International Journal of Engineering & Technology, 7 (4.5) (2018) 242-247

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

A Survey on Techniques Adopted in the Prioritization of Test Cases for Regression Testing

John Bruce. E[1]*, T. Sasi Prabha[2]

[1] Research Scholar, Faculty of Computing, Sathyabama Institute of Science and Technology
[2] Pro Vice-Chancellor, Faculty of Computing, Sathyabama Institute of Science and Technology
**Corresponding author E-mail:johnbruce@sathyabama.ac.in*

Abstract

Regression testing is testing the software with the intention to confirm that changes made on part of a module do not necessitate other parts of the module. Test case prioritization helps to reduce regression testing cost by ordering the test cases in such a way that it produces optimized results. Code Coverage and Fault detection being the factors behind the prioritization is dealt with techniques like Heuristic method, Meta Heuristic methods and Data mining techniques. The effectiveness of the techniques applied can be evaluated with the metrics like Average Percentage of Fault Detection (APFD) , Average Percentage Block Coverage (APBC), Average Percentage Decision Coverage (APDC) etc . In this paper,, a detailed survey on the various techniques adopted for the prioritization of test cases are presented.

Keywords: *Code Coverage ; Greedy ; Machine Learning; Meta Heuristic ; Prioritization, etc.*

1. Introduction

Regression testing accounts for 80% of the maintenance cost and thus optimizing regression testing is one of the objectives of testing team. It has been proved that prioritizing test cases based on the coverage criterion can achieve earlier fault detection by reducing the cost of testing. Test cases are assigned weights based on their ability to cover most part of the program to be tested. The test cases which cover most number of modified lines can be assigned with the highest priority and is executed first and the one with the least coverage of modified lines can be assigned with the least priority and is executed last for effective earlier fault detection. Prioritizing test cases based on Code Coverage and ignoring fault detection could produce some vague results. Taken a Program P and a Test Suite T, let PT be the set of permutations of T ; a function f from PT to the real numbers can be stated as follows :

Find T' € PT such that for all T" where (T" € PT) (T" ≠ T') [f (T") ≥ f(T)], where PT represents the set of possible prioritization of T and f is a function that applied to such ordering yields an fitness value for that ordering.

A super statement is formally defined as a set of statements that are executed by the same test cases in T. For any statement si and sj, if there exists a test case in T that executes one but only one of these two statements, the two statements must belong to different super statements.
Based on the concept of super statements, the statements of P may be divided into several sets of statements, each of which is a super statement. In order to validate the test cases, it is required to define the objective in a quantitative manner.

2. Motivation

Program testing can be a very effective way to show the presence of bugs, but it is hopelessly inadequate for showing their absence" [Dijkstra, 1972]. Most beta testers are "techies" who have higher tolerance of bugs and they do not report usability problems. Customers look for more reliable systems and organizations must perform cost/benefit analysis in order to determine how much to spend on testing. Software testing team save their test cases or test suites they have written for reusability and coverage. This type of re-use of test cases leads to regression testing. Test case prioritization techniques aim at defining an order of test cases that favor the achievement of a goal during test execution, such as revealing failures with minimum execution time. A number of techniques have been proposed in the literature such as the test case selection, test suite reduction and test case prioritization. The test case selection problem deals with selecting a subset of the test cases according to a specific criterion, whereas test suite reduction techniques deals with on selecting a cost effective subset of the test cases which covers its subsequent subsets..
Test case prioritization techniques, on the other hand address the problem of defining an execution order of the test cases according to a given testing goal, particularly, detecting failures as early as possible.

3. Literature Survey

Alessandro Marchetto et al (2015), has dealt deals with ordering of test cases using IR Traceability recovery that maximize the number of discovered faults that are both technical and business critical[3]. Dan Hao et al (2016) has conducted an empirical study on ten non-trivial object projects by representing as Integer Linear Programming (ILP) and it has been proved that Optimal coverage outperforms the additional coverage[5]. Automated test generation tool with high level petri nets to capture high level control and data requirements for functional testing has been devised by Dianxiang Xu (2015). Comparison of fault-detection capability, Time performance and scalability for different coverage are done[7].

It is also proved Latent Latent Semantic Analysis(LSI) outperforms Vector Space Model (VSM). Javier Tuya et al (2016) has revised a reduction algorithm which optimizes the query using cost and distance and executes the SQL query by splitting into various components [10]. Given a query q and a document Model V, the Lijun Mei et al (2015) has proposed refinement-oriented level-exploration strategy and a multilevel coverage method which locates a rule whose left hand side matches the query q by following the strategy for subsumption relation [12]. The experimental results have shown the model instance achieving a higher fault detection rate than a subsumed technique, which validates that the proposed hierarchy and have the potential to improve the cost-effectiveness of test case prioritization techniques.

Per Erik et al (2017) has developed a called SuiteBuilder which collects raw Test Suites and Assign Priorities . Since the alphabetical ordering of test cases has shown failures in the past, selection of test cases is done by selecting a subset of test cases given knowledge of the changes to the software [15].

Predicting the software components having more vulnerability using machine learning algorithms has been done by Riccardo Scandariato (2014). Identifying certain features as predictors, the authors using term frequencies have predicted whether a file is vulnerable or not using Naïve Bayes and Random Forest prediction methods. Java applications from f-droid.org were tested and experimental results have shown Naïve Bayes showing better performance than random forest[16].

A test case which has the highest coverage of not-yet-covered entities is called as a tie. Lexicographical ordering, which deals with ordering of lexicons following alphabetical order is proposed for breaking ties. A surrogate key is used which statistically or heuristically correlates with the faults, expecting that maximizing the surrogate will lead to maximizing the rate of fault detection [17]. Yi Bian et al (2017) have proved that the search space is reduced by combining knowledge of the application domain with a biological theory called Ant Colony optimization. The complexity of the problem and the availability of huge number of test cases is a challenge to find the relations between test cases [20].

Tingting Ma et al (2010) proposes a test case prioritization method based on requirement correlations based on the requirements given by users and developers. By readjusting prioritization of fault-related requirements, it can further optimize the order of test cases. Prioritization of test cases using artificial intelligence techniques like Simulated Annealing, Genetic Algorithm Swarm optimization, Ant colony optimization, Bee colony optimization has shown better performance than Greedy method [1] [4][6][11][21]. From the available literature, no evidence has been existing to prove whether there is a high correlation between coverage and fault detection.

4. Techniques Used

Optimization problems can be divided into two major categories as exact and approximate. Exact algorithms give exact solution whereas approximate algorithm may or may not give exact solution. The Taxonomy of Prioritization of test cases using different optimization problems is shown in Fig.1.

Algorithms are classified as Local and Global Search algorithms. Local search algorithms search the neighborhood of candidate solutions. Some examples of local search algorithms are simulated annealing, tabu search, iterated local search and variable neighborhood For example, a hill climbing search starts with a random solution and its neighbours are evaluated based on its defined fitness objective function. The search can easily get stuck in local optima, which can be overcome by restarting the search with new random values. Contrary to local search algorithms, global search algorithms try to overcome local optima in order to find more globally optimal solutions. With *evolutionary testing*, one of the most commonly applied global search algorithms is a *Genetic Algorithm* (GA).

A Memetic Algorithm (MA) is a hybrid form of global and local search, such that the individuals of a population in a global search algorithm have the opportunity for local improvement in terms of local and the global search. Test case selection techniques focus on covering the changed code between versions of the software under test.

Iterative and Greedy based Techniques are categorized by :

(a) Total Technique (TT) , which prioritizes test cases by maximizing the total number of covered entities.
(b) Additional Technique (AT), chooses the test case that covers the highest number of yet-cover entities

Greedy based algorithms may produce sub optimal results because they may get struck with local minima where as Meta-heuristic and evolutionary search algorithms aim to avoid such problems. They have been used for strategy advancement. Greedy algorithm may not give the optimal solution always. Given the test case in Tab 1, there are only two possible orderings:

Option 1 : 3 2 1 4or 3 2 4 1 based on the priority assigned

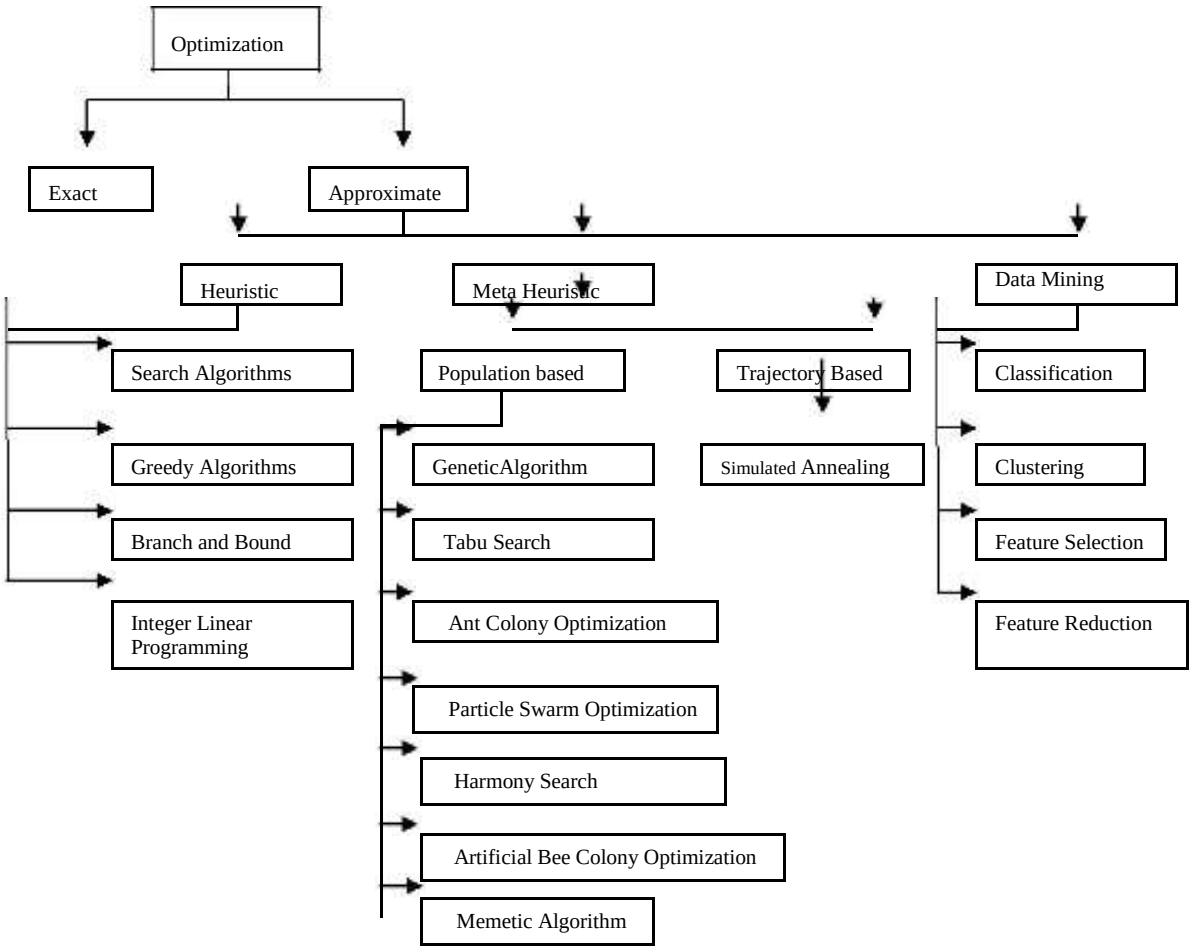

Fig. 1 Taxonomy of Prioritization of Test Cases using different Optimization Problems

Table 1 : Eg for local minima

	T1	T2	T3	T4	T5	T6	T7
1.	✓	✓	✓				
2.	✓		✓	✓	✓		✓
3.	✓	✓	✓	✓	✓	✓	✓
4.				✓	✓	✓	

Meta heuristic is a general algorithmic framework for addressing intractable problems. They are inspired by processes occurring in nature. Meta-heuristic algorithms are used to solve such problems which suffer at Local Minima and NP-Hard. A metaheuristic is formally defined as an iterative generation process which guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space, learning strategies are used to structure information in order to find efficiently near-optimal solutions (Osman and Laporte 1996). Metaheuristics are strategies that "guide" the search process. The goal is explore the search space in order to find the (near) optimal solutions. They cannot always produce optimal solutions, but they do have the potential to produce good solutions in short amount of time.

Genetic algorithm: In genetic algorithm, the initial population needs to be generated with the fitness function defined between 0 and 1 and it can provide significant reduction in the number of test cases. The higher amount of Fitness function, the more probability is the test frequency. GA and **Search algorithm Tabu search** divides the test cases into two groups: Short term and long term. The most frequently used test cases are in short term list where as all test cases will be kept in long term list. The long term test cases have more effects for error detection. The two lists are used for the creation of new test cases. These test cases can help in finding error in less time complexity and hence this method is considered as one of the best method.

Firefly is the next method used to find test cases. The radical idea is generated from luminescent relation of fireflies. The firefly pays

attention to luminosity of the other fireflies. Optimal solution can be obtained by Adjacency and Confusion matrix . In comparsion with ACO algorithm, this algorithm has better coverage.

Particle Swarm Optimization (PSO) optimizes a given problem by iteratively trying to improve a candidate solution against a measure of quality according to few simple formulae. Formally, let f: $\mathbb{R}n \rightarrow \mathbb{R}$ be the cost function which must be minimized. The function takes inputs in the form of a vector of real numbers and produces a real number as output which indicates the objective function. . The objective is to find a global solution a for which $f(a) \leq f(b)$ for all b in the search-space. This method does not guarantee that optimal solution is ever reached. Optimizations problems which are Np-Hard like Travelling Sales Person and Job sequencing with deadlines can be solved by PSO method with minimum execution time.

Ant Colony optimization (ACO) is a probabilistic technique used for solving problems which can be reduced to finding optimal paths through graphs. . The inspiring source of Ant Colony Optimization is the foraging behavior of real ant colonies. ACO algorithms are used for automatic test case generation. ACO algorithms may belong to different classes of approximation algorithms and these algorithms are better than Simulated Annealing and Genetic algorithm for Coverage capability, Convergence speed and stability.

Artificial Bee Colony (ABC) is based on the intelligent behavior of Bees. The Bee Colony has three categories of Bees as employed bees, onlookers and scouts. The employed bees search food around the food source in their memory and they convey information about these food sources to the onlooker bees.

Memetic algorithms (MA) is a form of evolutionary or any population-based approach with separate individual learning or local improvement procedures for problem search

Cuckoo Search (CS) is an optimization algorithm which was inspired by the obligate brood parasitism of some cuckoo species by laying their eggs in the nests of other host birds (of other species). It has both local and global search ability. The major advantage of CS is global optimization capability.

Harmony Search (HS) is a population based meta heuristic algorithm inspired from the musical process of searching for a perfect state of harmony. The pitch of each musical instrument determines the aesthetic quality, just as the fitness function value which determines the quality of decision variable. It is used to solve NP-hard optimization problems. This method is versatile to combine with other meta heuristic algorithms to produce Hybrid Meta Heuristics which can be applied in various applications.

Several relevant tasks in Data Mining such as clustering, classification, feature selection and data reduction are formulated as optimization problems, in which feature selection and feature reduction are used for test case selection. Table 2 lists the survey on prioritization.

5. Evaluation Metrics

The order in which the test cases are executed affects the Rate of Code coverage and Rate of Fault Detection. Heuristics are required to estimate the ability of a test case to reveal faults as shown in equations (1-3).

1. Average percent of faults detected (APFD)

Given T being the test tuple, *g* the number of faults in program under test, n the number of test cases, *reveal(i, T)* = position of the first test in *T* that exposes fault *i*

$$APFD(T,P) = 1 - \frac{\sum_{i=1}^{g} reveal(i,T)}{ng} + \frac{1}{2n} \quad \textbf{(1)}$$

2. **Total statement coverage and Decision Coverage**:

Test cases are prioritized by number of statements covered. The statement covers only the true conditions

$$\text{Statement coverage} = \frac{\text{Number of statements exercised}}{\text{Total number of statements}} \times 100\% \quad \textbf{(2)}$$

$$\text{Decision coverage} = \frac{\text{Number of decision outcomes exercised}}{\text{Total number of decision outcomes}} \times 100\% \quad \textbf{(3)}$$

3. Total Fault-Exposing-Potential (FEP) are one in which the test cases are prioritized by FEP.

$$FEP(t_i, s_j) = \frac{|\text{mutants}(s_j) \text{ killed by } t_i|}{|\text{mutants}(s_j)|}$$

$$FEP(t_i) = \sum_j FEP(t_i, s_j)$$

Table 2 : Survey on techniques used for prioritization

Factors for prioritization	Techniques	Author	Data sets	Results
Coverage based technique	1.Generalized AT using Lexicographical Ordering for breaking ties	Sepehr Eghbali and Ladan Tahvildari [17]	Java programs like Ant, Galileo, Jmeter, Toppas, Nano, XML	Reduced time complexity and increased the rate of fault detection
	2. Reducing a database with a set of SQL queries	Javier Tuya et al [10]	Helpdesk, Compiere, TPC-H	Maintenance operations can be done
	3. Memetic Algorithm	Gordon Fraser et al [9]	12,000 Java Classes	
	4. Reduction of test cases	Ahmad A. Saifan [2]	Java Source "Cinema"	Naïve Bayes is better than CBR and J48
Additional Coverage Based Technique	1.Test case prioritization is represented as an Integer Linear Programming	Dan Hao et al [5]	Datasets from Software-Infrastructure Repository (SIR)	Reduced time complexity and increased the rate of fault detection
	2. Genetic Algorithm and Ant Colony Optimization	Yi Bian [20]	Flex, space, bash, v8	Reduced time complexity
	3. Greedy algorithm	Lijun Mei et al [13]	Hotel Booking process	Test case prioritization
Code Coverage	1.An IR based traceability recovery	Alessandro Marchetto et al [3]	21 Java applications	Earlier detection of faults
	2. Genetic Algorithm	Zheng Li et al [21]	Print_tokens, print_tokens2, schedule, schedule2, space and sed	Genetic Algorithm is better than Greedy algorithm
Reach ability Coverage	1.Automatic Test Case generation	Dianxiang Xu et al [7]		Executable functions are automated
Fault Exposing potential	1.Predicts which component contains vulnerability	Riccardo Scandariato et al [16]	d-droid.org	Random Forest is better than Naïve Bayes
	2. Multi Objective Particle Swarm Optimization	Erum Ashraf et al [8]	Web Applications	Optimizes Fault Coverage and execution
	3. Test Case Prioritization by exposing faults	Lei Xiaoet al [12]	DCS (V1.0) and DCS (V2.0)	Clustering by DB index produces better performance than K-Means clustering
Path Coverage	1.Artificial Bee Colony optimization	Soma Sekara Baba Lam et al [18]	Triangle classification problem	Improved time and space complexity

6. Conclusion:

From the available literature, it is evident that Greedy algorithm runs in O (m n) time complexity, where as Additional Greedy algorithm takes $O(m\ n^2)$, 2-Optimal algorithm takes $O(m\ n^3)$ and Hill Climbing method runs in $O(n^2)$. It is proved that additional Fault Exposing Potential is more suitable than all other prioritization techniques that are based on coverage but with a marginal gain in APFD. Evaluation Metrics are discussed. Artificial Bee Colony optimization offers advantages over Tabu Search, GeneticAlgorithm and Ant Colony Optimization and recent studies have shown CS is more efficient than PSO and GA. Heuristics can be applied for the prioritization of test cases. All the heuristics outperform the untreated or orderly prioritized test suites.

References

[1] AdiSrikanth, Nandakishore J. Kulkarni, K. Venkat Naveen, PuneetSingh, and Praveen Ranjan Srivastava(2011), "Test Case Optimization Using Artificial Bee Colony Algorithm", ACC 2011, Part III, CCIS 192, Pp. 570–579.

[2] Ahmad A. Saifan (2016), "Test Case Reduction Using Data Mining Classifier Techniques", Journal of Software, 11(7), Pp. 656 -663

[3] Alessandro Marchetto, Md. Mahfuzul Islam, Waseem Asghar, Angelo Susi,Giuseppe Scanniello (2015), "A Multi-Objective Technique to Prioritize Test Cases", IEEE Transactions on Software Engineering, 42(10), Pp. 918 – 940

[4] Chengying Mao, YuXinxin, Chen Jifu, Chen Jinfu (2012)"Generating Test Data for Structural Testing Based on Ant Colony Optimization "12th International Conference on Quality Software, Xi'an, Shaanxi, pp. 98 – 101.

[5] Dan Hao, Lu Zhang, Lei Zang, Yanbo Wang, Xingxia Wu, and Tao Xie (2016), "To Be Optimal or Not in Test-Case Prioritization ", IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 22(5), Pp.490-503

[6] Deepak Rai and Kirti Tyagi (2014), "Regression Test Case Optimization Using Honey Bee Mating Optimization Algorithm with Fuzzy Rule Base", World Applied Sciences Journal, 31(4), Pp. 654-662

[7] Dianxiang Xu, Weifeng Xu, Michael Kent, Lijo Thomas, and Linzhang Wang(2015)," An Automated Test Generation Technique for Software Quality Assurance" , IEEE TRANSACTIONS ON RELIABILITY, 64(1), Pp. 247-268

[8] Erum Ashraf , Tamim Ahmed Khan, Khurrum Mahmood, Shaftab Ahmed (2017), "Value based PSO Test Case Prioritization Algorithm", International Journal of Advanced Computer Science and Applications, 8(1), Pp. 389-394

[9] Gordon Fraser, Andrea Arcuri, Phil Mcminn (2014), "A memetic algorithm for hole test suite generation", The Journal of System and Software Elsevier

[10] Javier Tuya, Member Claudio de la Riva, Marýa Jose Suarez-Cabal, and Raquel Blanco (2016), "Coverage-Aware Test Database Reduction", IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 42(10),Pp. 941-959

[11] Jun Wang, Yan Zhuang, Chen Jianyun (2011) "Test Case Prioritization Technique based on Genetic Algorithm", International Conference on Internet Computing and Information Services, Hong Kong , pp. 173 – 175.

[12] Lei Xiao, Huaikou Miao, Weiwei Zhuang, Shaojun Chen (2017), " An empirical study on clustering approach combining fault prediction for test case prioritization", 16th International Conference on Computer and Information Science (ICIS), 978-1-5090-5507-4/17, Pp. 815-820

[13] Lijun Mei, Yan Cai, Changjiang Ji, Bo Jiang, W.K. Chan, Zhenyu Zhang, T.H. Tse (2015), "A Subsumption Hierarchy of Test Case Prioritization for Composite Services ", IEEE TRANSACTIONS ON SERVICES COMPUTING, 8(5), Pp.658-673

[14] Marwah Alian,Dima Suleiman, Adnan Shaout(2016), "Test Case Reduction Techniques- Survey", International Journal of Advanced Computer Science and Applications, 7(5), Pp. 264-275.

[15] Per Erik Strandberg, Wasif Afzal, Thomas J. Ostrand, Elaine J. Weyuker, and Daniel Sundmark(2017), "Automated System-Level Regression Test Prioritization in a Nutshell ", IEEE Software,Pp. 30-37

[16] Riccardo Scandariato , James Walden , Aram Hovsepyan , Wouter Joosen(2014), "Predicting Vulnerable Software Components via Text Mining" , IEEE Transactions on Software Engineering, 40(10), pp: 993-1006

[17] Sepehr Eghbali and Ladan Tahvildari (2016), "Test Case Prioritization Using Lexicographical Ordering",IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 42(12), Pp. 1178-1195

[18] Soma Sekhara Baba Lam, ML Hari Prasada Raju, Uday Kiran M., Swarj Ch, Praveen Ranajin Srivatsav (2012), "Automated Generation of Independent Paths and Test Suite Optimization using Artificial Bee Colony, Proceedia Engineering, Elsevier 30, 191

[19] Tingting Ma , Hongwei Zeng , Xiaolin Wang(2016), "Test case prioritization based on requirement correlations", 17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 10.1109/SNPD.2016.7515934

[20] Yi Bian, Zheng Li, Ruilian Zhao, Dunwei Gong (2017), "Epistasis Based ACO for Regression Test Case Prioritization", IEEE Transactions on Emerging Topics in Computational Intelligence, 1(3), Pp.213-223

[21] Zheng Li, Mark Harman, and Robert M. Hierons(2007), "Search Algorithms for Regression Test Case Prioritization", IEEE Transactions on Software Engineering, Vol. 33(4)