Bug Triaging Mechanism for Non-Reproducible Bugs

Akshay Patil[1], Arpit Bobade[2], Vaishnavi Ingole[3]

[1,2,3]*Department of Computer Science and Engineering, Dr. D. Y. Patil College of Engineering, Pune, India*

***Abstract*—Programming organizations spend more than 45 percent of cost in managing programming bugs. An unavoidable stride of settling bugs is bug triage, which means to effectively appoint a designer to another bug. To diminish the time cost in manual work, content order procedures are connected to lead programmed bug triage. In this paper, we address the issue of information decrease for bug triage, i.e., how to lessen the scale and enhance the nature of bug information. We consolidate occurrence determination with include choice to at the same time diminish information scale on the bug measurement and the word measurement. To decide the request of applying case choice and highlight determination, we separate characteristics from recorded bug informational collections and construct a prescient model for another bug informational collection. We observationally research the execution of information lessening on absolutely 600,000 bug reports of two huge open source ventures, in particular Eclipse and Mozilla. The outcomes demonstrate that our information decrease can adequately diminish the information scale and enhance the precision of bug triage.**

Index Terms—Bug Data Reduction, Bug Triage, Data Management in Bug Repository, Feature Selection, Instance Selection, Mining Software Repository

I. INTRODUCTION

Bug triage is the methodology that assigns the bug reports to the appropriate developer. The data that is available in the real world is prone to noise and may also contain redundant values. Data that comprising noise may mislead the data analysis techniques while redundant data may increase the cost of data. Software bugs are inevitable and bug triaging is a difficult and time consuming task. Previous studies show that optimizing bug triaging is a non-trivial activity. Hence, automatic bug triaging techniques that can help triager in making strategic decision can be beneficial. One of the important factors that plays an integral role in developer selection is recency. This is due to the fact that knowledge of a developer is statistically correlated with time. The existing studies have used bug textual features with time decay for assignment. Since meta-fields are also important, we proposed a novel metafield oriented time decay model for bug triaging.

Previous studies propose varied bug triaging techniques considering all bug parameters on a same platform. In real time scenario, bug parameters can play a role with varying importance in decision making. Hence, we use phenomenon of parameter prioritization in bug triaging. Analytic hierarchy process (AHP) is a technique for decision making that involves parameter prioritization. We propose an AHP based bug triaging technique, W8Prioritizer, to optimize the efficiency of bug report assignment technique.

The main contributions of this research will be development of a proficient recommender system for bug report assignment. We characterize developer recommendation through various quantitative and qualitative models. In essence, these models utilize time decay and parameter prioritization. We plan to use these models to perform suitable developer selection for NR bugs. Our intuition is that before developer assignment for NR bugs if there is a prediction model that could judge the fixability of NR bugs then it could be beneficial for both bug triagers and software developers. With the use of such mechanism, developers & triager can actually devote their time and efforts only on those bugs that have high probability of getting fixed and are regarded as fixable by the proposed mechanism.

The order of applying the reduction techniques may affect the result of bug triage approach. In this paper, we propose a Predictive model in order to determine the order of bug data reduction techniques, i.e., FS to IS or IS to FS. To decrease the manual triager cost, text classification technique i.e., Naive Bayes is used to predict correct developer to solve and fix the bug reports. The proposed system performance is verified using Mozilla bug data set .After reducing the training set, the accuracy of bug data is measured as 78%. The result shows that the experiment on reduce training sets can obtain better accuracy than that on original training set.

The remainder section of this paper is organized as follows: Section 2 presents the proposed methodology. Section 3 presents the implementation. In Section 4 we briefly conclude this paper and present our future work.

II. PROPOSED METHODOLOGY

In the last two decades, researchers have addressed problem of bug triaging exhaustively. Various techniques such as machine learning (ML), information retrieval (IR), statistical approaches, fuzzy sets, and auction based approaches, social network & tossing graph based approaches have been used in literature. IR based approaches are most popular among all. This is due to the fact that IR based techniques consider overall expertise of developers towards bug reports for recommendation. Also, it is easy to comprehend with other techniques. Thus, we propose IR based recommender taking additional factors, time decay and parameter prioritization into consideration.NR bugs account for approximately 17% of all bug reports and 3% of these bugs are later marked as fixed. There could be various reasons behind this fixation of NR bugs. If we can have a mechanism that can provide information to developer beforehand that the bug report currently marked as

NR will be fixed in future or not, it will not only provide insights to triager but also helps developers by predicting the possibility that whether bug report marked as NR could get fixed in future or not.

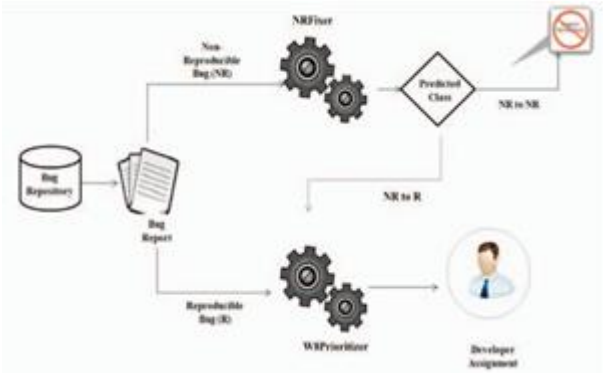

Fig. 1. Procedure of proposed methodology

This will save time, effort and cost incurred in those NR bugs which have less probability of getting fixed. With the use of such mechanism, developers & triager can actually devote their precious time and efforts on those bugs that are regarded as fixable by the proposed mechanism. This will also raise level of interest among developers towards NR bugs. Thus we are developing a machine learning based prediction model, NR Fixer to predict the probability of fixation of bug report currently marked as NR. The NR Fixer uses various bug meta-fields to predict the fixability of NR bug on the basis of past information. The NR bug reports predicted as fixable by the prediction model will be assigned with a new developer using the proposed bug assignment technique who will try to reproduce the bug and may fix it. The proposed prediction model, NR Fixer has been evaluated on Mozilla and Eclipse bug reports and achieves precision value up to 74.7% for Mozilla bug reports and 68% for Eclipse bug reports.

The bug details consist of bug repository and bug reports. In a bug repository, a bug is sustained as a bug report, which traces the textual illustration to repeat the bug and updates according to the status of bug fixing.

A) Bug Repository:

A bug repository is a typical software repository, for storing details of bugs, e.g., a popular and open source bug repository, Bugzilla .Large software projects deploy bug repositories is also called as bug or issue tracking systems, which is used to support information collection and to assist developers to handle bugs. Each bug is maintained as a bug report, which traces the documentary description of reproducing the bug and revises according to the significance of bug fixing. The use of bug repository can improve the development process and quality of software produced. It presents a data platform to sustain many forms of assignment on bugs, e.g., defect prediction, bug localization and reopened bug analysis.

B) Bug Report:

A recorded bug is called a bug report or bug data. It has multiple items for detailing the information of reproducing the bug. In a bug report, the outline and the report are two key items about the information of the bug, which are traced in natural languages. Summary denotes the general statement for identifying a bug and description gives the details to reproduce the bug. The bug report may also contain other items also, such as product, platform, and importance.

C) Bug Triage:

The method of allocating a correct developer for renovating the bug is called bug triage. Once the bug report is formed, a bug triager allocates the bug to a developer who can fix this bug and developer is recorded in an item assigned-to without any tossing.

D) Feature Selection:

Feature selection is a preprocessing method for choosing a diminished set of features for huge-scale data sets. The pre-processing techniques are tokenization, stop word removal, stemming process and vector space model. The tokenization method is used to tokenize the summary and description of the bug reports into word vectors. Non- alphabetic words and special character are removed to avoid the noisy bug words. Stop word removal technique remove the stop words in high frequency and provide no helpful information for bug triage. Stemming technique uses porter stemming algorithm for reducing inflected words their word stem/root form. Vector space model /Term vector model is an algebraic model for representing text document as vector of identifier. The minimized set is considered as the representative features of the original feature set. The four well-performed algorithms are chosen in text data and software data, namely Information Gain (IG), $\chi 2$ statistic (CH), Symmetrical Uncertainty attribute evaluation (SU), and Relief-F Attribute selection (RF).

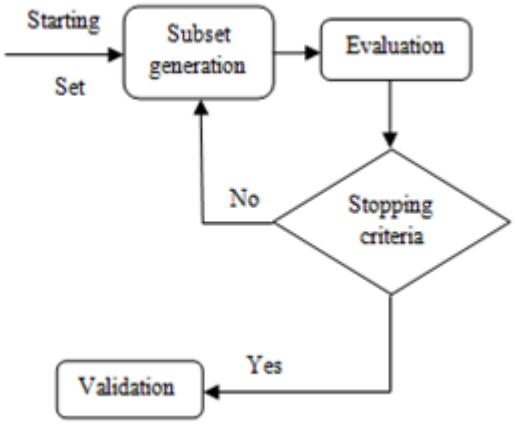

Fig. 2. General feature selection structure

Based on feature selection, words in bug reports are organized according to their feature importance and a given number of words with large values are selected as representative features.

E) Instance Selection:

Instance selection is methods to diminish the number of instances by eliminate noisy and redundant instances. An instance selection algorithm can give a condensed data set by eliminating non-representative instances. There are four instance selection algorithms, namely Iterative Case Filter (ICF), Learning Vectors Quantization (LVQ), Decremental

Reduction Optimization Procedure (DROP), and Patterns by Ordered Projections (POP).In the proposed the iterative case filter (ICF) algorithm defines local set L(X) which contains all cases inside largest hyper sphere centered in X such that the hyper sphere contains only cases of the same class as a instance X. The properties of ICF defined as Coverage of a case is the set of target problems that it can be used to solve.

Coverage (X) ={X'≤ T: X ≤ L (X')} (1)

Reachability of a target problem is the set of cases that can be used to afford a solution for the target.

Reachability(X) = {X ' ≤ T: X ' ≤ L(X)} (2)

III. Implementation

In this part, we introduce the information readiness for applying the bug information lessening. We assess the bug information lessening on bug stores of two vast open source ventures, to be specific Eclipse and Mozilla. Shroud is a multi-dialect programming improvement condition, including an Integrated Development Environment (IDE) and an extensible module framework; Mozilla is an Internet application suite, including some great items, for example, the Firefox program and the Thunderbird email customer. Up to December 31, 2011, 366,443 bug reports more than 10 years have been recorded to Eclipse while 643,615 bug reports more than 12 years have been recorded to Mozilla. In our work, we gather constant 300,000 bug reports for each undertaking of Eclipse and Mozilla ,i.e., bugs 1-300000 in Eclipse and bugs 300001600000 in Mozilla. As a matter of fact, 298,785 bug reports in Eclipse and 281,180 bug reports in Mozilla are gathered since some of bug reports are expelled from bug vaults (e.g., bug 5315 in Eclipse) or not permitted unknown access (e.g., bug 40020 in Mozilla). For each bug report, we download pages from bug stores and concentrate the points of interest of bug reports for tests. Since bug triage plans to foresee the designers who can settle the bugs, we take after the current work to evacuate unfixed bug reports, e.g., the new bug reports or will-not-settle bug reports. In this way, we just pick bug reports, which are settled and copy (in light of the things status of bug reports). Additionally, in bug vaults, a few designers have just settled not very many bugs. Such latent engineers may not give adequate data to foreseeing right designers. In our work, we expel the engineers, who have settled under 10 bugs.

A) Data Design:

A description of all data structures including internal, global, and temporary data structures, database design (tables), file formats of our project following diagrams shows data base design.

B) Internal software data structure:

We use xml files for store large data that data transfer among all other components.

C) Global data structure:

We are using java for development, we will design classes for store data globally.

D) Component Design:

Class diagrams, Interaction Diagrams, Algorithms. Description of each component description required.

TABLE I
BUGS AND ITS DESCRIPTION

Sr. No.	Table Name	Description
1.	Tbasg	Information about assign of Bug
2.	Tbbugttttttbbug Tbbug	Information about the Bug
3.	Tbbugdep	Information about
4.	Tbbugres	Information about Bug Register
5.	Tbclt	Information about the
6.	Tbemp	Information about the Employee
7.	Tbmsg	Information about the
8.	Tbprj	Information about the
9.	Tbprjtec	Information about the
10.	Tbtec	Information about the
11.	Tbusr	Information about the User

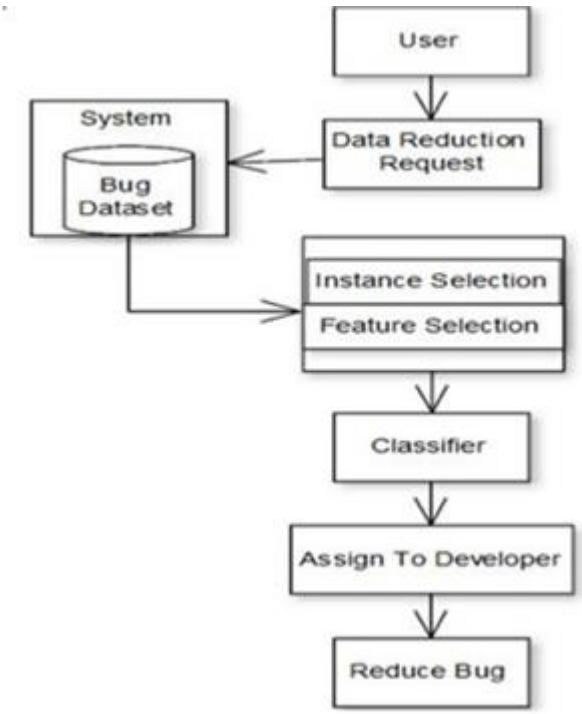

Fig. 3. Flow chart

IV. Conclusion

Bug triage is a costly step of software maintenance in both labor cost and time cost. The proposed system combines the feature selection algorithm (FS) with instance selection

algorithm (IS) in order to reduce the scale of bug data sets as well as improve the data quality. A comparative analysis of popular techniques used for bug triaging has been conducted. We proposed bug assignment approaches using time decay and parameter prioritization. The experimental result shows that both time based knowledge decay and parameter prioritization helps in more precise developer recommendation. We proposed NR Fixer, a prediction model to predict the fixability of bug reports marked as NR. In the future, we first plan to integrate the parameter prioritization model with time decay model. Second, we plan to evaluate the effectiveness of proposed bug report assignment techniques for NR bugs that are predicted as fixable by NR Fixer.

REFERENCES

[1] Y. Fu, X. Zhu and B. Li, "A survey on instance selection for active learning," *Knowledge and Information Systems*, vol. 35, no. 2, pp. 249-283, May 2013.

[2] C. Sun, D. Lo, S. C. Khoo and J. Jiang, "Towards more accurate retrieval of duplicate bug reports," *2011 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011)*, Lawrence, KS, 2011, pp. 253-262.

[3] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," *In Proceedings Of The Thirteenth International Conference On Machine Learning*, 1996.

[4] S. Artzi, "Finding Bugs in Web Applications Using Dynamic Test Generation and Explicit-State Model Checking," in *IEEE Transactions on Software Engineering*, vol. 36, no. 4, pp. 474-494, July-Aug. 2010.

[5] S. Breu, R. Premraj, J. Sillito and T. Zimmermann, "Information needs in bug reports: improving cooperation between developers and users," *CSCW*, 2010

[6] A. K. Farahat, A. Ghodsi and M. S. Kamel, "Efficient Greedy Feature Selection for Unsupervised Learning," University of Waterloo.

[7] V. Cerveron and F. J. Ferri, "Another move toward the minimum consistent subset: a tabu search approach to the condensed nearest neighbor rule," in *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 31, no. 3, pp. 408-413, Jun 2001.

[8] C. C. Aggarwal and P. Zhao, "Towards graphical models for text processing," *Knowledge and Information Systems*, vol. 36, no. 1, pp. 1-21, July 2013.