Systematic Literature Review in Software Test Data Generation

Gagan Kumar[1], Vinay Chopra[2], Dinesh Gupta[3]

Computer Science & Engineering Department, I.K.G. P.T.U Kapurthala, Punjab[1,3]
D.A.V. Institute of Engineering & Technology, Jalandhar, Punjab[2]

Corresponding author: Gagan Kumar, Email: gagan.daviet@gmail.com

In software development, software testing is an important practice which comprises of different activities. It is a time-consuming and cost-oriented process. In testing, it is very important to select the test data generation process wisely because testing efficiency is highly dependent on the data used and it may affect the cost and time. Soft computing algorithms explore test data in search-based software testing to optimize the coverage metric, which can be called an optimization challenge. Some Meta-Heuristics algorithms (Artificial Bee Colony, Particle Swarm Optimization, Genetic Algorithm, Firefly Algorithm and Ant Colony Optimization Algorithms) are selected in this paper for comparative study along with Artificial Immune Algorithms (Negative Selection Algorithm, Clonal selection, and Hybrid Negative Selection Algorithm). The Immune algorithm also has a significant impact in engineering applications and in the field of software test data generation. A survey on automated test data generation has been done on the various criteria such as type of objective function use, type and number of experiments performed for specific technique, comparison with other techniques, types of parameters used and the performance of the algorithm. From this survey it has been observed that the immune algorithms outperform meta-heuristic algorithms in terms of average coverage, average generation, cost, and average test data generated. But somehow the number of comparisons to generate test data in immune algorithms is more than the Metaheuristic algorithms.

Keywords: Test data generation, ACO, NSA, PSO, GA, ABC, FA.

2023. In Vikram Dhiman & Pooja Dhand (eds.), *Emerging Trends in Engineering and Management*, 91–107. Computing & Intelligent Systems, SCRS, India.
https://doi.org/10.56155/978-81-955020-3-5-11

1. Introduction

Testing is an important activity in the software development life cycle. It is the most time-consuming process in any software development process. Reliability of software depends on the various testing activities such as size of the software and test data[1]. As the size of the software grows, testing becomes a more laborious task to perform, of all testing activities test case generation includes significant bit of work since it influences the proficiency of the testing process before inception of the software[2]. When software becomes more complex, it is very difficult to test software to produce accurate test results. Many approaches have been employed and used in the state of the art for different programming languages, technologies, and environments to produce test cases that make the testing more effective and robust. The most influential techniques were classified by Anand et al into five groups such as symbolic execution and program structural coverage testing, model-based test case generation, combinational testing, adaptive random testing, and search-based testing. The most important technique in the category classified by Anand *et al.*,2015[3] is search-based research. The main objective of search-based software testing (SBST) is to explore successful test data that maximizes the software structure coverage metric.

Random testing [2] is a widely used and low-cost technique that randomly selects test inputs from valid range, its performance, however, is at lower side when the inputs are subject to complex constraints. SBST overcome the issues associated with random testing and generate quality test data even for large and complex problems. In SBST, by formulating the problem as a blended problem, some meta-heuristic search techniques have been used for test case generation [2].

Harman and Jones say that search-based software testing is an evolving field, and meta-heuristics are perfect for software engineering to be implemented by reformulating the problems of conventional software engineering [4] and a fitness function is defined to evaluate the quality of a solution in terms of the coverage metric while formulating the problem. The meta-heuristic does not make assumptions about the problem characteristics and provides rational results for problems of complexity that cannot be solved by empirical methods because of the dimensionality of the problem. Methods of search-based software test data generation that is being reviewed by some popular researches in the field of search based testing are (Harman , Mansouri, and Zhang) [5],Harman *et al.*, [6], McMinn[7]. Meta-heuristics algorithms provide an efficient way to solve the problem of test data generation and to locate the search space. Some of the widely applied meta-heuristics algorithms that are centered on various natural phenomena, are applied by the researchers in the field of test data generation such as Particle Swarm Optimization (PSO) algorithm[9] , Genetic algorithm [10], Artificial Bee Colony (ABC) [11], Ant Colony optimization (ACO)[12] , Firefly algorithm (FA) [13] are some examples of most popular meta-heuristic algorithms. It is mentioned in all these reviews that there are still many areas relevant to search-based software engineering and many interesting research challenges ahead.

2. Artificial Immune Systems (AIS)

Artificial Immune Systems are computational ideal models that have a place with the computational knowledge family and are enlivened by the natural safe framework. Over the past decade, researchers have attracted a lot of interest in designing immune based models and technique to solve complex software engineering problems.

- Negative Selection
- Clonal Selection

Are the three major AIS algorithms commonly used in the field of engineering!
One of the most important strategies in the Artificial Immune system (AIS) is the Negative Selection Algorithm (NSA) a branch of computer intelligence model. The biological action of the Natural Immune System (NIS), a compound biological network that uses fast and active techniques to protect the body against a particular foreign body called antigens, has inspired AIS [14]. AIS is one of the numerous forms of biological systems inspired algorithms, such as evolutionary algorithms, swarm intelligence and neural networks, which have attracted a lot of the attention of researchers. The goal is to develop immune- based methods to solve complicated computations [15].Forrest (1994) implemented NSA, and in fields such as computer security, pattern recognition,

anomaly detection, and faults detection have been added. The primary objective of the NSA is to differentiate between samples of self and non-self when only self-samples are available. Specifically, the objective is to cover non - self space with a specific number of detectors [16].Frank Macfarlane Burnet proposed the notion in 1957 to explain the wide variety of antibodies generated during the commencement of the immune response. Clonal selection theory is an immunological scientific hypothesis that explains how immune system cells (lymphocytes) respond to distinct antigens invading the body. It has been widely accepted in the field of various engineering applications.[16]

3. Related work on Test Data Generation

In recent years, some studies on PSO, ABC, GA, ACO and FA for test data/cases generation have been presented to literature.
Xiao-Mei Zhu & Xian-Fang Yang[17] projected a new approach based on PSO, in which inertia weight is adjusted according to fitness value. It uses branch Coverage as fitness criteria. It shows broad application prospective as compared to immune genetic algorithm and PSO. It outperforms IGA and PSO in terms of convergence speed, efficiency, and performance. Sanjay Singhal, Dharminder Kumar, H.M.Rai and Priti Singhal [18] projected a hybrid approach by combining GA and PSO(GPSCA). It uses data flow coverage by applying dominance concept between two nodes and multi-objective coverage criteria. They have conducted the comparison of the proposed approach with GA and PSO using seven benchmark programs, the proposed approach outperformed GA and PSO for coverage ratio and test data generation. Aiguo Li and Yanli Zhang[19] projected a new approach using traditional PSO with new objective function for all path coverage criteria. They have done the comparison of the projected approach with single path data for triangle classification problem, which shows the proposed approach outperformed the single path data for cost and time.

Shailesh Tiwari, K.K.Mishra and A.K Misra [20] projected new approach i.e PSO-TVAC (modified time varying acceleration) using code coverage objective criteria for five real world problems. They have conducted the comparison of the proposed approach with the exiting PSO approaches. The results show that the proposed approach has better coverage capability, control on local and global optimum. Chengying Mao , Xinxin Yu and Jifu Chen[9] projected approach based on PSO for test data generation i.e TDGen_PSO using branch coverage criteria for five real world programs. They have made comparison of proposed approach with TDGen_GA and CL-PS, the results shows that the proposed approach outperforms in terms of coverage and test data generation. Dan Liu,Xuejun Wang and Jianmin Wang[21] projected an approach IGA based on Genetic Algorithm(GA) for automatic test case generation. They have done comparison of IGA with traditional GA for triangle classification problem using branch fitness criterion. The improved algorithm adopts real number coding and principles of large coverage. The improved GA outperformed traditional GA in terms of convergence speed and higher test data generation efficiency. Moataz A. Ahmeda and Irman Hermadib[22] projected a GA based test data generator using multi path fitness. The approach can synthesize multiple test data to cover multiple target paths. They have performed the comparison of the proposed approach with Lin's & Pei's work based on GA using seven real world benchmark problems. The proposed approach is effective and efficient then the Lin's & Pei's work. Kewen Li Zilu and Zhang Jisong Kou [23] projected an approach GPSMA by using PSO inside GA. They have replaced the mutation operation in GA based on population division. The proposed approach is compared with GA and PSO for triangle classification problem. The proposed approach avoids premature generation and improved convergence speed. Gen.iana and Ioana [24] is projected a new approach based on three evolutionary approaches GA, PSO and SA. They compared proposed approach with GA, SA and PSO using ten benchmark problems.

In the proposed approach they have evaluated the distance between the actual paths. The proposed approach outperformed the GA, PSO and SA in terms of quality data generation and high convergence. In the proposed approach they have used annealing mechanism into GA along with similarity-based fitness function. They compare the proposed approach with GA and Random testing for triangle classification Problem. Soma Sekhara ,Babu Lam & M.L.Hari [25] projected an approach by combining the functionalities of scouts, employed and onlooker bees in ABC algorithm. They compared the approach with ABC, GA and ACO using benchmark triangle classification program by using independent test path coverage criteria. The proposed approach is a no pheromone-based approach which does not required to update the pheromone level which improve the time complexity and the number of tests required are also at low level. Surinder Singh Dahiya *et al.*[26] Proposed a static based symbolic execution approach using ABC algorithm with branch distance as objective function. They have compared the approach with average test cases generated path (ATCPP) and average percentage coverage (APC) metrics using ten real world benchmark programs. The said approach does not perform well for in the elevated value of ATCPP. Bharti Suri & Prabhneet kaur [27] proposed a regression augmentation testing approach based on ABC algorithm with branch distance as objective function. They have compared the functionality of the existing algorithm using eight real world benchmark programs for test suites. The proposed approach yields 100% path coverage in regression augmentation testing.

D.Jeya Mala,M.Kamalapriya [28] have proposed ABC based approach by applying heuristic in each test case with path coverage as objective function. They have compared the functionality of the proposed approach with ACO using six benchmark programs. The proposed approach generates optimal results and converges with a smaller number of test runs. Shunkun Yang, Tianlong Man, and Jiaqi Xu [29] proposed an approach based on ant colony optimization in which they have improved local pheromone strategy, pheromone volatilization co-efficient and global path pheromone with statement coverage, branch coverage and condition coverage as fitness value. They have compared the proposed approach with random algorithm and genetic algorithm suing benchmark program triangle classification. The proposed approach can effectively improve the search efficiency, restrain precocity, promote case coverage, and reduce the number of iterations. Chengying Mao, Lichuan Xiao, Xinxin Yu & Jinfu Chen [30] proposed an approach in which they reformed ACO into discrete version by redefining local transfer; global transfer and pheromone update rule with customize branch fitness function. They have compared the proposed approach with GA, SA and PSO using eight benchmark programs. The proposed approach outperforms the GA and SA and comparable to PSO. Pooja Sharma [31] has proposed an approach for automated software testing using meta heuristic technique based on improved ant algorithm in which she used statement, branch and modified decision/coverage as an objective function. The comparison of the proposed approach has been done with the existing RND, GA and different variants of ant algorithms for classic triangle classification and collision avoidance system. The proposed approach has better coverage and minimal generation as compared to RND, GA and different variants of ant algorithms. Praveen Ranjan Srivastava and KM Baby [32] proposed a meta-heuristic technique based on ACO for state transition testing. They have done experimentation with the proposed approach by selecting the enrolment statement machine and transition system state as experiments; they compared the proposed approach with GA and Software Transition testing. The proposed approach has better coverage than GA.

Faeghe Sayyari and Sima Emadi [33] has proposed an ACO and model based testing approach. They have used Markov model for the re-formation of ACO. The comparison of proposed approach has been done with ACO data flow testing and ACO Markov Chain for single telephone experiment. The Proposed approach generates quality data as compared to ACO data flow testing and ACO Markov Chain. Shayma Mustafa , MohiAldeen Radziah Mohamad and Safaai Deris [34] have proposed a new approach based on artificial immune system in which they have use the application of negative selection algorithm . They have compared the proposed approach with random testing, genetic algorithm and ant colony optimization using eleven benchmark programs. The proposed approach outperforms other methods in reducing the number of test data that covers all program paths by calculating the hamming distance. *Shayma Mustafa et al.*[35] projected a new hybrid approach based on NSA and GA for automated test data generation, the experimentation of the projected approach has been done on 11 real world programs, the projected approach is also compared with random testing approach and negative selection algorithm. The results show that the projected approach has high path coverage with minimum number of generations. *Ankit Pachauri and Gursaran* [36] has projected test data generation approach based on Clonal selection algorithm. They have used AI and NBD Approximation level with normalized branch distance as objective function to validate the test data. Sthamer triangle classifier problem has been used for experimentation with approximated experiment runs. The results show that the projected approach has poor generation and coverage ratio. *Poonam Saini and Sanjay Tyagi*[37] has also projected an approach based on Clonal Selection algorithm . They have used Korel Distance function for branch predicate as objective function to validate the test data. The projected approach is compared with random testing and genetic algorithm by considering nine real world programs for experimentation. The projected approach generates optimal test data and has an elite test data generation technique.

The main objective of this paper is to explore the search capabilities of Meta–heuristic algorithms ACO, GA, PSO, ABC, FA and Artificial Immune algorithm NSA on benchmark problems in software test data generation, including triangle classification, quadratic equation, even odd, largest number, telephone system etc. Meta-heuristic algorithms such as ACO, GA, PSO, ABC, FA has excellent search capabilities, but all these algorithms have somehow lag in complete coverage or may be sometimes struck in local optima. The Artificial Immune algorithms on the other side present significant improvement in the search capability Negative Selective algorithm, Clonal Selection and Hybrid NSA-GA are latest algorithms that explore the capability of test data generation in the significant manner. An immune algorithm has significant impact on the quality and coverage capabilities of test data generation. Besides, designing an objective function is another important subject that should be determined in the experiments [38]. Designing a good objective function allows the algorithm to monitor the optimum more accurately and quickly in the search space.
The rest of the paper is structured as follows. Software test data generation is formulated in the second section and then a section is devoted to the generation of search- based test data. A brief overview of Metaheuristic is given in the next section as pseudo codes. Survey of the Metaheuristics algorithms and AIS algorithms has been done on SBST and are recorded in section 4. Survey has been done is section 5 and Finally, the discussion and conclusion are dedicated to Section 6.

4.Search Based test data generation

Software testing is an essential task in software development to satisfy the requirements specification of the software under process. Testing takes the bigger portion of the development process. A well-designed test plan must be desired to align all the testing activities at different milestones.

In software testing main goal is to decide the issues given below [39]

- Maximum coverage should be achieved with minimal test cases.
- Design the well modeled structure of the developing system.
- Well planned testing activities to yield optimal results.

Opting for test data that yields maximum coverage can be attain either in manual or in automated way. Selection of the testing process depends upon the size of the software, to validate large size program automated testing is more preferred as compared to manual testing, as manual testing is a time consuming and laborious activity, for this reason the popularity for the automated testing has raised and being adopted by many researchers. Many new techniques have been proposed for automated test data generation, which have shown the significant impact on the quality of the product being developed. In, literature different test data generation architectures have been used [40].

In most of the studies, program code is converted into a control flow graph (CFG), which represents the graphical flow of the sequence of statements related to source code of the relevant program[41].

A CFG is defined as a directed graph

G = (N, E, s, e)
N – Set of nodes
E - No of edges
s- Starting node
e – Exit node

CFG must have a unique Entry s and Exit node e. That the program is going to start to execute and terminate the control flow graph for minmax problem is presented in fig. 1.

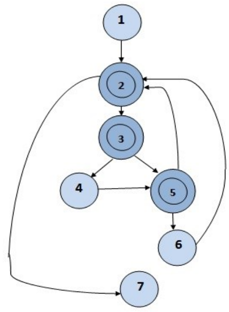

Fig. 1: Control Flow Graph for Minmax Problem

Selecting an appropriate input that passes through the different predicates (Statement, Branch and Path) in CFG can be considered as an optimization problem, which aspires to maximize the coverage of the source code. Therefore, Search based software testing techniques based on optimization algorithms and refined by a fitness function for improved quality has received attention of the researchers in past years [42]

Harman [5] has proposed a generic search based test data generation technique. The flow of the working of test data generation is represented in the form of sequence diagram in Fig 2.

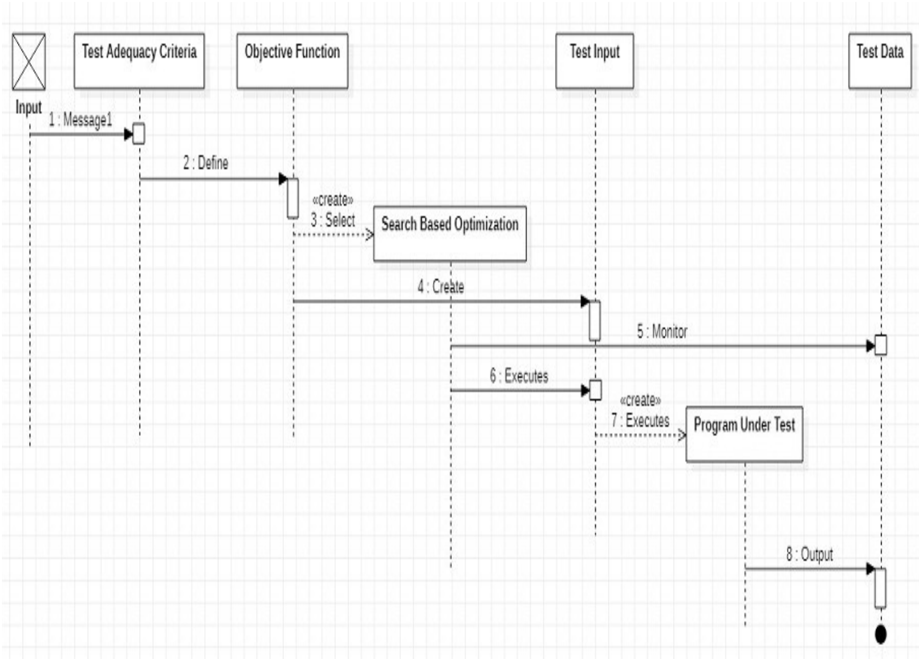

Fig 2 Sequence diagram for Test Data Generation

5. Meta–heuristic & Artificial Immune Algorithms for Test Data Generation

Researchers have applied the functionality of various meta-heuristic algorithms based on different natural phenomena in the field of test data generation. Algorithms who show significant impact in test data generation are Particle Swarm Optimization (PSO) algorithm, Ant Colony optimization, Artificial Bee Colony (ABC), Genetic algorithm (GA) and Firefly algorithm (FA).

The Immune algorithms are also gaining popularity in various engineering applications, even in the field of test data generation. The outcome of immune algorithms is having upped edge than Meta – heuristic algorithms. Negative Selection algorithm (NSA) & Clonel Selection algorithm are widely applied in the field of software testing. The Pseudo Codes of meta – heuristic and artificial algorithms are presented as follows:

5.1 PSO algorithm : introduced by Eberhart and Kennedy in 1995[9], is a swarm-based meta-heuristic that models the social behavior of bird flocking or fish schooling. The Pseudo Code of PSO is as follows:

- Begin
- Initialize Particles randomly
- Repeat until $_P$Best != target_reached
- Evaluate $_P$Best for each particle
- If current position > $_P$Best than
- Update: $_P$Best
- Else
- Assign: $_g$Best = $_P$Best
- endif
- Compute velocity
- Update particle position
- If target_reached
- End
- Else
- goto step 4

5.2 Artificial Bee Colony (ABC): algorithm developed in 2005 by Karaboga [11] mimics the for aging behavior of honey bees and has been applied to many problems encountered in different research areas . The Pseudo Code of ABC is as follows:

1. Begin
2. Initialize population randomly
3. Repeat steps 4 to 10 until convergence! = expected
4. Employed bee phase
5. Onlooker bee phase
6. Scout bee phase
7. If convergence== expected
8. End
9. Else
10. Goto step

5.3 The ant colony optimization algorithm (ACO): introduced by Marco Dorigo[12], in the year 1992 and it is a paradigm for designing Meta heuristic algorithms for optimization problems and is inspired by the foraging behavior of ant colonies. Ant Colony Optimization targets discrete optimization problems and can be extended to continuous optimization problems which is useful to find approximate solutions[12]. The Pseudo code of ACO as follows:

1. Begin
2. Initialize parameters
3. Repeat while iterations < n
4. Generate random population

5. Calculate fitness
6. Update pheromone
7. Apply transition
8. create new_path
9. If iteration=n
10. End
11. Else
12. Goto step 4

5.4 Genetic Algorithms (GA) is proposed by Professor Holland in Michigan University of the United States in 1975, which was inspired by the biological evolution. GA is based on the principle of natural selection and universal search optimization algorithm of the genetic mechanisms[10][43]. The Pseudo Code of GA is as follows:

1. Begin
2. Initialize Population
3. Repeat while Population! = quiet
4. Select Population
5. If new_population=quiet
6. End
7. Else
8. Cross overs
9. Mutations
10. Goto step 4

5.5 Firefly Algorithm: FA was developed by Xin-She Yang, inspired by the flashing behavior of fireflies. Mechanisms of firefly communication via bioluminescent flashes and their synchronization have been imitated effectively in various techniques of wireless networks design , dynamic market pricing , mobile robotics , economic dispatch problem , and structural optimization problems.[13] . The Pseudo Code of Firely is as follows:

1. Begin
2. Repeat steps 3 to 9 until iteration 1=max_generation
3. Initialize Firefly
4. Evaluate objective function
5. Generate rank of Firefly
6. Compute best_function
7. Compare best_function= firefly_movement
8. If iteration=max_generation
9. Display result
10. Else
11. Goto step 3
12. End

5.6 Negative Selection Algorithm: Negative Selection Algorithm (NSA)'is one of the most important methods in an Artificial Immune System' (AIS)[14], It was introduced by Forrest (1994) which is a branch of computational intelligence models. AIS was inspired by the biological behavior of Natural Immune System'(NIS), which is a compound biological network using fast and active techniques to defend the body versus a specified foreign body called antigens. AIS are one of the different kinds of algorithms inspired by biologic systems. Its objective is to develop immune-based techniques for solving complicated computation[14]. The Pseudo Code of NSA is as follows:

Generation_Stage

1. Begin
2. Generate Random_Detectors
3. If Detectors=Self
4. Goto step 2
5. Else
6. Accept as New_Detector
7. If Detectors=Enough

8. Goto Detection_Stage
9. Else
10. Goto step 2

Detection_Stage
1. Begin
2. Input New_Sample
3. If (New_Sample=Detector)
4. Self
5. Else
6. Non-self
7. End

5.7 **Clonal Selection Algorithm:** Clonal Selection algorithm belongs to the field of artificial immune system. The clonal selection algorithm is inspired by the clonal selection theory of acquired immunity.[16] The clonal selection theory credited to Burnet was proposed to account for the behaviour and capabilities of antibodies in the acquired immune system.[44] The theory suggests that starting with the initial repertoire of general immune cells, the system is able to change itself the CSA was designs as a general machine learning approach and has been applied to pattern recognition, functional optimization , combinational optimization and test data generation domain[36]. The Pseudo code of Clonal algorithm is as follows:

1. Begin
2. Initialize Antibodies Randomly
3. Repeat until antibodies > max_antibodies
4. Evaluation_Chamber = Antibodies
5. Compute Affinity_extent
6. If Affinity_extent > threshold
7. Print mature antibody group
8. Else
9. Arrange antobody based on affinity
10. If antibodies > max_antibodies
11. Goto step 6
12. Else
13. Print first antibody
14. Goto step 3

The main objective of this paper is to explore the search capabilities of Meta–heuristic algorithms ACO, GA, PSO, ABC, FA and Artificial Immune algorithm NSA on benchmark problems in software test data generation, including triangle classification, quadratic equation, even odd, largest number, telephone system etc. Meta-heuristic algorithms such as ACO, GA, PSO, ABC, FA has excellent search capabilities, but all these algorithms have somehow lag in complete coverage. The Artificial Immune algorithm NSA is new approach in generation of test data. An immune algorithm has significant impact on the quality and coverage capabilities of test data generation. Besides, designing an objective function is another important subject that should be determined in the experiments [4]. Designing a good objective function allows the algorithm to monitor the optimum more accurately and quickly in the search space
.

6. Software Test Data Generation: Survey

Table 1 Comparative Study of different test data generation techniques

Author	Technique Adapted	Fitness/Objective function used	No. of Experiments	Comparison with other techniques	Findings	Performance Measure Parameter's adopted
(Sekhara et	ABC Based by	Independent Test Path	Triangle Classificati	ABC, GA & ACO	Less No. of Test	Path Coverage,

al., 2012) [25]	combining scouts, employed & onlooker bees	Coverage Criteria	on		Required, Low Time Complexity for Test data Generation, Faster & efficient	Path Sequence Comparison.
(Dahiya, Chhabra and Kumar, 2010) [26]	Static Based Symbolic Execution	Branch Distance	10 Real World's Problem	Average Test cases Generated Path (ATCPP) and Average Percentage Coverage (APC) metrics	Does not Perform Well for High Value of ATCPP	Average Test Cases Generated Path (ATCPP) & Average Percentage Coverage (APC) Metrics
(Malhotra, 2014) [45]	Comparison	-----------	9 real world Programs	GA & ACO	Yields Better Results for Large and Complex Problems	No. of Paths Covered, Number of Iterations, Number of Test Cases. Time taken for number of Generation
(Suri and Kaur, 2014)[27]	Regression Augmentation testing	Branch Coverage	8	Test Suites	Yields 100 % coverage	Path Coverage
(Mala and Nadu, 2009) [28]	ABC based approach with heuristic in each test case.	Path Coverage	6	ACO	Generate Optimal Results and Converges with a smaller Number of Test Runs.	Time Complexity, Path Coverage
(Zhu, 2010)[17]	Improved PSO	Branch Coverage Fitness	Triangle Classification	Immune Genetic Algorithm and PSO	Outperform IGA & PSO in terms of Convergence Speed, Efficiency & Performance	Average Iteration Time, Convergence Rate.
(Singla et al., 2011) [18]	Hybrid GA & PSO(GPSCA)	Multi - Objective	7	GA & PSO	High Coverage Ratio, Less Generation.	Coverage Ratio No. of Test Cases. No. of Generation
(Li and	Traditional PSO	New All Path Objective	Triangle Classificati	Single Path Data.	Cost of Time is	Cost and time of test

Zhang, 2009) [19]	With new objective function	Function	on & Binary Search		Half as compared to Single Path	data generation
(Tiwari, Mishra and Misra, 2013)[20]	PSO-TVAC (modified Time Varying Acceleration.	Code Coverage.	5 Benchmark Programs	PSO Variants	Better Performance and Great Code Coverage Capability, Control on Local and Global Optimum	Code Coverage and test case Generation.
(Mao, Yu and Chen, 2012) [9]	PSO Based Test Data Generation (TDGen_PSO)	Branch Coverage	5 Real World Programs	TDGen_GA, CL-PS	Outperforms TDGen_GA and CA-PSO in terms of Coverage & Generations	Average Coverage, Successful Rate, Average Generations and Average Time
(Liu, Wang and Wang, 2013)[21]	Modified Genetic Algorithm	Branch Fitness	Triangulation network	Traditional GA	Avoid Premature Convergence. Fast Convergency, High test data generation efficiency	Total No. of Coverage Time Coverage Rate
(Ahmed and Hermadi, 2008) [22]	GA Based Test data Generator	Multi Path Fitness	7	Lin;s & Pei's work based on GA	Synthesize multiple test data, More Effective & Efficient than similar tools	Average Generation Average Coverage
(Gupta and Applications, 2014)[46]	GA	Branch	11	Random Testing	Find more error prone paths, reduce Development Cost & Improve Efficiency	Paths Identification, Cost & efficiency
(Li, Zhang and Kou, 2010)[23]	PSO used inside GA	Individual Sa test Case	Triangle Classification	GA and ACO	Maintain Colony Polymorphism, Avoid Premature Convergence. Improve Convergence Speed	Test Data Generation, Test Data Convergence, Colony Maintenance
(Zhang and	Anneal Mechanism	Similarity Based	Triangle Classificati	GA and Random	Preserve the best	Selection & Elitist

Wang, 2011)[47]	into GA	Fitness Function (Hamming Distance)	on	testing	probability, Effective & Efficient than other techniques	Crossovers Mutation Simulated Annealing and Convergenc e
(Latiu, Cret and Vacari u, 2012)[24]	Based on Three Evolutionar y Approaches GA, SA & PSO	Approximati on Level and Branch Distance (Evaluating the Distance between actual path)	10	GA, SA & PSO	SA generated quality Data	Convergenc e
(Mao et al., 2015)[30]	Local transfer, global transfer, pheromone update has been re defined.	Customize Branch fitness function	8 Benchmark s programs has been used	Genetic Algorithm, SA and PSO	Outperform Genetic Algorithm and Simulated Annealing, Comparabl e to PSO	Average Coverage, Successful rate, Average Convergenc e, Average time
(Faeg he sayya ri and sima emadi , 2015)	Markov model	NA	Single Telephone Experiment	ACO Data Flow testing & ACO Markov Chain	Generate quality	Pheromone -factor, Cost, and user Parameters
(Mao et al., 2012)[48]	Local Transfer, Global transfer, pheromone update has been re - defined.	Branch fitness function	5 Benchmark s programs has been used	GA, SA, ACO	Outperform s	Average Coverage, Successful rate, Average Convergenc e, Average time
(Yang, Man and Xu, 2014)[29]	Improved Local Pheromone strategy and pheromone volatilizatio n Co-efficient and Global Path Pheromone	Statement Coverage, Branch Coverage and Modified Condition/D ecision Coverage.	Triangle classificatio n and collision detection.	Random Algorithm and Genetic Algorithm	Improved coverage and generation.	Average Coverage and Average Generation
(Sriva stava, 2010)[32]	State Transition based testing and its coverage level	NIL	The Enrolment state machine, transition system state machine	GA & STT (Software Transition testing)	Better coverage than GA	Complete Coverage, Generation of Optimal Test Sequence, Enhanceme nt of the

						Tool.
(Sharma, 2014) [31]	Ant Colony Optimizatio n	Statement, Branch & modified decision/cov erage	Classic triangle Classificati on, & collision avoidance system	RND, GA, SACO & ACO	Average Coverage, Average minimal generation	Generation of Optimal & Minimal Test Sequence for Complete Coverage.
(Mohi-Aldeen, Mohamad and Deris, 2017)[49]	NSA	Hamming Distance	Benchmark Program Triangle Classifier	Random testing & GA	NSA is efficient in time of execution & effective in generation of test data.	Test Data Generation, Execution Time
(Mohi-Aldeen, Mohamad and Deris, 2016)[34]	Application of NSA	Hamming Distance	11 Real world Benchmark programs	Random Testing, GA & ACO	Outperform s other methods in reducing the number of test data that covers all program paths.	Path Coverage, Effectivene ss & Efficiency.
(Mohi-aldeen, Mohamad and Deris, no date)[50]	Negative Selection Algorithm	Hamming Distance	Benchmark Program Triangle Classifier	Random Testing	Outperform s Random Testing for Path Coverage.	Automated Test Case Generation, Effectivene ss & Efficiency.
(Pachauri, 2012)[36]	Clonal Selection Algorithm (Immune Algorithm)	Al & NBD, Approximati on Level with normalized Branch Distance	Sthamer Traingle Classifier Problem	Hundred Experiments (run)	Poor Generation & Coverage	Mean Number of Generation, Mean Percentage Coverage.
(Saini and Tyagi, 2014)[37]	GA and CSA	Korel Distance Function for Branch Predicate	9 Benchmark Program	Random, GA & CSA	Elite Test Data Generation Technique, Generate Optimal Test Data	Performanc e of Test Data Generation
(Id, Mohamad and Deris, 2020) [35]	Hybrid NSA & GA	Hamming Distance	11 Benchmark Program	Random, NSA, NSA-GA	High Path Coverage with minimum number of Generation.	Average Coverage, Average test Data, Average Generation.

Figures 3(a), 3 (b), 3 (c) , 3(d) presents the graphical survey of Automated test data generation.

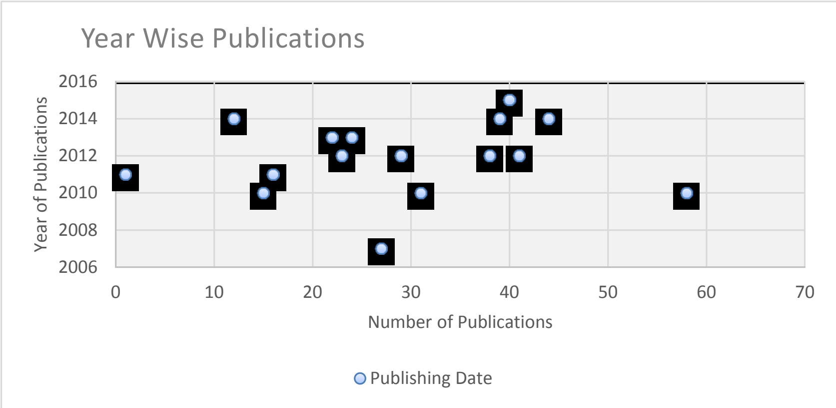

Fig. 3: (a) Publication frequency of test data generation articles

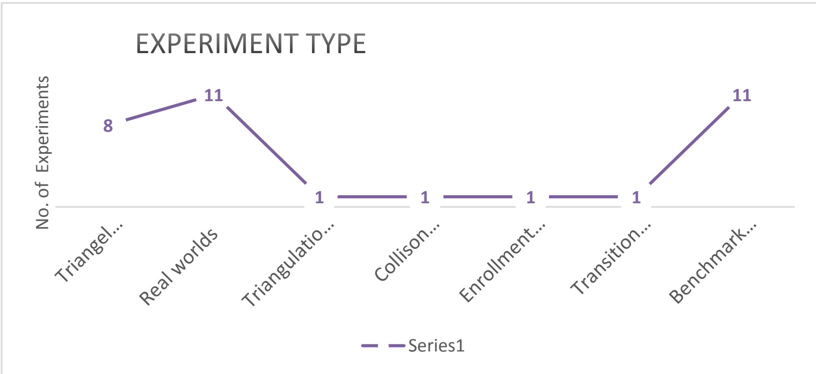

Fig. 3: (b) Type of Experiments used in Test data generation

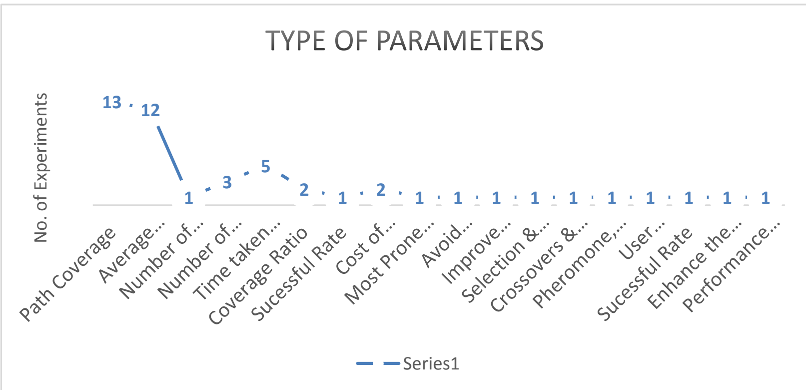

Fig. 3: (c) Type of Parameters used in Test data generation

Figure 3 (d) No. of comparison conducted by various approaches

7. Discussion & Conclusion

In this paper the survey on Metaheuristics and Artificial Immune algorithms have been done on generation of software test data, preferably for structural testing because structural testing in general treated as the favored method for detecting errors and bugs in software code, however, to produce test data with much greater code coverage capability is still an open question. Particle Swarm Optimization, Artificial bee Colony, Genetic Algorithm, Ant Colony Optimization algorithm and Firefly algorithm has been selected for relative study due to their enormous applicability in the field of automated software test data generation and in different engineering application. Negative Selection Algorithm (NSA) and Clonal Selection algorithm has been also selected from the class of artificial immune system, just because of his superiority on Meta–Heuristics algorithms. From the survey it has been observed that most of the work carried out in automated software test data generation has been done for metrics like average coverage, success rate, average generation, and average time. Objective function also plays a crucial role to validate the test data. Different objective functions such as statement coverage, single path coverage, branch coverage and multipath coverage has been projected in this study, out of all branch coverage is widely preferred objective function which also represents significant enhancement in the quality of test data.

In NSA hamming distance is used in place of objective function to validate the test data. Some benchmark programs such as triangle classification, even odd, largest number, leap year, quadratic equation and telephone system has been widely applied for experimentation to validate the efficiency and effectiveness of the technique for generating test data. We notice that adapted ACO methodology is stronger than ACO, IACO, ABC and GA in coverage capability, convergence speed and consistency, and is comparable to the PSO-based method in few experimental setups, according to the survey the experimental results generated in various research papers showed that NSA is more efficient and more competitive than meta-heuristic algorithms. It has better locating capability with lesser number of generations. The findings shows that NSA has the potential to minimize the amount of test data. In this paper we have explored some promising method for generating test results.

References

[1] M. A. Jamil, M. Arif, N. Sham, A. Abubakar, and A. Ahmad, "Software Testing Techniques : A Literature Review," 2016, doi: 10.1109/ICT4M.2016.40.

[2] N. Anwar and S. Kar, "Review Paper on Various Software Testing Techniques & Strategies," vol. 19, no. 2, 2019.

[3] S. Anand *et al.*, "The Journal of Systems and Software An orchestrated survey of methodologies for automated software test case generation Orchestrators and Editors ," vol. 86, no. 2013, pp. 1978–2001, 2015, doi: 10.1016/j.jss.2013.02.061.

[4] M. Harman and B. F. Jones, "Search-based software engineering," vol. 43, pp. 833–839, 2001.

[5] M. Harman, S. A. Mansouri, and Y. Zhang, "Search Based Software Engineering : A Comprehensive Analysis and Review of Trends Techniques and Applications," pp. 1–78, 2009.

[6] M. Harman, P. Mcminn, J. T. De Souza, and S. Yoo, "Search Based Software Engineering : Techniques , Taxonomy , Tutorial."

[7] M. Harman and P. Mcminn, "A Multi – Objective Approach To Search – Based Test Data Generation."

[8] P. Mcminn, M. Harman, G. Fraser, and G. M. Kapfhammer, "Automated Search for Good Coverage Criteria : Moving from Code Coverage to Fault Coverage Through Search-Based Software Engineering," pp. 17–18, 2016.

[9] C. Mao, B. X. Yu, and J. Chen, "Swarm intelligence-based test data generation for structural testing," *Proc. - 2012 IEEE/ACIS 11th Int. Conf. Comput. Inf. Sci. ICIS 2012*, no. May 2012, pp. 623–628, 2012, doi: 10.1109/ICIS.2012.103.

[10] M. S. Al-zabidi, "Study of genetic algorithm for automaatic software test data generation," vol. 1, no. 2, pp. 65–74, 2013.

[11] D. Karaboga, B. Gorkemli, and N. Karaboga, "A comprehensive survey : Artificial bee colony (ABC) algorithm and applications A comprehensive survey : artificial bee colony (ABC)," no. June, 2012, doi: 10.1007/s10462-012-9328-0.

[12] M. Dorigo and T. Stützle, *Optimization*. .

[13] P. Ranjan, B. Mallikarjun, and X. Yang, "Optimal Test Sequence Generation Using Firefly Algorithm ", Swarm and," *Swarm Evol. Comput.*, vol. 8, no. 1, pp. 44–53, 2014, doi: 10.1016/j.swevo.2012.08.003.

[14] T. Intelligent, C. Systems, and A. Science, "Conceptual Frameworks for," vol. 1, pp. 315–338, 2005.

[15] J. Timmis, A. Hone, T. Stibor, and E. Clark, "Theoretical advances in artificial immune systems," *Theor. Comput. Sci.*, vol. 403, no. 1, pp. 11–32, 2008, doi: 10.1016/j.tcs.2008.02.011.

[16] D. Dasgupta, "Advances in artificial immune systems," *IEEE Comput. Intell. Mag.*, vol. 1, no. 4, pp. 40–43, 2006, doi: 10.1109/CI-M.2006.248056.

[17] X. Zhu, "Software Test Data Generation Automatically Based on Improved Adaptive Particle Swarm Optimizer," pp. 1300–1303, 2010, doi: 10.1109/ICCIS.2010.321.

[18] S. Singla, D. Kumar, H. M. Rai, and P. Singla, "A Hybrid PSO Approach to Automate Test Data Generation for Data Flow Coverage with Dominance Concepts," vol. 37, pp. 15–26, 2011.

[19] A. Li and Y. Zhang, "Automatic generating all-path test data of a program based on PSO," *2009 WRI World Congr. Softw. Eng. WCSE 2009*, vol. 4, pp. 189–193, 2009, doi: 10.1109/WCSE.2009.98.

[20] S. Tiwari, K. K. Mishra, and A. K. Misra, "Test Case Generation for Modified Code using a Variant of Particle Swarm Optimization (PSO)Algorithm," 2013, doi: 10.1109/ITNG.2013.58.

[21] D. A. N. Liu, X. Wang, and J. Wang, "Automatic test case generation based on genetic algorithm" vol. 48, no. 1, pp. 411–416, 2013.

[22] M. A. Ahmed and I. Hermadi, "GA-based multiple paths test data generator," vol. 35, pp. 3107–3124, 2008, doi: 10.1016/j.cor.2007.01.012.

[23] K. Li, Z. Zhang, and J. Kou, "Breeding software test data with Genetic-Particle Swarm Mixed Algorithm," *J. Comput.*, vol. 5, no. 2, pp. 258–265, 2010, doi: 10.4304/jcp.5.2.258-265.

[24] G. I. Latiu, O. A. Cret, and L. Vacariu, "Automatic Test Data Generation for Software Path Testing Using Evolutionary Algorithms," *2012 Third Int. Conf. Emerg. Intell. Data Web Technol.*, pp. 1–8, 2012, doi: 10.1109/EIDWT.2012.25.

[25] S. Sekhara, B. Lam, M. L. H. Prasad, U. K. M, and S. Ch, "Procedia Engineering Automated

Generation of Independent Paths and Test Suite Optimization Using Artificial Bee Colony," vol. 00, no. 2011, 2012, doi: 10.1016/j.proeng.2012.01.851.
[26] S. S. Dahiya, J. K. Chhabra, and S. Kumar, "Application of Artificial Bee Colony Algorithm to Software Testing," *Softw. Eng. Conf. (ASWEC), 2010 21st Aust.*, pp. 149–154, 2010, doi: 10.1109/ASWEC.2010.30.
[27] B. Suri and P. Kaur, "I nternational journal of research in applied science and engineering technology (ijraset) Path Based Test Suite Augmentation using Artificial Bee Colony Algorithm identifying the element," vol. 2, no. Ix, pp. 156–164, 2014.
[28] D. J. Mala and T. Nadu, "A Non-Pheromone based Intelligent Swarm Optimization Technique in Software Test Suite Optimization," pp. 1–5, 2009.
[29] S. Yang, T. Man, and J. Xu, "Improved ant algorithms for software testing cases generation," *Sci. World J.*, vol. 2014, 2014, doi: 10.1155/2014/392309.
[30] C. Mao, L. Xiao, X. Yu, and J. Chen, "Adapting ant colony optimization to generate test data for software structural testing $," vol. 20, pp. 2014–2016, 2015.
[31] P. Sharma, "Automated Software Testing Using Metahurestic Technique Based on Improved Ant Algorithms for Software Testing," pp. 3505–3510.
[32] P. R. Srivastava, "Automated Software Testing Using Metahurestic Technique Based on An Ant Colony Optimization," 2010.
[33] F. Sayyari and S.Emadi, "Automated generation of software testing path based on ant colony," no. Ictck, pp. 11–12, 2015.
[34] S. M. Mohi-Aldeen, R. Mohamad, and S. Deris, "Application of Negative Selection Algorithm (NSA) for test data generation of path testing," *Appl. Soft Comput. J.*, vol. 49, pp. 1118–1128, 2016, doi: 10.1016/j.asoc.2016.09.044.
[35] S. M. M. Id, R. Mohamad, and S. Deris, "Optimal path test data generation based on hybrid negative selection algorithm and genetic algorithm," pp. 1–21, 2020, doi: 10.1371/journal.pone.0242812.
[36] A. Pachauri, "Use of Clonal Selection Algorithm as Software Test Data Generation Technique," vol. 2, no. 2, pp. 1–5, 2012.
[37] P. Saini and S. Tyagi, "Test Data Generation for Basis Path Testing Using Genetic Algorithm and Clonal Selection Algorithm," vol. 3, no. 6, pp. 2012–2015, 2014.
[38] O. Sahin and B. Akay, "Comparisons of metaheuristic algorithms and fitness functions on software test data generation," *Appl. Soft Comput.*, vol. 49, pp. 1202–1214, 2016, doi: 10.1016/j.asoc.2016.09.045.
[39] A. Bertolino, A. Bertolino, and I. A. Faedo, "Software Testing Research : Achievements , Challenges , Dreams Software Testing Research : Achievements , Challenges , Dreams," no. September, 2007.
[40] H. Tahbildar and B. Kalita, "Automated software test data generation : direction of research," vol. 2, no. 1, 2011.
[41] V. Elodie, "White Box Coverage and Control Flow Graphs," pp. 1–33, 2011.
[42] P. Mcminn, R. Court, and P. Street, "Search-based Software Test Data Generation : A Survey," pp. 1–58, 2004.
[43] J. Lin and P. Yeh, "Automatic test data generation for path testing using GAs," vol. 131, pp. 47–64,
[44] P. Agarwal, "Nature-Inspired Algorithms : State-of-Art , Problems and Prospects," vol. 100, no. 14, pp. 14–21, 2014.
[45] R. Malhotra, "Comparison of Search based Techniques for Automated Test Data Generation," vol. 95, no. 23, pp. 4–8, 2014.
[46] M. Gupta s, "Effective Test Data Generation using Genetic Algorithms," no. November 2012, 2014.
[47] B. Zhang and C. Wang, "Automatic Generation of Test Data for Path Testing by Adaptive Genetic Simulated Annealing Algorithm," *IEEE Int. Conf. Comput. Sci. Autom. Eng.*, pp. 38–42, 2011, doi: 10.1109/CSAE.2011.5952418.
[48] C. Mao, X. Yu, J. Chen, and J. Chen, "Generating Test Data for Structural Testing Based on Ant Colony Optimization," *2012 12th Int. Conf. Qual. Softw.*, no. May, pp. 98–101, 2012, doi: 10.1109/QSIC.2012.12.
[49] S. Mustafa, R. Mohamad, "Automated path testing using the negative selection algorithm," no. April, 2017, doi: 10.1504/IJCVR.2017.10001815.
[50] S. M. Mohi-aldeen, R. Mohamad, and S. Deris, "Automatic Test Case Generation for Structural Testing Using Negative Selection Algorithm."