A Survey On Automatic Bug Assignment To Developer Using Data Reduction Techniques.

Ashwini Khandagale¹, Pranali Bhujbal², Pragati Kokane³, Prof. Wavhal D.N.⁴

¹ Department of Computer Engineering, Jaihind College Of Engineering, Kuran, Savitribai Phule Pune University, Pune, Maharashtra, India
² Department of Computer Engineering, Jaihind College Of Engineering, Kuran, Savitribai Phule Pune University, Pune, Maharashtra, India
³ Department of Computer Engineering, Jaihind College Of Engineering, Kuran, Savitribai Phule Pune University, Pune, Maharashtra, India
⁴ Head Of Department of Computer Engineering, Jaihind College Of Engineering, Kuran, Savitribai Phule Pune University, Pune, Maharashtra, India

Abstract—Software industries spend more than 45 percent of expense in managing programming bugs. A software bug is an error, flaw, failure, or fault in a computer program or system that causes it to produce an incorrect or unexpected result, or to behave in unintended ways. Most bugs occur from mistakes and errors made by human being in either a program's source code or its design, or in frameworks and operating systems used by such programs. Few bugs arise do to a compilers producing incorrect code. Software repositories are large-scale databases for storing the output of software development example source code, bugs, emails, and specifications. To reduce the time cost in manual work, content characterization procedures are connected to direct programmed bug triage.

The process of fixing bug is bug triage to assign a expect developer to a new bug. The aim of the technique is to addresses the problem of data reduction for bug triage. An unavoidable stride of altering bugs is bug triage, which expects to effectively allocate a designer to another bug. The techniques show that data reduction can effectively improve the accuracy of bug triage by reducing the data scale and manual work. The systems combine instance selection with feature selection to simultaneously reduce data scale on the bug dimension and the word dimension. To determine the order of applying instance selection and feature selection, the system extract attributes from historical bug data sets and build a predictive model for a new bug data set.

Index Terms—Bug, Bug-triage, Source code, Instance selection, Feature selection, Failure, Software Repositories.

I. INTRODUCTION

A program that contains a large number of bugs or bugs that seriously interfere with its functionality is said to be buggy. Reports detailing bugs in a program are commonly known as bug reports, defect reports, fault reports, problem reports, trouble reports, change requests and so on. In modern software development industries, the large databases are required to store bug repositories. The data mining is turn out as promising to handle data. Using arbitrage data mining repositories, real world software problems can be solved. In the experiments, evaluation on the reduction of data for bug triage on bug reports were applied over two large open source projects, namely Eclipse and Mozilla. Experimental results show that applying the instance selection technique to the data set can reduce bug reports but the accuracy of bug triage may be decreased; applying the feature selection technique can reduce words in the bug data and the accuracy can be increased[8]. Thus if both the preprocessing techniques are combined, can increase in the accuracy, as well as reduction in the size of the bug reports and words can be easily obtained. For example, when 60 percent of bugs and 80 percent of words are removed, the accuracy of Naïve Bayes on Eclipse improves by 3 to 15 percent and the accuracy on Mozilla improves by 2 to 7 percent. Based on the attributes from historical bug datasets, constructed predictive model can provide accuracy of 80.8 percent for predicting the reduction order. Based on top node analysis of the attributes, results show that all attributes are essential for prediction and an individual attribute cannot predict this without other attributes. Bugs trigger errors that can in turn have a wide variety of ripple effects, with varying levels of inconvenience to the user of the program. Some bugs have only a delicate effect on the program's functionality, and may thus lie unnoticed for a long time[2]. More serious bugs may cause the program to crash or freeze. Others qualify as security bugs and might for example enable a malicious user to bypass access controls in order to obtain illicit privileges. In this report, the system address the problem of data reduction for bug triage, i.e., how to reduce the bug data to save the labor cost of developers and improve the quality bug data set to facilitate the process of bug triage.

Data reduction for bug triage aims to construct a small-scale and high-quality set of bug data by removing bug data set and words, which are redundant or non-instructive. In our work, we combine existing techniques of instance selection and feature selection to simultaneously reduce the bug dimension and the word dimension[8]. The reduced bug data set contain smaller quantity bug reports and words than the original bug data and provide similar information over the original bug data. We evaluate the reduced bug data according to two criteria: data set scale and the accuracy of bug triage. To avoid the bias of a single algorithm, we empirically examine the results of four instance selection
algorithms and four feature selection algorithms. This project presents the problem of data reduction for bug triage. This problem aims to augment the data set of bug triage in two aspects, namely to simultaneously reduce the scales of the bug dimension and word dimension and to improve the accuracy of bug triage. We propose a combination approach to addressing the problem of data reduction. This can be viewed as an application bug repositories.

II. LITERATURE SURVEY

In this paper, introduces a dynamic test generation technique for the dynamic Web applications. The technique uses both combined concrete and symbolic execution and explicit-state model checking. This technique generates tests automatically, by capturing logical constraints on inputs it runs the tests, and minimizes the conditions on the inputs so that resulting bug reports are small and useful in finding and fixing the underlying faults[1].

In this paper, author surveyed the problem of the developer prioritization, which ranks the contributions of developers. We mainly discover two aspects, namely modeling the developer prioritization in a bug repository and assisting predictive tasks with our model. First based on social network technique, we model how to assign the priorities of developers. Second, we consider the developer prioritization to improve bug repository [2].

In this paper, author deals with Software quality which is vital for the success of a software project. Although many software quality assurance activities such as testing, inspection, static checking, etc have been proposed to improve software quality, in reality software systems are often shipped with bugs. For a large and evolving software system the project team could receive a large number of bug reports over a long period of time. For example, around 4414 bugs were reported for the Eclipse project in 2009. Once a bug report is received and confirmed, the project team should locate the source code files that need to be changed in order to fix the bug. However, it is often costly to manually locate the files to be changed based on the initial bug reports. For a large project consisting of hundreds or even thousands of files. The manual bug localization is a painstaking and time-consuming activity [4].

The paper [6] tells us about how program repair. It is a tedious and difficult activity that requires lots of resources spent on locating and fixing program bugs. They proposed automated program repair, reducing the cost of software maintenance. In large-scale programs consumes a lots of time for recompiling and reinstalling the modified program which is the result of automatically repairing a bug by modifying the program source code. They introduce a recompilation technique called weak recompilation. Recompile is used for suppressing the time cost and to make the repair process more efficient. The advantage of weak recompilation is that reinstallation cost will be cut down completely and redundant recompilation cost can be also suppressed. They also built WAutoRepair which is a system that enables scalability to fix bugs in large-scale C programs with high efficiency. The results of the experiments confirmed that their repair system significantly outperforms Genprog which was a famous approach for automatic program repair[6].

In this paper, proposed how to make a good report. They are conducting a survey between developers and users on open source projects such as APACHE, ECLIPSE, and MOZILLA. The analysis the betray of information between what developers need and what users supply. Almost all developers are uses terms such as reproduce, stack traces, and test cases.

For simplification but which are, the toughest to contribute a users. So this kind of perception is need to designing new bug tracking tools which are used to collecting and proving helpful information. Author proposes a CUEZILLA prototype. Which is measures the quality of new bug reports. It also recommends which elements should be added to improve the quality. They trained CUEZILLA on a 289 bug reports and rates by developers[7].

In this paper used data reduction technique is used to reduce the data set scale and improve the quality of data set. They combine two reduction techniques instance selection and feature selection. And predict which techniques used first. These techniques are used to reduce the data scale into text matrix having two dimensions. The result produce that bug data set can effectively reduce by the data reduction techniques and also increase the accuracy of bug triage[8].

III. MOTIVATION

A small-scale and high-quality set of bug data can be obtained by removing bug reports and words, which are redundant or non-informative. This is called the Data reduction in Bug triage. Bug triage is an age-old technique that is used to handle the software bugs, whose utilization can often be time consuming. Bug triage aims to assign a correct developer to fix a new bug. In traditional software development, new bugs are manually triaged by an expert developer, i.e., a human triage. Due to the large number of daily reported bugs and the lack of knowledge of all the bugs, manual bug triage is expensive in time cost and also results in low accuracy. In proposed system avoid expensive cost of manual bug triage and it can be used to assist human triggers rather than replace them. It is helpful to improve the data quality and the cost of prediction is not expensive. This system is helpful for analysts for analysis of results. It is helpful for better decision making and more accurate prediction.

IV. OBJECTIVE

The data reduction along with bug triage has this objective, 1) Reducing the data scale

2) Improving the accuracy of bug triage

3) Assuring new bug is assigned to expert developer

4) To reduce the number of instances by removing noisy and redundant instances. (Instance selection use)
5) A reduced data set by removing non-representative instances.

V. EXISTING SYSTEM

A time-consuming step of handling software bugs is bug triage, which Objective is to assign a correct developer to fix a new bug. In conventional software development, new bugs are manually triaged by an expert developer, i.e., a human triage. Due to the large number of daily bugs and the lack of expertise of all the bugs, manual bug triage is costly in time cost and low in accuracy. To reduce the time cost in manual work, text classification techniques are applied to conduct automatic bug triage in proposed system.

VI. PROPOSED SYSTEM

In fig1. we propose how to reduce the bug information to save the work cost of developers and improve the quality to facilitate the process of bug triage. Data reduction for bug triage expects to build a small-scale and high-quality set of bug data by removing bug reports and words which are not informative and redundant. We use existing techniques of instance selection and feature selection to decrease the bug dimension and the word dimension. The reduced bug data contain less bug reports and fewer words than the original bug data and provides similar information over the original bug data. Proposed system is evaluating the reduced bug data, which are size of a data set and the correctness of bug triage.

In the proposed system the input is in the form of bug data set. The bug data set consists of bug report and the details of the developer who have worked on the respective bug. The bug report is mainly separated in two parts:
1. Summary and
2. Description

The proposed system gives predicted results in the form of output. There are two types of users in the proposed system:
1. Developer
2. Tester

In bug dimension we reduce noisy or duplicate bug report to decrease the number of historical bug. In word dimension to reduce noisy and duplicate words. In the system bug triage use to predict the correct developer who can fix the bugs. We follow existing work of developer to solve the problem of fixing bugs. We predict a new bug to the developer by his/her expertise. In system bug repositories, several developers only fixed very few bugs. Such inactive developer does not provide sufficient information for assignment correct developers. In proposed system we eliminate the developers, who have fixed less than 10 bug.

1. Bug triage:
In bug triage, a bug data set is converted into a text matrix with two dimensions, namely the bug dimension and the word dimension. Each row of the matrix indicates one bug report while each column of the matrix indicates one word.

2. Classification:
We are going to use classification techniques to take the input data set and divide it under different classification label to form the sets. We can do classification using supervised and unsupervised learning.

3. Data reduction
To reduce the scale of bug data sets as well as improve the data quality.

4. Word dimension:
To remove noisy duplicate words in a data set. By removing uninformative words we can improves the accuracy of bug triage.

5. Bug dimension:
To remove uninformative bug reports as soon as the accuracy may be decreased by removing bug reports.

VII. COMPARITIVE STUDY

<table>
<thead>
<tr>
<th>Existing System</th>
<th>Proposed System</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. In Existing System guaranteed compute resources always ready to accommodate job execution under reserved instance pricing.</td>
<td>1. In Proposed System aim to present the problem of data reduction for bug triage. This problem aim to augment the data set of bug triage in two aspects, namely to simultaneously reduce the scale of bug dimension and the word dimension and to improve the accuracy of bug triage.</td>
</tr>
<tr>
<td>Existing Technique:- Text classification techniques.</td>
<td>Proposed Algorithm:- Data reduction based on FS-IS</td>
</tr>
<tr>
<td>Drawbacks:- Software bugs are</td>
<td>Advantages:- Maintain in Quality in bugs.</td>
</tr>
</tbody>
</table>
predictable and fixing bugs is expensive in software development.
- Bug decrease taking too much of time.
- Manually developer have to clear the bugs.
- Automatically clear the bug.
- Cost is low.

V. CONCLUSION

The goal was to achieve the system which will reduce the human effort in software development to assign bugs to the respective developers. Our proposed Bug Triage System approximately try to achieve the same one. Proposed system focused on reducing bug data set in order to have less scale of data and storage. For that we have used feature selection and instance selection techniques of data mining as well as we have used Naïve bayes classifier for classification. Our experimental results showed that this data reduction technique will give correct data as well as it will reduce the data scale.

REFERENCES


Ashwini R. Khandagale:
She studying in B.E computer in Jaihind College of Engineering, Kuran. Her area of interest are programming language like Java,C++
Email:101996ash4@gmail.com

Pranali G. Bhujbal:
She studying in B.E computer in Jaihind College of Engineering, Kuran.. Her area of interest is Developing Android Application.

Pragati S. Kokane:
She studying in B.E computer in Jaihind College of engineering,kuran. Her area of interest is Programing language C,C++,Java

Prof. D. N. Wavhal:
Head Of Department of Computer Engineering, Jaihind College of Engineering, kuran, Maharashtra, India.His area of interests are Computer programming languages ,Computer Network , Data structure, Software Engineering,JOT.
Email: