-parametric modeling.

REFERENCES

1. M. R. Lyu, *Handbook of Software Reliability Engineering*, McGraw Hill, 1996
2. Pham H, *Software reliability*, Singapore: Springer-Verlag, 2000
3. Hu Q.P, Xie M, Ng SH, "Robust recurrent neural network modeling for software fault detection and correction prediction," *Reliability Engineering & System Safety*, Vol. 92, 2007, pp. 332-340
4. E. O. Costa, A. T. R. Pozo, and S. R. Vergilio, "A genetic programming approach for software reliability modeling," *IEEE Transactions on Reliability*, Vol. 59, 2010, pp. 222-230
5. B. Yang, X. Li, M. Xie, F. Tan, "A generic data-driven software reliability model with model mining technique," *Reliability Engineering and System Safety*, Vol. 95, 2010, pp. 671-678
6. Q.P. Hu, M. Xie and S.H. Ng, "Software Reliability Predictions using Artificial Neural Networks," *Computational Intelligence in Reliability Engineering*, Vol. 40, 2007, pp. 197-222
7. N. Karunanithi, D. Whitley and Y. K. Malaiya, "Prediction of software reliability using connectionist models," *IEEE Transactions on Software Engineering*, Vol.18, 1992, pp. 563-574
8. Sitte, R, "Comparison of software-reliability-growth predictions: Neural networks vs parametric recalibration," *IEEE Transactions on Reliability*, Vol.48, 1999, pp. 285-291
9. Cai, K.Y., Cai, L., Wang, W.D., Yu, Z.Y., Zhang, D, "On the neural network approach in software reliability modeling," *The Journal of Systems and Software*, Vol. 58, 2001, pp. 47-62
10. Ho SL, Xie M, Goh TN, "A study of the connectionist models for software reliability prediction," *Computers & Mathematics with Applications*, Vol. 46, 2003, pp. 1037-1045
11. Tian L, Noore A, "Evolutionary neural network modeling for software cumulative failure time prediction," *Reliability Engineering & System Safety*, Vol. 87, 2005, pp. 45-51
12. J. Zheng, "Predicting software reliability with neural network ensembles," *Expert Systems with Applications*, Vol. 36, 2009, pp. 2116-2122
13. S. H. Aljahdali, A. Sheta, D. Rine, "Prediction of software reliability: A comparison between regression and neural network non-parametric models," in *ACS/IEEE International Conference on Computer System and Application*, 2001, pp. 470-473
14. Y S Su, C Y Huang, "Neural-network-based approaches for software reliability es-

timation using dynamic weighted combinational models," *The Journal of Systems and Software*, Vol. 80, 2007, pp. 606-615
15. Emad A. El-Sebakhy, "Software reliability identification using functional networks: A comparative study," *Expert Systems with Applications,* Vol. 36, 2009, pp. 4013-4020
16. P F Pai, W C Hong, "Software reliability forecasting by support vector machines with simulated annealing algorithms," *The Journal of Systems and Software*, Vol. 79, 2006, pp. 747-755
17. Yang B, Tan F, Huang HZ, "Data selection for support vector machine based software reliability models," in *Proceedings of international conference on reliability engineering and safety engineering*, 2007, pp. 299-307
18. Fei Xing, Ping Guo, Michael R. Lyu, "A novel method for early software quality prediction based on support vector machine," in *Proceedings of the 16th IEEE International Symposium on Software Reliability Engineering*, 2005, pp. 213-222
19. Chen K, "Forecasting systems reliability based on support vector regression with genetic algorithms," *Reliability Engineering and System Safety*, Vol. 92, 2007, pp. 423-432
20. P F Pai, "System reliability forecasting by support vector machines with genetic algorithms," *Mathematical and Computer Modeling*, Vol. 43, 2006, pp. 262-274
21. Tian L, Noore A, "Dynamic software reliability prediction: an approach based on support vector machines," *Journal of Reliability, Quality and Safety Engineering*, Vol. 12, 2005, pp. 309-321
22. Eduardo Oliveira Costa, Silvia R. Vergilio, Aurora Pozo, Gustavo Souza, "Modeling software reliability growth with Genetic Programming," in *Proceedings of the IEEE International Symposium on Software Reliability Engineering*, 2005, pp. 1-10
23. Eduardo Oliveira Costa, Gustavo Souza, Aurora Pozo, Silvia R. Vergilio, "Exploring Genetic Programming and Boosting Techniques to Model Software Reliability," *IEEE Transactions on Reliability*, Vol. 56, 2007, pp. 422-434
24. Ferreira C, "Gene expression programming: A new adaptive algorithm for solving problems," *Complex Systems*, Vol. 13, 2001, pp. 87-129
25. K.K. Xu, Y.T. Liu and etal, "A novel method for real parameter optimization based on gene expression programming," *Applied Soft Computing*, Vol. 9, 2009, pp. 725-737
26. Ferreira C, *Gene expression programming: Mathematical modeling by an artificial intelligence*, Springer, 2006
27. Liliana T, Daniel S, "High energy physics event selection with gene expression programming," *Computer Physics Communications*, Vol. 178, 2008, pp. 409-419
28. Adil B, Mustafa G, "Gene expression programming based due date assignment in a simulated job shop," *Expert Systems with Applications*, Vol. 36, 2009, pp. 12143-12150
29. Vasileios K.K, Andreas S, "Efficient evolution of accurate classification rules using a combination of gene expression programming and clonal selection," *IEEE Transactions on Evolutionary Computation*, Vol. 12, 2008, pp. 662-678
30. http://www.gepsoft.com
31. S. Yamada, K. Tokuno, S. Osaki, "Imperfect debugging models with fault introduction rate for software reliability assessment," *International Journal of Systems*

Science, Vol. 23, 1992, pp. 2241-2252
32. C. Y. Huang, C. T. Lin, "Software reliability analysis by considering fault dependency and debugging time lag," *IEEE Transactions on Reliability*, Vol. 55, 2006, pp. 436-450
33. H. Pham, "An imperfect-debugging fault-detection dependent-parameter software," *International Journal of Automation and Computing*, Vol. 4, 2007, pp. 325-328
34. H. F. Li, Q. Y. Li, M. Y. Lu, "Software reliability modeling with logistic test coverage function," in *Proceedings of the IEEE International Symposium on Software Reliability Engineering*, 2008, pp. 319-320
35. C. Y. Huang, S. Y. Kuo, M.R. Lyu, "An assessment of testing-effort dependent software reliability growth models," *IEEE Transactions on Reliability*, Vol. 56, 2007, pp. 198-211
36. X. Teng, H.Pham, "A software Cost Model for Quantifying the Gain with Considering of Random Field Environments," *IEEE Transactions on Computers*, Vol. 53, 2004, pp. 380-384
37. Pham L, Pham H, "Software reliability models with time -dependent hazard function based on Bayesian approach," *IEEE Transactions on Systems, Man, and Cybernetics*, 2000, pp. 25-35
38. Pham L, Pham H, "A Bayesian predictive software reliability model with pseudo-failures," *IEEE Transactions on Systems, Man, and Cybernetics*, 2001, pp. 233-238
39. Singpurwalla, N. D, Soyer, R, "Assessing (software) reliability growth using a random coefficient autoregressive process and its ramifications," *IEEE Transactions on Software Engineering*, 1985, pp. 1456-1464
40. J. Musa, *Software reliability data*, Data & Analysis Center for Software, 1980
41. Yashwant K., Malaiya, Michael Naixin Li, "Software reliability growth with test coverage", *IEEE Transactions on Reliability*, Vol. 51, 2002, pp. 420-426
42. X. Cai, M. R. Lyu, "Software Reliability Modeling with Test Coverage Experimentation and Measurement with a Fault-Tolerant Software Project", in *The International Symposium on Software Reliability Engineering*, 2007, pp. 17-26