Middle-East Journal of Scientific Research 24 (9): 2896-2899, 2016
ISSN 1990-9233

DOI: 10.5829/idosi.mejsr.2016.24.09.23670

Identification of Effective Website Maintainability Metrics Through Genetic Algorithm Based Classification

[1]Mrs. N. Thangammal and [2]A. Pethalakshmi

[1]Mother Teresa Women's University, Kodaikanal, Tamil Nadu, India
[2]Dept of Computer Science,
M.V.M.Govt Arts College fro Women, Dindigul, Tamil Nadu, India

Abstract: With the exponential growth in the number of web sites being engineered, the quality of the engineered web sites has come to become a paramount concern to web developers. Quality of web sites is a multi faceted concept. One important aspect of quality is maintainability. Given the dynamic nature of business enterprises today, their websites have to undergo regular maintenance to ensure that they are up-to-date and interesting for the viewers. It is highly desirable to engineer websites that are easy to maintain and this would yield substantial cost savings for the organization. In order to control maintainability, it has to be measured and this is a non-trivial task. Various metrics have been proposed which claim to quantify the maintainability aspect of web sites. Given a set of metrics and the actual maintainability of some websites, identification of a subset of these metrics that effectively quantify the maintainability is a classification problem. Genetic algorithms can be effectively used to handle such type of complex problems. The paper proposes the use of GA to identify the effective subset of metrics that quantify the maintainability of web sites.

Key words: Website maintainability · Metrics · Genetic Algorithms · Classification

INTRODUCTION

The last decade has seen a steep spurt in the number of websites engineered. Websites have become inevitable for organizations that endeavor to improve profitability and customer management. Quality of the engineered web sites has become a top concern for organizations engaged in website development. As Fairley claims [1], software development is a labor intensive activity. Website development is no exception. The quality of the website developed tends to lean heavily on the nature and quality of personnel involved in its development.

Maintainability: Quality is a multi-faceted concept which makes it a difficult problem to be confronted by developers and organizations. One important aspect of quality is maintainability that cab be defined as the "effort required to maintain the software" [2]. Given the dynamic nature of the manner in which businesses of the day are conducted, maintenance of websites has to be done on a regular basis and this entails cost. Cost savings can be obtained if the websites engineered require minimum effort to maintain them.

The Challenge of Measurement: In order to effectively control one aspect, it is very crucial to have a mechanism for measuring the aspect and express it in numbers [2]. This is the importance of metrics. In this regard, the challenge posed by maintainability is that there is not yet one metric which can give an accurate, objective estimate of maintainability of websites. On the other hand, various metrics exist to measure other aspects and it has been claimed that some of these can be used to quantify the effort required to maintain a website.

Background: Given the values for a set of metrics on one side and an indication of their maintainability on the other side, identification of a subset of these metrics that effectively quantify the maintainability is a classification problem. Classification is a data mining technique where the class labels for a set of data are known apriori and these are used to build a model that can be used for

Corresponding Author: Mrs. N. Thangammal, Research Scholar, Mother Teresa Women's University, Kodaikanal, Tamil Nadu, India.

classifying data for which class labels for unknown. Various algorithms exist in the literature for classification [3]. Some of these are Decision tree induction, Bayesian classification and Support Vector Machines.

This paper proposes the use of Genetic Algorithms which are inspired by Darwin's theory of natural evolution. The basic idea behind genetic algorithms is as follows: solutions to problems are encoded as genes in chromosomes. An intial random population of solutions is assessed for fitness and based on the fitness chromosomes are crossed over to produce new off-springs. These off-springs are subject to mutation or changes in some of the genes. The fitness of the off-springs is calculated and the less fit solutions are replaced by fitter ones. The procedure stops when a stopping criterion is reached.

Motivation: The motivation behind this research is the work of Vivanco and Pizzi [4]. In their work, they use a Java application from the biomedical domain. A system architect is asked to rank the quality of objects in terms of their maintainability. They use values for 16 metrics collected from the java application and the rankings to train a genetic algorithm based classifier. They have proposed the use of genetic algorithms for identifying effective software metrics that quantify maintainability of software in general. Since web sites have many aspects strikingly different from conventional software, the idea was to apply the method for identifying the effective metrics that quantify maintainability of web sites. The identification will greatly aid the web developers in acquiring knowledge about the aspects which they need to focus on to develop maintainable websites.

Metrics Considered for the Study: Various metrics have been proposed for websites and some of these can be found in [5,6]. For the purpose of the study, the following set of 12 metrics was taken into consideration.

Table 1: Metrics Considered for the Study

Metric	Meaning
NserverP	Total Number of Server Pages
NClientP	Total Number of Client Pages
NTotP=NServerP + NClienP	Total Number of pages
NClientScriptComp	Total Number of Client Script Components
NServerScriptComp	Total Number of Server Script Components
NC	Total Number of Classes
NM	Total Number of Methods
NFormE	Total Number of form elements
Web Data Coupling	NFormE / NServerP
NForwardR	Total number of forward relationships
CSTM	Number of statements of server components executed in response to requests coming from a client side form
ICC	Number of independent paths in the inter-procedural control flow graph

Proposed Methodology: 50 websites developed by a local software organization were given to a web developer who has a rich experience in web development and he was asked to rank the web sites as low, medium and high in terms of maintainability. Web sites ranked as "high" are those that will entail minimum effort for modifying them and those ranked as "low", entail a huge effort even to make seemingly small changes.

Solutions are encoded as 12 bit strings with each bit indicating the presence or absence of one of the 12 metrics. Position of a particular metric in the string is fixed. A random population of initial solutions is formed and the fitness of a solution is calculated using the Linear Discriminant Analysis (LDA) technique adopted by Vivanco and Pizzi with the leave one out method of training and testing. LDA is detailed in section 5.1. With this approach, training is done with 49 websites and it is checked whether the remaining web site is classified correctly. The overall classification rate is the number of times a web site is correctly classified.

The following were the values of the various GA parameters chosen. Number of generations was set in the first run to 50 and in the second run to 100. One generation was set to contain 50 possible solutions and 25% was the number of elite genes – genes with high fitnesses that are passed to the next generation without mutation.

Linear Discriminant Analysis: Linear Discriminant Analysis is a classifier strategy used to determine linear decision boundaries between groups while taking between-group and within-group variances into account

[7-10]. Linear discriminant analysis can be proven to generate the optimal linear decision boundary between groups provided the error distributions for each group are the same.

Mathematically, a feature vector x should be allocated to the group for which the probability distribution $p_i(x)$ is greater than any other distribution while taking into account prior-known probabilities. $x \in \omega_i$ If $q_i p_i(x) \geq q_j p_j(x) \forall j = 1,2,...,k$ where q_i is the prior probability of observing x from the group ω_i. Substituting the proportional probabilities N_i / N for each $\omega_i, D_{ij}(x) = L_i(x) - L_j(x) = 0$ defines the hyperplane that separates ω_i from ω_i where the discriminant function is given by: $L_i(x) = \log(q_i) + \mu_i^T W^{-1}(x - \frac{1}{2}\mu_i)$. Here μ_i is the mean for ω_i. and W is the co-variance matrix of the dataset.

RESULTS AND DISCUSSION

The best classification rate obtained with 50 generations was 60.23% and with 100 generations was 62.21%. The classification rates obtained by considering all the metrics, only the metrics contained in the top gene and the common metrics contained in the top 5 genes are tabulated.

Table 2: Best Classification Rates

	Best Classification rate
All Metrics	62.21%
Best gene	69.43%
Common Metrics in the top 5 genes	68.92%

The following table gives the metrics represented in al the top 5 genes

Table 3: Metrics Represented in the Top 5 Genes

Metric	Meaning
NserverScriptComp	Total Number of Server Script Components
NC	Total Number of Classes
NM	Total Number of Methods
Web Data Coupling	NFormE / NServerP
ICC	Number of independent paths in the inter-procedural control flow graph

As can be seen, the total number of server script components has a significant impact on the maintainability of websites. This captures the intuition that web sites with little or no server side script components require little, if any, effort for maintenance. The Number of classes, number of methods also seem to have an impact on maintainability. The most expected candidate is the ICC metric as this is more or less tied to cyclomatic complexity and hence can impact maintainability.

To uncover the potential of GA in tacking complex problems such as the problem chosen, a comparison was attempted with 2 other well known classification techniques, SVM and C4.5. It was observed that GA outperformed both these techniques [11-16].

Table 4: Comparison with SVM and C4.5

Technique	Best Classification Rate
SVM	61.01%
C4.5	60.22%
GA (Proposed Method)	62.21%

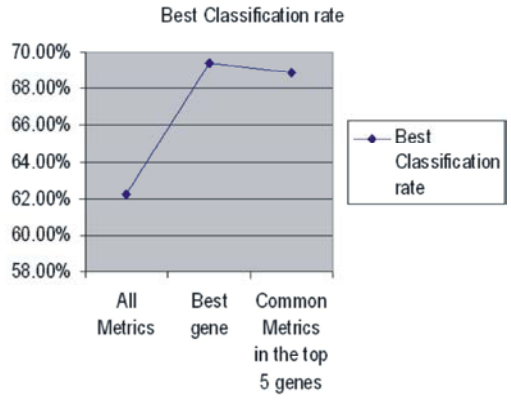

Fig. 1: Best Classification Rates Obtained

Fig. 2: Comparison with SVM and C4.5

CONCLUSION

The paper proposed a method for identifying the effective subset of metrics that accurately characterize the maintainability of web sites. As web developers often have to work under limited resources and tight schedule, this identification can throw light on the crucial areas of

website development that impact its maintainability. The proposed method also throws light on the potential of Genetic Algorithms in tacking complex problems encountered in software engineering.

As a part of future work, many more metrics can be included in the study and more advanced techniques like particle swarm optimization can be allied to uncover even more accurate classifiers.

REFERENCES

1. Fairley, R.E., 2001. Software Engineering Concepts, Tata McGraw Hill.
2. Pressman, R.S., 2001. Software Engineering-A Practitioner's Approach, Fifth Edition.
3. Han, J. and M. Kamber, 2006. Data Mining: Concepts and Techniques, Second Edition, Morgan Kaufmann Publishers.
4. Vivanco, R.A. and N.J. Pizzi, 2003. Identifying Effective Software Metrics Using Genetic Algorithms, IEEE Canadian Conference on Electrical and Computer Engineering,
5. Ghosheh, Emad, Black, Sue, Kapetanios, Epaminondas and Baldwin, Mark, 2009. Exploring the relationship between UML Design Metrics for Web Applications and Maintainability, Journal of Object Technology, 2(3).
6. Nadia Alshahwan and Harman Mark, 2012. Crawlability Metrics for Web Applications, IEEE 5th International Conference on Software Testing, Verification and Validation.
7. Weston, J., 2014. Support Vector Machine Tutorial, Retrieved from: www.cs.columbia.edu/~kathy/ cs4701/documents/jason_svm_tutorial.pdf.
8. Ruggieri, Salvatobe, 2015. Efficient C4.5, Retrieved from: idb.csie.ncku.edu.tw/tsengsm/ COURSE/DM/Paper/ec45.pdf.
9. Sree Ram Kumar, T. and K. Alagarsamy, 2013. Finding Effective Software Security Metrics Using Genetic Algorithm, International Journal of Software Engineering.
10. Duda, C.D., P.E. Hart and D.G. Stork, 2001. Pattern Classification, Wiley and Sons, New York, USA.
11. Back, T., D.B. Fogel and Michalewicz, 2000. Evolutionary Computation 1, Basic Algorithms and Operators, Institute of Physics Publishing, Bristol, UK.
12. Artzi, S., A. Kiezun, J. Dolby, F. Tip, D. Dig, A. Paradkar and M.D. Ernst, 2010. Finding bugs in web applications using dynamic test generation and explicit-state model checking, IEEE Transactions on Software Engineering, 36: 474-494.
13. Artzi, S., J. Dolby, F. Tip and M. Pistoia, 2010. Practical fault localization for dynamic web applications, in Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering Volume 1, ser. ICSE '10. New York, NY, USA: ACM, pp: 265-274.
14. Warren, P., C. Boldyreff and M. Munro, 1999. The evolution of websites, in IWPC '99: Proceedings of the 7th International Workshop on Program Comprehension. Washington, DC, USA: IEEE Computer Society, pp: 178.
15. Palmer, J.W., 2002. Web site usability, design and performance metrics, Info. Sys. Research, 13(2): 151-167.
16. Di-Lucca Giuseppe, Anna Fasolino and Porfirio Tramontana, 2004. Reverse engineering web applications: the WARE approach," Journal of Software Maintenance, 16(2): 71-101.