International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 9 (2015) pp. 22919-22926

http://www.ripublication.com

# Application of Gene Expression Programming In Predicting Software Reliability

**Shailee Lohmor[a], Dr. B B Sagar[b]**
[a]*Research and Development Centre, Bharathiar University, Coimbatore, India*
[b]*Assistant Professor, Birla Institute of Technology, Mesra- Ranchi, India*

## Abstract

Major component in quality of software is reliability of software. It refers to the ability of a component or system for performing its needed functions under mentioned conditions for particular time period. Mostly all software industries desire to generate software which is error free. Software Reliability Growth Models (SRGMs) are assisting the software industries for developing software which is reliable and error free. This paper aims to combine techniques of machine learning with algorithm of Gene Expression Programming (GEP) namely Adaboosting and explore the GEP for modeling the defect function of predicting on the basis on test coverage. It was analyzed that predictive capability of GEP models was better than other models. When predicted values are compared with traditional SRGM models, Non-parametric SRGMs (NPSRMs), Support Vector Machine (SVM), Genetic Programming (GP), and Artificial Neural Network (ANN), the proposed model outperform well.

**Keywords:** Software Reliability; Software Reliability Growth Model(SRGM); Gene Expression Programming (GEP); Adaboosting; Genetic Programming (GP)

## Introduction

Reliability of software is an integral part in quality of software along with performance, functionality, serviceability, documentation, usability and maintainability. Software reliability is referred as the probability that a system would proceed to function successfully for a particular period in particular environment. Study of reliability of software can be classified into three parts such as measurement, modeling and enhancement. Modeling of software reliability has matured to the extent that meaningful outcomes can be acquired by adopting a model appropriate for the problem stated. Numerous models are there to solve the stated problems but there is no single model can identify a needed quantity of characteristics in the software.

Abstractions and assumptions needs to be made for simplifying the problems. Measuring the reliability of software is naive [1].

There are different approaches that can be applied for enhancing the software reliability, thus, it is complicated to balance budget and development time with reliability of software [2]. The main reasons of poor reliability are determined in an integration of non-compliance with good coding and architectural practices. Such non-compliance could be identified by gauging the static attributes of quality of an application. In order to assess the static attributes that underlie on reliability of application that gives an estimation of business risk level and likelihood of potential failures in the applications and so on [3] [4]. To assess reliability of the software it is required to checks the following technical attributes and best practices in the software such as architecture practices of application, coding practices, algorithms of complexity, dirty programming, compliance with structured programming and object-oriented best practices, complexity in the practices of programming, multi-layer design compliance and error and exception handling [5]. Apart from these, it was also noted that if any system with high complexity degree in the software then it would be difficult to reach a specific reliability level, developers of system would push complexity into the layer of software with rapid development of size of system and ease of performing by updating the software [6].

## Related Work

Software Reliability Growth Models (SRGMs) are the models in the mathematics that deals with the properties of fault detection process during failure or testing reports when performing operation in the software. The SRGMs assist in deciding time of project release and to manage resources in the project. According to the difference between modeling theory, most SRGMs can be classified into two categories namely Parametric SRGM (PSRM) and Non-parametric SRGM (NPSRM).

The PSRMs are based on assumptions on the nature of software faults and the stochastic behavior of testing process. PSRMs have explicit expressions form and physical interpretation, therefore can be easily understood and used. However, because the assumptions of PSRMs are usually not consistent with the real conditions, the fitting and prediction accuracy of PSRMs can’t be kept satisfactory across various projects [21].

The NPSRMs utilize Machine Learning (ML) techniques for learning the inherent patterns of failure process. NPSRMs don’t require any prior assumptions; they usually have adaptive and self learning performance which improves the prediction and fitting accuracy compared with PSRMs. Many NPSRMs were proposed in recent years based on ML techniques such as Artificial Neural Networks (ANN), Support Vector Machine (SVM), and Genetic Programming (GP) [11-13, 21, 22].

Gene Expression Programming (GEP) is the extension of Genetic Algorithm (GA) and Genetic Programming (GP) in order to combine their advantageous features and overcome certain drawbacks of both algorithms compared with GA and GP. The fundamental difference between the three algorithms resides in the nature of the individuals. In Genetic Algorithm(GA) the individuals are linear strings of fixed

length (chromosomes) whereas in GP the individuals are the non-linear entities of different sizes and shapes (parse tree) whereas in GEP the individuals are encoded as linear strings of fixed length (the genome or chromosomes) which are afterwards expressed as nonlinear entities of different sizes and shapes.

It was applied to build a system that involves simple and common models of mathematics of behavior of temporal data [18]. GEP was referred as genotype system wherein computerized programs were evolved in unique size and shapes and they are encoded especially in linear chromosomes of fixed length [19]. In GEP, computerized programs are constituted as fixed-length linear character strings are known as chromosomes and after that evaluating the fitness are represented as ET (Expression Trees) of unique shapes and sizes. Isolation of phenotype and genotype had provided GEP with more power as well as flexibility of establishing the whole space of search compared with conventional genetic programming. Methods of GEP would help in solving various issues namely classification, cellular automata, optimization, analysis of time series, symbolic regression, logic synthesis and so on [20].

GEP-based non-parametric approach of reliability of software modeling involves certain significant characters in modeling of reliability in major components in algorithm of GEP which would result in the GEP-non-parametric reliability of software modeling by adopting GEP for mining the failure set of data for identifying the link between the observed faults and test coverage or failure time directly prior any assumptions [12]. In addition to this, experimental outcome indicate that when compared with parametric SRGMs, non-parametric SRGMs (NPSRMs), SVM (Support Vector Machine), ANN (Artificial neural networks), GP (Genetic Programming), GEP algorithm to non-parametric modeling of reliability was novel attempt and effective which can be very prominent.

## Proposed Work

This study aims to combine techniques of machine learning with algorithm of GEP namely Adaboosting and explore the GEP for modeling the defect function of predicting on the basis on test coverage. We introduce techniques of machine learning with the algorithm of GEP namely Adaboosting, simulated annealing which concentrates on how to adopt Adaboosting for obtaining the required NPSRMs from Training Failure Data-Set (TFDS). In Adaboosting, five main components are significant such as function set, fitness function, terminal set, termination criterion and control parameters has to be find out prior adopting Adaboosting.

TFDS $SD_0$ could be expressed as two input forms $(a_1, b_2)$..... $(a_j, b_j)$..... $(a_n, b_n)$..... or $(b_1, a_2)$..... $(b_j, a_j)$..... $(b_n, a_n)$.....where n denotes the number of data of $SD_o$, $b_j$ is denote cumulated faults and $a_F$ denotes the interval or cumulated time (failure time). If the NPSRM form is presented as A(b), then 1st form or input has to be applied. If the NPSRM form is presented as B($a_F$) then the 2nd form of input has to be preferred.

**Pre-processing of Data:**

During the process of testing there exists complexity and uncertainty, original Failure data set (FDS) unavoidably have more noise would affect accuracy of predictive

capability. Thus, the initial FDS needs to be pre-processed first. If the data of time B in $SD_0$ and denoted as interval time and it has to be transformed to the cumulated time and abolish the noise more efficiently than time for interval.

**Process of Modeling**

**Step 1:** Initialize population as $R_0$ based on some strategy for initialization. If $R_0$ involve the dominant characters that is genes in such characters would be more appropriate and diversified for the object of modeling and then efficiency of evolution and modeling quality could be efficiently enhanced. Hence, for developing $R_0$ with dominant characters, some functions in the elementary are considered as the elements of set in the function $F_R$ that are mostly adopted for modeling the reliability of the software and expressed in equation 1.1

$$F_R = \{+, *, /, -, \log, \text{Sqrt}, \exp(z)\} \quad 1.1$$

In order to expand validation, function set elements $F_R$ represented in equation 1.1 was more suitable one for NPSRM and next to that $F_R$ was compared with set in the function $F_R$' as represented in equation 1.1 that consists of few elementary and general functions. These fundamental functions are often adopted in modeling of mathematics

$$F_R' = \{/, *, +, -, 10z, \cos, \sin\} \quad 1.2$$

The terminal set has to be counted by either failure time or quantity of cumulated faults in data set training and random constant between 0 and 9.

**Step 2:** Focus is on encoding chromosomes.

**Step 3:** Aims to evaluate fitness. Function for evaluating fitness strongly relies on the problem stated type and needs to take into consideration that Adaboosting was generated for increasing the fitness. Therefore, it was suggested to follow two functions in the fitness which is mostly adopted as the predicting the power of SRGMs or comparison fitting criteria.

$$\text{MSE} = \frac{1}{h} - \sum_{u=0}^{h} (w_u - w_u')^2 \quad 1.3$$

$$\text{S or R-square} = 1 - \sum_{u=1}^{h} (w_u' - w_u)^2 / \sum_{u=1}^{h} (w_u - w_{av})^2 \quad 1.4$$

MSE stands for Mean Squared Error, $w_u'$ denotes the fitting data, $w_u$ denotes the observed data, $w_{av}$ and denotes the average value of $w_u$. MSE value is smaller or R-square or S is closer to 1, then the chromosomal fitness will be better.

**Step 4:** Aims to validate that if the chromosomal fitness does not reach the criteria of terminal C then shift to step 5. Otherwise iteration has to be stopped and shift to output. When fulfilling the terminal criteria C we have to consider three criteria's

- whether chromosomal fitness reaches the optimal value
- whether evolutionary process acquires a needed quantity of generation
- Whether values in the fitness are constant at the given quantity of generations.

**Step 5:** Develops new generation by operators of genetic series and selection.
**Step 6:** Moves to step 2 for new iterative process.

Output in the process of modeling needs Adaboosting to fulfill the terminal criterion C.

## Comparative Analysis

The comparative analysis concentrates on the applicability of genetic programming in the software reliability modeling. Codes are discussed with step by step instructions with screen shots.

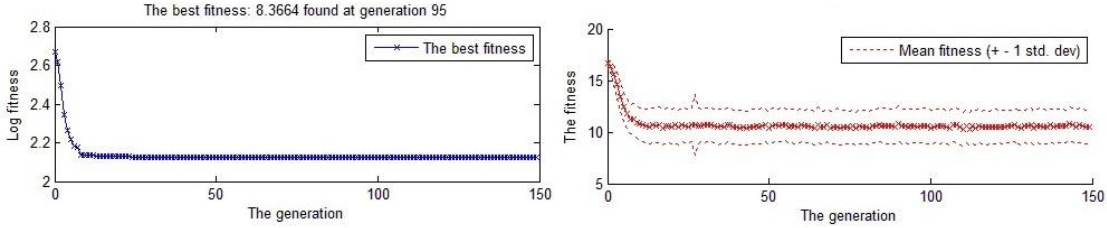


Figure 1.1 : Fitness Function

Figure 1.1 describes the fitness function. Red color dotted lines indicate the mean fitness of gene. Blue color dotted lines indicate best fitness of gene was 6.7399 found at generation 109. Control parameters or size of population adopted in this study are 400. Number of generation to run was 150 shown in X axis. Log fitness was shown in the y axis.

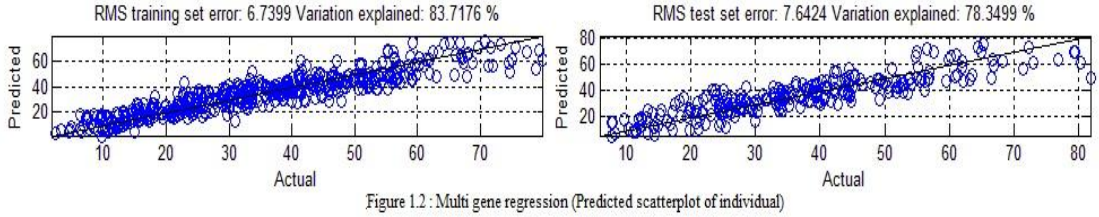


Figure 1.2 : Multi gene regression (Predicted scatterplot of individual)

This Figure 1.2 explains multi-gene regression, prediction scatterplot of individual. X axis shows the actual values and Y axis shows the predicted values. Actual root mean square (RMS) set error was 7.0988 variations were explained at 81.9375 %. Actual RMS set error was 7.934 variations were explained at 76.6663 %. Actual RMS validation set error was 7.6233 variations were explained at 79.6346 %.

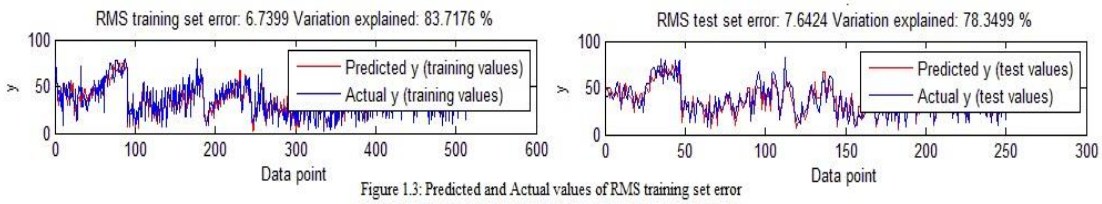


Figure 1.3: Predicted and Actual values of RMS training set error

This figure 1.3 depicts the predicted and actual values of RMS training set error, RMS test set error. X axis represents the actual y training, test and validation values. Y axis shows the predicted y training, test and validation values. Red color line denotes predicted training values and test values and blue color line denotes actual values and test values. From the analysis, it was found out that predicted values and actual values in gene were almost similar.

## Concluding Remarks and Future Work

From the analysis, it was noticed that GEP model is a novel and effective attempt which would be prominent for reliability of the software. From the analysis of the study, it was observed that predicted values and actual values in gene were almost similar. When predicted values are compared with traditional SRGM models, Non-parametric SRGMs ~~(NPSRMs),~~ SVM, GP, ANN and the proposed model outperform well. In addition to that, from the analysis of the research it was noticed that predictive capability of GEP models was better than other models. In future work, machine learning techniques would be combined with expanded Adaboosting algorithm. Likewise, in future extended Adaboosting will be used for predicting the function both in parametric modeling and non-parametric modeling.

## References

[1] M R. Lyu, "Handbook of software reliability engineering", New York: McGraw-Hill, 1996.

[2] Fenton, N. and Neil, M.. "Software metrics: Successes, failures and new directions". Journal of Systems and Software Vol47, 1999, pp.149-157.

[3] R. Sitte, "Comparison of software-reliability-growth predictions: neural networks vs. parametric recalibration". IEEE Transactions on Reliability Vol. 48, 1999, pp.285- 291~~.~~

[4] H. Pham, "System Software Reliability", Reliability Engineering Series, Springer, 2006.

[5] C. Y. Huang, M. R. Lyu, and S. Y. Kuo, "A Unified Scheme of Some Nonhomogenous Poisson Process Models for Soft ware Reliability Estimation", IEEE Trans. Software Eng., Vol. 29, March 2003, pp. 261-269.

[6] A. E. Emad, "Software reliability identification using functional networks: A comparative study", Expert Systems with Applications Vol. 36, 2009, pp.4013-4020.

[7] Mohd. Anjum, Md. Asraful Haque, Nesar Ahmad ~~et. al~~. ,"Analysis and Ranking of Software Reliability Models Based on Weighted Criteria Value,I.J". Information Technology and Computer Science, 02, ~~2013~~, pp. 1-14.

[8] T. Liliana and S. Daniel, "High energy physics event selection with gene expression programming, Computer Physics Communications", Vol. 178, 2008, pp.409-419.

[9] S.L., Ho, M., Xie, and T.N., Goh, "A study of connectionist models for software reliability prediction. Computers and Mathematics with Applications", Vol 46 issue no (7), 2003, pp.1037–1045.

[10] Afzal.W and Torkar.R , "A Comparative Evaluation of using GP for Predicting Fault Count Data, Third International Conference on Software Engineering Advances",2008, pp. 407-414.

[11] Aljahdali.H and Telbany.E , "Software Reliability Prediction using Multi-objective genetic algorithm", Retrieved on:6th March 2015, Retrieved from:*http://www.researchgate.net/profile/Mohammed_El-Telbany/publication/221429524_Software_reliability_prediction_using_multi-objective_genetic_algorithm/links/09e4150c8e30dde465000000.pdf*, 2009.

[12] Li H F, Lu M Y, Zeng M and Huang B Q , "A non parametric software reliability modeling approach by using gene expression programming", Journal of Information Science and Engineering 2012, pp.1145-1160.

[13] Vamsidhar.Y, Raju.P and Kumar.T , "Performance Analysis of Reliability Growth Models using Supervised Learning Techniques", International Journal of Scientific and Technology Research, Vol 1:1, 2012, pp.1-7.

[14] Kotaiah.B and Khan.R.A , "A Survey on Software Reliability Assessment by using different machine learning techniques", International Journal of Scientific, Engineering Research, Vol 3: issue 6, 2012, pp.1-7.

[15] Juan.S, Jing,Z and Shan.C et al , "Research on Software Reliability Assessment with Optimum Reserved Strategy Genetic Programming", Journal of Convergence Information Technology Vol 7: No 23, 2012, pp.317-323.

[16] Thamarai.I and Murugavalli.S, "Prediction of Software Performance using GP", Proceedings of International Conference on Emerging Research in Computing, Information, Communication and Applications, 2013, pp.660-664.

[17] Gaba.N and Ahuja.T , "A Review on PSO for Software Reliability", International Journal of Emerging Trends and Technology in Computer Science, Vol 3: Issue 3, 2014, pp.213-214.

[18] Al-Rahamneh Z., Reyalat M. Sheta A. F., Bani-Ahmad S., and Al-Oqeili S., "A New Software Reliability Growth Model: Genetic-Programming-Based Approach", Journal of Software Engineering and Applications, Vol 4, 2011; pp.476-481.

[19] C. Ferreira, "Gene Expression Programming: Mathematical Modeling by an Artificial Intelligence", Springer, Germany, 2006.

[20] Xie, Z., Li, X., Eugenio, B. D., Xiao, W., Tirpak, T. M. and Nelson, P. C. "Using Gene Expression ,Programming to Construct Sentence Ranking Functions for Text Summarization". Proceedings of the 20th International Conference on Computational Linguistics (COLING-2004). Geneva, Switzerland, 2004.

[21] E.O Costa, A.T.R Pozo, and S.R. Vertiligio," A genetic programming approach for software reliability modeling", IEEE Transactions on Reliability, Vol 59, 2010, pp.222-230.

[22] B.Yang, X.Li, M.Xie, and F.Tan, " A generic data driven software reliability model with model mining technique", Reliability Engineering and System Safety, Vol 95, 2010, pp.671-678.