ISSN:0975-9646

Performance And Result Analysis Of Hybrid Adaptive Random Partition Testing For Improved Bug Detection And Test Coverage

Reetika Patidar,Mahesh Malviya
Department of CSE, Jawaharlal Institute of Technology vidhya Vihar Borawan (M.P), India

***Abstract*- Testing is a very crucial process of assuring products reliability and trust dependencies. Thus for achieving maximum coverage the test must be generated from overall distributed regions of input domains and known as partition testing. But it gives optimal results for homogeneous regions. Random testing serve better than partition testing but is also generating high computation overheads. Another technique is Adaptive Random technique having adaptive nature of regenerating and passing some of the test cases back to the input for correcting the next test cases lined to be passed. Apart from the above benefits of ART the complete problem solution of PT and RT is not provided. Thus this work proposes a Hybrid Adaptive Random Partition Testing (H-ARPT). The effective test cases can be determined if the test comes from complete regions and covers at least once each type of input. But all of certain such heavy numbers of inputs are not tested with some minimum attempts. Hence, a new mechanism is required which reduces the test size but increases the code coverage. It works towards assuring the reliability of the system. This paper proposed a Hybrid ARPT, which works towards effective and early identification of bugs according even with their priority levels also. Means the module which is most critical should be tested more. It overcomes the existing issues of high testing cost and computation complexity. The paper also suggests some analytical evaluation factors and compares the RT, PT, ART with hybrid ARPT. On the preliminary evaluations the works seems to provide effective solution of testing domains.**

Keywords- Software Testing, Blackbox Testing, Adaptive, Random and Partition Testing, Hybrid Adaptive Random Partition Testing, Coverage analysis and Fault Detection Rate;

I. INTRODCUTION

The software industry had suggested so many different types of testing over the last few decades by which the detection accuracy and quantity is improved. They are focusing on different parameters and use various tools for bug's removal. But still some question remains to be answered like the selection of testing and which one is best. Normally, the programs testing criteria for analysing a code is its subset of inputs with a very smaller range. These small set of input test criteria is termed as test cases and combination of these cases according to different operating conditions and program codes are called as test suites. An aim is towards feeling confident about the test cases and assuring that all the defects are removed. The testing technique uses some of the information about the programs or its structure for guiding the generation of test cases and suites. This information might be related to functional behaviour, flow of data, common hitting errors or their respective combinations.

Partition testing

Partition testing is one of the testing mechanisms which divide the inputs domains into various sub-domains categorized according to some separation conditions of test cases [1]. Here the test cases are selected from every sub- domain where the input sustains some of the equivalence partitioning properties for encountering the errors. It could be made feasible by modelling the test cases using some knowledge about the test criteria and program. In *random testing*, the pseudo random number is used for selecting the test case from input domain according to some random condition. Some of the researchers found that the partition based knowledge based testing is lower performance and accuracy while comparing with random testing. Also the failure rates covered by random testing are more than the partition testing.

The higher probability of error detection in random testing with complete coverage of inputs using partition testing can be used combining for better serving the defect removal. Even though such testing would have the benefit that consistency data could be formed, in follow with most software of any difficulty, it is not possible to model precisely enough the sharing of real input data [2].

The term "partition testing," in its broadest sense, alludes to an exceptionally general family of testing strategies. The primary characteristic is that the program's input domain is divided into subsets, with the tester selecting one or more element from every subdomain. In the testing literature, it is normal not to limit the expression "partition" to the formal mathematical importance of a division into disjoint subsets, which together span the space being considered. Instead, testers for the most part utilize it as a part of the more informal sense to allude to a division into (perhaps overlapping) subsets of the domain. The objective of such a partitioning is to make the division in such a path, to the point that when the tester chooses test cases based on the subsets, the subsequent test set is a good representation of the whole domain. Ideally, the partition divides the domain into subdomains with the property that inside every subdomain, either the program produces the right response for each element or the program produces an erroneous response for each element.

In spite of the fact that code coverage strategies are not generally viewed thusly, they can be considered to be

partition testing strategies. Case in point, statement testing obliges that sufficient test data be selected so that each statement of the program is executed in any event once. This divides the input domain into nondisjoint subdomains with every subdomain comprising of all test cases that cause a specific statement to be executed. Since a given test case will for the most part cause numerous statements to be executed, it is an individual from the subdomain determined by every such statement. Extension testing likewise divides the domain into non-disjoint subdomains. Way testing obliges that sufficient test data be selected so that each way from the program's passage statement to the program's exit statement is traversed in any event once. This technique does divide the input domain into disjoint classes since a given test case causes precisely one way to be traversed.

Adaptive testing

Adaptive testing is a novel mechanism covering some different aspect of testing. It is a feedback oriented testing presenting its strong nature against the random and partition testing. Though its application cost and complexity is higher than others. It provides the effective test selection criterion with reduced test cases. Some of the analytical evaluations over all the above testing is found as: partition testing performs well with equal size partitions, random testing gives better results in random test generation with higher coverage, and adaptive testing serves better then both with high computational complexity. Partition testing with proportional allocation is shown to perform at least as well as random testing in terms of all of these criteria [3]. Some empirical answer of exact comparison of these testing methodologies is not yet present. Also, if all the three mechanisms are used simultaneously then their working operation and behavior is not controlled as suggested many times.

Component Testing

The application of randomly constructed test sets to software components would appear to offer the same benefits as for compilers. Hence the author considered the implications of adding random testing to the British Computer Society component testing standard . The use of such testing is certainly accepted in the sense that published material uses the approach. Then, random argument values and parameters are computed for the application of tests of the standard mathematical functions (sin, cos etc) in the Pascal Validation Suite . However, the use of this technique is effective due to the ease with which automatic acceptance checking can be applied. With such a process, a large number of tests can be run automatically.

Hand analysis of the few that fail (if any) can then be undertaken into the account. Based on the strength of the acceptance logic, the testing can be very thorough. Even when the acceptance criterion is merely that the program does not crash, confidence in this property is obtained at modest cost. The strength of conventional component testing is assessed by metrics such as statement coverage. With random testing, the number of random tests is of little value unless the distribution is a reasonable fraction of the entire input domain principal. Here we finds a problem with the BCS standard, as a key aspect of the standard is the testedness metric. The approach taken below is that the coverage of the input domain is measured by means of the equivalence partitions used for equivalent partition testing, together with the number of tests run.

Random testing

For Better understanding let N be the total number of elements in the input domain, and suppose we want to randomly select n inputs for testing the system. It may be possible on the basis of any probability distribution, i.e., the n inputs can be selected independently with the Although sampling without-replacement is obviously more efficient, the practical implementation of a without-replacement sampling scheme is difficult and typically not cost-effective. Moreover, for the applications we are interested in particularly, input domain size and the partitions are large enough relative to the sample size so that the two are essentially similar. Random testing has generally been viewed as being easy to implement and therefore cost-effective. However, to actually ensure that every test case is selected independently according to a given probability distribution is not at all easy. This involves a careful definition of the input domain and the use of appropriate sampling schemes. If one tries to accomplish this other part, one runs risks such as incompletely covering the domain, selecting inputs from some parts of the domain more intensively, or introducing other biases.

After studying the various articles related to random testing, partition testing and adaptive testing. Partition testing deals with dividing the input space into several partition. But it works well only with equal sized partition. In random testing the use of pseudo random generator decreases the test case count but increase the computational complexity. While adaptive nature support some revert logic of further test selection. It is based on feeding back the derived results for further correcting the next input. Though most of the changes is made and various improvements is performed in these areas; still some issues which remains unsolved. Here this work focuses these issues as problem areas like selection criteria of random and partition test is not clear at the point of testing. The criteria must contain coverage rate and fault rate for better analysis and error detection. Adaptive nature with random and partition behavior is having high computation overheads. This it requires conservable higher time for detecting the faults. If the sub areas or components of partitions are not balanced the performance are not desired and the results are not clear. It also not gives any idea about the coverage of the functional blocks.

This work proposes a novel hybrid adaptive random testing using partitioning logic. Here the technique generates the test data which is thoroughly distributed in overall region of the divided partition. Thus, the suggested method can turn formal detection to complete analysis and the probability of fault detection is also increased. Traditionally the adaptive random testing is of only two forms; distance based and partitioning based. The suggested hybrid approach is combination of both the measures for effective determination of faults with minimum number of test cases.

II. BACKGROUND

The testing can be categorized according to their working style and behaviour. Mainly it is categorizes as black box and white box testing. In black box testing the codes are tested as a combined functional blocks and its internal structure is not known.[15]

Software size and its handling complexity is increased as the development process proceeds and modular designs gets into shape. Lots of interaction needs to be taken over for accessing and measuring the strength of bounds and branches. For effectively quantifying the behaviour of the software some reliability improvements and judgment methods is used along with development or after development. This process is termed as software testing which is used to identify the bugs in program codes parallel as the code progress. Testing involves various resource utilizations and traversing the code blocks at least once for analysing the behaviour and integration actions. Measuring the strength of code and estimating the testing resources for removing the bugs may increase the reliability and hence it is included in software development life cycle.[15]

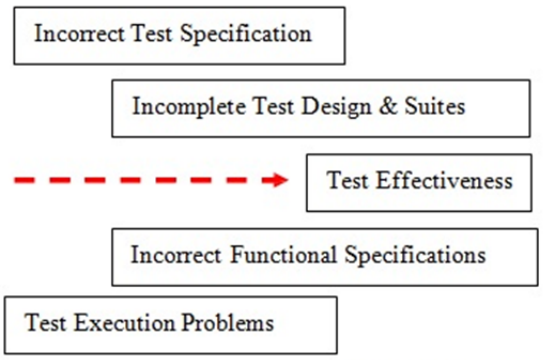

FIGURE 1: FACTORS OF TEST EFFECTIVENESS

The inputs ranges are used for applying multiple executing conditions and their behaviour. It serves a parametric model which works as a finite time events and calculates the responses of the overall software's. Te white box testing as an approach whose aims is towards detecting the internal structure bugs and problems. Its goal is reliability detection and estimation by completely analysing the internal structure a logic handling by the use of control graphs. It is having very large input sp[ace and hence it is quite complicated to handle such methods. Totally the aim is towards the identification of faults that leads to failure the means of above testing strategies.

Now, the tester requires effective testing which identifies all the defects with minimum efforts. It can be made feasible by effective uses of both black box and white box testing. This work specifically focuses on random testing, partition testing and adaptive testing. Understanding the domain completely requires a clear view of each before analyzing and applying any measures.

(i)Random Testing [4]: It is used for testing the code blocks by applying a set of input condition selected randomly for the domain. It is black box testing and hence it does not requires any internal structure information. It is practically formed by the use of pseudo random generator because pure random values are not provides correct mapping of input domains. It is easy and widely applicable and its implementation and execution time is also very less. Here are some types of random testing:

- *Pure random:* Test cases are randomly generated until appear to be enough.
- *Guided by number of cases:* Test cases are randomly generated until a given number of cases has been reached.
- *Error guessing:* Test cases are generated by the subject's knowledge of what typical errors occur during programming. It stops until they all appear to have been covered.

(ii) Partitioning [5]: This technique divides the input ranges into various sub domains and classes according to different conditions. These classes are known as equivalence partitions to reduce the total number of test cases because in this the test cases are selected from each partition. It is part of functional testing which represents pure black box nature. These classes are the set of some valid and some invalid inputs with range limits. The partition logic uses the sampling theory of statics for categorizing the population. Applying the testing of software by selecting the random test samples from the population is called as partition testing. In some cases where simple random function is used the partition testing outperforms the random testing in terms of failure detection rates.

(iii) Adaptive Testing: It works towards selecting the adaptive idea of testing for minimizing the test oracle problems and optimizing the results. It minimizes the total cost of error and faults detection by the use of partition logics.

(iv)Adaptive Random Testing [6]: It is based on a simplistic intuition that if the partition and random logic fails to detect bust the condition which reduces the faults occurrence should places closes to next inputs. Thus, the inputs must be placed nears to last successful test. It preserves the randomness property and will serve the feedback to the system. ART is developed as an enhancement to random testing with an objective of using fewer test cases to detect the first failure.

Some detailed study and their nature is clearly defined in the next section of literature survey. It also deals with all the relative changes taken places over the last few years for improving the above testing strategies. Aim is towards minimizing the test cases and its cost.

III. RELATED WORK

During the last few years software testing had grown tremendously with their techniques. Lots of new and overwhelming methodologies are developed which improves the testing performance and decreases their costs. Among them, some approaches shows their strong presence in the respective areas and are related to their work are taken here as literature. These are:

In the paper [7], a dynamic partitioning strategy is presented for selecting the test cases through some online

feedback mechanism. Here the approach is focuses on online medium for generating the test sequences and starts with selecting criteria of online partition. Also the testing is not based on the codes or internal structure of the programs instead it uses only some metadata information and passing and failing condition of previously executed test cases. It does not requires any program code thus can be used anywhere with direct software bundles. The paper also evaluated the cost effectiveness of the dynamic partitioning approach and presents the comparison with some traditional testing. For evaluating some basic programs are used and the results shows the minimum number of test cases requires for detecting the complete faults.

Although, solving all problems related to testing oracle is not feasible each time because of their high complexity. In the paper [8], some of the evaluation is perform on random testing method of black box testing. Here the random test cases are generated for identifying the total bugs from test oracles. It also enables the coverage, if it is high the probability of error detection is more and if the lesser coverage is achieved then the test cases quantity is increased. The paper deals with complete analysing the random testing through a mathematical model which includes identification of effectiveness of random testing, comparison of random strategies, scalability measurement predictability of two runs and threats to validity.

In the paper [9], some more classification is provided on the adaptive random testing (ART). Here the ART is completely studied for demonstrating the behaviour of technique with higher detection rates of faults in comparing with normal random testing. The paper also suggests couple of new ART algorithms fir further increasing the effectiveness. The suggested algorithm provides similar working but the overhead associated with the testing gets reduced. Here the overall test is subdivided into several domains and test is selected from the largest partition. As the partition process is operates on the basis of a randomly preferred test case, we call this process ART by random partitioning. It ensures that test cases maintain to be widely broadened by only selecting new cases from partitions which enclose no preceding test case.

After studying the cores of random testing some of the authors had tries their work with anti random testing. This anti random testing improves the fault detection capability of existing random test by selecting the test which is different from the previous test. The test cases generated by anti random test are more evenly distributes in input partition than the random test. It basically applies form specific numerical input ranges because of their measurable property. Apart from all the above benefits some more modification is provided in antirandom testing in [10]. The proposed techniques basically free the dependency of only numerical inputs of antirandom testing. The suggested technique is more fault detection rates than any of the random testing variants and is tested on various applications. Numbers of test cases are functional and test for detection of faults inserted by using transformation testing.

The paper [11] covers some aspect of partition testing logics and overcome its existing problem. The work detected that if the sub-domains of the testing inputs for the partition is not homogeneous than their performance are not as desired and their success also not contribute so much confidence. Though the code coverage parameters in testing is taken for grated as best practice always. The author's main target is to develop strategies for the automatic progression of a test suite that does inspire assurance. The work suggested is a combination of test suite augmentation and reduction which assures changes handling and repetition of testing logic. The paper identifies the semantic difference between the overlapping transform partitions with comparable behaviours of programs. The generated test cases are witnesses of the behavioural difference of both programs. Semantic change is defined formally based on the notion of change partitions overlapping for original and changed program.

The paper [12] focuses towards further improving the performance and error detection probability of adaptive random testing. It mainly increases the fault revealing ability of random testing by introducing the ART based on two point partitioning. According to the new algorithm of ART-TPP the given are of testing inputs are divided into two or more section based on midpoint theory rather than direct division of\r equal division. Here the first point of division initialization is randomly generated. The selection of second point is through candidate set according to the maximum criteria distance. The experimental evaluation is also given with some existing ART algorithms: ART-RP and ART-BP. The partition can be iteratively performed until the potentia faults are found or the size of test data set reaches the pre-set limit. Analytical evaluation proves the effectiveness of the suggested approach.

The paper [13] focuses on one of the major difficult with testing which is its automated generation. This automatic generation is performed by prior making some of the generation criterion. It reduces the efforts and cost of the testing makes the process truly automated. The paper focuses o generation of test cases from the use of genetic algorithms. The results are compared thoroughly with the random testing. Here the designed algorithms make the use of population and equivalence conditions. If the selection of equivalence class is correct then the potential testing complexities is gets reduced. Here the suggested genetic based equivalence class partitioning detect the best selection of fitness function and overcomes the test generation complexity issues. At the primary level of work of author the approach is serving all the need of testing.

IV. PROBLEM IDENTIFACTION

Testing aims towards detecting the bugs and faults in the code of functional behaviour of the code. It always requires generation of test cases and suites for verifying their input ranges. It comes under the test estimation process which is guided by various mechanism names as testing methodologies. Basically the division is of black box and white box. Among both the paper had worked towards improving the black box testing by suggesting

some modification of unsolved problems. After studying the various articles related to random testing, partition testing and adaptive testing. Partition testing deals with dividing the input space into several partition. But it works well only with equal sized partition. In random testing the use of pseudo random generator decreases the test case count but increase the computational complexity. While adaptive nature support some revert logic of further test selection. It is based on feeding back the derived results for further correcting the next input. Though most of the changes is made and various improvements is performed in these areas; still some issues which remains unsolved. Here this work focuses these issues as problem areas. Some of the major working areas are identified as:

(i) Selection criteria of random and partition test is not clear at the point of testing. The criteria must contain coverage rate and fault rate for better analysis and error detection.

(ii) Adaptive nature with random and partition behaviour is having high computation overheads. This it requires conservable higher time for detecting the faults.

(iii) If the sub areas or components of partitions are not balanced the performance are not desired and the results are not clear. It also not gives any idea about the coverage of the functional blocks.

(iv) Generated test always starts with initial condition, it should be continuously changing with the coverage results of test cases and hence the improved feedback of test generation is required with adaptive nature.

(v) Testing must consider the severity level and criticality of application and its data. It is not provided with any of the above mechanism. Thus test prioritization is required.

Thus by taking the above problems as a base, this works aims towards generating a improved solution than any of the combination or individual effort of partitioning, random test or adaptive testing[15].

V. PROPOSED SOLUTION

This Software testing involves compete evaluation of code with minimum number of test cases. If the tests belong to the internal structure then it is white box testing and if the tests are taken from the working input conditions then it is black box testing. Among the black box testing this work focuses on here basic testing: Partitioning testing, Random Testing and Adaptive Testing. Aim is towards improving their performance with reduction in test cases generation complexity. This work proposes a novel hybrid adaptive random testing using partitioning logic. Here the technique generates the test data which is thoroughly distributed in overall region of the divided partition. Thus, the suggested method can turn formal detection to complete analysis and the probability of fault detection is also increased. Traditionally the adaptive random testing is of only two forms; distance based and partitioning based. The suggested hybrid approach is combination of both the measures for effective determination of faults with minimum number of test cases.[15]

Descripton:

The approach starts with identification of the systems specification which needs to be tested. These specifications might belong to their functionality and working structures. It cloud be of control and data flow and applies with a detailed analysis for generating the codes functional blocks where separate input can be passed. The complete project is the combination of these individual components. Now, after the clear separation of each functional block their operating conditions are determined for generating the individual test codes. This test codes are based on the property of inputs. Now before applying here any testing some verification is planned which rechecks the functional module dependencies and if the dependencies are zero then only a module is passed for further process. Now after this some components start their operations.[15] These are:

(i) **Partitioning Process:** The individual functional blocks with zero dependencies is passed in this module identifies the categories of the input data. These categories are measures by the behaviour analysis of the data and their respective functional blocs. After the category division partitioning of input space is applied. In this the inputs are divided to various sub domains. Later on the basis of this several regions are formed. As partition process gives best results in case of homogeneous division of regions. But in this work, the hybrid nature uses heterogeneous division of different sizes regions. [15]

(ii**) Pseudo Random Generation:** This component is used for applying the random behaviour of the testing. Here the heterogeneous generation of regions not serves as needed. Thus the random generation of test from different regions has more probability of detecting the faults on time. But it also serves the higher computational cost. Thus the partition of regions is made in such a way that each partition size reduces with fifty percent with next creation. Means if the first region is of 50 percent then the next should be of 25 and later will be of 12.5. Thus by this logic a test frame is constructed and passed for next module. [15]

(iii) **Decisional Conditions:** Now this module compares the randomly generated partitions and their test frames. Here the comparison condition checks whether the coverage achieved is maximum with closes distance of relativity for test inputs. If the condition is met then the test is taken to be successful. And if the test frames are not up to the mark then the readjustment of logics are made. After this, proceeds with the next candidate selection for test. [15]

(iv) **Adaptive Logic:** This nature shows the adjustment probability of the partitions and inputs when the desired output is not met. By this always some guiding results are feed back to the inputs regions so next time the regions and their combinations of input passing for testing can be changed. In this ways some guidance regarding to this testing is always available with adaptive nature. [15]

(v) **Coverage and Distance Comparison:** This module deals with comparing the empirical result. It measure the coverage achieved from the test generated and distance of the test with faults. If the test cases are achieving maximum coverage with minimum counts the testing is serving their best. [15]

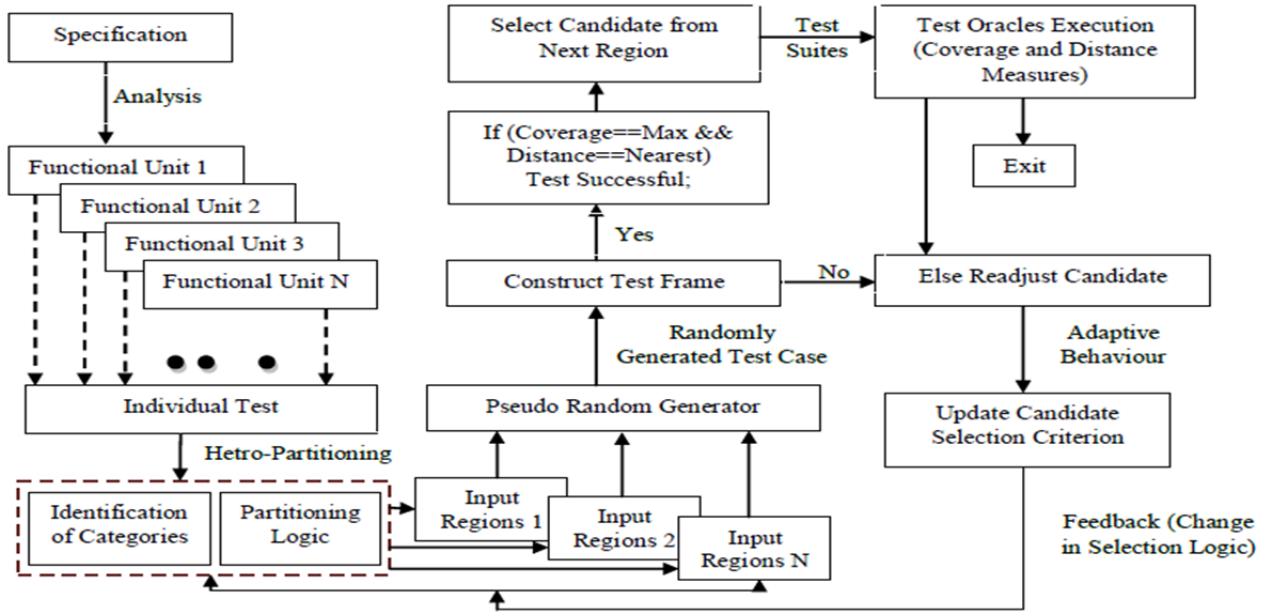

FIGURE 2: PROPOSED HYBRID ARPT BASED TESTING

Later on the generated test cases are tested thoroughly on various scripts for detecting the bugs. Here the works aims towards maximum bugs removal with near optimal efforts and costs. Thus the effective combination of all the above black box testing is used.The above process is used for creating the specification according to the semantic requirements analysis. Here the testers, main job is to identify the categories by rationing the correct separation conditions. Once the regions are divided variable test frames are generated using random sequence generation functions. And later on the criteria are verified for coverage and distance measure conditions.

***Branch and Flow statement Coverage Rate*:**

It is used to identify the code coverage achieved by generated test cases. If the coverage is maximum and the number test cases are minimum then it represents effective testing.

Now to calculate the coverage of functional bock i is following:

Number of executable statements executed=n_i

Total number of executable statements = t_i

$$\text{Coverage } C_i = (n_i / t_i) \text{ X } 100$$

$$\text{Which means } n_i = (t_i \text{ X } C_i)/ 100$$

Now the total statement is calculated using following formula:

$$C=([\sum (t_i \text{ X } C_i)/100] / \sum t_i) \text{ X } 100 = \sum (t_i \text{ X } C_i)/ \sum t_i$$

At the analytical level of evaluation, approach seems to be effective and well performed than existing mechanism. The approach also reduces the computation load with better monitoring and resource consumption analysis.

VI. RESULT EVALUATION

The suggested system is completely implemented on the .NET framework which provides various features for serving the complete feature in the form a tool view. Here the tool is also been able to analyzed the generation process on the basis of some of the well known factors such as number of generated test, compete coverage achieved by the generated test, generation time, the system resources such as CPU and RAM utilized etc. The robust experimental analysis shows the tool behaviour setting a milestone in the field of test case generation and coverage analyses. The partition and random test will give the effective results after a complete evaluation and demonstration of the developed tool. Result obtained is in the form of Comparison Table's, Graphs, Utilities Functions, Features, Parameter Covered tables.

Result Set Table 1: Normal Test Generation

S. No	Test Data	Test Type	No Testing (Test Count)	Reduced Test Counts	Generation Time	CPU Utilization	RAM Utilization	Page Faults
1	Test Set 1	Static	**48**	**18**	**0.577**	**12%**	**45.204**	**55%**
2	Test Set 2	Static	**54**	**9**	**0.53**	**9%**	**45.39**	**55%**
3	Test Set 3	Dynamic	**26**	**12**	**0.42**	**7%**	**45.50**	**54%**
4	Test Set 4	Dynamic	**99**	**21**	**3.96**	**27%**	**46.67**	**54%**

Result Set Table 2: Selective Test Generation

S. No	Test Data	Test Type	Partition Logic	Selected Attributes	CPU Utilization	RAM Utilization	Page Faults
1	Test Set 1	Static	**CSV**	**02**	**6%**	**43.49**	**33%**
2	Test Set 2	Static	**CSV**	**03**	**16%**	**43.53**	**35%**
3	Test Set 3	Dynamic	**CSV**	**02**	**5%**	**43.29**	**33%**
4	Test Set 4	Dynamic	**CSV**	**04**	**19%**	**44.03**	**36%**

Result Set Table 3: Pseudorandom Selective Test Generation

S. No	Test Data	Test Type	Reduced Test Counts	CPU Utilization	RAM Utilization	Page Faults
1	Test Set 1	Static	**18**	**12%**	**47.25**	**31%**
2	Test Set 2	Static	**09**	**18%**	**48.36**	**32%**
3	Test Set 3	Dynamic	**09**	**20%**	**49.89**	**34%**
4	Test Set 4	Dynamic	**21**	**24%**	**50.44**	**37%**

Result Interpretation(1)

Table shows the results of multiple test combinations of attributes along with the fixed and variable number of attributes. Table is concluding the reduction of the test set counts after use of Hybrid Adaptive Random Partition Testing and increase in the accuracy of calculation of test cases. Also that the process of calculation the test cases involve lesser system resources like CPU and RAM utilization. we found the maximum coverage with reduced test cases generated using proposed approach.

Result Interpretation(2)

The test case generation process always gives all possible combinations of test cases. Table 2 shows the facility of selecting the number of attributes that must included in the resultant test cases mandatorily or by choice. Our tool follows comma separated values for the giving number of selected attributes. Also the system performance while generating selective results.

Result Interpretation(3)

Selective results (Table 3)shows the combinations of the test cases that must include the selected attributes only, but also we can randomize the results so to get the coverage of the test case maximum. This can be done using Pseudorandom generator.

Coverage Anaysis:

Now the above generated comma separated value (csv) files are passed as an input to the ComCoverage tool which analyses the coverage achieved without partition logic, with partition logic and with pseudorandom generation stages of the solution phases.

VII. CONCLUSION

Software testing with adaptive behaviour will always allows some open process for testing and its re-execution. The effective test cases can be determined if the test comes from complete regions and covers at least once each type of input. But all of certain such heavy numbers of inputs are not tested with some minimum attempts. Hence, a new mechanism is required which reduces the test size but increases the code coverage. It works towards assuring the reliability of the system. This paper proposed a novel Hybrid ARPT, which works towards effective and early identification of bugs according even with their priority levels also. Means the module which is most critical should be tested more. It overcomes the existing issues of high testing cost and computation complexity. The paper also suggests some analytical evaluation factors and compares the RT, PT, ART with hybrid ARPT. On the preliminary evaluations the works seems to provide effective solution of testing domains.

REFERENCES

[1] Elaine J. Wicker and Benching Jing, "*Analyzing Partition Testing Strategies*", In IEEE Transaction of Software Engineering, doi: 009%5589/91/0700-0703$, July 1991.

[2] B A Eichmann, "*Some Remarks about Random Testing*", in National Physical Laboratory, Teddington, Middlesex, UK, May 1998.

[3] V. N. Nair, D. A. James,W. K. Ehrlich and J. Zevallos, "*A Statistical Assessment of Some Software Testing Strategies and Application of Experimental Designs Techniques*", in Journal of Statistica Sinica, University of Michigan, Bell Labs and AT&T Laboratories, Vol 8, 1998.

[4] Richard Hamlet "*Random Testing*", in international Journal of Software Engineering, Hexawise, doi:0018-9162/5527/2-22/2000, March 2000.

[5] Kirk Sayre and J.H. Poore, "*Partition Testing With Usage Models*", in University of Tennessee, 2001

[6] Arindam Chakrabarti and Patrice Godefroid, "*Software Partitioning for Effective Automated Unit Testing*", in EMSOFT'06 Conference by ACM, doi: 1595935428/06/0010, 2006.

[7] Sinaga, A., Zhou, Z., Susilo, W., Zhao, L. & Cai, K. 2009, *"Improving software testing cost-effectiveness through dynamic partitioning*", in B. Choi (eds), Proceedings of the 9th International Conference on Quality Software, IEEE, Los Alamitos, USA, pp. 249-258.

[8] Andrea Arcuri, Muhammad Zohaib Iqbal and Lionel

Briand,"*Random Testing: Theoretical Results and Practical Implications*", in International Symposium on Software Testing and Analysis (ISSTA), ACM, 2010.

[9] T.Y. Chen, G. Eddy, R. Merkel and P.K. Wong, "*Adaptive Random Testing Through Dynamic Partitioning*", in Proceedings of the Fourth International Conference on Quality Software (QSIC'04), IEEE, doi:0-7695-2207-6/04, 2010

[10] Kulvinder Singh, Rakesh Kumar and Iqbal Kaur, " *Effective Test Case Generation Using Anti Random Software Testing*", in International Journal of Engineering Science and Technology Vol. 2(11), 2010, 6016-6021.

[11] Marcel Böhme, "*Software Regression as Change of Input Partitioning*", in ICSE Doctoral Symposium , IEEE, Zurich, Switzerland , doi:978-1-4673-1067-3/12, 2012

[12] Chengying Mao, "*Adaptive Random Testing Based on Two-Point Partitioning*", in International Journal of Informatica, Volume 36, 2012

[13] Rakesh Kumar, Surjeet Singh, Girdhar Gopal, "*Automatic Test Suit generation with Genetic Algorithm*", in IJETCAS, ISSN (Online): 2279-0055, 2013

[14] Junpeng Lv, Hai Hu, Kai-Yuan Cai, and Tsong Yueh Chen, "*Adaptive and Random Partition Software Testing*", in IEEE Transaction of Systems , Man and Cybernetics: Systems, ISSN 2168-2216 ,doi: 10.1109/TSMC.2014.2318019, 2014

[15] Mahesh Malviya,Reetika Patidar "*H-ARPT: A Hybrid Adaptive Random Partition Testing for Improved Bug Detection and Test Coverage* "International Journal of Computer Science and Information Technologies, Vol. 5 (5) , 2014, 6796-6801