Indian Journal of Science and Technology, Vol 8(35), DOI:10.17485/ijst/2015/v8i35/81102, December 2015
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Combined Genetic and Simulated Annealing Approach for Test Case Prioritization

R. Uma Maheswari[1*] and D. Jeya Mala[2]

[1]Department of Computer Applications, K.L.N. College of Engineering, Sivagangai - 630612, Tamil Nadu, India; uma05raj@gmail.com

[2]Department of Computer Applications, Thiagarajar College of Engineering, Madurai - 625005, Tamil Nadu, India; djmcse@tce.edu

Abstract

Background: The test case prioritization of regression Testing is described. **Methods:** A new test case prioritization algorithm is proposed to get better the rate of fault detection and cost reduction. Heuristic Techniques like Genetic Algorithm (GA) and Simulated Annealing (SA) are employed. It prioritizes the test cases depends on fault detection ability and execution time taken. **Findings:** The implementation of proposed algorithm in JAVA is found to produce optimal or near optimal results. The efficiency of proposed regression testing technique is proved by comparing it with GA and SA methods individually. **Applications:** A complete automation tool for the complete usage of the algorithm is being developed. It will also be analyzed on larger projects with large number of test cases and faults.

Keywords: Fault Coverage, Genetic Algorithm, Heuristic Techniques, Regression Testing, Simulated Annealing, Test Case Prioritization

1. Introduction

Regression testing is the process of testing changes to computer programs to make sure that the older programming still works with the new changes. It is a chic activity and devours huge amount of exertion and effort. The test suites are already offered from previous stages of Software Development Life Cycle (SDLC). Regression testing does not involve rerunning the whole suite however choosing a set of it that may observe all the faults. There are various regression techniques: Retest all, Regression test selection and test case prioritization and hybrid approaches.

Prioritization orders the test cases so that the most beneficial are executed first according to some fitness function such as fault coverage, Path coverage and statement coverage. The combined Genetic and simulated annealing can be utilized to tackle issues of time-obliged environment productively. The GASA has been used before to solve traveling salesman problem, Sub synchronous Damping Control in Electrical Power Transmission Systems[1]. Simulated Annealing is based upon the analogy with the simulation of the annealing of solids. It is designed to accept candidates with higher cost to escape from local minimum. The Temperature schedule must be selected in such a way that Initial temperature to consider non-best solutions for selection and final temperature to consider only best solutions. Genetic algorithms are heuristic technique based on the natural process of evolution.

Regression Testing has been implemented and investigated from numerous points of view. Numerous methods are proposed by scientists for lessening expense identified with regression testing. Routhermel et al.[2] investigated several prioritizing techniques based on various criteria like total statement or branch coverage prioritization and additional statement or branch coverage prioritization to improve the rate of fault detection. Greedy Algorithm and an Additional Greedy Algorithm based on code coverage is used in above techniques.

Rajib Mall et al.[3] proposed a novel regression test case prioritization technique based on an analysis of

*Author for correspondence

a dependence model for object-oriented programs. It represents control and data dependencies, association, inheritance and aggregation. Yuen Tak Yu et al.[4] proposed and developed the fault-based prioritization of test cases based on fault-detecting ability and the relationships among the test cases. Krishna moorthy et al.[5] proposed models that prioritizes the system test cases based on the six factors: customer priority, changes in requirement, implementation complexity, completeness, traceability and fault impact. Prabha et al.[6] proposed a novel regression testing method based on criticality measure calculated by means of dependability metrics and internal Complexity Metrics of Components.

Mala et al.[7] proposed a test suite optimization method based on artificial bee colony technique. It was found to outperform existing GA Technique. Karur et al.[8] used HPSO algorithm which combines the features of Genetic Algorithm and Particle Swarm Optimization to make regression testing efficient.

Alexandre et al.[9] built the test suites prioritization method using string distances (Hamming, Cartesian, Levenshtein or Edit distance, Manhattan) and found they are more efficient in detecting faults have a better APFD than randomly ordered test suites. Harman et al.[10] Provides the survey of search based techniques for Test Case Prioritization. Arnaldo singo et al.[11] applies adaptive random testing for Prioritization. Singh et al.[12] proposed a new variable based algorithm works for programs with multiple modules using the hybrid technique. Uma et al.[13] proposed Test case Prioritization to achieve maximum fault coverage at the earliest. It Uses Hamming Distance to find the next test case in Prioritized Order designed to reduce time and cost.

Suri et al.[14] presented a new test case reduction hybrid technique based on Genetic Algorithms (GA) and Bee Colony Optimization (BCO). Varun et al.[15] proposed a new approach which considers the severity of faults based on requirement prioritization. Aim is to find the severe faults early in the testing process and hence to improve the quality of the software according to customer point of view.

2. Genetic Algorithm

Genetic algorithms are heuristic technique based on the natural process of evolution. The survival of the fittest among individuals over consecutive generation for solving a problem is simulated by this algorithm.

The basic process for a genetic algorithm is:

- Initialization – Initial population of desired size is created randomly.
- Evaluation – Fitness of each Member is calculated to identify how well it fits with the desired requirements.
- Selection – In this step, Fitter individual will be selected for next generation by discarding the bad individuals to improve the overall fitness of population.
- Crossover – Here Selected Individuals are combined to create new individuals. It is done with the intention to inherit the best traits from each of its parents to its offspring.
- Mutation – Little bit randomness into the population's genetics is added.
- And repeat – Now Start next generation again from step two until a termination condition is reached.

3. Simulated Annealing

Simula1ted Annealing is based upon the analogy with the simulation of the annealing of solids. It is designed to accept candidates with higher cost to escape from local minimum. The Temperature schedule must be selected in such a way that Initial temperature to consider non-best solutions for selection and final temperature to consider only best solutions.

The generic flow of SA algorithm is given below.

- Generate an initial solution.
- Get an initial temperature T0, where T0>0.
- Execute the following steps till temperature does not freeze.
- Perform the following loop K times.
- Select a random neighbor solution Solp of Sol.
- Calculate $\Delta = F(Solp) - F(Sol)$.
- If $\Delta < 0$ (i.e. downhill move) then Sol = Solp.
- If $\Delta \geq 0$ (uphill move) then Sol = Solp with probability $P(\Delta, T)$.
- If $F(Sol) \geq F(SolBest)$ then SolBest = Sol.
- Upgrade the temperature (T) using cooling rate.
- SolBest is best solution.

4. Proposed Technique

It is troublesome for typical strategies to resolve Test Case Prioritization problem due to its large solution space. So, a technique which combines Genetic Algorithm (GA)

and Simulated Annealing (SA), termed GASA has been proposed. The Proposed Techniques takes the advantage of both algorithms to make regression Testing Efficient. Quick Processing and exploration of large solution space property of GAs and Efficient local solution Improvement Property by SA are taken in to consideration. The solutions are randomly generated by GA and they are further refined by SA.

Input

- T - Test Suite to be prioritized.
- N - No of Test Case in T.
- Fi I = 1 to N – No of faults Covered by each Test Case.
- For Each Test Case ti (I = 1 to N).
- F(ti) – Set of Faults covered by ti.
- E(ti) – Execution time of ti.

Output

T'- Prioritized Test Suite

Steps:

- Initial population is set by randomly generating n feasible chromosomes.
- Define the initial temperature for SA.
- Calculate fitness value of each chromosome.
- Apply the parallel SA algorithm: It involves two things. First a Chromosome generated by GA is taken and is mutated to generate new Chromosome. Let fi and fi+1 be the fitness values of original and new Chromosome respectively. If fi+1 >fi, new chromosome is accepted as the starting point for the next iteration.
- As the optimization proceeds, the temperature is updated so as to consider non-best solutions for selection during initial temperature and only best solutions during final Temperature.
- GASA terminates if the maximum number of generations allowed is reached. Otherwise, go back to Step 3.

5. Algorithm Explanation

The GASA is a heuristic based technique, where some population of chromosomes is taken and is refined to approach the solution. The population plays major role to decide the manner in which the solution will approach. The process flow of the proposed technique GASA is shown in Figure 1. In our procedure for test case prioritization, Total Execution time to cover all faults is taken as a fitness function. The regression testing which is followed in example is total fault covered – in less time. The

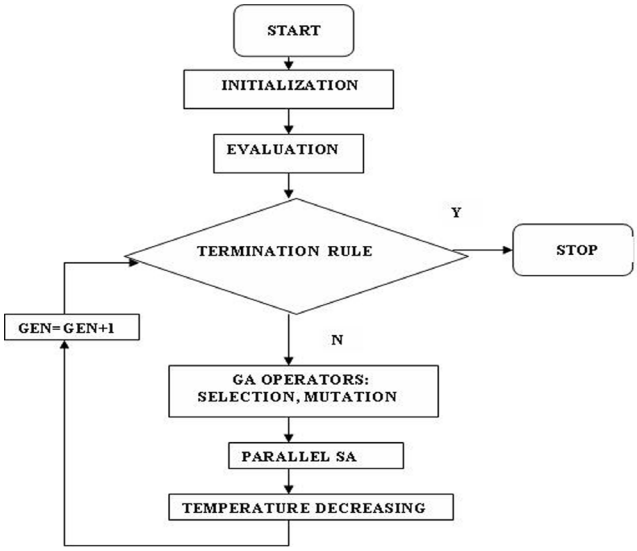

Figure 1. Flow of Proposed Method.

procedure will be terminated after the maximum number of iterations. The criteria considered here is full optimized solution. The initiation of GASA algorithm starts with random generation of population and initialization of Annealing Parameters. Once first random population is generated, the fitness of each is assessed. At that point every chromosome is refined by SA and new population is generated. State transition and acceptance are repeatedly used throughout the SA. In mutation, the operation of state transition is identical to that of GA mutation. The random GA mutation theory has been used. GA-SA terminates if the maximum number of generations allowed is reached. This process will continue until the stopping criterion is met.

6. Implementation

Simulated Annealing for test case Prioritization has been implemented in Java. The Figure 2 and Figure 3 Shows Execution of the Program.

7. Experimental Results

This section discusses an evaluation result of the above experiment for test suite with 7 test cases covering a total of 11 faults. The regression test suite contains seven test cases {T1, T2, T3, T4, T5, T6, and T7}. The algorithm assumes the knowledge of the faults detected by each test case and its execution time in T as shown in Table 1.

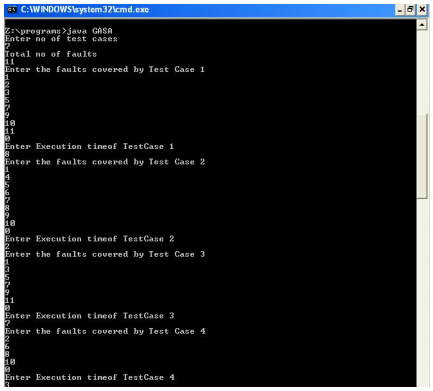

Figure 2. Execution of GASA.

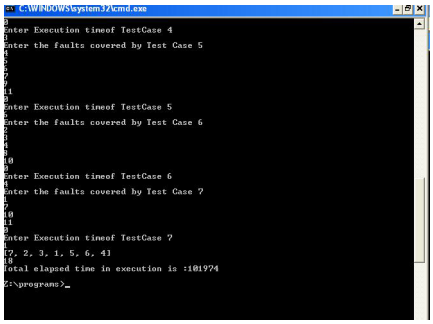

Figure 3. Execution of GASA.

Table 1. Fault Covered by Test Case and Execution Time

Test Case	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	Execution Time
1	X	X	X		X		X		X	X	X	8
2	X			X	X	X	X	X	X	X		2
3	X		X		X		X		X		X	7
4		X				X		X		X		3
5				X	X	X	X		X		X	6
6		X	X	X				X		X		4
7	X						X			X	X	1

The Genetic Algorithm, Simulated Annealing Algorithm and Proposed hybrid GASA algorithm are executed against this example for 10 times and results achieved are represented in Figure 4, Figure 5 and Figure 6 respectively.

Table 2 shows that the proposed method is able to produce optimal or near optimal results for test case Prioritization Problem. Figure 7 visualizes the results of GASA for various Test Data.

Figure 8 presents a graph that compares the above proposed method to GA, SA test case prioritization

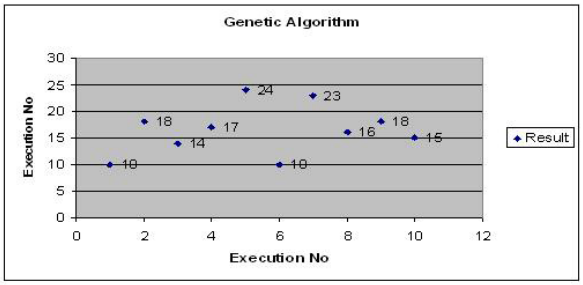

Figure 4. Execution of Genetic Algorithm.

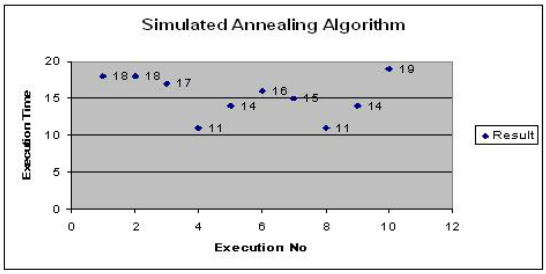

Figure 5. Execution of Simulated Annealing.

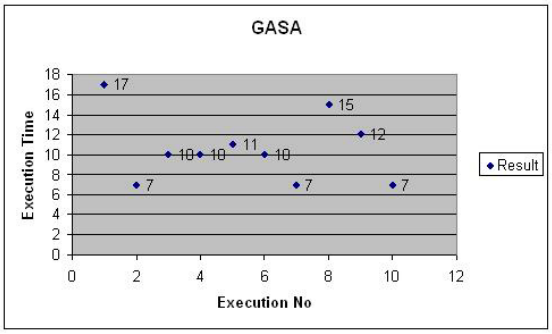

Figure 6. Execution of GASA.

Table 2. Result Comparison

Test Data	Best Result	Optimum Result
1	8	6
2	8	7
3	22	21
4	7	7

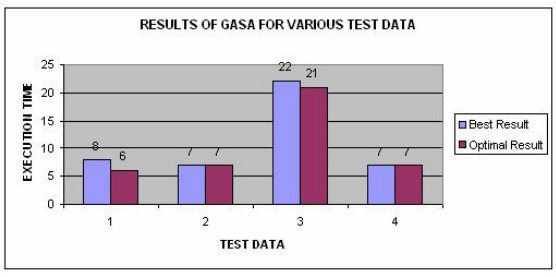

Figure 7. Results of GASA for various Test Data.

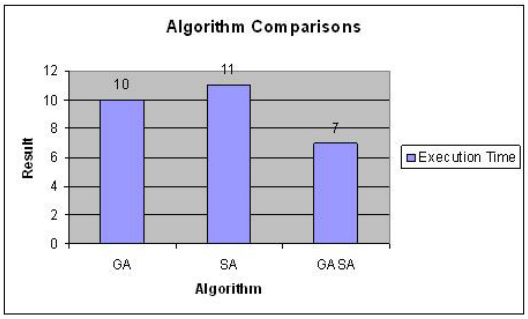

Figure 8. Comparison of GASA over GA and SA.

techniques. Also, the graph shows that the "GASA" method consumes the least total time during a prioritization process, comparing to other techniques.

8. Conclusion

This work proposed a new methodology by combining simulated annealing and a genetic algorithm (GASA) in order to reap the advantages of SA and lessen the time that GA spends stuck at local minima. The analysis for the examples of fault coverage shows encouraging results. All these factors indicate that the GASA technique could be used to prioritize the regression test suite. In spite of the fact that the algorithm has been implemented successfully, it obliges manual interface to input test suite information which makes the utility of the implemented part restricted to small sized test suite.

9. References

1. Xie X. Genetic algorithm and simulated annealing: A Combined Intelligent Optimization Method and Its Application to Subsynchronous Damping Control in Electrical Power Transmission Systems. 2012; 245–70.
2. Malishevsky AG, Rothermel G, Elabaum S. Test case prioritization: A family of empirical studies. IEEE Transactions on Software Engineering. 2002; 28(2):159–82.
3. Panigrahi CR, Mall R. An approach to prioritize regression test cases of object-oriented programs. J. CSI Trans ICT. 2013; 1(2):159–73.
4. Yu YT, Lau MF. Fault-based test suite prioritization for specification-based testing. Information and Software Technology. 2012; 54(2):179–202.
5. Krishnamoorthi R, Mary SA. Factor oriented requirement coverage based system test case prioritization of new and regression test cases. Information and Software Technology. 2009; 51(4):799–808.
6. Prabha MR, Mala DJ. Critical Components Identification and Verification for effective Software Test Prioritization. IEEE 3rd International Conference on Advanced Computing (ICoAC); Chennai. 2011 Dec. p.181–6.
7. Mala DJ, Mohan V. On the use of intelligent agents to guide test sequence selection and optimization. International Journal of Computational Intelligence and Applications. 2009; 8(2):155–79.
8. Kaur A, Bhatt D. Hybrid Particle Swarm Optimization for Regression Testing. University School of Information Technology. IJCSE. 2011; 3(5):1815–24.
9. Ledru Y, Petrenko A, Boroday S, Mandran N. Prioritizing test cases with string distances. Automated Software Engineering. 2012; 19(1):65–95.
10. Harman M, McMinn P. A theoretical and empirical study of search-based testing: Local, global, and hybrid search. IEEE Transactions on Software Engineering. 2010; 36(2):226–47.
11. Zhou ZQ, Sinaga A, Susilo W. On the fault-detection capabilities of adaptive random test case prioritization: Case studies with large test suites. IEEE 45th Hawaii International Conference on Systems Sciences (HICSS); Maui, HI. 2012. p. 5584–93.
12. Singh Y, Kaur A, Suri B. A hybrid approach for regression testing in interprocedural program. Journal of Information Processing Systems. 2010; 6(1):21–32.
13. Maheswari RU, Jeya Mala D. A novel approach for test case prioritization. IEEE International Conference On Computational Intelligence and Computing Research; Enathi. 2013. p. 1–5.
14. Suri B, Mangal I, Srivastava V. Regression test suite reduction using an hybrid technique based on BCO and genetic algorithm. IJCSI. 2011; 2(1-2):165–72.
15. Kumar V, Kumar Mohit S. Test Case Prioritization Using Fault Severity. IJCST. 2010; 1(1):67–71.
16. Indhumathi S, Venkatesan D. Improving coverage deployment for dynamic nodes using genetic algorithm in wireless sensor networks. Indian Journal of Science and Technology. 2015; 8(16):1–6.
17. Maheswari RU, Mala DJ. WOV based multi objective test case prioritization. International Journal of Applied Engineering Research. 2015; 10(55):1165–9.