Asian Journal of Information Technology 15 (18): 3556-3561, 2016
ISSN: 1682-3915

An Novel Approach On Software Reliability Growth Modelin Using the Data Mining Techniques

G. Nandini and G. Sridevi
Department of Computer Science and Engineering, K L University, Pradesh, India

Abstract: Software is directly a key part of many safety-critical and life-critical function systems. People consistently need easy- and instinctive-to-use software but the colossal challenge for software engineers is how to advance software with high accuracy in a appropriate manner in a appropriate manner. To assure quality and to assess the authenticity of software products, many Software Reliability advance Models (SRGMs) have been expected in the past three decades. The constructive problem is that consistently these selected SRGMs by association or software professional disagree in their reliability forecast while no single exemplary can be trusted to administer consistently accurate results across assorted applications. Consequently, some investigator have expected to use combinational models for develop the prediction capability of operating system reliability. In this study, appreciate weighted-combination, namely adequate arithmetic combination are expected. To solve the dilemma of determining proper burden for model combinations, we farther study how to assimilate Enhanced Genetic conclusion (EGAs) with several efficient engineer into weighted assignments. analysis are performed based on absolute software breakdown data and numerical conclusion show that our expected models are malleable enough to depict assorted software development climate. Finally, some administration metrics are conferred to both assure software aspect and complete the optimal release approach of software amount under development.

Key words: Software reliability, genetic programming, modeling, software faults, conferred

INTRODUCTION

Reliability in the accustomed engineering sense is the anticipation. It gives basic or system in a define situation will operate correctly for a stated period of time. Since the program systems impregnate every corner of modern life and any bankruptcy of those systems brunt us. An important issue in advance such software arrangement is to produce high aspect software system that amuse user requirements. As part of the operating system engineering process, developers attack to gauge the accuracy of their software and analyze the current level of accuracy with the past history of that operating system. If a software arrangement is experiencing fewer bankruptcy as time goes on. The accuracy of that system is spoken to be growing. acknowledge two inquiry of when the software allow be shipped and what its accuracy will be at that time are based on the use of spreadsheet reliability figure. The basic acceptance in software reliability create is that software bankruptcy are the result of a debatable process, having an unknown anticipation distribution. Software accuracy models cite some reasonable form for this circulation and are fitted to data from a software activity. Once a exemplary demonstrates a good fit to the accessible data, it can be used to complete the modern reliability of the software and predict the accuracy of the software at eventual times. The complication is that program systems are so complicated such that software architect are not directly able to test software well abundant to insure its appropriate operation. This may be due to the expectation made by separate software reliability portrayal or due to there is confidence among consecutive software runs. The debatable dependence of consecutive software runs also depends on the amount to which constitutional state of spreadsheet has been concerned and on the nature of transaction undertaken for execution continuation. Addressing these problems is: By finding instrument or accord to more accurately complete the nature of software systems,without visiting a large chunk of their achievable states. Taking in examination the failure correlation and; inspecting there is no single model adequately authentic in most or all applications freshly many ways of using parametric models, nonlinear time course analysis and data drilling to model software honesty and quality have been checked. The se examination point the way towards using computational understanding technologies to support personal developers in build software systems by

Corresponding Author: G.Nandini, Department of Computer Science and Engineering, K L University, Pradesh, India

manipulate the different forms of ambiguity present in a program system results from infrequent and erratic occurrence of human omission and incomplete or estimated data, in order to model complicated systems and support agreement making in uncertain climate (Dick and Kandel, 2005). These computational acumen methods are evolving assemblage of methodologies, which adopt patience for imprecision, uncertainty and partial truth to obtain meat, tractability and low cost. furry logic, neural networks, genetic algorithm, genetic register and developmental calculation are the most important key procedure. In this study, genetic-based access as one of the computational understanding techniques is followed in anticipate software reliability by predicting the indiscretion during the software examination process using software wrong classical data. Moreover, a multi-detached genetic algorithm is practiced to solve the three problems recorded previously by absorb the possible dependence among consecutive software run and use altogether of forecasting models by advance arrangement for estimating the model(s) framework with multiple and challenge objectives, through the groundwork of GA progress.

Literature review: Computationally astute technologies find its use software engineering as its focus on system create and decision making in the existence of uncertainty. In the last years many analysis studies has been borne out in this area of software accuracy modeling and forecasting. They admitted the application of auditory networks, fuzzy logic models; Genetic Algorithms (GA) based auditory networks, recurrent neural networks, fragment Swarm Optimization (PSO), Bayesian neural networks and backing Vector Machine (SVM) established techniques (Tian and Noore, 2007). Cai *et al.* (1991) defend the development of fuzzy software accuracy models in place of Probabilistic Software Accuracy Models (PSRMs). Their altercation was based on the proof that software authenticity is fuzzy in nature. A demonstration of how to advance a fuzzy model to characterize software accuracy was also conferred. Karunanithi *et al.* (1992) carried out a accurate study to explain the use of connectionist models in software reliability advance prediction. It was shown through factual results that the connectionist models acclimate well across different datasets and display better predictive accuracy than the well-known analytic software accuracy growth models. Aljahdali *et al.* (2001), made donation to software reliability growth forecast using neural networks by predicting accrue faults in a decisive time interval. They use a feed ahead neural network in which the number of neurons in the input blanket represents the number of problem in the input data. For the analysis, they used 4 delays: βi-1 , βi-2 , βi-3 and βi-4 , representing the number of collapse observed in the earlier days before βi. Ho *et al.* (2003) achieve a comprehensive study of connectionist models and their application to software accuracy prediction and found them to be improved and more malleable than the traditional models. A provisional study was achieve between their expected modified Elman frequent neural network, with the more attractive feed forward neural network, the Jordan recurrent model and some classical software accuracy growth models. Numerical results show that the expected network architecture achieve better than the other image in terms of forecast. Despite of the recent advance in the software reliability growth models, it was attended that different models have different anticipating capabilities and also no single exemplary is suitable under all assets. Tian and Noore (2005a, b) expected an on-line flexible software reliability forecasting model using mutative connectionist access based on multiple-delayed-input single-output building. The expected approach as shown by their results had a improved performance with respect to next-step monotony compared to existing auditory network model for failure time forecast. Tian and Noore (2005a, b) expected an evolutionary neural network design access for software cumulative failure time forecasting. Their results were found to be better than the actual neural network models. It was also shown that the neural network building has a great brunt on the performance of the network. Pai and Hong (2006) have enforced Support Vector Machines (SVMs) for forecasting software accuracy where assumed annealing (SA) algorithm was used to select the criterion of the SVM model. The empirical results show that the expected model gave better predictions than the other correlated methods. Su and Huang (2006) showed how to apply neural networks to anticipate software reliability. Further they made use of the neural network access to build a Dynamic Weighted Combinational Model (DWCM) and empirical results show that the proposed model gave significantly better forecast. Costa *et al.* (2005) expected the using of Genetic Programming (GP) to obtain software accuracy model for forecasting the reliability and continued this work by boosting the GP algorithm using re-weighting. The reweighting algorithm calls many times the literature algorithm with accredit weights to each example. Each time, the weights are gauge according to the error (or loss) on each illustration in the learning algorithm. In this way, the learning algorithm is employ to look closer at examples with bad indicator functions. Sheta (2006) uses genetic algorithms to appraisal the COCOMO model criterion for NASA Software Projects. The same idea is achieve for

guessing the parameters of different SRGM image using PSO (Sheta, 2006a, b). In this study, we analyze the use of GA to predict the faults during the software testing action using software faults historical data. Detailed results are arrange to explore the advantages of using GA in solving this problem.

Genetic algorithms: Genetic algorithms are machine culture and optimization blueprint, much like neural networks. However, genetic conclusion do not appear to endure from local minima as badly as auditory networks do. Genetic algorithms are based on the model of enlargement, in which a population evolves against overall fitness, even though entity perish. Evolution precept that superior entity have a better chance of reproducing than inferior entity and thus are more likely to pass their admirable traits on to the next generation. This acontinuity of the fittest criterion was first transformed to an optimization algorithm by Holland in 1975 and is today a dominant optimization approach for complex, nonlinear problems. In a ancestral algorithm, each original of a population is one achievable solution to an escalation problem, encoded as a double string called a chromosome. A group of these entity will be generated and will compete for the right to duplicate or even be carried over into the next bearing of the population. Competition consists of exercise a fitness function to every particular in the population; the entity with the best result are the fittest. The next bearing will then be constructed by bring over a few of the best individuals, breeding and mutation. Reproduction is carried out by a Acrossover activity, similar to what appear in an animal embryo. Two chromosomes commerce portions of their code, thus assemble a pair of new individuals. In the simplest form of crossover, a crossover point on the two chromosomes is elected at random and the chromosomes change all data after that point, while charge their own data up to that point. In order to announce additional variation in the community, a mutation operator will randomly adjustment a bit or bits in some chromosome (s). Usually, the alteration rate is kept low to permit good explanation to remain stable. The two most critical aspect of a genetic algorithm are the way solutions are defined and the fitness function, both of which are problem-reliant. The coding for a solution must be designed to perform a possibly convoluted idea or sequence of steps. The fitness function must not only interpret the cipher of solutions but also must authorize a ranking of different solutions. The fitness function is what will drive the entire community of solutions towards a globally best (Goldberg, 1989).

Figure 1 illustrates the basic steps in the approved genetic algorithms. Most GAs has been used for single detached problems, although several multi-objective access have been expected. There are three different

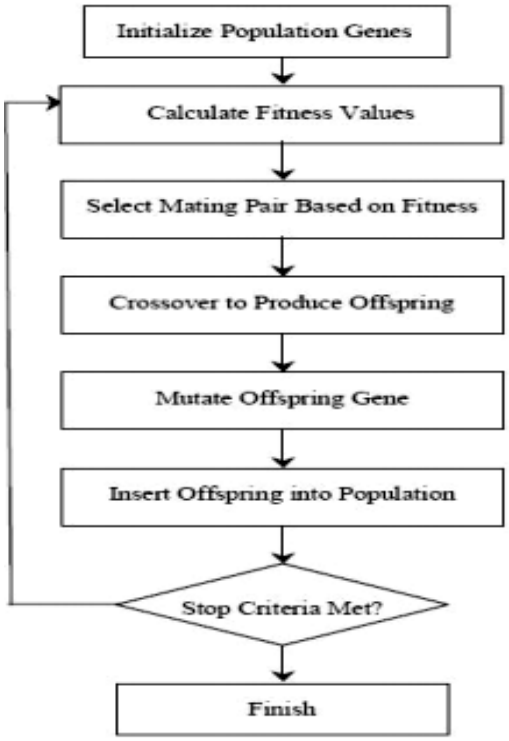

Fig. 1: Canonical genetic algorithm

access to cope with multi-objective problems, namely: convert the original multi-objective problem into a single detached problem by using a weighted function, the lexicographical access where the objectives are ranked in order of arrangement and the Pareto approach which subsist of as many non-dominated solutions as possible and retracing the set of Pareto front to the user. The main completion are that the weighted formula access which is by far the most used in the data mining biography is an ad-hoc access for multi-objective optimization, whereas the lexicographic and the Pareto approaches are more principled approaches and therefore earned more attention from the computer science association (Mitchell, 1998).

MATERILAS AND METHODS

Predicting models: In the past three decades, hundreds of models were imported to estimate the accuracy of software systems (Xie, 2002; Dietterich, 2000). The issue of architecture growth models was the subject of many analysis researches (Musa, 2004, 1975) which helps in guessing the reliability of a software system before its release to the market. There arrive to be three major trends in software reliability research: the use of Non Homogeneous Poisson Process (NHPP) models, Bayesian

inference and time series inquiry. An NHPP is a Poisson process with a time-varying mean value function. Bayesian inference in software accuracy models actually consists of treating the parameters of a accuracy model as random variables instead of criterion to be estimated. Some reasonable prior circulation are assumed for these parameters and Bayes = theorem is then invoked to regulate the posterior circulation using accuracy data. Finally, time series analysis uses an auto-regressive process and an Auto-regressive Unified Moving Average (ARIMA) model. In addition to these three large-scale trends, there are many other introduce software reliability models that are somewhat unique. In this study, the auto-relapse models are approve.

RESULTS AND DISCUSSION

Implementation

Clustering: Clustering is a form of unsupervised information in which no class labels are administer. It is often the first data mining task enforced on a given collection of data. In this, data records need to be arranged based on how similar they are to other records. It is a task of organism data into groups such that the data objects that are analogous to each other are put into same cluster. The groups are not predefined. It is a course of partitioning a data in a appointed of meaningful sub-classes called clusters. Clusters are subsets of article that are similar. Clustering helps users to understand the essential grouping or structure in a data set. Its blueprint are evaluated based on the analogy of objects within each clusters. This approach offers asset to the expert who must choose the labels. Instead of inspecting and labelling software modules one at a time, the adept can investigate and label a given cluster as a whole; he or she can assign all the schedule in the cluster the clone quality label.

Software Defect Prediction using Clustering In (Sheta, 2006a, b; Goldberg, 1989) they have used k-mean approach of clustering for Software Defect Prediction. K-mean clustering is a non-hierarchical clustering agenda in which items are moved amid sets of clusters until the covet set is reached. It has certain drawbacks, so to affected those drawbacks Quad Tree-based k-mean clustering method was expected. The objective was: first, Quad-trees are enforced for finding initial cluster centres for k-mean algorithm. Second, the Quad tree-based algorithm is applied for predicting faults in curriculum modules. They have appraise the effectiveness of Quad tree-based k-mean bundle algorithm in predicting faulty software modules as correlated to the authentic k-mean algorithm. The result of this Quad tree-based k-mean algorithm is correlated with other approaches and found that the number of emphasis of k-means algorithm is less in case of Quad tree-based k-mean save for other approaches, as well as percent error also give fairly adequate values.

Classification: Classification is a course of finding a set of models that describe and categorize data classes or approach. It is the organization of data in given classes accepted as supervised learning, where the class labels of some training fragment are given. These samples are used to administer the culture of a classification model. Classification access normally use a training set where all objects are earlier associated with known class labels. The classification conclusion learns from the training set and builds a model. The model is used to allocate new objects. Fraud detection and credit risk function are particularly well suited to this type of analysis. This access frequently employs agreement tree or neural network-based classification algorithms. The data classification process associate learning and classification. In Learning the training data are consider by classification algorithm. In classification test data are used to appraisal the efficiency of the classification rules. Software Defect forecast using classification A large number of classification methods have been advised to build software crack prediction models. Hashem (1997), Holland (1975) an company rule classification method is expected which derives a comprehensible decree set from the data. They have compared CBA2 (Musa, 1975) with other rule established classification method to check whether classification algorithms based on association rules are advisable for software fault forecast or not. They have also tried to find out whether rule sets learned on one data set are germane to others data sets or not. They have considered the performance of an association rule based classification approach for software crack prediction. The experiments were attend on the data sets and the result was correlated with other classifiers and access the satisfactory performance without losing comprehensibility.

Association mining: The Association mining task subsist of identifying the frequent itemsets and then assemble conditional conclusion rules among them. It is the task of conclusion correlations between items in data sets. Association Rule conclusion need to be able to generate rules with assurance values less than one. Association rule mining is desultory or unsupervised data mining past variable-length data and it harvest clear, understandable results. The task of association rules mining subsist of two steps. The first associate finding the set of all continual item sets. The second step associate testing and generating all high certainty rules among item sets. It has a quiet problem affidavit that is to find the set of all

subsets of items that occur generally in many database records or transactions and to fragment the rules telling us how a subgroup of items influences the continuation of another subset.

Software Defect forecast using association mining In association rule mining approach we use defect type data to anticipate software defect company that are the relations among contrasting defect types. The crack associations can be used for three ambition: First, Find as many analogous defects as achievable to the detected crack and make more effective alteration to the software. Second, it helps to appraise reviewer's results amid an inspection. Third, it helps in assisting managers in developing the software process through inquiry of the reasoning some defects frequently occur calm. Association rule mining ambition at discovering the decoration of co-occurrences of the aspect in the database. They begin that higher support and higher assurance levels may not result in higher forecast accuracy.

Criteria value matrix: In this matrix each element aij shows the value of jth criteria of ith model. Let us consider n number of SRGMs with m criteria:

$$\text{Criteria value matrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \\ (\text{Amin})_1 & (\text{Amin})_2 & \cdots & (\text{Amin})_m \\ (\text{Amax})_1 & (\text{Amax})_2 & \cdots & (\text{Amax})_m \end{bmatrix}$$

Where:
(Amax)j = Maximum value of jth criteria
(Amin) j = Minimum value of jth criteria
A_{ij} = Value of j th criteria of I th model
Else

$$\text{Criteria rating} = \frac{\text{Criteria value in the database} - \text{criteria value}}{\text{Criteria maximum value in the database} - \text{Criteria minimum value in the database}}$$

When bigger value of the criterion represents fitting well to the actual data i.e. is the best value:

$$\text{Criteria rating} = \frac{\text{Criteria minimum value in the database} - \text{criteria value}}{\text{Criteria maximum value in the database} - \text{Criteria minimum value in the database}}$$

Criteria weight: Let us consider Xij represent the rating of jth criteria of ith model. Weight of criteria will be Wij = 1-Xij:

$$X_{ij} = \frac{(\text{Amax})_i - a_{ij}}{(\text{Amax})_j - (\text{Amin})_j}$$

$$X_{ij} = \frac{a_{ij} - (\text{Amin})_j}{(\text{Amax})_j - (\text{Amin})_j}$$

Where, i = (1, 2, 3... n) and j = (1, 2, 3... n).

Weight matrix: The weight matrix can be represented as:

$$\text{Weight matrix} = \begin{bmatrix} W_{11} & W_{12} & \cdots & W_{1m} \\ W_{21} & W_{22} & \cdots & W_{2m} \\ \vdots & \vdots & \cdots & \vdots \\ W_{n1} & W_{n2} & \cdots & W_{nm} \end{bmatrix}$$

Weighted criteria value matrix: Weighted criteria value is calculated by multiplying the weight of each criterion with the criteria value i.e. $A_{ij} = W_{ij} * ai$

$$\text{Weighted criteria value matrix} = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1m} \\ A_{21} & A_{22} & \cdots & A_{2m} \\ \vdots & \vdots & \cdots & \vdots \\ A_{n1} & A_{n2} & \cdots & A_{nm} \end{bmatrix}$$

Permanent value of model: Permanent value of model is the weighted mean value of all criteria

$$Z_i = \frac{\sum_{i=1}^{m} A_{ij}}{\sum_{i=1}^{m} AW_{ij}}$$

where, i = (1, 2, 3... n). Ranking of models are done the basis of permanent value of model as calculated above. Smaller permanent value of model represents good rank as compare to bigger permanent value of model.

CONCLUSION

In this research, we have measured the predictability of software reliability using ensemble of models trained using GA. The study is applied on three study sets; military, real time control and operating system. As far as, the predictability of the single AR model and ensemble of AR models trained by GA algorithm over the trained and test data is concerned, the ensemble of models performed better the single model. Also, we find that the weighted

average combining method for ensemble has a better performance in a comparison with average method. This due to the GA learned weights which decide the contribution of each model in the final results. However these models are linear in the future, we plan to use non-liner models like neural networks and other form of ensemble combinations.

REFERENCES

Aljahdali, S.H., A. Sheta and D. Rine, 2001. Prediction of software reliability: A comparison between regression and neural network non-parametric models. Proceedings of the ACS/IEEE International Conference on Computer Systems and Applications, June 25-29, 2001, IEEE, Beirut, Lebanon, ISBN: 0-7695-1165-1, pp: 470-473.

Cai, K.Y., C.Y. Wen and M.L. Zhang, 1991. A critical review on software reliability modeling. Reliab. Eng. Syst. Safety, 32: 357-371.

Costa, E.O., S.R. Vergilio, A. Pozo and G. Souza, 2005. Modeling software reliability growth with genetic programming. Proceedings of the 16th IEEE International Symposium on Software Reliability Engineering (ISSRE'05), November 1-1, 2005, IEEE, Chicago, Illinois, ISBN: 0-7695-2482-6, pp: 10-180.

Dick, S. and A. Kandel, 2005. Computational Intelligence in Software Quality Assurance. Vol. 63, World Scientific, Singapore, ISBN: 781-256-172-2, Pages: 179.

Dietterich, T.G., 2000. Ensemble methods in machine learning. Proceedings of the 1st International Workshop on Multiple Classifier Systems, June 21-23, 2000, Cagliari, Italy, pp: 1-15.

Goldberg, D.E., 1989. Genetic Algorithms in search optimization and Machine Learning. Pearson Education Pte. Ltd., Singapore.

Hashem, S., 1997. Optimal linear combinations of neural networks. Neural Networks, 10: 599-614.

Ho, S.L., M. Xie and T.N. Goh, 2003. A study of the connectionist models for software reliability prediction. Comput. Math. Appl., 46: 1037-1045.

Holland, J., 1975. Handbook of Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, Michigan.

Karunanithi, N., D. Whitley and Y.K. Malaiya, 1992. Prediction of software reliability using connectionist models. IEEE Software Eng. Trans., 18: 563-574.

Mitchell, M., 1998. An Introduction to Genetic Algorithms. The MIT Press, USA., ISBN-10: 0262631857, Page: 221.

Musa, J.D., 1975. A theory of software reliability and its application. IEEE. Trans. Software Eng., 1: 312-327.

Musa, J.D., 2004. Software Reliability Engineering: More Reliable Software Faster and Cheaper. Tata McGraw-Hill Education, Noida, India, ISBN-13: 978-0-07-060319-6, Pages: 611.

Pai, P.F. and W.C. Hong, 2006. Software reliability forecasting by support vector machines with simulated annealing algorithms. J. Syst. Software, 79: 747-755.

Sheta, A., 2006a. Reliability growth modeling for software fault detection using particle swarm optimization. Proceedings of the 2006 IEEE International Conference on Evolutionary Computation, July 16-21, 2006, IEEE, Vancouver, British, ISBN: 0-7803-9487-9, pp: 3071-3078.

Sheta, A.F., 2006b. Estimation of the COCOMO model parameters using genetic algorithms for NASA software projects. J. Comput. Sci., 2: 118-123.

Su, Y.S. and C.Y. Huang, 2006. Neural-network-based approaches for software reliability estimation using dynamic weighted combinational models. J. Inform. Softw. Technol., 80: 606-615.

Tian, L. and A. Noore, 2005a. Evolutionary neural network modeling for software cumulative failure time prediction. Reliab. Eng. Syst. Saf., 87: 45-51.

Tian, L. and A. Noore, 2005b. On-line prediction of software reliability using an evolutionary connectionist model. J. Syst. Software, 77: 173-180.

Tian, L. and A. Noore, 2007. Computational Intelligence Methods in Software Reliability Prediction. In: Computational Intelligence in Reliability Engineering. Gregory, L. (Ed.). Springer Berlin Heidelberg, Berlin, Germany, ISBN: 978-3-540-37367-4, pp: 375-398.

Xie, M., 2002. Software Reliability Models Past Present and Future. In: Recent Advances in Reliability Theory: Methodology. Limnios, N. and M. Nikulin (Eds.). Practice and Inference, Oswego, New York, USA., pp: 323-340.