IJREAT International Journal of Research in Engineering & Advanced Technology, Volume 4, Issue 2, April - May, 2016
ISSN: 2320 – 8791 (Impact Factor: 2.317)
www.ijreat.org

A Survey on Regression Testing Technique and their Metrics

Jyoti Shinde, Privi Dubey, D.Sirisha, Dr. Manjula R.
School of Computing Science, Vellore Institute of Technology, Tamil Nadu.

Abstract- Regression testing is a fundamental maintenance and testing activity that ensures that the current changes made doesn't affects the previous functionality of the code. But the regression testing tends to be expensive while reusing the whole test suites that were used on initially developed programs. There are various algorithms that exist to make regression testing more optimal. We have surveyed such algorithms and have compared and analyzed these algorithms on perceptible and subjective metrics such as execution time, precision, type of testing and so on. This survey provides an insight about the above algorithms with their strengths and weakness, and which one will be an ideal choice for regression testing.

Keywords- Regression testing, Test selection, Prioritization, Minimization, Optimization

1. Introduction

The process of examining the changes that were introduced to the systems, in order to check if the older version is still functioning provided it produces the same efficiency. It is an exclusive process and exhausts the system with intense execution time. Although regression testing do not necessarily initiates the rerun of the whole suite but choosing a set that has the least fault is tricky. Software modification usually affects the functionality that has been optimally working up until now. Thus regression verification is one of the demanding steps in software testing [1] [2]. Basically it is done in two ways; one of them is to re-execute all the test case in the system; second is to re-execute a part of test suite. Quintessential method is to test the whole suite anew, but insufficient resources and time period is the major flaw in this approach. As executing all the test cases each time a modification occurs, will cost extensively. Thus second method of executing a each of the test suite is a better method. An effectual software development process [2] expedient three important objectives: scope, cost and time. The software development process contains several steps, all of them holds up different measures, like time and costs. The essential goal for all organization is to reduce the time and cost as much as possible.

The software testing is a crucial procedure as it checks the exactness and effectiveness of the software to resolve if all the customer's demands or requirements are fulfilled. Many steps are involved in correcting the software defects.

1. Initializing test plans,
2. Test case designing,
3. Software testing (as per the test cases),
4. Recording the test consequences and error reporting

To get good quality software, requisites of the customers should be fullfill before the delivery of product or software. After the delivery of the product, if the user requests any change in the product or software to recorrecct the current bugs of software or the updation is made in the software to support the new requisites or changes in the technology [2], in order to acquiesce the software to support those rquisites, the process may require update process.A trivial defect might result in considerable amount of side effect, whereas a secondary bug of a major defect can stimulate zero to a just a minor side effect[3].
Thus to make a better and more feasible software we optimize the regression testing procedure.

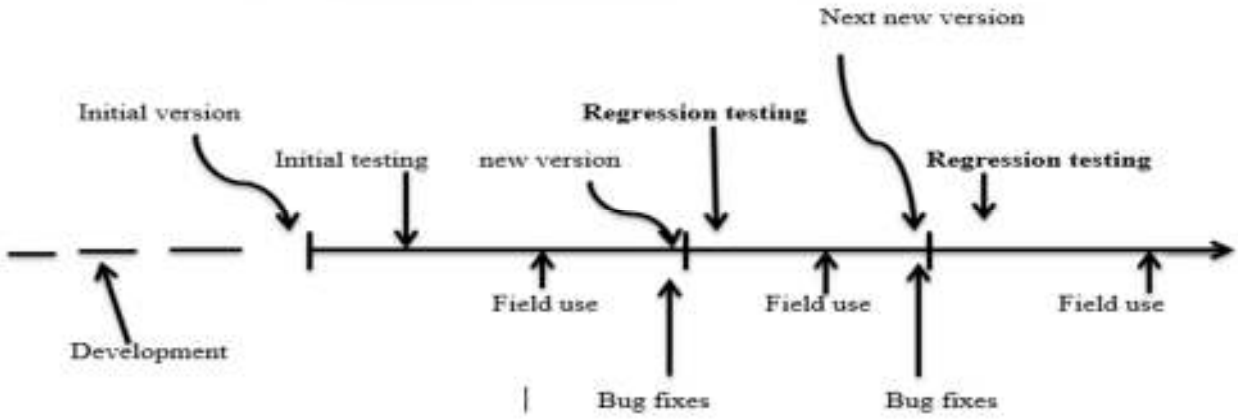


Figure 1) Life Of A Software.

There are various techniques such as Test case selection, Test case prioritization technique and Test case minimization technique. They are used so as select only those test cases that is compatible with the new changes. In test case prioritization the execution orders are aligned so as to increase the efficiency. The essential objective of this technique is the execution of those test cases that were used in earlier process of testing and whichever efficiently gets maximum testing coverage they are given higher priority. Whereas Test suite minimization attempts to cost

reduction saving and test reusage during maintenance .And eliminating test suites that are redundant.

2. Preliminaries

Software that has been changed must be validated with focus in the following objectives:

a) Assure that the new requirements have been implemented correctly.
b) Ensure that new requirements do not affect previous functionalities.
c) Test those parts of the software that have not been checked before.

The process of retesting the software to ensure that modified program still work correctly with all of the test cases used to test original program is known as regression testing.

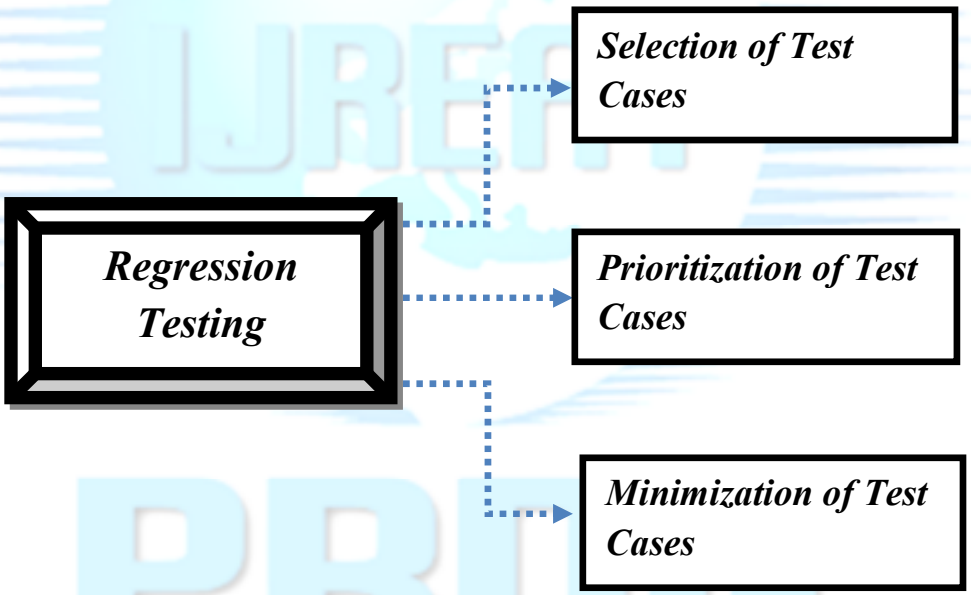


Figure 2) The Regression Testing Methodology.

In recent years, the software testing association has given sufficient importance to the subject of regression testing. A few approaches have been presented to maximize the test suite of each SUT: minimization of test cases, selection of test cases and prioritization of test cases. Test cases minimization is the approach to denote what test cases are redundant and then remove them from test suite. Test suite selection chooses a subset of test case that will be used to test the parts that have been changed in software. Finally, test suite prioritization identifies the best order of test case that maximizes some property, such as fault detection.

As seen in Figure 1 describes a common illustration of a timeline during the life of a software system. As shown in figure, the execution of regression testing is a considerable fraction of the system's lifetime. Unfortunately, regression testing cannot always be completed because of the frequent changes and updates of a system. When a program has been modified, it is essential to guarantee that the changes which works correctly.

Unmodified modules of the program have not been influenced by the modification. It is exceptionally necessary because even small modifications, in one part of a program, may have a negative impact in other independent parts of the program. However the modified program can produce correct outputs with test cases particular constructed for that version. But it can produce incorrect outputs to other test cases which original program returns correct outputs. So, during the regression testing, the changed program is executed with all regression tests to check if it maintains the same functionalities present in the original program.

2.1.Test Case Selection

Number of regression test selection techniques is described. A survey describes several families of techniques [4]. Here the approaches of each have been described and examples are provided for each technique.

2.1.1 Minimization Techniques

In minimization test selection technique the main aim is to select the minimal sets of test cases from T so as it yield covers the affected area of P [4]. As for an example, to define the relationship between test case and basic blocks, some systems make use of systems of linear equations. To identify a subset T' of T and it makes sure that each segment that is statically reachable from a changed segment is applied by at least one test case in T that also exercises the modified segment.

2.1.1.1 Dataflow Techniques

Data flow regression test technique select test cases that practice data interactions that have been changed by modification. For an example, some approach needs that every definition use pair that is deleted from P, new in P, or modified for P be tested. The approach selects every test case in T, that when executed on P, exercised modifies definition pairs [4].

2.1.1.2 Safe Techniques

Among all selection techniques, Minimization and data flow techniques are not designed to be safe. The unsafe techniques tend to fail to select the test case that would have revealed a fault in the modified program. Conversely, safe test selection techniques guarantees that the select subset, T', contains all test cases in the primary test suite T that reveals the faults in P', this

happens when an explicit set of safety conditions are satisfied. Many safe test selection techniques are proposed.

2.1.1.3 Ad Hoc/Random Techniques

When a time restraint prevents the use of retest all the approaches, but selection tools are not available. Another easy approach is to arbitrary selecting a predefined number of test cases from T. All existing test cases simply make use of the retest all techniques. It adequately selects all test cases present in the suit. To lower the cost of regression testing, regression test selection selects a subset of the test suite that was used at the time of development.

After that it uses that subset for testing the modified program. By using this approach a subset of test cases are elected and all the cases of test suite are rerun. By using this approach the existing test suites are divided into-

a) Reusable test cases;
b) Re-testable test cases;
c) Obsolete test cases[5][6];

2.1.2 Test Case Prioritization

The techniques of test case prioritization gets the input from the above step and by keeping in check various criteria's schedules the cases for the execution process in such a way that it endeavors to reach the goal performance with a striking increase in efficiency. There exist various goals that can be achieved by optimizing the regression testing. One of them being, faster rate of fault detection dimension, to deduce of how swiftly faults can be detected during the process of testing.

Test case prioritization techniques have a basic inability to eliminate various test cases. This characteristic flaw of the technique is veiled by other techniques that we have surveyed in this paper, i.e. selection of test cases and minimization of test suite, which easily discard the test cases that don't effectuate the requirement. Preferably, in scenarios where the elimination of test cases from the previous test suite is admissible, this technique can be utilize in alliance with the techniques like regression test selection or minimization of test suite to priorities those test cases that are selected or minimized in test suite. In addition to this, prioritization of test case increases the possibility if for any reason regression testing process unusually terminates, thus we can conclude that the time spent on testing has been wasted more profitably than if test cases were not prioritized.

Input: Test suite T_i, number of faults detected by a test case FD, and cost to run each test case TC. Output: Prioritized Test suite T_i'.

1: Begin

2: Set T_i' empty

3: For each test case t_i ε T_i do

4: Calculate average faults found per minute as FD/TC

5: End for

6: Sort T_i in descending order based on the on the value of every case.

7: Let T_i' be T_i

8: End

Test case prioritization techniques validates for code coverage that involves block coverage, decision (branch) coverage, and statement coverage as well. With differing objective of the functions, these techniques will differ in complexity and search-space characteristics too.

2.1.3 Test Case Minimization

When a program under goes changes it leads to test cases being archaic, thereby removal of those test cases from the test suites is a suitable option. A redundant test cases in the test suite that provides the same coverage of the program might exists, this reduces the efficiency and consumes [7][8]. Reducing the test suite size will easily decrease the effort for the test suite maintenance and the remaining test are re-executed for subsequent modification in the software.

***Definition*:** A test suite TS, and a test requirements set of r_1, r_2… r_n, they are to be satisfied to provide the desired testing coverage of the program and subsets of t_1, t_2....t_n. Each test suite is associated with each of the r_i, such that any one of the test cases t_j belonging to t_i can be used to test r_i [13].

Test suite minimization approaches are used to identify duplicate test cases and minimizing the size of test suite by removing duplicate test cases. Test cases become redundant because [9] their input/output relation is no longer meaningful due to changes in program,[10][11]these tests were developed for a specific program that has been modified or [12] their structure is no longer in assent with the software coverage. Minimization or "test suite reduction" means a constant elimination; even so, the two concepts are related since all techniques acts as a momentary subset of the test suite, in spite of only minimization can always remove test cases.

3. Regression Testing Algorithms

In this paper we have surveyed and studied in detail about various algorithms that have consistently been used for improving regression testing. And we have come across the following three algorithms that have been under keen eye of study.

3.1. Genetic Algorithm

A Genetic algorithm [14] dawn with guesses and tries to improve the guesses by evolution. A Genetic algorithm has five parts:

a) A delegation of guess is called as chromosome.
b) A basic pool of chromosomes.
c) Fitness function
d) Selection function
e) Mutation operator and crossover operator.

A chromosome can be defined as a set of binary string or a detailed data structure. We can manually create the initial pool of chromosomes. To meet a specified objective that is the propriety of chromosomes, the fitness function is used. In the genetic algorithm which function is going to participate is decided by the selection function. Exchange of genes from two chromosomes and creation of two new chromosomes is done by crossover operator. The mutation operator changes the gene in a particular chromosome and creates a new one. Genetic algorithm is one of the evolutionary problems. Previously in real life problems evolutionary algorithms have been applied. Genetic algorithm has come up as an efficient, optimization search method. Genetic algorithm is a type of search algorithm. Genetic algorithm is inspired by the way nature expands species using natural selection of the fittest individuals.

Step 1: A society has M individuals which are arbitrary and are generated by pseudo random generators where an individual among them, might represent a solution that is feasible. Such is known as primary solution, which is obtained from vector solution in the solution space. This encourages the search to be authentic, starting from broad range of points in the solution space.

Step 2: Some elite members of the current population easily estimate an appropriate function value.

Step 3: Mapping of the function is done to a fitness function that approximates value of fitness for each member of the society. Hence, following the implication faced by genetic algorithm operators.

The solutions to the problem that are solved are represented by a population of chromosomes. Chromosomes are strings of binary digits and every digit that makes up a chromosome is known as gene. This primary population can be created manually using greedy algorithm.

The pseudo code of GA is: [14][15]:-

Load (population)

Assess (population)

While (stop when not satisfied)

{

Select (population)

Cross (population)

Alter (population)

Classify (population)

}

The algorithm emphasizes until the population has originates to form a solution to the query, or until a maximum number of iterations have taken place. Genetic algorithm makes use of three operators that are described below –

Selection

To choose the individuals for mating that is based on the fitness, the selection scheme is applied. Wellness of a man can be determined as a capacity of a person to sustain and imitate in an environment. To start a new generation, selection procedure generates a new population from the old population. To find out the value of fitness every chromosome is assessed in present era. This value of fitness is utilized for selecting the better chromosomes from the populace for the coming generation.

- **Crossover**

Subsequent to applying the operation of selection, the next operation that is the crossover is enforced on the chromosomes that were selected. It includes interchanging of genes between the two individual [16]. This procedure is rerun among various parent individuals till the coming generation has got sufficient individuals. After the completion of the crossover operation the next

operation that is the mutation operation is applied. Subsequent to this subgroup of population is selected in a random way and on that the mutation operation is applied.

- **Mutation**

Changing of DNA sequence is known as mutation. It is a natural process. To describe something undesirable or broken, the word mutant or mutation is commonly used. To propose new good traits, mutation process alters the chromosomes in short way. To bring the diversity in the population the mutation process is applied.

The major idea trailing all these particular algorithms is to depict the nature changeability. This requisite that the procedure expected must act like a natural system. The nature is contemplated by the Genetic Algorithm to a large extent. Genetic algorithms yields a population in such a manner that the feature which is very much popular, that means, it's value of fitness is depicted higher, as is wrought by the nature [17] [18]. This is the elementary theory after transformation. In this way, these algorithms are likewise shown as the mutative algorithms.

3.2. Simulated Annaealing Algorithm

Simulated Annealing actually is a metaphor of annealing in metallurgy that is a chemical process that involves cooling of any solid material in bath with heated temperature. The solid material which is heated beyond its melting point, and again brought to the normal solid state after heating, tends to undergo changes in the structural property. This leads to the structural properties of the material, which is now cooled, depend on the cooling rate [19][20][21].

The P being probability for acceptance of solution with inferior properties that change as the search is in progresses, and is calculated as:

$$P = E- \mu t$$

Where μ stands for the difference of the desired value and the current solution as well the next inferior solution that is considered, and t symbolizes control parameter i.e. temperature. The temperature is cooled down as per the cooling schedule. Originally the temperature is in higher range such that the dependency on the initial solution is lost. This algorithm is an adaptation of Metropolis –Hastings Algorithm.

In software regression testing the simulated annealing test is implemented by taking candidate solutions. Implementation on simulated annealing algorithm is done where a candidate solution is being represented by configuration of $[X_1, X_{2...}X_n,]$ .Whereas, the energy that has to be minimized is represented by Z which is the cost function in Equation 1. The algorithm starts with a random initial configuration at a high temperature and reduces the temperature gradually to a

freezing point [21]. At each temperature points, regions of modification in the solution space are searched using Metropolis-Hastings algorithm.

Metropolis algorithm starts with an iteration which initializes by introducing a random commotion to the configuration of the system and assessing the resultant value of the change. The change is considered to be a downhill move if the value is negative or zero, the perturbation is approved and later the latest configuration of lower- energy is titled as the initial point [22]. Whereas, if the change turns out to be positive an uphill move, then this proposed perturbation might be validated with a probability that is temperature-dependent so as to prevent the system to be stuck in the loop that leads to inefficiency and least productive state. The algorithm defined below of Simulated Annealing integrates a feasible technique for negative change value or downhill moves. Its focus is to search the feasible solution space regions. The system's configuration yields a feasibly optimal or nearly-optimal re-tests subsets. The SA algorithms generic flow is given below. Initial solution is generated.

a) Deduce an initial temperature T_0, where $T_0>0$.
b) Following steps are to be executed until temperature does not start freezing.
c) The following loop is to be performed K times.
d) Random neighboring solution is selected i.e. Solp of Sol.
e) Calculate ϕ= F(Solp)−F(Sol).
f) When ϕ <0 (i.e. downhill move); Sol = Solp.
g) When $\phi \geq 0$ (uphill move) ; Sol = Solp (P)
h) P (ϕ, T). If F (Sol) ≥F (SolB) then SolB = Sol.
i) The temperature (T) using cooling rate is upgraded
j) SolB is best solution.

3.3. Incremental Approach

Software development and testing are influenced not by the model of new software, but by the network and management or maintenance of existing software [23]. Such software management activities may account for as much as two-thirds of the overall cost of software production. In particular, computerization of the testing process should be beneficial because the typical testing process is a human-demanding activity and as such, it is generally costly, time consuming, error prone and often partially done.

Maintenance testing phase of a program's life is managed by regression testing, which can be defined as, Software's or program's behavior is unchanged if there is repetition of tests except

seeing as recommended by the change to the software. It is observed to provide assurance that changed or modified cod behaves as designed, and that modification have not negligently disturbed the behavior of unchanged code [5]. In regression testing, to validate changed code, creating test set T' and then executing changed or new modified on test set T'. In tradition, test suite T' generally contains all cases of T plus new test cases designed to test modified code or new functionality [24]. Executing modified on these all test cases us generally too expensive.

Regression testing alters from initial test cases in suite and the outcomes from their execution on validated are available to the tester. Incremental regression testing endeavors to accomplishment this information in two ways [25]. (1) If existing test cases applied on new components for testing then new construction of test cases can be avoided which is costly. Because of this reason, many techniques endeavor to maximize the reuse of test cases from suite T. (2) if it may be contended that, for a subset of test cases T, changed code will produce the same result as validated, then the costly execution of changed on this subset can be avoided. This happens, if unchanged program components are executed by a test case.

Steps for the Incremental regression testing:

(1) Find a set of affected program components of changed.

(2) Find a subset of test case T that tests the affected components.

(3) Find untested components of changed that must be enclosed by new tests.

(4) Create new test cases for the uncovered components.

(5) Run changed on T', the union of the test cases produced in steps (2) and (4).

4. PROPOSED APPROACH

As we know they verifying the behavior of modified program is unchanged, except where needed by the modifications, we apply various approaches that find test cases in the actual or existing test suite on which the original and changed programs may produce different results.

4.1 Assumptions for Modified Program

We assume that: No statements are deleted from or added to the program as well as no transitions are made to the left-hand side of assignment statements. By doing so we have observed:

(1) If a statement is not executed by a test case, it cannot alter the program output of that test case.

(2) Each test case is not applied on all statements in the program.

4.1.1 Execution Technique

The set of statements of an execution slice is executed by a test case in a program. Therefore the main idea is if a test case does not execute any changed or modified statement, it need not be rerun. And the approach is to measure the execution slice of each test case, then re-run only those test cases whose execution slices contain a modified statement. This technique is based on observation (1) and (2) above. Suppose the program modification consists of changing a single statement in the program.

```
S1: read (a, b, c);
S2:          Class: = scalene;
S3: if a == b or b == a
S4:           Class: = isosceles;
S5:  if a*a ==b*b + c*c
S6:          Class: = right;
S7: if a==b and b ==c
S8:           Class: = equilateral;
S9: case class of
S10:     right: Area: = b*c/2;
S11:      equilateral: Area: = a*2*sqrt (3)/4;
S12:       otherwise: s: = (a+b+c)/2;
S13:         Area: = sqrt(s*(s-a)*(s-b)*(s-c));
          end
S14: write (class, area)
```

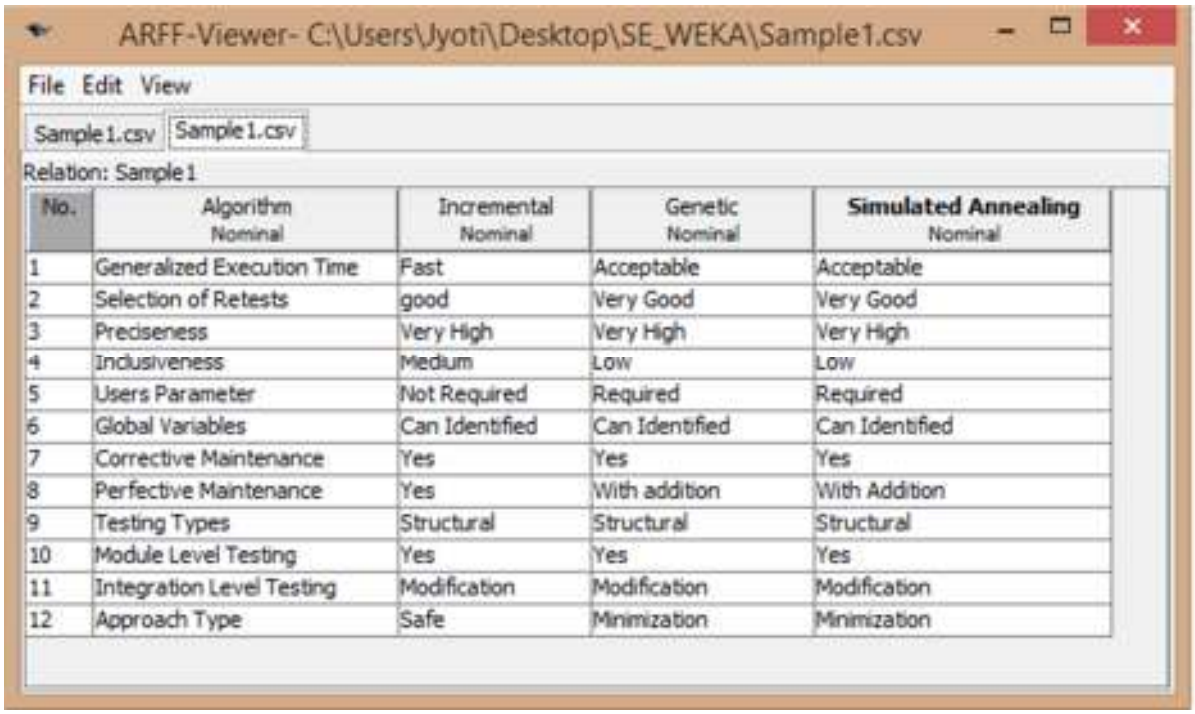
Figure 3) Algorithm used for incremental approach

Test Case	Input			Output	
	a	b	c	Class	area
T1	2	2	2	Equilateral	1.73
T2	4	4	3	isosceles	5.56
T3	5	4	3	Right	6.00
T4	6	5	4	Scalene	9.92
T5	3	3	3	Equilateral	2.60
T6	4	3	3	Scalene	4.47

Figure 4) Observation table of Execution techniques

5. Regression Using WEKA

Regression involves building a standard or model to predict the dependent variable based on one or more independent variables. A simple example of regression would be to analyze the behavior of incremental approach, genetic algorithm and simulated annealing. The behavior of algorithms is analyzed on the basis of some parameters, which are shown as below-



No.	Algorithm Nominal	Incremental Nominal	Genetic Nominal	Simulated Annealing Nominal
1	Generalized Execution Time	Fast	Acceptable	Acceptable
2	Selection of Retests	good	Very Good	Very Good
3	Preciseness	Very High	Very High	Very High
4	Inclusiveness	Medium	Low	Low
5	Users Parameter	Not Required	Required	Required
6	Global Variables	Can Identified	Can Identified	Can Identified
7	Corrective Maintenance	Yes	Yes	Yes
8	Perfective Maintenance	Yes	With addition	With Addition
9	Testing Types	Structural	Structural	Structural
10	Module Level Testing	Yes	Yes	Yes
11	Integration Level Testing	Modification	Modification	Modification
12	Approach Type	Safe	Minimization	Minimization

Figure 5) Surveyed analysis of the performance of each algorithm

In weka data is imported in the native (Attribute-Relation File Format) arff format. Weka also supports .csv format. Open weka GUI and then click on 'Explorer'. After clicking on explorer, it will open new window where you can choose your file.

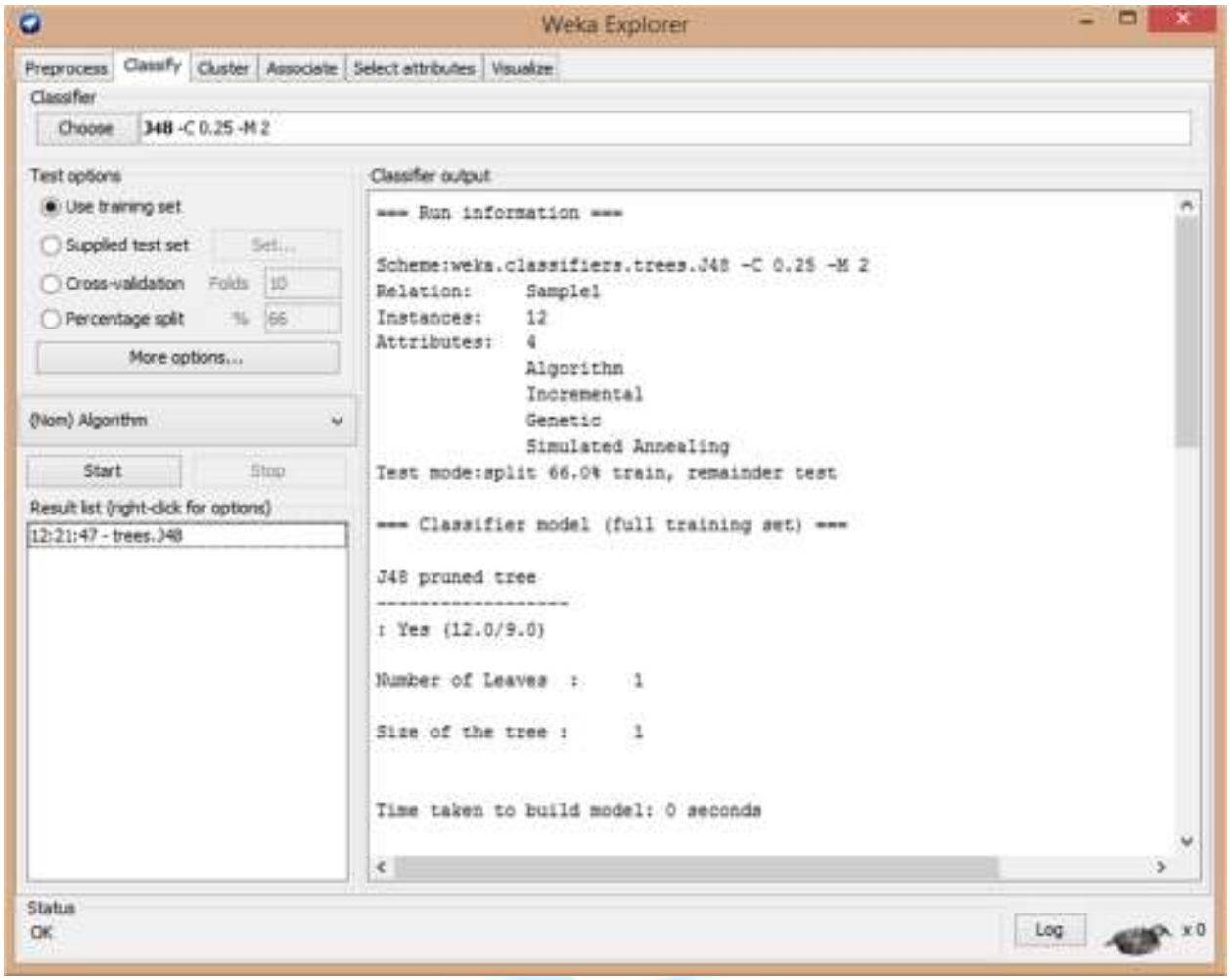


Figure (6) Values generated using WEKA tool

As shown in figure (6), next step is to click on classify tab and choose your file. There are test options for building the regression standard. The options are as given below:

- Use training set: The classifier is calculated on how well it predicts the instances of class, on which it trained on.
- Supplied test set: The classifier is calculated on given file and predicts how set of instances of class are loaded. When you click on set button, brings up a dialogue, which allow to choose file to test on.
- Cross-validation: The classifier is calculated by cross-validation
- Percentage split: The classifier is calculated on how it predicts a certain percentage of data which is used for testing.

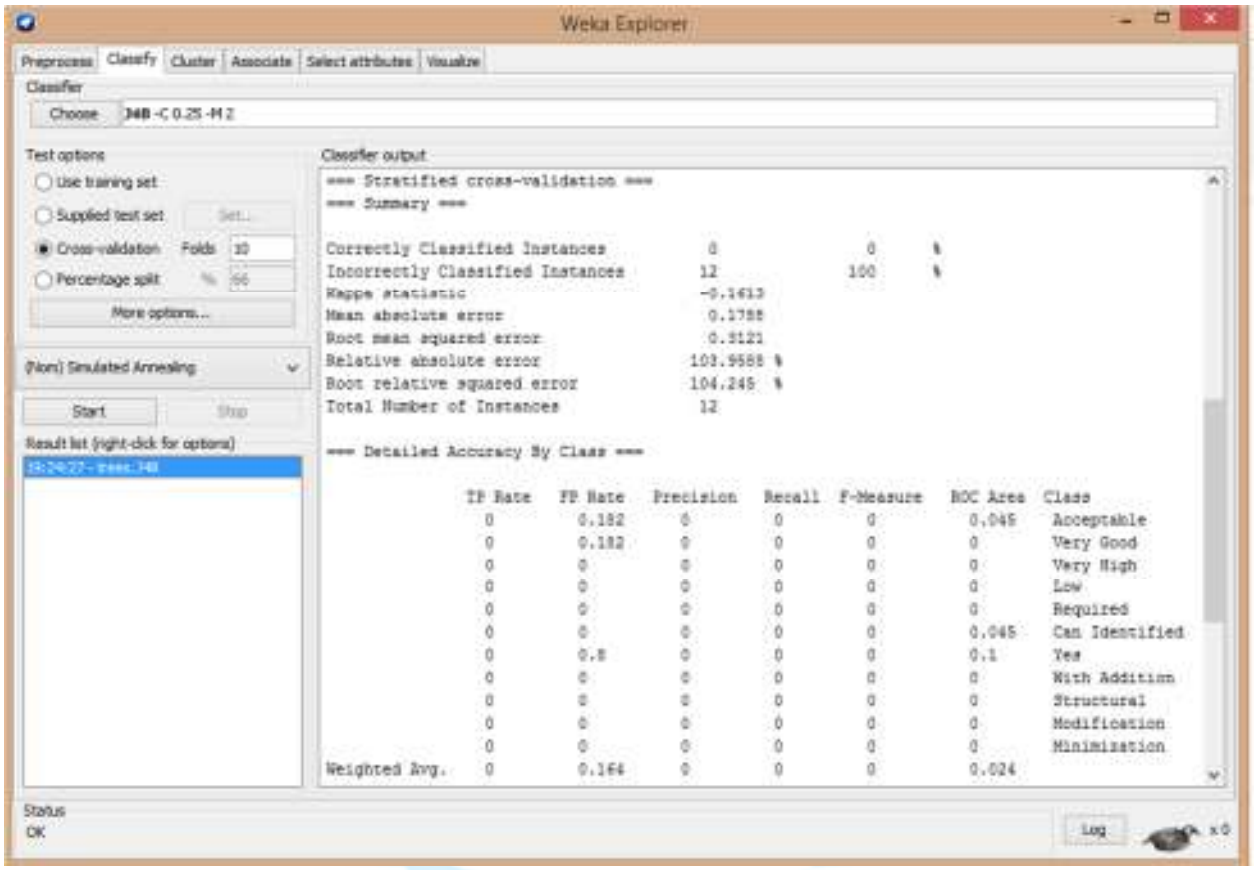


Figure 7) Percentage split of the values generated

The performance of the algorithms in regression testing is analyzed as per various surveys that we have conducted.

5) Result And Inferrences

As per the study done in this paper we have generalized the performance of the three algorithms on the basis of perceptible and subjective metrics.

5.1 Perceptible Metrics

- **Execution time**: The modules that were chosen are differentiated with the size. The execution varies if size is smaller or bigger. The papers we surveyed shows that for small module the execution time of simulated annealing and genetic algorithm is not fast enough,whereas the incremental approach is fast enough. For modules with medium size, the sa algo runs slower than genetic algorithm.

- **Test cases selected**: simulated annealing and genetic algorithm takes in least amount of test cases for re-test. Whereas the incremental approach takes a fair share of test cases.
- **Preciseness**: All the three algorithms gives a fair of precisions.
- **Inclusiveness**: simulated nnealing algorithm and genetic algorithm have lower rate of inclusiveness.

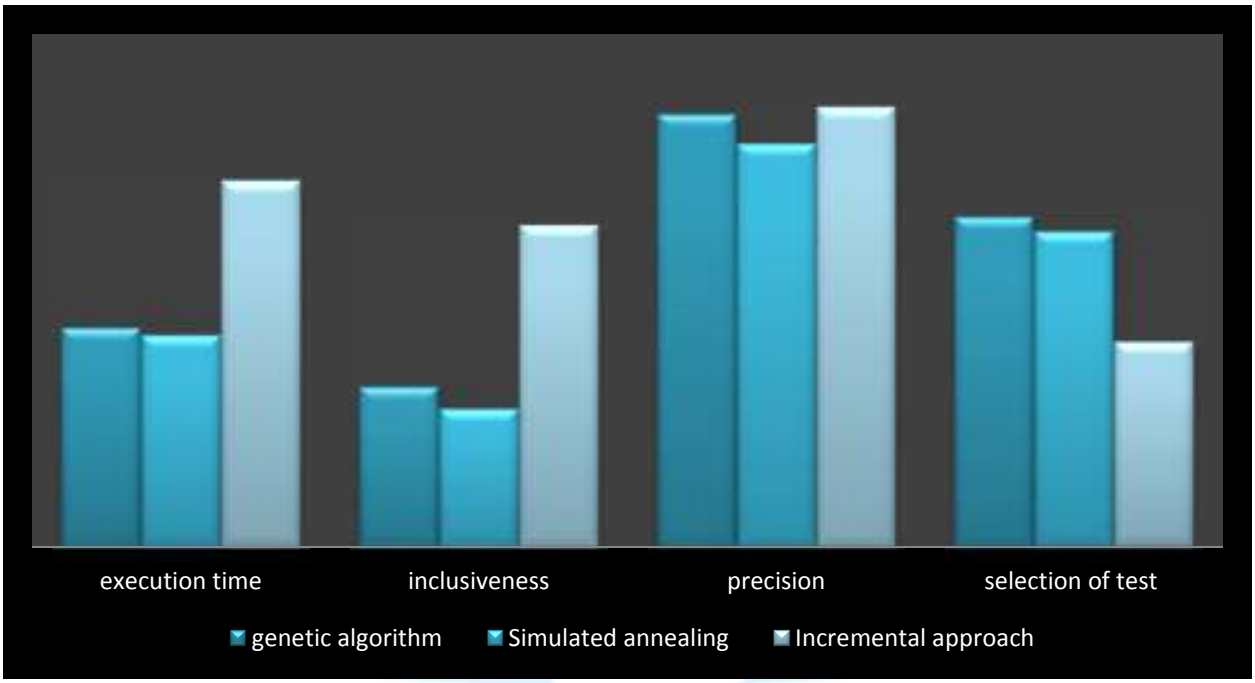


Figure 10) Comparitive Result

5.2 Subjective Metrics

- **Parameter's settings**: Simulated annealing algorithms need parameter's settings from user, so does genetic algorithms. Whereas the Incremental approach doesn't require any of it.
- **Testing methodology** : Simulated annealing algorithms and Genetic algorithms require structural methodology. But incremental approach is is compatibility with both structural and functional methodology.
- **Approach methodology**: Simulated annealing and Genetic algorithm required minimization as it is their execution time is slower. Incremental approach methodology is safe.

6. Conclusion

From this paper we conclude that the algorithms used for optimizing the regression testing have given best results if they satisfy the requirement of the user. If we need to perform the tests with smaller module as per the requirement we should go for genetic algorithm and simulated annealing algorithm. Whereas, if the tester's main motive is to find the modified part of the system then incremental algorithm is the best choice. The entire three algorithms are efficient in their own respect but as per survey we can conclude that incremental algorithm has so far yielded the best performance.

References

[1]A Comparative Study of Five Regression Testing Techniques : A Survey, International Journal of Scientific & Technology Research Volume 3, Issue 8, August 2014.

[2]Regression Testing Using Coupling and Genetic Algorithms, HarchBhasin et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 3 (1) , 2012, 3255 – 3259.

[3]D. Li, C. Sahin, J. Clause and W. G. J. Halfond, "Energy-Directed Test Suite Optimization," IEEE International Workshop on Green and Sustainable Software, pp. 62-69, 2013.

[4] S.Yoo and M. Harman, "Regression Testing Minimization, Selection and Prioritization: A Survey," ACM Software Testing, Verification & Reliability, vol. 22, no. 2, pp. 67-120, 2012.

[5] Gregg Rothermel, Mary Jean Harrold. Analyzing RegressionTest Selection Techniques, IEEE Transactions on SoftwareEngineering.

[6] Dr.Arvinder Kaur and Shubhra Goyal. Article: A GeneticAlgorithm for Fault based Regression Test Case Prioritization.International Journal of Computer Applications 32(8):30-37,October 2011. Published by Foundation of Computer Science, New York, USA.

[7] Parul Gandhi and Pradeep Kumar Bhatia. Article: Optimization of Object-Oriented Design using Coupling Metrics.International Journal of Computer Applications 27(10):41-44,August 2011. Published by Foundation of Computer Science, New York, USA.

[8] Harsh Bhasin, Surbhi Bhatia: Use of Genetic Algorithms forFinding Roots of Algebraic Equations. International Journal ofComputer Science and Information Technology, 2011. Volume2, Issue 4, pages 693-696.

[9] Harsh Bhasin, Surbhi Bhatia. Application of Genetic Algorithmsin Machine learning, 2011. IJCSIT Volume 2 Issue 5, pages :2412-2415.

[10] S. Kadry, "A New Proposed Technique to Improve Software Regression Testing Cost," International Journal of Security and Its Applications vol.5 no. 3, 2011.

[11] D. Jeffrey and N. Gupta, "Experiments with Test Case Prioritization using Relevant Slices," Journal of Systems and Software, vol. 81, no. 2, pp. 196-221, 2008.

[12] P. Tonella, P. Avesani, and A. Susi.Using the case-based ranking methodology for test case prioritization.In Proceedings of the 22nd International Conference on Software Maintenance (ICSM 2006), pages 123{133. IEEEComputer Society, July 2006.

[13] R. A. Santelices, P. K. Chittimalli, T. Apiwattanapong, A. Orso, and M. J. Harrold.Test-suite augmentation forevolving software. In Proceedings of 23rd IEEE/ACM International Conference on Automated Software Engineering,pages 218{227. IEEE, September 2008.

[14] S. Sampath, R. C. Bryce, G. Viswanath, V. Kandimalla, and A. G. Koru.Prioritizing user-session-based test cases forweb applications testing. In Proceedings of the 1st International Conference on Software Testing Veri_cation and Validation (ICST 2008), pages 141{150. IEEE Computer Society, April 2008.

[15] M. Ruth and S. Tu. A safe regression test selection technique for web services. In Proceedings of the 2nd InternationalConference on Internet and Web Applications and Services (ICIW 2007), pages 47{47. IEEE Computer Society Press,May 2007.

[16] M. Ruth and S. Tu. Concurrency issues in automating rts for web services. In Proceedings of the IEEE International Conference on Web Services (ICWS 2007), pages 1142{1143. IEEE Computer Society Press, July 2007.

[17]M. Ruth, S. Oh, A. Loup, B. Horton, O. Gallet, M. Mata, and S. Tu. Towards automatic regression test selection forweb services. In Proceedings of the 31st International Computer Software and Applications Conference (COMPSAC2007), pages 729{736. IEEE Computer Society Press, July 2007.

[18]L. Zhang, S.-S. Hou, C. Guo, T. Xie, and H. Mei.Time-aware test-case prioritization using Integer LinearProgramming.In Proceedings of the International Conference on Software Testing and Analysis (ISSTA 2009), pages212{222. ACM Press, July 2009.

[19] S. Yoo, M. Harman, P. Tonella, and A. Susi.Clustering test cases to achieve e_ective& scalable prioritisationincorporating expert knowledge.In Proceedings of International Symposium on Software Testing and Analysis (ISSTA2009), pages 201{211. ACM Press, July 2009.

[20] Y. Yu, J. A. Jones, and M. J. Harrold.An empirical study of the e_ects of test-suite reduction on fault localization.InProceedings of the International Conference on Software Engineering (ICSE 2008), pages 201{210. ACM Press, May2008.

[21]H. Zhong, L. Zhang, and H. Mei. An experimental study of four typical test suite reduction techniques. Informationand Software Technology, 50(6):534{546, 2008.

[22]J. Zheng, L. Williams, B. Robinson, and K. Smiley. Regression test selection for black-box dynamic link library components. In Proceedings of the 2nd International Workshop on Incorporating COTS Software into SoftwareSystems: Tools and Techniques (IWICSS 2007), pages 9{14. IEEE Computer Society, January 2007.

[23] J. Zheng, L. Williams, and B. Robinson.Pallino: automation to support regression test selection for COTS-basedapplications. In Proceedings of the 22nd IEEE/ACM international conference on Automated Software Engineering(ASE 2007), pages 224{233. ACM Press, November 2007.

[24]J. Zheng, B. Robinson, L. Williams, and K. Smiley.. In Proceedings of the 5th International Conference on Commercial-o_-the-Shelf(COTS)-Based Software Systems (ICCBSS 2006), pages 137{146.IEEE Computer Society, 2006.