A Survey on various Techniques for Bug Triage

Vaishnavi B. Sawant[1], Nilesh V. Alone[2]

[1] *ME 2nd year Student, Department of Computer Engineering, Gokhale Education Society's R. H. Sapat College of Engineering, Nashik, Maharashtra, India*
[2] *Assistant Professor, Department of Computer Engineering, Gokhale Education Society's R. H. Sapat College of Engineering, Nashik, Maharashtra, India*

---***---

Abstract - ***A software bug is a problem which causes a computer program or system to crash or produce invalid output or to behave unintended way. Software bugs are unavoidable. Many software companies have to face large number of software bugs. Bug Triage consumes more time for handling software bugs. It is the process of assigning a new bug to the correct potential developer. There are various existing techniques for bug triage. In this paper, we will review some of these techniques. It includes Text categorization, Tossing Graph, Recommendation, Role-Based, Text Mining etc. Most of these techniques provide automatic bug triage. Some of these techniques are further classified.***

Key Words: *Bug Triage, Text categorization, Tossing Graph, Recommendation, Role-Based, Text Mining.*

1. INTRODUCTION

Large no of software projects have bug repository. A Bug Repository is a software repository which contains all the information related to software bugs. A software bug is a problem, which causes a computer program or system to crash or produce invalid output or to behave unintended way. In bug repository, software bug is maintained as bug report. It consists of textual description regarding the bug and updates related to status of bug fixing.

After the formation of bug report, a human triager assigns this bug to a developer, who will try to fix this bug. If the assigned developer cannot fix this bug, then new developer is assigned for fixing that bug. This process of assigning a correct potential developer to fix a new bug is called bug triage.

2. LITERATURE SURVEY

In this paper, we will review some of the existing techniques for bug triage, and some of these techniques are further classified.

2.1 Text Categorization Techniques

Bug Triage consumes more time for handling software bugs. In traditional software development, a human triager is used i.e. expert developers were manually triaged the new bugs. But manual Bug Triage is expensive in time and accuracy because of large number of daily bugs and the lack of expertise of all bugs [1].

To reduce this expensive cost of manual bug triage, Cubrani and Murphy [2] first propose the problem of automatic bug triage. They apply machine learning techniques to assist in bug triage by using text categorization. Text categorization is also known as text classification which is a technique of automatically sorting a set of documents into categories from a predefined set. In this paper, developers will get predicted using the bug's description [2]. This paper used supervised machine learning technique using Naïve Bayes classifier to predict the correct developer.

Anvik et al. [3] present a semi-automated approach for the assignment of bug reports to a developer. This paper uses a supervised machine learning algorithm. For the bug assignment problem, the text documents are the bug reports and the label of the documents are the names of developers suitable to resolve the report. In machine learning, the documents are called instances and the attributes of an instance are called features [3]. A supervised machine learning algorithm takes a set of instances as input with known labels and generates a classifier. Then this generated classifier can be used to assign a label to an unknown instance. In this way this paper used supervised machine learning for bug assignment.

Xuan et al. [4] present a semi-supervised approach for automatic bug triage using text classification. This approach is used to avoid the deficiency of labeled bug reports in existing supervised approaches. This approach combines naive Bayes classifier and expectation maximization [5] (EM) to take advantage of both labeled and unlabeled bug reports. Xuan et al. [4] trains a classifier with a fraction of labeled bug reports. Then the approach iteratively labels numerous unlabeled bug reports. Then they train a new classifier with labels of all the bug reports. From the result of [4], this semi-supervised approach improves the classification accuracy of bug triage by up to 6% and it avoids low-quality bugs.

M. Alenezi et al. [6] propose an automatic approach using text mining to reduce time and cost of bug triaging. Existing techniques also address the problem of bug triaging [7], [8] but these are not efficient. M. Alenezi et al. [6] predict a developer which has relevant experience to solve the new coming report. In this approach, First step is text processing. A Bug report contains unstructured data. Therefore, by using traditional text processing technique text data is transformed into meaningful data. Here, they use summary of bug reports as a description of bugs [6]. After that, bug-term matrix is formed. And it is weighted by term frequency. Then, to reduce the dimensionality and the sparseness of data, different term selection methods are applied. In the next step, using Naïve Bayes approach, classifier gets build. This classifier is then trained using the training data set i.e. bug reports. When a new report arrives, then this bug report follows the same steps to produce the reduced bug-term vector, and then bug report is assigned to a developer using the predictive model [6].

2.2 Tossing Graph Techniques

When a bug report has been assigned, then if the assigned developer cannot fix this bug, then developers can reassign the bug to other developers for fixing that bug. This process of reassignment of bug is called "Bug Tossing". Jeong et al. [9] find out that in manual bug triage, 37 percent - 44 percent of bug reports are "tossed" i.e. reassigned to other developers. Bug tossing is the same as ticket routing [10]. Only a few studies research about the reassignment of bug reports. D'Ambros et al. [11] visualized the life cycle of bugs, with the assignment of developers. Halverson et al. [12] defined patterns in bug reports; from them one was the reassignment of developers. This paper investigates both, assignment and reassignment of developers empirically. In addition, Jeong et al. [9] Builds a model of bug tossing; which reduces the number of reassignment of bug reports. They proposed a tossing graph approach based on Markov chains which captures past bug tossing history to improve bug assignment and reduce unnecessary tossing steps

There are several techniques for bug triage like: Machine learning & IR techniques, Incremental learning, Tossing Graph. These techniques are good for triaging and reducing tossing path; but their accuracy is decrease by various issues like outdated training sets, inactive developers, and imprecise etc. P. Bhattacharya et al. [13] improve triaging and reduce tossing path lengths by using several techniques. They propose three novel extensions to existing techniques. They achieve higher prediction accuracy using richer feature vectors. In previous work, bug title and summaries are used; here they add attributes corresponding to the product–component information for a bug. Next, they apply fine-grained, intra-fold updates which keep the classifier up-to-date at all times. Next step is to constructing multi-feature tossing graphs.

The existing systems machine learning techniques are ineffective for large project. Therefore P. Bhattacharya et al. [14] proposed a method for bug triage. Goal of this paper is to find the optimal set of machine learning techniques to improve bug assignment accuracy in large projects. This paper used a comprehensive set of machine learning tools as well as a probabilistic graph-based model (bug tossing graphs) that lead to highly-accurate predictions, and laid the foundation for the next generation of machine learning-based bug assignment [14]. They used methodology like Choosing effective classifiers and features, Incremental learning, Multi-featured tossing graphs to achieve their goal.

The current technique of Bug Triaging involves modeling the reassignment of bugs as a goal-oriented path model. V. Akila et al. [15] proposed a new framework with the additional capabilities. This models the reassignment of bugs as Enriched Adaptive Bug Triaging System (EABTS) which is based on actual path model. The proposed graph structure captures the relationship among developers as the number of tosses and also captures the propinquity exists among developers. Therefore, this graph structure is enriched. The proposed technique is based on Ant routing. Ant routing is inherently adaptive in nature. The proposed work gives a sub graph that consists of developers who are frequently involved in bug resolution [15].

2.2 Assigning Bug Report through Recommendation

To assist triager, J. K. Anvik et al. [16] propose a presents a machine learning approach to create triage-assisting recommenders. They mention it as ML_{Triage}. The goal of this is to reduce the human involvement in triage. In this ML_{Triage} process, Firstly, reports are automatically selected from a project's issue tracking system. Then, from these selected reports, features i.e. specific piece of data are collected and reports with similar features are grouped under a label. The label shows the class or category to which the features belong. Then, these extracted data and labels are fed to a supervised machine learning algorithm. Then, recommender is created for specific development-oriented decision. Next, when recommender ask to make a prediction for new bug report, that time features are extracted from the new bug report and are fed to the recommender. Then, recommender gives a list of potential labels. This is The ML_{Triage} [16].

2.4 Bug Triage using Vocabulary-based expertise model of developers

D. Matter et al. [17] present an approach to automatically suggest developers who have the appropriate expertise to handle a bug report. D. Matter et al. [17] model the developer expertise using the vocabulary which was found in their source code contributions. Then they compare this vocabulary to the vocabulary of bug reports. They then recommend developers whose contribution vocabulary is lexically similar to the vocabulary of the bug report [17]. An advantage of this approach is that, it doesn't need a record of previous bug reports. D. Matter et al. [17] are able to recommend developers who did not work on bugs previously.

2.5 A Framework for Automated Assignment of Bugs

Olga Baysal et al. [18] present a framework for automated assignment of bug. They proposed framework which is able to conclude a developer's level of expertise by tracking the history of the bugs previously resolved by this developer. Preference elicitation means the problem of developing a decision support system which is capable of generating recommendations to a user, which then assist in decision making. Olga Baysal et al. [18] Approach employs preference elicitation [19] to learn developer predilections in fixing bugs within a given system. Olga Baysal et al. [18] apply a vector space model to recommend experts for fix that bugs. When a new bug report arrives, the system automatically assigns it to the correct developer by considering his or her expertise, current workload, and preferences.

2.6 Cost-Aware algorithm for Bug Reporting System

Existing techniques Treats bug triage as a recommendation problem and propose a solution which is an instance of content-based recommendation (CBR). But CBR suffer from over-specialization i.e. it recommends only the types of bugs that each developer has solved in the past. Therefore, Park et al. [20] propose a new Bug Triage Technique which (1) converts the bug triage into an optimization problem optimizing accuracy and cost (2) adopt a content-boosted collaborative filtering (CBCF) which combines an existing CBR with a collaborative filtering recommender (CF). To achieve these two goals, a key challenge is sparseness. To address this challenge, Park et al. [20] develop a topic-model to reduce the sparseness and enhance the quality of CBCF.

2.7 Fuzzy Set-based Automatic Bug Triaging

A. Tamrawi et al. [21] propose Bugzie, a novel approach for automatic bug triaging. Bugzie is based on fuzzy set-based modeling of bug-fixing expertise of developers. It considers a system to have multiple technical aspects. Each of this is associated with technical terms. Then, Bugzie uses a fuzzy set to represent the developers who are capable of fixing the bugs which are relevant to each term. The membership function of a developer in a fuzzy set is calculated via the terms extracted from the bug reports that (s)he has fixed [21]. When new fixed reports are available then the function gets updated. For a new bug report, its terms are extracted and as per the terms corresponding fuzzy sets are union'ed. Based on their membership scores in the union'ed fuzzy set, Potential fixers will be recommended [21]. Bugzie achieves higher accuracy and efficiency than other approaches.

2.8 Role Analysis-based Automatic Bug Triage

To reduce the workload of Bug Triage, Previous research has focused on various solutions like duplicate detection, automatic bug triage. But they did not analyze the different roles of developer which they play in the Bug Fixing Process. This paper continues previous work to guide the bug fixing process. T. Xang [22] proposes a new bug triage algorithm that recommends appropriate developers to fix the given bugs. T. Xang [22] first, analyzes the different roles that the developers play in the bug fixing process for extracting the further characteristic features. Proposed system considers four roles of developer as: a reporter, as a triager, an assignee, a commenter. Next, they calculate the developers' experience for understanding and fixing submitted bugs. For example, if the developer posted some good quality comments on the bug reports as a commenter, she or he may have the potential ability to fix related bugs that are similar to commented bugs due to good understanding for them [22]. Therefore, the developer may be an appropriate fixer.

3. CONCLUSIONS

Bug Triage is a time-consuming step of handling software bugs. It is the process of assigning a new bug to the correct potential developer. This paper reviews various techniques of bug triage. Some of them are of machine learning, incremental learning, tossing graph, fuzzy-based, role-based etc. Most of these techniques provide automatic bug triage. Apparently all bug triage techniques certainly have their advantages and some drawbacks. So they can be used by considering their strengths and drawbacks.

ACKNOWLEDGEMENT

We would like to thanks Gokhale Education Society's R. H. Sapat College of Engineering for valuable support. We gratefully acknowledge Computer department of their helpful support and guidance for writing this paper.

REFERENCES

[1] Jifeng Xuan, He Jiang, Yan Hu, Zhilei Ren, Weiqin Zou, Zhongxuan Luo, and Xindong Wu, "Towards Effective Bug Triage with Software Data Reduction Techniques", *IEEE transactions* on knowledge and data engineering, vol. 27, no. 1, january 2015

[2] D. Cubrani and G. C. Murphy, "Automatic bug triage using text categorization," in Proc. 16th Int. Conf. Softw. Eng. Know l. Eng., Jun. 2004, pp. 92–97.

[3] J. Anvik, L. Hiew, and G. C. Murphy, "Who should fix this bug?" in Proc. 28th Int. Conf. Softw. Eng., May 2006, pp. 361–370.

[4] J. Xuan, H. Jiang, Z. Ren, J. Yan, and Z. Luo, "Automatic bug triage using semi-supervised text classification," in Proc. 22nd Int. Conf. Softw. Eng. Knowl. Eng., Jul. 2010, pp. 209–214.

[5] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using EM," *Machine Learning.* Hingham, MA, vol. 39, pp. 103-134, May 2000.

[6] M. Alenezi, K. Magel, S. Banitaan "Efficient Bug Triaging Using Text Mining" *Academy Publisher*, 2013.

[7] W. Zou, Y. Hu, J. Xuan, and H. Jiang, "Towards training set reduction for bug triage," in Proc. *IEEE 35th Annual Computer Software and Applications Conference*, Washington, DC, USA: IEEE Computer Society, 2011, pp. 576–581.

[8] A. Tamrawi, T. Nguyen, J. Al-Kofahi, and T. Nguyen, "Fuzzy set and cache-based approach for bug triaging," in Proc. 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering, 2011, pp. 365–375.

[9] G. Jeong, S. Kim, and T. Zimmermann, "Improving bug triage with tossing graphs," in Proc. Joint Meeting 12th Eur. Softw. Eng. Conf. 17th ACM SIGSOFT Symp. Found. Softw. Eng., Aug. 2009, pp. 111–120.

[10] Q. Shao, Y. Chen, S. Tao, X. Yan, and N. Anerousis, "Efficient ticket routing by resolution sequence mining," in Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, Aug. 2008, pp. 605–613.

[11] M. D'Ambros, M. Lanza, and M. Pinzger. "A Bug's Life" Visualizing a Bug Database. *In Proceedings of IEEE International Workshop on Visualizing Software for Understanding and Analysis* (VisSoft 2007), pages 113–120, Banff, Alberta, Canada, 2007. IEEE Computer Society.

[12] C. A. Halverson, J. B. Ellis, C. Danis, and W. A. Kellogg. "Designing task visualizations to support the coordination of work in software development." In CSCW '06: Proceedings of the 2006 20th Anniversary Conference on Computer Supported Cooperative Work, pages 39–48, 2006.

[13] P. Bhattacharya P. and Neamtiu I.: *Fine-grained incremental learning and multi-feature tossing graphs to improve bug triaging*, in Proc. of ICSM'10, pp.1-10 (2010).

[14] P. Bhattacharya, L. Neamtiu, C. R. Shelton "Automated, highly-accurate, bug assignment using machine learning and tossing graphs" , 2012.

[15] V.Akila, Dr.G.Zayaraz, Dr.V.Govindasamy "Effective Bug Triage – A Framework", International Conference on Intelligent Computing, Communication & Convergence, 114 – 120, 2015

[16] John Karsten Anvik "Assisting Bug Report Triage through Recommendation" The university of British columbia November, 2007

[17] D. Matter, A. Kuhn, and O. Nierstrasz, "Assigning bug reports using a vocabulary-based expertise model of developers," in Proc. 6th Int. Working Conf. Mining Softw. Repositories, May 2009, pp. 131–140

[18] O. B. Michael and G. C. Robin, "A Bug You Like: A Framework for Automated Assignment of Bugs.," *IEEE 17th international conference*, 2009.

[19] L. Chen and P. Pu. "Survey of preference elicitation methods." Technical report, Swiss Federal Institute of Technology in Lausanne (EPFL), 2004.

[20] J. W. Park, M. W. Lee, J. Kim, S. W. Hwang, and S. Kim, "Costriage: A cost-aware triage algorithm for bug reporting systems," in Proc. 25th Conf. Artif. Intell., Aug. 2011, pp. 139–144

[21] A. Tamrawi, T. Nguyen, J. Al-Kofahi, and T. Nguyen, "Fuzzy sat based automatic bug triaging" 2011

[22] T. Zhang, G. Yang, B. Lee, I. Shin "Role Analysis-based Automatic Bug Triage Algorithm", 2012