BS-GEP Algorithm for Prediction of Software Failure Series

Yongqiang ZHANG
Hebei University of Engineering/ The information and electricity-engineering institute, Handan, China
Email: yqzhang@hebeu.edu.cn

Jing XIAO and Shengjuan SUN
Hebei University of Engineering/ The information and electricity-engineering institute, Handan, China
Email: {xiaojing8785, sunshengjuan}@163.com

***Abstract*—This paper introduces GEP(Gene Expression Programming) fundamental. Aimed at prediction of software failure sequence, an improved GEP(GEP based on Block Strategy, BS-GEP) is presented, in which the population is divided into several blocks according to the individual fitness of each generation and the genetic operators are reset differently in each block to guarantee the genetic diversity. The algorithm complexity and convergence of BS-GEP is analyzed in the paper. Furthermore, BS-GEP is applied in the solution of prediction in software failure sequence. The simulation results show that the model found by BS-GEP, which is proved widely used for many other time series, is more accurate than the one of classic GEP.**

***Index Terms*—BS-GEP; Complexity Analysis; Convergence Analysis; Software Failure; Time Series Prediction**

I. INTRODUCTION

GEP(gene expression programming) is a newly proposed genetic algorithm with separate genotype and phenotype, which evolves 2 to 4 orders of magnitude faster than GP(genetic programming)[1].It has become an international new hotspot and been applied to many fields of data mining[2~4].

In this paper, we adopted BS-GEP(GEP based on block strategy) to predict software cumulative failure time (the next failure time) sequence, and clarified the research methods. We particularly analyzed the software testing case of Armored Force Engineering Institute[5,6], and completed BS-GEP model and its prediction. Lastly, we calculated the model reliability parameters and compared the short-term prediction ability with GEP model and other classic probability models. All what we did is to testify the feasibility and availability of model fitting and predicting by BS-GEP algorithm.

II. GEP FUNDAMENTAL

The implementation techniques of GEP include encoding, fitness function selection, genetic operators, transposition operators, recombination operators, and numerical variables. Now we just introduce the parts that will be improved in this paper.

A. Fitness Function Selection

Individuals that represent problem solutions need to be evaluated in all evolutionary algorithms. In GEP the solution is a computer program, or more exactly an expression. So the evaluation is to be completed by the fitting degree of data calculated by the expression and the training data. The following three ways are usually adopted[1].

$$f_i = \sum_{j=1}^{C_t}\left(M - \left|C_{(i,j)} - T_j\right|\right). \tag{1}$$

$$f_i = \sum_{j=1}^{C_t}\left(M - \left|\frac{C_{(i.j)} - T_j}{T_j} * 100\right|\right). \tag{2}$$

$$if\ \ n \geq \frac{1}{2}C_t,\ then\ \ f_i = n,\ else\ \ f_i = 1. \tag{3}$$

where M is the range of selection, and $C(i,j)$is the value returned by the individual program i for fitness case j (out of Ct fitness cases), and Tj is the target value for fitness case j, and n is the number of correct cases. Note that formula (1) and (2) can be used to solve any symbolic regression problem, but formula (3) to logic problems. In the design of fitness function, the goal is very clear that is to make the evolutionary direction of the system in accordance with requirements.

B. Mutation Operator

According to Candida's experiments[1], we know that the mutation operator is the most basic and most efficient operator among all genetic operators. Mutation operator can adjust parts of gene values of the individual encoding string, to make GEP search the local space and improve the local search ability. Besides, mutation operator can change encoding structure, to maintain the population diversity, and prevent or reduce premature and jump out of local optimal solution.

Mutation operator acts on a single chromosome, and tests randomly on each code of the chromosome. When

doi:10.4304/jsw.7.1.243-248

the mutation probability P_m meets a certain value(typically is 0.044), the code is re-generated. To ensure the same organizational structure, the code can be varied to any symbol of the function set and terminal set if mutation occurred in the head. Conversely, the code could be symbol of terminal set when in tail. It is can be predicted the structure of new individual generated through mutation is always correct.

III. BS-GEP ALGORITHM

A. Block Strategy

Genetic operators play an important role in the evolutionary results quality. If they are designed unreasonably, some extraordinary individuals generated in the early evolutionary could multiply rapidly and fill the population positions after several generations. So the local optimal solution, also called premature phenomenon is coming. Another way, the algorithm is close to convergence in the later stage of evolutionary, and the fitness difference between individuals is smaller. So the potential of optimization reduced, and the result tends to purely random selection and hardly a global optimal solution. In this paper, we adopt blocking population to make sure the population diversity of each generation. The scheme is as follows.

Step 1, suppose f_i , $i = 1,2,\cdots,n$ is the fitness of individual x_i ,order individuals by f_i ,a block of 20, the population is divided into m blocks B_j, $j = 1,2,\cdots,m$ (number of B_m is permitted less than 20), $f_{j-\max}$ (the fitness maximum of B_j)is less than the fitness minimum of B_{j+1} ($f_{(j+1)-\min}$),that is $f_{j-\max} < f_{(j+1)-\min}$;

Step 2, as in the individual fitness of each block are very close, linear or power function transformation method is adopted for scaling the fitness function, and then individuals are selected to genetic operations follow the roulette wheel or tournament method;

Step 3, since the individuals' goodness differences in the blocks, mutation operator is reset respectively to each other block, like a smaller mutation probability set to individuals in the block with a high goodness and larger to low goodness, in order to ensure high population diversity.

In view of this scheme, we need to redesign fitness function and improve mutation operator.

- Fitness Function

On GEP-based symbolic regression problems, the two evaluation models proposed by Candida own their inherent shortcomings[7]. In statistics, it is more usually to employ R^2(Coefficient of Determination) to evaluate the fit degree of two sets of data. The calculation formula is as below.

$$R^2 = 1 - SSE / SST . \qquad (4)$$

in which $SSE = \sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2$, $SST = \sum_{i=1}^{n}\left(y_i - \bar{y}\right)^2$, y_i is the real observed value, and $\bar{y}_i$ is the average one of observed values, and $\hat{y}_i$ is the regressed value. SSE is residual sum of squares of the observed values and the regressed values, and summation of SSE and SSR(regression sum of squares $\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2$).

So, we design the fitness function like this:

$$f = n \times 100 \times R^2 .(n \text{ is the sample size}) \qquad (5)$$

$\because SSE < SST$, $\therefore 0 < R^2 < 1$. It can be known the range of f is $(0, n \times 100)$. When the individual fitness of each block are very close, fitness of the next generation can hardly be improved obviously, which would lower evolutionary efficiency. So we make fitness linear amplified by multiplying the factor $n \times 100$ (n is the sample size).

- Mutation Operator

We set dynamic mutation probability in this paper, in order to make mutation operator self-adaptive. Mutation probability function is designed as follows.

$$P_{im} = P_M \times e^{\frac{\bar{f}_i + \left(\bar{f}_i\right)_{\max}}{\bar{f} - C}} . \qquad (6)$$

where P_{im} is mutation probability of the current block, and P_M is a constant set before evolutionary with a range of (0, 0.15), and $\bar{f}_i$ is average fitness of the current block and its maximum is $\left(\bar{f}_i\right)_{\max}$, while C= $n \times 100$ is the maximal fitness.

It can be easily learned from formula (6) that P_m of each block is in inverse ratio to the average fitness, also to generations (or $\left(\bar{f}_i\right)_{\max}$). The value range of P_m is $\left[0, \frac{1}{e} P_M\right]$.

B. BS-GEP Algorithm Description

Every individual mutates on a fixed probability in the classic GEP algorithm, which affect population diversity seriously. We brought out a scheme based on block strategy to the mutation operator. BS-GEP algorithm structure is shown in Figure1.

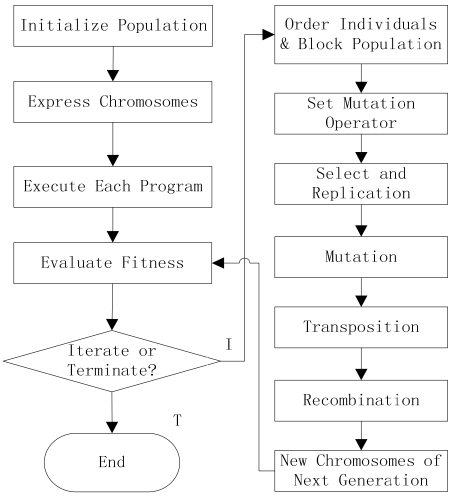


Figure 1. Flow Chart of BS-GEP.

From Figure 1, it is apparently that the new algorithm adds the mutation rate reset in every generation contrast to the classic GEP.

The pseudocode for BS-GEP is shown as follows, where $P(t)$ represents the tth population, and the initial population $P(0)$ is designed randomly, and Pim is the mutation probability and $Q= P(t)$ or Φ:

Procedure BS-GEP:
{
Begin
t =0;
Initialize(*P*(*t*));
Evaluate(*P*(*t*)):
While(not terminate condition)*Do*
Begin
Order individuals according to *fi*;
Block(*Pi*(*t*)) into *Bj*;
Set $\wp mj(t)$ (that is *Pim*);
Pc(*t*)= *Crossover* (*P*(*t*));
Pm(*t*)=*Mutation*(*Pc*(*t*));
Pt(*t*)=*Transposition*(*Pm*(*t*));
Pi(*t*)=*Recombination*(*Pt*(*t*));
Evaluate(*Pi*(*t*));
P(*t*+l)=*Select*(*Pm*(*t*) $\cup$*Q*);
t=*t*+1;
End
End
}

C. BS-GEP Complexity Analysis

Theorem 1: the algorithm complexity is $O(P \times G \times n)$, in which *P* is population size, *G* is the total generations, *n* is the sample size.

Demonstration: in the algorithm, the calculative complexity of population initialization from n samples is $O(n)$; the fitness of each individual need to be calculated, so the calculative complexity of population fitness is $O(P \times n)$; as the maximum of generations is *G*, so the algorithm complexity is $O(P \times G \times n)$.

D. BS-GEP Convergence Analysis

Theorem 2: the probability of convergence to the optimal solution using BS-GEP is less than 1.

Demonstration: all possible status of population is divided into two kinds, one is S_0 including the optimal individual, and another is S_n that does not have the optimal individual. $S = S_0 \bigcup S_n$, $S_0 \bigcap S_n = \phi$. Wishing to demonstrate the stable probability that P_1 runs to S_0 is less than 1, we take proof by contradiction: Assuming the probability is equal to 1, the probability that P_1 runs to S_n is 0, that is $\lim_{t \to \infty} P\{P_t \in S_n\} = 0$. In the process of BS-GEP evolutionary, if the population mutate from a status $i \in S_m$ to another status $j \in S_m$, and the mutation probability is m_{ij} ,the stochastic matrix $M = \{m_{ij}\}$ is the population status transfer matrix of BS-GEP.

M is a stochastic matrix, and $m_{ij} = P_m^{H(i,j)}(1 - P_m)^{1-H(i,j)} > 0$ ($H(i,j)$ is the Hamming distance between i and j), so M is positive definite. At the moment t the probability that the population is in status j is $P_j(t) = \sum_{i \in I} P_i(0) \cdot m_{ij}^t$, $t = 0,1,2, \ldots$. Learning from the characteristics of the homogeneous Markov chains[8],the stable probability distribution of $P_j(t)$ is independent with that of initial, that is $P_j(\infty) = P_i(\infty) m_{ij} > 0$. At this moment $j \in S_m$, that is to say, j is the status of S_n .So $\lim_{t \to \infty} P\{P_t \in S_n\} > 0$ .This is contradictious with the previous assumption. Therefore, *Theorem 2* is tenable.

It can be known from the above analysis that, the problem solving based on BS-GEP has convergence to the global optimum in probability, but not the strong convergence to the global optimum. So it can not rule out the possibility of convergence to local optimum.

Ⅳ. SOFTWARE FAILURE TIME SERIES ANALYSIS BASED ON GEP AND BS-GEP

The data series selected are the former 16 data of the software testing case in Armored force Engineering Institute, which are given in TABLE I as follows.

TABLE I.
FAILURE DATA SERIES

x	1	2	3	4	5	6	7	8
t_i	1	1	1	5	4	24	6	14
T_i	1	2	3	8	12	36	42	56
x	9	10	11	12	13	14	15	16
t_i	33	1	30	22	13	22	77	7
T_i	89	90	120	142	155	177	254	261

where, $t_i = T_i - T_{i-1}, i = 1,2,\cdots,16$ and $T_0 = 0$ (t_i is the mean time between failures(*MTBF*), T_i is the cumulative time of failures, also means the next failure time).In this paper we have formed GEP model and BS-GEP model just on T_i . Parameters of the algorithms in the test are set as shown in TABLE II.

TABLE II.
PARAMETERS SETS OF GEP & BS-GEP

Parameters	Span Solution
Population Size	60
Maximum of Generations	1000
Gene Number	5
Head Length	6
Function Set(*F*)	+、-、×、/、^
Terminal Set(*T*)	$\{t, 0, 1, \cdots, 9\}$
Select Operator	roulette wheel
Mutation Operator	0.044
Transposition Operator	0.1
Recombination Operator	0.3
Fitness Function	GEP with Formula (2)(M=100) BS-GEP with (5)
Terminal condition	Maximum of Generations

(Note: To make algorithms are more suitable for software reliability modeling, in consideration of the software reliability growing characteristic, we add exponential function to *F*, which also owns growing feature. Both of the fitness maximums are 1600.)

Run the evolutionary program in the mixed environment of *VC++* and Mathematica. After 1000 generations of evolution, we get preferable adaptive models and their structures expressions are as follows.

$$\begin{aligned} T_{GEP}(x) &= -0.592059 + 2.67892\,x^2 \\ &+ \frac{0.386808\,(0.203589 + x)}{0.250649 + x} - 1.54051\,x(1+x) \\ &+ (0.559069 + x)e^{-x} \end{aligned} \tag{7}$$

$$\begin{aligned} T_{BS-GEP}(x) &= -0.162737 + 10^{0.272627\,(-0.541917 + x)} + 10^{0.086973\,x} \\ &- \frac{6.11894}{x} + x^2 + \frac{(0.037978 - x)(0.863582 + x)}{x} \\ &- 0.588002\,x^2 e^{-x} \end{aligned} \tag{8}$$

A. The Calculation of Software Reliability Model Parameter--MTBF

The prediction of T at the 17th failure by the models (7) and (8) are T_{GEP17} =302.6031, $T_{BS-GEP17}$ =300.7515, while the real value is 300. Accordingly, t(*MTBF*) at the moment T_{17} are $MTBF_{GEP}$ =41.6031, $MTBF_{BS-GEP}$ =39.7515, while 39 is the real result. In TABLEIII the appraisal results on t_{17} and T_{17} of GEP and BS-GEP models are compared with several traditional reliability models.

TABLE III.
CALCULATION RESULT OF *MTBF*

Models	*MTBF*	Next Failure Time
GEP Model	**41.6031**	**302.6031**
BS-GEP Model	**39.7515**	**300.7515**
Exponential Model	90.5000	351.5000
J-M Model	108.5019	369.5019
G-O(NHPP)	50.2572	311.2572
Moranda Model	72.4638	333.4638
S-W Model	126.7990	342.7990

From the table above, we can see that the distances of *MTBF* and Next Failure Time values between the result by these traditional models and the real result are much larger. However, the results calculated by GEP and BS-GEP models are more suitable and accurate, and the BS-GEP model is the best. All of above can testify that the software reliability of the new models represent better than other traditional models on one-step-ahead prediction capability.

B. Failure Rate Curve

Having calculated the *MTBF* value, the current failure rate of the software system can be brought out by $\lambda = 1/MTBF$. So the current reliability function is $R(t) = e^{-\lambda t}$. By models (7) and (8) the initial failure rates are 0.86592 and 0.90668 separately, and the current failure rates at T=261 are 0.0240367 and 0.0251563 respectively. The failure rates curves of the two models are shown in Figure 2 and Figure 3.

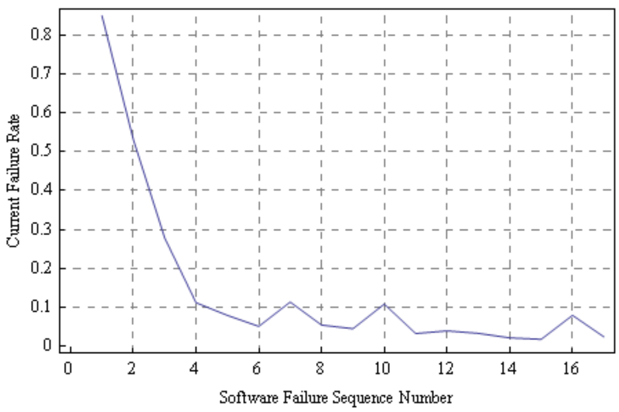


Figure 2. Failure Rate Curve of GEP Model

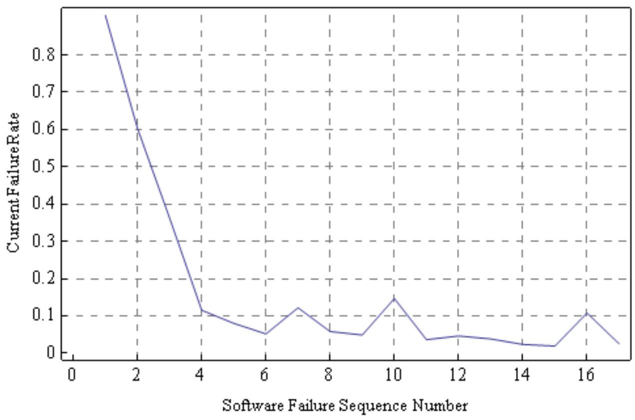


Figure 3. Failure Rate Curve of BS-GEP Model

From above figures, it is learned that the change tendency of software failure rates from the two models is similar, and tends to monotone decreasing as a whole.

C. The Short-Term Prediction Capability Comparison of Models

In order to testify the prediction capability of new models, we adopt the short-term range error (*SRE*) in the reference[9] for scaling the short-term prediction capability. Its formula is shown as follows.

$$SRE = \frac{\sum_{i=1}^{n-1} \frac{|x_r(i+1) - x_p(i+1)|}{x_r(i+1)}}{n-1} . \qquad (9)$$

where $x_r(i+1)$ represents the real value of next *MTBF* and $x_p(i+1)$ is the next *MTBF* predicted by the model using the former i failure data. The smaller the *SRE* value is, the stronger and better models' short-term prediction capability will be, meanwhile, the more accurate the one-step-ahead prediction capability will be gotten.

In the view of our testing case above, we can get the prediction results of failure data series from the 13th point to the 17th one, which are calculated by the seven models above. Their calculated results and the *SRE* values are given in TABLEⅣ.

TABLE IV.
PREDICTION RESULTS AND *SRE* VALUES

Prediction Results	Exponential Model	J-M	G-O (NHPP)	Moranda
The 13th point	50.0833	211.1405	34.7047	37.8788
The 14th point	58.1540	84.0211	28.6110	30.3030
The 15th point	66.6430	70.0565	30.2247	37.0370
The 16th point	79.1330	81.5659	75.1856	55.8659
The 17th point	90.5000	108.5000	50.2572	72.4638
SRE	2.3520	3.1275	2.5214	2.2198
Prediction Results	S-W	**GEP**	**BS-GEP**	
The 13th point	78.5848	**30.1566**	**26.0533**	
The 14th point	38.3494	**46.3531**	**42.0821**	
The 15th point	48.1770	**55.8263**	**51.7706**	
The 16th point	124.0762	**12.5763**	**9.26688**	
The 17th point	126.7990	**41.6031**	**39.7515**	
SRE	5.0278	**0.7130**	**0.5175**	

Comparing with these short-term prediction results and the *SRE* values, we can draw the conclusion that $SRE_{BS\text{-}GEP} < SRE_{GEP} < SRE_{Moranda} < SRE_{\text{Exponential Model}} < SRE_{G\text{-}O} < SRE_{J\text{-}M} < SRE_{S\text{-}W}$. It is these values that prove the short-term prediction capability of new models much more superior to others. So their predictive effectiveness is testified.

D. Model Simulation

Figure 4 and Figure 5 give out the cumulative time simulation figures of the two models, of which it can be easily learned that both GEP and BS-GEP models fit failure data quite well. GEP executes to the 900th generation when program finds the optimal solution, and the fitness is 1182.285551 and the time-consume is 10.5seconds. But to BS-GEP, the optimal one is found just at the 350th generation with fitness value of 1576.162104, and it only takes 3 seconds. It is very clear that the BS-GEP model has a higher predictive efficiency and can fit better than GEP. (Fitness represents error between the predictive value and the real one.)

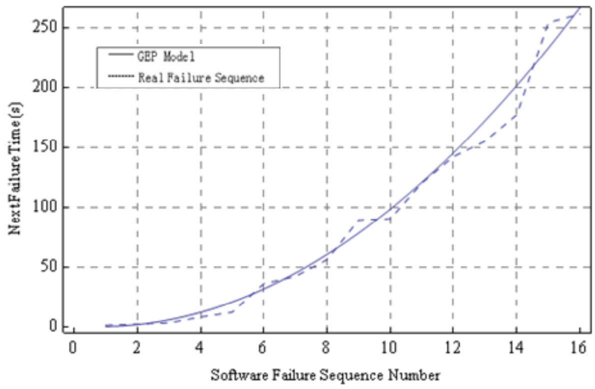


Figure 4. Simulation Result of GEP Model

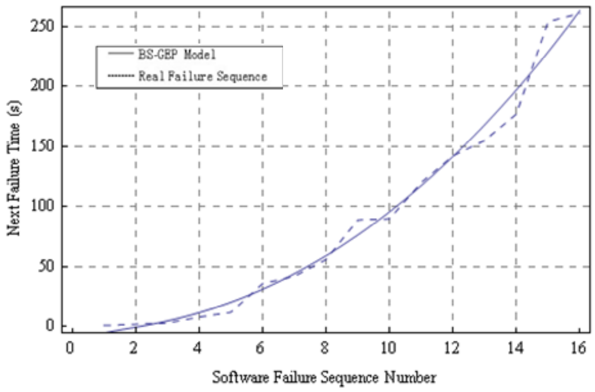


Figure 5. Simulation Result of BS-GEP Model

In addition, we have created the reliability model with software *MTBF* series, as well as the error statistical data of NTDS (Naval Tactical Data System) of America Navy tactical systems as well as the error statistical accumulative failure data series of SYS1, SYS2 and SYS3[5] from Musa in 1979. We also analyzed and appraised some criteria, which can all testify the applicability of BS-GEP. All what we have done have testified the feasibility and availability of this algorithm on both theory and applications.

Ⅴ. CONCLUSIONS

Using GEP for prediction, it is unnecessary to learn the target function and the causal relationship among various factors, and it can predict precisely just with enough experiments or experimental data supplied[10]. The innovation of this paper is that a new GEP algorithm based on Block Strategy (BS-GEP) was proposed, by which a model with high predictive accuracy and fitting degree has been established. Having experimented and analyzed several cases, we can find that BS-GEP model is better than the classic GEP model, as well as the several other traditional probability models, also faster than GEP on speed of solutions.

ACKNOWLEDGMENT

The authors thank the National Natural Science Foundation of Hebei Fund (F2010001040) for supporting this project.

REFERENCES

[1] C. Ferreira, “Gene Expression Programming: A New Adaptive Algorithm for Solving Problems,” *Complex Systems*, vol. 13(2), pp. 87-129, 2001.

[2] C. Ferreira, “Discovery of the Boolean Functions to the Best Density-Classification Rules Using Gene Expression Programming,” Proceedings of the 4th European Conference on Genetic Programming, *Berlin: Springer-Verlag*, pp. 51-60, 2002.

[3] L. Duan, C.J. Tang and J. Zuo, et al, “An Anti-noise Method for Function Mining based on GEP,” *Journal of Computer Research and Development*, vol. 41(10), pp.1685-1689, 2004.

[4] D. Satchidananda and Sung-Bae Cho, “Multi-objective Classification Rule Mining Using Gene Expression Programming,” Third 2008 International Conference on Convergence and Hybrid Information Technology, *ICCIT*. vol.27, pp.754-760, 2008.

[5] G.W. He, “The Software Reliability Growth Experiment and Software Reliability Testing Platform,” *Software engineering technology*. 1997(04).

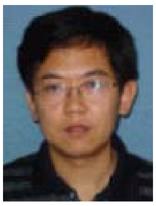
[6] Y.Z. Gong and Q.H. Zhou, “A Software SRTP Testing Report,” *Armored force Engineering Institute*, 1995.

[7] J. Zuo, “Research of GEP Core Technology,” *Doctoral Dissertation in Sichuan University*, 2004.

[8] J. Qin, L.S. Kang and Y.P. Chen, “The Convergence Analysis and Algorithm Improvement of Computation Algorithm,” *Computer Engineering and Applications*, vol.19(179), pp. 91~92, 2003.

[9] Michael R.Lyu, *Handbook of Software Reliability Engineering*. McGraw-Hill publishing, 1995, ISBN 0-07-039400-8.

[10] X.S. Qian and C.H. Yang, “Improved Gene Expression Programming Algorithm Tested by Predicting Stock Indexes,” *CAAITransaction on Intelligent Systems*.vol.5(4), pp.303-307, 2010.



Yongqiang ZHANG (1966-), professor of Hebei University of Engineering who is studying on software reliability engineering and so on.

Jing XIAO (1987-), candidate for master degree who is studying on the GEP Algorithm and the software reliability modeling.

Shengjuan SUN(1981-), researcher(master) who is studying on software reliability engineering and so on.