Research on Software Reliability Assessment with Optimum Reserved Strategy Genetic Programming

SUN Sheng-Juan, ZHAO Jing, CHEN Hua-Shan

Abstract

The software failure data can be analyzed by Genetic Programming (GP for short). Meanwhile, the Particle Swarm Optimization (PSO for short) algorithm is taken to find the rational parameters during the dynamic modeling and finally the optimized model structure of software reliability can be obtained, all of which can make sure the accuracy of software reliability forecasting can be dramatically improved. This paper has also analyzed and validated Genetic Programming has the convergent characteristics, which make use of optimum reserved strategy. And the results testify that Genetic Programming can meet the best solution needs of the variety disciplinary of the failure behaviors, which can further testify Genetic Programming is able to apply for software system testing as well as guarantee the availability of the data. The application inconsistency of new model shows obviously comparing with other traditional models, which has much more theory research merit and practical application meaning to the analysis of software reliability models as well as the forecast of software failure behaviors.

Keywords: Software Reliability Model, Genetic Programming (GP), Failure Data, Convergence

1. Introduction

Various software reliability prediction models were built in the process of the development of software reliability, using the failure data collected in the test procedure to estimate the software reliability. Due to the traditional software reliability models are based on some assume, and the assume condition is hard to proving. So when the assume condition tally with the software attribute, the forecast result is very accurate; otherwise the result will go flooey, meanwhile the forecast results of different models will differ discordance. This will lead to many problems in the practical application, such as which model to choose, and whether the forecast result is credible or not. For the few past years, the researchers have been seeking reliable prediction models which abandon the model assume and with general applicability. Some researchers have been researching the relationship between the software failure data set[1-2], software engineering practical activity and the software attribute. Quantizing the influence on failure data set, caused by some of software attributes and engineering practical activity, improved prediction precision. But it is difficult to quantize because the engineering practical activity and software attribute which may lead the software disabled are so many, and the effect degree is different. So in this case, we can analyse the software failure data as time series of time between software failure or software failure frequency changing with time to explore the law of software failure, so we can avoid the problem to quantize so much influencing factor, and express the synthetic action as well.

Genetic Programming (GP for short) simmers down different fields’ issues as seeking computer program which satisfy the given constrained to discover the issue. That is to say, using the program we can find the optimal solution among the programming space. If we adopt the Genetic Programming (GP) to fit the data, it needs only a precision, not the equation’s structure. After Evolutionary Computation, we can obtain the equation. It has been widely applied in every field of engineering technology, and the effect of fitting and forecasting is very outstanding. Considering this, this paper uses GP to analyze the sequence of the software failure time, to seek the real failure data’s best fitting equation, and to imitate the failure process. And then based on this we can forecast software failure, building the software reliability GP model to evaluate the software reliability.
2. Software reliability GP model

2.1. parameters of the GP model

Parameters of the evolutionary algorithm in the test are as shown in Table 1.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>function set (F)</td>
<td>+, -, ×, /, log, sin, exp, cos, tan, sqrt</td>
<td>choose method</td>
<td>optimum reserved</td>
</tr>
<tr>
<td>termination character set (T)</td>
<td>independent variable x</td>
<td>terminal condition</td>
<td>the max evolution time</td>
</tr>
<tr>
<td>group size</td>
<td>30</td>
<td>max evolution algebra</td>
<td>100</td>
</tr>
<tr>
<td>probability of crossover</td>
<td>0.90</td>
<td>max depth of crossover</td>
<td>7</td>
</tr>
<tr>
<td>mutation rate</td>
<td>0.05</td>
<td>max depth of mutation</td>
<td>7</td>
</tr>
<tr>
<td>initial group generation method</td>
<td>grow method</td>
<td>max depth of initial individual</td>
<td>5</td>
</tr>
</tbody>
</table>

In order to explain GP algorithm’s feasibility in solving software failure data sequence modeling, and to verify that using this algorithm in software reliability modeling can get the optimal solution, which could reflect the software failure mechanism. The detailed argument of the effect on astringency, caused by the GP algorithm parameters’ setting, will show as follow.

2.2. the convergence analyses about GP with Optimum Reserved Strategy

**Theorem 1.** (GP’s astringency): take \( p_m > 0 \) as the mutation rate, the growth method as approach, and the optimum reserved strategy, then the GP algorithm solving optimizing issue convergence, and the convergence has nothing to do with the initial group.

**Theorem proving:** If we consider every generation of the GP algorithm evolution process as a status, then we can test the whole evolution process as a random process, and analyze its convergence with Markov chain. First let’s introduce the concept of realizable history.

If a strategy \( \pi = \{ \pi_0, \pi_1, \ldots \} \in \Pi \), for any history \( h_t = \{ i_0, a_0, i_1, a_1, \ldots \} \in H_t \), \( P_{H_t}^\pi [h_t | h_0] > 0 \). That is to say, under the strategy \( \pi \) proce, the probability of event \( h_t \) is positive. Then \( h_t \) can be called the realizable history under the strategy \( \pi \). Namely, that is under the strategy \( \pi \) starting from \( i_0 \), after taking action \( a_0 \) the status transfer to \( i_1 \), then taking action \( a_1 \), go on until to the moment \( t \)’s status transfer to \( i_t \). The whole event’s probability measure induced by strategy \( \pi \) is not 0, then the history is a realizable history under the strategy \( \pi \).

The optimal action set is defined as follow. To every status \( i \) in \( S \), the following is \( i \)’s optimal action set.

\[
A'(j) = \arg \max_{a \in A(i)} \left\{ r(i,a) + \beta \sum_{j \in S} p(j|i,a) \psi^*(j) \right\} \tag{1}
\]

The necessary and sufficient condition of the strategy \( \pi = \{ \pi_0, \pi_1, \ldots \} \in \Pi \) defined as an optimal strategy is that for \( \forall t \geq 0 \), if history \( h_t = \{ i_0, a_0, i_1, a_1, \ldots \} \in H_t \) is realizable under strategy \( \pi \), then \( \pi(a|h_t) = 0 \) [3] when \( a \in A(i_t) - A'(i_t) \).

From the above theorem we can conclude that GP retains the best individual in every generation to next generation, that is \( A(t+1) = \max \{ A(t), A_{best} \} \), in this \( A_{best} \) indicate the best individual of the
At generation, and \( A(t) \) the best of \( t \) generation. Therefore, the GP with optimal strategy remains to be a Qi Markov chain. It means that the probability is greater than 0 transferring from any status to the other that contains the optimal solution, and the probability is 0 transferring from the status that contains the optimal solution to any one that don’t. That’s to say, GP with optimal strategy incomplete coverage, always converges to the optimal solution with probability 1.

From the above analysis we can conclude that the genetic programming algorithm is feasible in solving software failure data array model, and modeling with the algorithm and its parameter in this paper can get the optimal solution reflecting the software failure mechanism.

### 2.3. Instance analysis

Using GP algorithm, we respectively made evolution modeling on the failure accumulated time sequences from some software test case [4] of Armoured Engineering Institute (taking the first 16 of data only) SYS3[5] offered by Musa data center, and error statistical data of the development and test process(data of 1–30 groups) of Naval Tactical Data System from U.S. Navy Fleet Computer Programming Center.

Running the evolution procedure at MATLAB6.5 environment, we got adaptive model for the above three sets of failure accumulated time, after 100 times evolutionary computation.

\[
T_{AEI} = f(x) = x \cdot \sqrt{x} \cdot \sqrt{x - \cos[\tan x \cdot (x - \ln x + \sin(x)])}
\]  
\[ (2) \]

\[
T_{sys3} = f(x) = x \cdot \sqrt{x} \cdot \ln x + \sin x
\]  
\[ (3) \]

\[
T_{NTDS} = f(x) = x + \sqrt[(x - \sin([x - e^x]/\cot x)] \cdot \tan x \cdot \ln x
\]  
\[ (4) \]

### 2.4. Optimization model with PSO

The software failure characteristic function can be regarded as multiple objective function awaiting to be optimized, and the PSO algorithm has many advantages such as good stability, easy encoding and strong optimizing skill. So we adjust and optimize the software failure characteristic function, using the PSO algorithm, to increase the predicted precision.

Here \( f \) is the model built by using genetic programming, and the model structure is defined as \( g = ax + b \) optimized by the PSO algorithm. Then we can gain the model optimized from GP model, which is corresponding with the above three failure data sets, by the PSO.

\[
T_{AEI-PSO} = f(x) = 1.0112 \cdot x \cdot \sqrt{x} \cdot \sqrt{x - \cos[\tan x \cdot (x - \ln x + \sin(x)]\}} - 2.9678
\]  
\[ (5) \]

\[
T_{sys3-PSO} = f(x) = 0.9993 \cdot x \cdot \sqrt{x} \cdot \ln x + \sin x + 2.9964
\]  
\[ (6) \]

\[
T_{NTDS-PSO} = f(x) = 1.1752 \cdot x + \sqrt[(x - \sin([x - e^x]/\cot x)] \cdot \tan x \cdot \ln x + 2.9988
\]  
\[ (7) \]

Meanwhile, the initial and final status for PSO of Armoured Engineering Institute are showed as follows in figure1 and figure2. The tiny particles shown in the figure are parameter a and b, through evolution computation and at last converge to the optimal solution suits to the model.

Using PSO algorithm optimizes the models’ parameter, and the models are the newly built software reliability ones, which are built on the software failure data sets in 2.3. The errors before and after optimization are listed in Table 2. (If the truthful data are \( y \), the GP model \( f \), and PSO-GP model \( g \),

then the errors are defined as follow: \( \text{error}_1 = \sum_{i=1}^{m} (f_i - y_i)^2 \), \( \text{error}_2 = \sum_{i=1}^{m} (g_i - y_i)^2 \). Here \( i \) is the sequence
number, and \( m \) is the data’s total number.) Figure 3 shows the MATLAB simulation of the corresponding model of the three data sets.

![Figure 1. The initial status of the PSO evolution](image1.png)

![Figure 2. The final status of the PSO evolution](image2.png)

**Table 2. The error compare**

<table>
<thead>
<tr>
<th>error</th>
<th>Failure data of Armoured Engineering Institute</th>
<th>Failure data of Musa-SYS3</th>
<th>Failure data of NTDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{error}_1 ) (GP)</td>
<td>1.6873e+003</td>
<td>5.2278e+007</td>
<td>4.5545e+004</td>
</tr>
<tr>
<td>( \text{error}_2 ) (PSO-GP)</td>
<td>1.4761e+003</td>
<td>5.1997e+007</td>
<td>2.0180e+004</td>
</tr>
</tbody>
</table>
3. The evaluation of the new model

The software reliability model needs to resolve two problems, the improvement of software development process and the test of software reliability. Relatively, the research on software reliability model and its application focuses on the following two aspects: phase forecast model and prediction model. The former stand-by the software product itself and its development process measure the software reliability under the consequence of the failure data unknown. Such models have significant meaning in improving software development process, conducting software test, and enhancing software reliability. The later focus on the future, forecasting the software reliability, including the software failure at the moment and the next failure time. The software reliability GP model built in this paper is belonging to the later. So, we evaluate the model from accuracy of MTBF and the ability of short-time predictive.

3.1. MTBF prediction accuracy comparison

Here taking the test case’s failure data of Armoured Engineering Institute for example, we calculate the forecasting accumulative total failure time, that is the next failure time $t_{17}$, is 292.1355, and the actual datum is 300. Accordingly, the mean time between failure of the moment $t_{16}$ is 31.1355, and the actual datum is 39. The assessment result of the mean time between failure and the next failure moment to the 17th moment applying the failure data in Table 1 to several traditional models and the GP model, is listed in Table 3.

![Figure 3](image.png)  
*Figure 3* the model’s simulation result

<table>
<thead>
<tr>
<th>evaluation model</th>
<th>MTBF</th>
<th>next failure time</th>
<th>evaluation model</th>
<th>MTBF</th>
<th>next failure time</th>
</tr>
</thead>
<tbody>
<tr>
<td>exponential model</td>
<td>74.3098</td>
<td>335.3098</td>
<td>Moranda</td>
<td>72.5822</td>
<td>333.5822</td>
</tr>
<tr>
<td>J-M model</td>
<td>108.5019</td>
<td>369.5019</td>
<td>S-W model</td>
<td>infinite</td>
<td>Infinite</td>
</tr>
<tr>
<td>G-O (NHPP)</td>
<td>63.4742</td>
<td>324.4742</td>
<td>GP new model</td>
<td>31.1355</td>
<td>292.1355</td>
</tr>
</tbody>
</table>

From the data in the above table, we can conclude that the MTBF and the next failure moment of the traditional models are far way from the actual result, while the software reliability calculation result gained from the GP model is much better. And this proves software reliability GP model is superior to the several traditional models in one step predictive power.
3.2. short-time predictio ability comparison

In order to inspect the assessment ability of the model based on the GP algorithm, apply the MTBF time array listed in 3.1 to the GP new model and five traditional software reliability models such as G-O model, J-M model, to compare their short-time predictive power. We use SRE, mentioned in application literature [6], to measure models short-time predictive power, judging GP new model and several traditional models’ short-time predictive power from predicted values and actual observed values. The following is the definition of SRE:

\[
SRE = \frac{\sum_{i=1}^{n} |x_r(i+1) - x_p(i+1)|}{n-1}
\]

We Built the model with the preceding failure data until the moment \( t_{i+1} \), to calculate the next failure interval time \( x_{i+1} \) and the next failure moment \( t_{i+1} \), here \( t_{i+1} = t_i + x_{i+1} \). \( x_r(i+1) \) indicates the next failure interval time observed actually, and \( x_p(i+1) \) means the predicted next failure interval time of the model built with the preceding failure data. The smaller the value of SRE, the better the short-term prediction power of the model, and one step forecasting is more accuracy.

In order to compare the short-time predictive power of the above six models, we calculate them respectively, and get the next failure interval time from the first few failure interval data. As to the test case for this paper, the predictive result of the failure data and the value of SRE are listed in Table 4.

<table>
<thead>
<tr>
<th>sequence number</th>
<th>J-M model</th>
<th>G-O model</th>
<th>GP model</th>
<th>PSO-GP model</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>28.1</td>
<td>21.6078</td>
<td>26.31</td>
<td>30.92</td>
</tr>
<tr>
<td>28</td>
<td>34.8</td>
<td>35.7585</td>
<td>35.06</td>
<td>41.19</td>
</tr>
<tr>
<td>29</td>
<td>45.6</td>
<td>46.9422</td>
<td>20.29</td>
<td>23.86</td>
</tr>
<tr>
<td>30</td>
<td>66.4</td>
<td>50.3197</td>
<td>52.47</td>
<td>61.65</td>
</tr>
<tr>
<td>SRE</td>
<td>2.5286</td>
<td>2.1234</td>
<td>1.6181</td>
<td>1.9016</td>
</tr>
</tbody>
</table>

The result of the Compare short-term forecast ability metrics is \( SRE_{GP} < SRE_{PSO-GP} < SRE_{G-O} < SRE_{J-M} \), it explains that the new model got from using GP and evolution is superior to the two traditional models J-M and G-O in short-term predictive power.

4. Conclusion

This paper used GP to forecast the software failure data sequence, and analyzed parameters’ setting of the algorithm, proving in theory that the GP algorithm which used the optimum reserved strategy having astringency, and that we can get the optimal solution which could satisfy the failure action’s change rules, and practical application. It has improved the algorithm’s forecasting precision to select particle swarm optimization to do parameter estimation in the building process of dynamic models, and finally got an optimizing model structure of software reliability. At last, applied the new model to actual data set, and the result shows that the new model having preferable universality and higher forecast precision, proving that the new model’s feasibility and effectiveness further. The usage of software reliability model has certain practical application of significance in analyzing and forecasting the software’s fault behavior.

Sometimes the influence of population size or the setting of genetic operation and so on, could slow down the speed to get the optimal solution, namely the convergence rate, and the operation time is too long. At the same time, there is certain irrationality in PSO algorithm caused by optimizing the model structure. And that leads to the final model’s forecast effect is not so good, for example the case in Table 4 PSO-GP model the SRE value wax. So the further research will focus on speeding convergence and setting up reasonable optimizing model structure, designing and improving the
combination of GP and PSO algorithm, to make it more effectively while analyzing software reliability model and forecast software’s fault behavior.

5. Acknowledgement

Thanks to Natural science foundation grants of Hebei province (No.F2010001040) and all the teachers’ guidance.

6. References


