ISSN: 2348 - 2117

International Journal of Engineering Technology and Computer Research (IJETCR)
Available Online at www.ijetcr.org
Volume 3; Issue 3; May-June; Page No. 01-04

A Review of Cluster Based Mutant Reduction Technique in Mutation Testing

Neetu Rani[1], Jyoti Chaudhary[2]

[1]M.Tech Scholar, TIT&S[1], Bhiwani, Haryana

neeturathee22@gmail.com,

[2]Assistant Professor, TIT&S[1], Bhiwani, Haryana

jyotiseptember20@gmail.com

Abstract:
Mutation Testing is a white box software testing technique, that involves modifying a program's source code or byte code in order to test sections of the code that are seldom or never accessed during normal tests execution. Mutation testing improves the quality& reliability of the software but still it is an exhaustiveand computationally expensive testing technique. So, in order to reduce this burden a number of cluster based techniques are being widely used to reduce the number of mutants generated. This paperreviews and analyses these techniques and focuses on the most optimized technique of reducing the computation cost.

Keywords: Mutation, reliability mutant, cluster, optimized, advanced environment.

INTRODUCTION:

Mutation Testinga type of white box unit testing technique is considered to be the most efficient in detecting faults.Mutation testing or (Program mutation or Mutation analysis) is used to evaluate the quality of existing software tests. It involves modifying the program with small changes. These Small changes are said to be mutants. This form of testing deals with mutating parts of the program intentionally and then detecting them.
Mutation testing, in spite of being most powerful and effective, is not widely used because of high computational cost associated with it.

Mutation testing was originally proposed by Richard Lipton [1] as a student in 1971, and first developed and published by De Millo, Lipton and Sayward. The first implementation of a mutation testing tool was by Timothy Budd as part of his PhD work (titled Mutation Analysis) in 1980 from Yale University. Mutation testing is based on two hypotheses. The first is the competent programmer hypothesis. This hypothesis states that most software faults introduced by experienced programmers aredue to small syntactic errors. The second hypothesis is called the coupling effect. The coupling effect asserts that simple faults can cascade or couple to form other emergent faults.

Mutation Testing can be used for testing software at the unit level, integration level and thespecification level.
While mutation testing is a powerful and general technique, a number of associated problems have limited its practical impact. Mutation Testing has three main problems in our view.

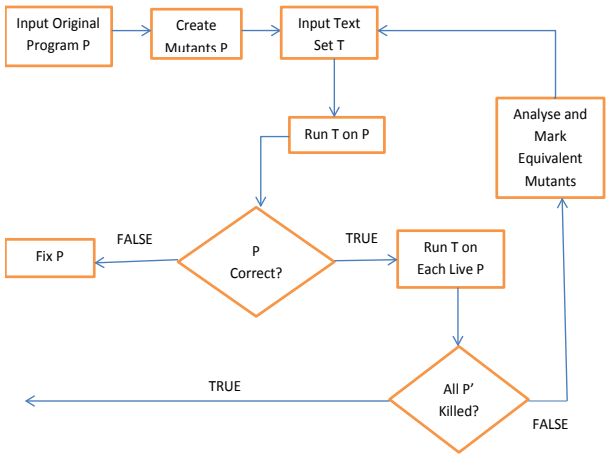

Figure1: Generic Process of Mutation Testing

The first limitation of mutation testing is the large number of mutants, because program may have a fault in many possible places and with only one inserted semantic fault we will have one mutant. Thus, a large number of mutants will be generated in the mutant generation phase of mutation testing. Typically, this is a large number for even small program.This problem leads to a very high execution cost because the test cases are executed on not only original program but also each mutant.

The second limitation of mutation testing is realism. Mutations are generated by single and simple syntactic

Corresponding author: Neetu Rani

changes; hence they do not denote realistic faults. While according to Langdon et al. 90 percent of real faults are complex. In addition, it is not sure that we have found a large proportion of real faults present even if we have killed all the killable mutants this is one of big limitations for applying mutation testing.

The third limitation of mutation testing is equivalent mutant problem. In fact, many mutation operators can produce equivalent mutants which have the same behavior as original program. In this case, there is no test case which could "kill" that mutants and the detection of equivalent mutants often involves additional human effort. These are main limitations of the mutation testing i.e. mutant reduction technique & the execution reduction technique. In this paper we focus on the mutant reduction technique briefly.

Mutant Reduction Techniques

The Computational Cost is large due to the large number of mutants being generated and executed. Even a small program can have hundreds of mutants. There are mainly four techniques for reducing the number of mutants [2] i.e. Mutant Sampling, Mutant Clustering, Selective Mutation and Higher Order Mutation.

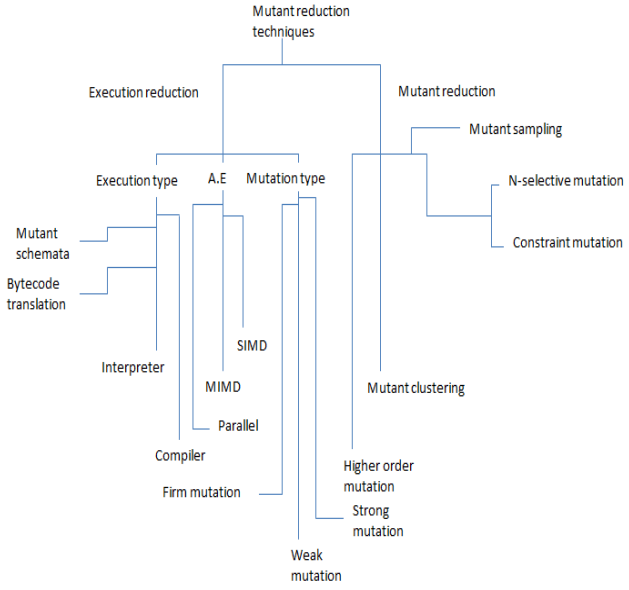

Figure 2: Overview of the mutant reduction techniques

1. **Mutant Sampling**:-This approach is very simple and fast that selects randomly a small subset of mutants from the whole set.
2. **Mutant Clustering**:-This approach uses the clustering algorithms for mutant's selection instead of random selection. The idea of Mutant Clustering was firstly proposed in Hussein's master's thesis [3].
3. **Selective Mutation**:-This approach initially called "constrained mutation" [5] reduces the number of mutants by selecting a subset of mutation operators—called sufficient mutation operators.
4. **Higher Order Mutation**: Initially this approach considers only first order mutants, created by the injection of a single fault. Often these first order mutants denote trivial faults that are easily killed. Higher order mutants are created by the insertion of two or more faults

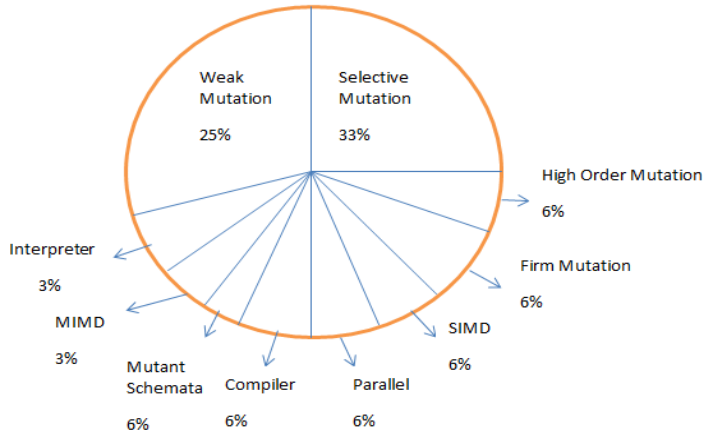

Figure 3: Percentage of publications on Mutant Reduction Techniques (Jia and Harman, 2011)

Among these mutant reduction techniques we focus in this paper on Mutant Clustering. Subsequentsection 2 explains the detailed study of clustering. The applications of mutation testing are explained in section 3. Section 4 gives the concluding remarks to the paper.

2. Mutant Clustering

The idea of mutant clustering was first proposed in Hussain's thesis [3]. It chooses a subset of mutants using clustering algorithms [8]. The process starts from generating all first order mutants after than a clustering algorithm is applied to classify the first order mutants into different clusters based on the killable test cases. Each mutant in the same cluster is guaranteed to be killed by a similar set of test cases. Only a small number of mutants are selected from each cluster to be used in mutation testing, and the remaining mutants are discarded. Clustering is the process of organizing objects into groups whose members are similar in some way. Therefore a *cluster* is a collection of objects which are "similar" between them and are "dissimilar" to the objects belonging to other clusters for exploratory mining, and technique for statisticaldata analysis including fields machine learning, pattern recognition, image analysis, information retrieval, and bioinformatics.So, we can say that unsupervised learning is the problem of clustering. The distance between two

clusters involves some or all elements of the two clusters. The clustering method determines how the distance should be computed.

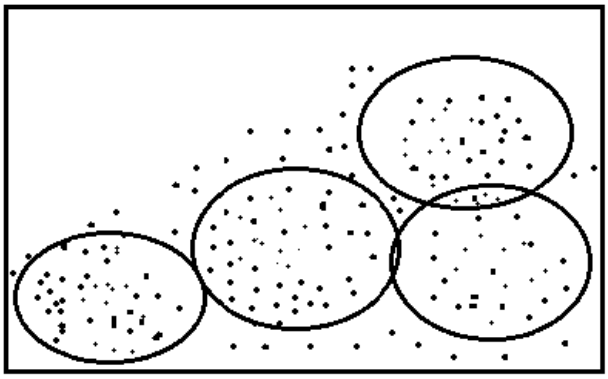

Figure 4: Cluster diagram, each one cluster is different from other clusters

There are many clustering methods available, and each of them may give a different grouping of a dataset. The choice of a particular method will depend on the type of output desired, the known performance of method with particular types of data, the hardware and software facilities available and the size of the dataset. In general, clustering methods [4] may be divided into two categories based on the cluster structure which they produce. The non-hierarchical methods divide a dataset of N objects into M clusters, with or without overlap. The main classifications of the clustering methods are:

1. Partitioning Methods
2. Hierarchical methods

Inpartitioning method [7] the classes are mutually exclusive, and the less common clumping method, in which overlap is allowed. Each object is a member of the cluster with which it is most similar; however the threshold of similarity has to be defined. It can be further divided into K-means algorithm and fuzzy c-means algorithm. The hierarchical methods [6] produce a set of nested clusters in which each pair of objects or clusters is progressively nested in a larger cluster until only one cluster remains. It can be further divided into agglomerative algorithm and divisive algorithm.

2.1 Previous Work

In order to handle the computational problem of mutation testing, researchers have proposed a number of strategies. Wong and Mathur [11] in proposed a way to reduce the computational cost, by reducing the number of mutants. Mathur suggested using the approximation technique to reduce the number of the mutants and another technique of random mutant selection. These two approaches are referred as 'do fewer' approaches in or alternate mutation. There are other approaches as well such as 'do smarter and do fewer'. Shamila Hussian in proposed also a technique for the reduction of computational cost incurred by large number of mutants. According to Shamila [3] by cluster the similar mutants together, so by selecting mutants from each cluster we can reduce the number of mutants required without reducing the power of the set of mutants. Mutant power will measured in terms of the set of test cases required to kill the entire mutant set.

Although researchers have proposed several strategies to control the number of mutants, still they are not precise. It is still a wide research area, and therefore attracts and invites more research to handle the problem.

3. The Application of Mutation Testing

Mutation testing is used to "test your test cases". To make small mutations to your application and then run your tests to make sure they catch the bugs added to these mutations. Mutation testing is widely applied in design models, specifications, databases, tests, xml schemas, and other types of stack overflow, software artifacts, and web applications. Mutation testing is applied to test both i.e.

1. Program mutation and
2. Specification mutation.

Program mutation [12] is strictly applied on the source code of program and it comes into white box testing. It applied on both the unit level and the integration level. In spite of this specification mutation [13] is applied on the program specification and it comes in the consideration of black box testing. Mutation testing is applied into a variety of languages these are:

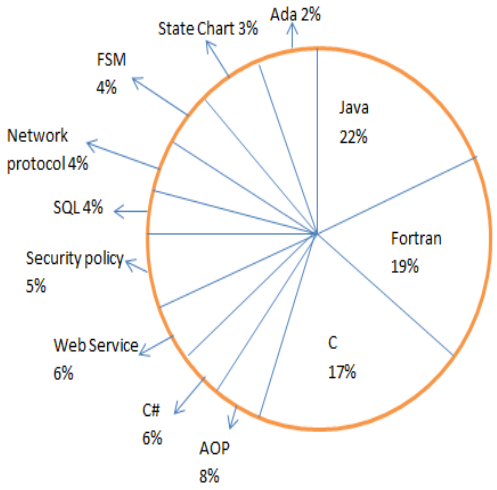

Figure 5: Percentage of publications addressing Mutation testing.

As shown in above figure mutation testing [2]is available for Fortran, Ada, C, Java, C#, SQL, Aspect-Oriented programming, Other program mutation application is Lustre programs, PHP, Cobol programs, Matlab and spreadsheets. And mutation testing for Formal specifications, Running environment, Web services, Networks Security policy. In spite of these mutation testing support other testing activities i.e. Test data generation and Regression testing.

4. Conclusion and Scope

In this paper, the generic process of mutation testing is described. The paper has explored the cluster based techniques used by different researchers. The previous work shows that till date this technique is not being widely used for mutation testing. So, this area needs a lot of research to be done and is attracting a number of researchers for their future research.

REFERENCES:

1. R. Lipton, “Fault Diagnosis of Computer Programs,” Student Report, Carnegie Mellon University, 1971.
2. Yue Jia, Mark Harman “An Analyses and Survey of the Development of Mutation Testing”, IEEE Transactions of Software Engineering, 2010.
3. S. Hussain, “Mutation Clustering,” Master’s Thesis, King’s College London, UK, 2008.
4. Athman Bouguettaya "On Line Clustering", IEEE Transaction on Knowledge and Data Engineering Volume 8, No. 2, April 1996.
5. Euripides G.M. Petrakis and Christos Faloutsos "Similarity Searching in Medical Image Databases", IEEE Transaction on Knowledge and Engineering Volume 9, No. 3, MAY/JUNE 1997.Data
6. Rob Short, Rod Gamache, John Vert and Mike Massa "Windows NT Clusters for Availability and Scalability" Microsoft Online Research Papers, Microsoft Corporation.
7. Jim Gray "QqJim Gray’s NT Clusters Research Agenda" Microsoft Online Research Papers, Microsoft Corporation.
8. Bruce Moxon "Defining Data Mining, The Hows and Whys of Data Mining, and How It Differs From Other Analytical Techniques" Online Addition of DBMS Data Warehouse Supplement, August 1996.
9. Willet, Peter "Parallel Database Processing, Text Retrieval and Cluster Analyses" Pitman Publishing, London, 1990.
10. S.Bandyopadhyay, U.Maulik, “An Evolutionary Technique based on k-Means Algorithm for Optimal Clustering in RN”, Information Sciences, Vol. 146, pp.221-237, 2002.
11. P. Mathur, W. E. Wong, Reducing the cost of mutation testing: an empirical study, Journal of Systems and Software, Volume 31, Issue 3 (December 1995), Pages: 185-196, Year of Publication: 1995, ISSN: 0164-1212
12. R. A. DeMillo, “Program Mutation: An Approach to Software Testing, “Georgia Institute of Technology, Technical Report, 1983.
13. A. S. Gopal and T. A. Budd, “Program Testing by Specification Mutation,” University of Arizona, Tucson, Arizona, Technical Report 83-17, 1983