Journal of Theoretical and Applied Information Technology
31st December 2017. Vol.95. No 24

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

NATURE INSPIRED SOFT COMPUTING BASED SOFTWARE TESTING TECHNIQUES FOR REUSABLE SOFTWARE COMPONENTS

[1]PREETI GULIA, [2]PALAK

[1]Assistant Professor, Department of Computer Science and Applications, MDU, Haryana, India

[2]Research Scholar, Department of Computer Science and Applications, MDU, Haryana, India

E-mail: [1]preeti.gulia81@gmail.com, [2]palak.aug6@gmail.com

ABSTRACT

Software is the inseparable part of today's human life. Each and every gadget that we use is dependent on some or other kind of software. Component based software engineering (CBSE) has provided an effective software development paradigm which allows selection of domain specific components from component repository and assemble them into a modular and scalable application. The reliability of software and its components depends on the amount of effective testing carried on it during its development life cycle. We cannot deny the fact that exhaustive testing is not possible. Selection of appropriate test suite is a combinatorial problem. Soft computing provides a promising solution for the same. Emergence of artificial intelligence over years has added fuel to nature inspired testing techniques. This paper is a comparative study of various soft computing approaches inspired by nature for reusable software components such as artificial neural network, genetic algorithm, fuzzy logic and other swarm based techniques. A comparative analysis is presented to discuss pros and cons of different soft computing techniques for software testing of reusable components along with their preferences in recent years. A future dimension is also proposed to develop hybrid techniques for optimization of testing techniques.

Keywords: *Soft computing, Test Case Prioritization, Testing, Reusable Components.*

1. INTRODUCTION

Optimized techniques and smart algorithm designs have always attracted potential programmers. Since its inception in 1990s, the vast scope of soft computing based algorithms has attracted various researchers and practitioners from various fields over years. Soft Computing (SC) is a multi- disciplinary system of bio inspired approaches such as Fuzzy Logic (FL), Neural Network (NN), Genetic Algorithm (GA), Swarm Intelligence (SI) etc. It provides a large range of techniques which are inspired by natural phenomenon and are capable to deal with imprecision and uncertainty [1].Unlike hard computing which uses two valued logic, soft computing uses multivalued logic. Although hard computing provides deterministic crisp results but it requires exact input data and algorithm in advance. Soft computing provides approximate results with noisy or incomplete data which get improved iteration after iteration. Within the last decades, substantial amount of growth has been noticed in the application of soft computing techniques in various fields of industries such as communication, manufacturing automation, robotics, power systems, process engineering etc. [2]. Besides offering simplicity, Soft computing techniques are self-adaptive and able to handle non-linearity. It is suitable for conflicting multi objective problems where one parameter is optimized at the cost of other. In contrast to hard computing, soft computing provides inexact solutions which can be used to solve uncertain, vague and NP complete problems. Three most important benefits of such techniques are: rich knowledge representation, scalable and fast knowledge acquisition and flexible knowledge processing which reduces the overall cost of developing intelligent systems [2].

Software testing is the most challenging and crucial phase of software development life cycle. It decides the overall quality and user satisfaction regarding the final software product. Choosing an optimal test suite is a combinatorial problem which makes exhaustive testing impossible. The

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

faults in any software are un-deterministic. It requires a large pool of input values to be tested and many rounds of testing for finding faults before the software is released. This makes the process of testing more costly and time consuming. The development paradigm has shifted from object oriented to component oriented [3]. Domain specific components are selected from large pool of third party components and assembled using glue code to make working software. This scenario increases the challenge of testers to ensure fault free product. Thus soft computing provides a promising solution to assist the selection of optimal test suite. This paper presents various SC techniques that are utilized in the field of software testing by various practitioners over years. Modifications in the existing SC techniques are making them more efficient day by day that attracts many researchers in this field. A future dimension is also proposed to develop hybrid techniques for optimization of testing techniques.

Rest of the paper in organized as follows: Section II summarizes components of soft computing techniques along with their application area; Section III presents related work carried out over years by various researchers in chronological order and tabular form. Various CBSE testing challenges are given in section IV. A comparative analysis of each technique and their preferences by current researchers is given in Section V.

2. COMPONENTS OF SOFT COMPUTING TECHNIQUES

Soft computing itself is a separate branch of computing techniques which provides numerous bio inspired meta-heuristic techniques, but to make the study easy we tried to classify the various components of soft computing techniques. These are genetic algorithm, artificial neural network, fuzzy logic, swarm intelligence, probability based approach etc. Two or more such approaches can be cascaded to get better results and we name it hybrid approaches.

Soft computing has spread its roots to each and every field of science, engineering and mathematics. Soft computing techniques can be utilized to study, model and analyze very complex phenomena which are not solved by conventional methods completely. We chose soft computing techniques to generate and prioritize effective test cases which provide meta-heuristic techniques to choose a population of promising test cases. Some of the area in which soft computing can be applied for software testing are: optimal selection of test suite, automated test case generation, test case prioritization, increasing code coverage etc. Below is a brief introduction to all the SC components.

2.1. Fuzzy Logic

Fuzzy Logic (FL) is a powerful tool to deal with vagueness which provides many valued logic between 0 and 1. Although the concept came into existence a long back in 1965, but the range of applications [4] covered by it make it suitable for optimizing modern real life problems. Fuzzy logic works on linguistics variables and the results are represented as IF-THEN rules and help in decision making. The linguistics variables are the variables which take values in common language words such as *young, middle aged, old*. Fuzzy logic works in three important steps. First step is fuzzification which involves choosing membership function and converting the problem into fuzzy sets. Next step is to generate rules and evaluating them to choose the best rule. The third step involves de-fuzzification to convert the problem back into crisp set. The beauty of FL is that it is very close to human language and really easy to interpret.

2.2. Genetic Algorithm (GA)

It is a soft computing technique inspired by evolution of organism in their natural environment. It is inspired from Darwin's theory of natural selection and survival of the fittest. It is best suited for solving various search problems which demands searching a set of promising solutions out of given pool of candidate solutions. The problem parameters are represented as a binary string of 0's and 1's. The iterative process is applied to these candidate strings to obtain optimized set of solutions. The basic operations of genetic algorithm are Mutation, Cross over, Reproduction, Evaluation and Selection. GA can be effectively used to prioritize regression test suite to produce improved population of test cases in black box testing as researched by [5]. It is also widely used for automated path testing and improving coverage criteria in white box testing [6].

2.3. Swarm Intelligence

The swarm of organisms living in their natural habitat shows peculiar properties and intelligently handles its survival. Inspired from

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

coordination between the members of society of such organisms, swarm intelligence techniques are becoming popular day by day. They are capable to handle non linearity and are mostly adaptive [7] [8]. Ant Colony Optimization (ACO) is a widely used technique to search promising test cases as shown by [9-14] [18]. Swarm Techniques use self-adaptive parameters that improve the results in next iteration taking input from previous step. They are mostly suitable for searching optimal test suite and automation of testing process. Artificial bee colony (ABC) is another such technique based on the synergic social behavior of bee colony. Firefly Algorithm, Cuckoo search and Particle Swarm Optimization (PSO) also belong to this category of nature inspired techniques.

2.4. Artificial Neural Network (ANN)

ANN is imitation of human nervous system for solving problems. The inspiration behind ANN is human brain. It is a highly connected network of processing elements in which output of one unit becomes input of next unit and an activation function is applied at each layer. These systems are self-learning and trained. The basic components of ANN are Input unit, one or more hidden units and output unit. The most important aspect of ANN is its learning (training). The available domain specific knowledge is collected and a part of it is used to train the network and weights are adjusted according to it. Once the NN is trained and tested it is ready to solve problems. A multilayered perceptron is used to select and prioritize test data using various classifications and clustering techniques [15].

2.5. Hybrid Approach

There is no limitation to human thinking. We always strive to move one step forward in optimization problems by hybridizing above soft computing techniques to form a hybrid model [19] [20]. Neural networks are often linked with fuzzy logic to make a neuro-fuzzy model [15] enclosing the benefits of the two. Similar kind of hybrid models can be developed and applied one after the other to improve results.

3. SURVEY OF EXISTING TECHNIQUES

Soft computing techniques have attracted many researchers over years due to its potential to solve uncertain problems. Soft computing when used for optimizing testing techniques offers vast research domain for the researchers such as test data generation, test case selection, test automation etc. Moreover, the field of component testing in component based software is also attractive to work upon. We tried to extract some of important research from various sources over past few years that utilize the potential of soft computing techniques in software testing. The following section presents some latest ongoing research in the field of CBSE testing in chronological order.

Authors in [9] (2017) proposed new pheromone update strategy in ACO that is inspired from correlation between genes popularly known as epistasis theory. They applied the modified algorithm for prioritizing regression test cases. Authors and researchers in [10] also used swarm intelligence for test case optimization. They used chaotic behavior of firefly to improve branch coverage as compared to other swarm based and SC techniques. Some researchers as in [11] utilized GA and PSO for automated test data generation and PSO for test data prioritization.

Ahlam Ansari et. al (2016) in [12] referred to the problem of test case prioritization for optimizing regression testing. They utilized the well-known ant colony optimization approach to select optimal set of test cases on the basis of certain criteria. They also compared their proposed technique with manual testing and found that ACO is capable of reducing the overall testing cost and effort. ACO can be used to optimize test sequence generation. Sumesh Agarwal et. Al (2016) proposes a method of optimal test-sequence generation using ACO on control flow graph (CFG) [13]. They claimed to reduce coverage redundancy when compared with other existing techniques. Their results shows that after using a modified ACO approach, better test sequence can be generated.

Authors in [14] presented a comprehensive study for test case generation paradigms. They classified the test case generation techniques in four categories: Code based, Model based, Search based and Requirement based. They also proposed a new testing technique by combining search based and model based techniques by using UML and EACO (Enhanced Ant Colony Optimization). Hybrid techniques are equally important. Gaurav Kumar et. Al. (2015) [15] proposed a neuro-fuzzy model to estimate & optimize quality of components. Their approach is divided into two phases. First phase analyzes

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

and evaluates reusability for Component Based Software Engineering with the help of a series of design patterns and self-organizing map neural network technique. They used CK metrics for evaluation. Second phase empirically categorizes reusability on the basis of range of reusability calculator and Fuzzy-Back Propagation Neural Network technique. Baseline of their paper is CK metrics. Their model is tested on MATLAB and the results show that their model is an effective tool which can be further exploited for effective selection of components and removing low quality components from the system. Besides ACO and other swarm based techniques, Fuzzy logic is also exploited by many researchers for optimizing the various steps in software testing as shown by the authors in [16].

Samaila Musa et. al. (2015) [17] proposed test case prioritization technique for regression testing using modified genetic algorithm for object oriented software. They used dependence graph to select only those test cases which reveal software modifications. They evaluated their proposed algorithm on Average Percentage of rate of Fault Detection (APFD) metric. Authors in [18] (2014) proposed improved ant algorithm for test case generation by improving pheromone update strategy and volatilization coefficient. Ahmed S. Ghiduk (2014) [5] uses variable length chromosome to generate automated basis test paths that are independent of each other. He redefined the basic GA terms and proposed a basis test paths generation tool. The length of each chromosome is dependent on the length of the path. He also checked the feasibility of each test path. GA can be used for minimizing test suite for software product lines as shown by authors in [6]. They applied random weight based GA by defining appropriate fitness function and achieved better performance.

T. P. Jacob et al. (2013) [21] employed GA for test case prioritization for selecting test cases for regression testing. Researchers in [22] (2013) presented a survey on applications of GA in different types of software testing. It was found that by using GA, the results and the performance of testing can be improved. They tabulated various GA parameters used in different types of software testing such as cross over, mutation etc. Soma Sekhara Babu Lam et. al. (2012) [23] applied modified artificial bee colony (ABC) for test suite optimization and generating independent paths. They also compared their technique with other SC based techniques. Ali M. Alakeel (2012) in [24] utilized fuzzy logic for test case prioritization for the programs with assertions. He proposed a two-step process to measure the effectiveness of test suite using fuzzy logic. S. Raju et. al. (2012) employed GA for the same by keeping prioritization factors in mind such as customer assigned priority, complexity, dynamic requirements etc. [25]. Sebastian Bauersfeld et. al. (2011) in [26] used ACO for generating input sequence for GUI testing. Sangeeta Sabharwal et. al. (2011) applied GA for test case prioritization derived from UML (Unified Modeling Language) diagrams [27]. F. Saglietti et. al. (2010) focused on automation of unit and integration testing for CBS [28]. They utilized operators of Genetic Algorithm and state based interaction techniques to improve testing efficiency. Praveen Ranjan et. al (2010) [29] optimized test sequence using ACO by applying it on state transition diagrams to improve overall coverage. They compared their results with GA based testing techniques. Qian Zhongsheng (2010) worked on optimizing test case generation for web based applications by collecting user session and logs and further improved them by using GA [30].

The study of the literature shows that various algorithms exist to select and prioritize test cases and automate test case generation. Some of them are not fully optimized and can be explored further. Some are computationally complex and expensive. The above section shows that ACO and GA are the most commonly used techniques for optimizing testing process by reducing effort and increasing efficiency. A combination of soft computing techniques is also used for testing but for CBSE based applications they have some limitations. Complexity of CBSE based software is higher than conventional software so traditional testing techniques are not sufficient. Only a hybrid testing approach can provide a promising solution.

Software testing has always been an attractive research topic from many decades. The importance of testing can be felt be looking at the vast literature available over years and still the research is going on. We extracted the recent research in the field of software testing that is inspired from soft computing techniques. Each researcher utilized various available techniques at various phases of software testing. Table 1 shows few relevant publications in this field over last seven years.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

4. CBSE TESTING CHALLENGES

Although the field of CBSE is heavily researched over last two decades but still there are some challenges which the practitioners has to face. Some of them are listed below:

4.1. Heterogeneity of Components

Components are heterogeneous in terms of programming language, platform, data structure, naming conventions etc. Testing of such components may lead the testers into a dilemma during designing of test cases as each language and platform has its own notation and specification.

4.2. Need of Dummy Test Modules

Testing is not an isolated task. It requires dummy modules called stub and drivers to simulate the behavior of dependent modules. Building and configuring separate stub and drivers for each component is a cumbersome task for testers.

4.3. Test Suite Prioritization

The problem of test selection and prioritization is the most challenging problem faced by potential testers. It is considered as NP hard problem.

4.4. Test Automation

Testing is a time consuming task which requires test case generation, execution and many iterations of the same. Most of the testing activities can be automated. Test automation is a potential field of research that is attracting many researchers.

4.5. Measuring Test Coverage

Measuring test coverage is an important aspect to test the efficiency of testing. The Higher the coverage more the efficiency is. Maintaining higher test coverage is always expected from a good tester. Thus test coverage measurement is always given priority and itself a challenging task.

4.6. Complex Interface specification

Each component has some interfaces through which it interacts with other components. Interface specifications are the entry points for defects. They need to be tested thoroughly. But the problem arises when these specifications are complex.

4.7. Continuous Evolution and Versioning

CBSE based systems are highly dynamic and evolve continuously to meet the changing requirements. A component may exist in various versions with slight modifications. Testing such dynamic and evolving systems is itself a problem. Regression testing after every modification becomes an important task.

5. COMPARISON OF VARIOUS TECHNIQUES

So far we have discussed various nature inspired techniques. Each technique has its own pros and cons. Their performance varies from scenario to scenario. It is the responsibility of the practitioner to choose the best suitable technique on the basis of given requirements. ANN needs extensive training but once trained they provide fast results. On the other hand fuzzy systems require proper choice of membership function. Each technique has some limitations and advantages. Table 2 summarizes pros and cons of each technique.

Figure 1 shows the comparative use and preference to each SC technique by different researchers against the number of publications mentioned in this paper over last few years.

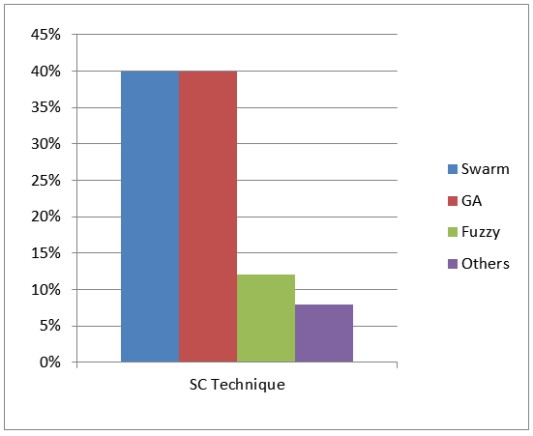

Figure 1: Comparative Use of Various SC Techniques over Past few years

It is clear from intensive literature survey that swarm based techniques are attracting researchers for current ongoing research in the field of software testing. Genetic algorithm is equally preferred for optimizing testing process. About 80% research in software testing is based on swarm based techniques and genetic

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

algorithm. Other techniques like fuzzy, neural etc. are also important due to the fact that hybrid approaches are being developed.

7. RESEARCH CONTRIBUTION

This paper presents a comprehensive survey to show the current trend of nature inspired techniques in the field of software testing and that too in the recently flourished and widely adopted component based development paradigm. Existing research in the field focuses on test case generation, test case selection, prioritization and test automation. It is interesting to note that swarm based techniques and genetic algorithm are preferred over other nature inspired techniques. We found out that the hybrid techniques are more attractive and efficient as compared to sole technique.

8. CONCLUSION

Nature provides inspiration in almost every field of computing and optimization. In this paper, we discussed various soft computing techniques inspired from nature which can be exploited for optimization of software testing process. A summary of all the basic soft computing techniques is given along with a new dimension of fusion of two or more techniques to make a hybrid approach. The flexibility provided by these techniques are tolerant to imprecision and uncertainty. Through our extensive literature survey that is published in recent years we found that 80% software testing optimization techniques are based on swarm based techniques and genetic algorithm. The full potential is yet to be exploited which inspires researchers to make hybrid techniques.

REFERENCES

[1] D. Ibrahim, An Overview of Soft Computing, 12th International Conference on Application of Fuzzy Systems and Soft Computing (ICAFS), Vienna, Austria, Procedia Computer Science 102, pp. 34 – 38, August 2016.

[2] Y. Dote and S. J. Ovaska, Industrial Applications of Soft Computing: A Review, Proceedings of The IEEE, Vol. 89, No. 9, September 2001.

[3] T Vale, I. Crnkovic, E. S. de Almeida , P. S. Neto, Y. C. Cavalcanti and S. R. D. L. Meira, "Twenty-eight years of Component-Based Software Engineering," Journal of Systems and Software, vol. 111, pp. 128–148, January, 2016.

[4] H. Singh, M. M. Gupta, T. Meitzler, Z. G. Hou, K. Garg, A. M. G. Solo and L. A. Zadeh, "Real-Life Applications of Fuzzy Logic", Advances in Fuzzy Systems, Hindawi Publishing Corporation, Article ID 581879, 3 pages, Volume 2013.

[5] A. S. Ghiduk, "Automatic generation of Basis Test Paths Using Variable Length Genetic Algorithm", Information Processing Letters, 114, pp. 304–316, 2014.

[6] S.Wang, S. Ali and A. Gotlie, Minimizing Test Suites in Software Product Lines Using Weight-Based Genetic Algorithms, Proceedings of the 15th annual conference on Genetic and evolutionary computation (GECCO'13), Amsterdam, The Netherlands, pp. 1493-1500, July 6–10, 2013.

[7] A. Abraham, H. Guo, and H. Liu, "Swarm Intelligence: Foundations, Perspectives and Applications", Swarm Intelligent Systems, Springer Berlin Heidelberg, pp. 3-25, 2006.

[8] X. S. Yang, Z. H. Cui, R. Xiao, A. Gandomi and M. Karamanoglu, Swarm Intelligence and Bio-Inspired Computation: Theory and Applications, Elsevier, 1st ed. 2013.

[9] Y. Bian, Z. Li, R. Zhao and D. Gong, "Epistasis Based ACO for Regression Test Case Prioritization," , IEEE Transactions on Emerging Topics in Computational Intelligence, Vol. 1, no. 3, pp. 213-223, June 2017.

[10] A. Pandey and S. Banerjee, Test Suite Optimization Using Chaotic Firefly Algorithm in Software Testing. International Journal of Applied Metaheuristic Computing (IJAMC), 8(4), 41-57. doi:10.4018/IJAMC.2017100103, 2017.

[11] M. Mann, P. Tomar and O. P. Sangwan, Bio-inspired Metaheuristics: Evolving and Prioritizing Software Test Data, Applied Intelligence, pp 1–16, 2017.

[12] A. Ansari, A. Khan, A. Khan and K. Mukadam, Optimized Regression Test using Test Case Prioritization, 7th International Conference on Communication, Computing and Virtualization, Procedia Computer Science 79, pp. 152 – 160, 2016.

[13] S. Agarwal, S. Gupta, and N. Sabharwal, Automatic Test Data Generation-Achieving Optimality Using Ant-Behavior, International Journal of Information and Education Technology, Vol. 6, No. 2, February 2016.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[14] R. Elghondakly, S. Moussa and N. Badr, "A Comprehensive Study for Software Testing and Test Cases Generation Paradigms", ACM, ICC '16, March 22-23, 2016.

[15] G. Kumar and P. K. Bhatia, "Neuro-Fuzzy Model to Estimate & Optimize Quality and Performance of Component Based Software Engineering", ACM SIGSOFT, Software Engineering Notes, March 2015.

[16] P. K. Singh, O. P. Sangwan, A. P. Singh and A. Pratap, An Assessment of Software Testability using Fuzzy Logic Technique for Aspect-Oriented Software, International Journal of Information Technology and Computer Science, Vol.7, No.3, Feb. 2015.

[17] S. Musa, A. B. M. Sultan, A. A. B. A. Ghani and S. Baharom , Software Regression Test Case Prioritization for Object-Oriented Programs using Genetic Algorithm with Reduced-Fitness Severity, Indian Journal of Science and Technology, Vol 8(30), DOI: 10.17485/ijst/2015/v8i30/86661, November 2015.

[18] S. Yang, T. Man, and J. Xu, Improved Ant Algorithms for Software Testing Cases Generation, Hindawi Publishing Corporation. Scientific World Journal, Article ID 392309, 9 pages, Volume 2014.

[19] P. P. Bonissone, Y. T. Chen, K. Goebel, And P. S. Khedkar, Hybrid Soft Computing Systems: Industrial and Commercial Applications, Proceedings of The IEEE, VOL. 87, NO. 9, 1999.

[20] H. Bhaumika, S. Bhattacharyyaa, M. D. Natha and S. Chakraborty, Hybrid Soft Computing Approaches to Content Based Video Retrieval: A brief review, Applied Soft Computing, Vol-46, pp-1008–1029, 2016.

[21] T. P. Jacob and T. Ravi, Optimization of Test Cases by Prioritization, Journal of Computer Science 9 (8): 972-980, 2013.

[22] C. Sharma, S. Sabharwal and R. Sibal, "A Survey on Software Testing Techniques using Genetic Algorithm", International Journal of Computer Science Issues, Vol. 10, Issue 1, No 1, January 2013.

[23] S. S. B. Lam, M L H. P. Raju, U. K. M, S. Ch and P. R. Srivastav, Automated Generation of Independent Paths and Test Suite Optimization Using Artificial Bee Colony, International Conference on Communication Technology and System Design-2011, Procedia Engineering 30, pp.191 – 200, 2012.

[24] A. M. Alakeel, A Fuzzy Test Cases Prioritization Technique for Regression Testing Programs with Assertions, ADVCOMP 2012: The Sixth International Conference on Advanced Engineering Computing and Applications in Sciences, 2012.

[25] S. Raju, G. V. Uma, Factors Oriented Test Case Prioritization Technique in Regression Testing using Genetic Algorithm, European Journal of Scientific Research, Vol.74 No.3, pp. 389-402, 2012.

[26] S. Bauersfeld, S. Wappler and J. Wegener, An Approach to Automatic Input Sequence Generation for GUI Testing using Ant Colony Optimization, *GECCO'11*, Dublin, Ireland. ACM 978-1-4503-0690-4/11/07, July 12–16, 2011.

[27] S. Sabharwal, R. Sibal and C. Sharma, Applying Genetic Algorithm for Prioritization of Test Case Scenarios Derived from UML Diagrams, IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 3, No. 2, May 2011.

[28] F. Saglietti and F. Pinte, "Automated Unit and Integration Testing for Component-based Software Systems", S&D4RCES, Vienna, Austria, ACM, 2010.

[29] Praveen, K Baby, "Automated Software Testing Using Metahurestic Technique Based on An Ant Colony Optimization"., Proceedings - 2010 International Symposium on Electronic System Design (ISED), 2010.

[30] Q. Zhongsheng, "Test case generation and optimization for user session-based web application testing." *Journal of Computers*, pp. 1655-1662, Nov 2010.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 1: Existing Soft Computing Based Testing Techniques: A Survey

Year of Publication	Title	Authors	Basic Soft Computing Technique Used
2017	Epistasis Based ACO for Regression Test Case Prioritization [9]	Y. Bian, Z. Li, R. Zhao and D. Gong	Ant Colony Optimization
2017	Test Suite Optimization Using Chaotic Firefly Algorithm in Software Testing [10]	Abhishek Pandey and S. Banerjee	Firefly Algorithm
2017	Bio-inspired Meta-heuristics: evolving and prioritizing software test data [11]	Mukesh Mann, Pradeep Tomar, Om PrakashSangwan	Genetic Algorithm, Particle Swarm Optimization
2016	Optimized Regression Test using Test Case Prioritization [12]	Ahlam Ansari, Anam Khan, Alisha Khan, Konain Mukadam	Ant Colony Optimization
2016	Automatic Test Data Generation-Achieving Optimality Using Ant-Behavior [13]	Sumesh Agarwal, Shubham Gupta, and Nitish Sabharwal	Ant Colony Optimization
2016	A Comprehensive Study for Software Testing and Test Cases Generation Paradigms [14]	Roaa Elghondakly, Sherin Moussa, Nagwa Badr	Ant Colony Optimization
2015	Neuro-Fuzzy Model to Estimate & Optimize Quality and Performance of Component Based Software Engineering [15]	Gaurav Kumar, Pradeep Kumar Bhatia	Neuro Fuzzy
2015	An Assessment of Software Testability using Fuzzy Logic Technique for Aspect-Oriented Software [16]	Pradeep Kumar Singh, Om Prakash Sangwan, Amar Pal Singh, Amrendra Pratap	Fuzzy Logic
2015	Software Regression Test Case Prioritization for Object-Oriented Programs using Genetic Algorithm with Reduced-Fitness Severity	Samaila Musa, Abu-BakarMd Sultan, Abdul-Azim Bin Abd-Ghani and Salmi Baharom	Genetic Algorithm
2014	Improved Ant Algorithms for Software Testing Cases Generation [18]	Shunkun Yang, Tianlong Man, and JiaqiXu	Ant Colony Optimization
2014	Automatic generation of basis test paths Using variable length genetic algorithm [5]	Ahmed S. Ghiduk	Genetic Algorithm
2013	Minimizing Test Suites in Software Product Lines Using Weight-based Genetic Algorithms [6]	Shuai Wang , Shaukat Ali, Arnaud Gotlieb	Genetic Algorithm
2013	Optimization of Test Cases by Prioritization [21]	T. Prem Jacob and T. Ravi	Genetic Algorithm
2013	A Survey on Software Testing Techniques using Genetic Algorithm [22]	Chayanika Sharma, Sangeeta Sabharwal, Ritu Sibal	Genetic Algorithm
2012	Automated Generation of Independent Paths and Test Suite Optimization Using Artificial Bee Colony [23]	Soma Sekhara Babu Lam, M L Hari Prasad Raju, Uday Kiran M, SwarajCh, Praveen Ranjan Srivastav,	Artificial Bee Colony Technique

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

2012	A Fuzzy Test Cases Prioritization Technique for Regression Testing Programs with Assertions [24]	Ali M. Alakeel	Fuzzy Logic
2012	Factors Oriented Test Case Prioritization Technique in Regression Testing using Genetic Algorithm [25]	S. Raju , G. V. Uma	Genetic Algorithm
2011	An Approach to Automatic Input Sequence Generation for GUI Testing using Ant Colony Optimization [26]	Sebastian Bauersfeld, Stefan Wappler, Joachim Wegener	Ant Colony Optimization
2011	Applying Genetic Algorithm for Prioritization of Test Case Scenarios Derived from UML Diagrams [27]	Sangeeta Sabharwal, RituSibal and Chayanika Sharma	Genetic Algorithm
2010	Automated Unit and Integration Testing for Component-based Software Systems [28]	F. Saglietti, F. Pinte	Genetic Algorithm
2010	Automated Software Testing Using Meta- heuristic Technique Based on An Ant Colony Optimization [29]	Praveen Ranjan, Km Baby	Ant Colony Optimization
2010	Test case generation and optimization for user session-based web application testing [30]	Qian Zhongsheng	Genetic Algorithm

Table 2: Pros and Cons of Various SC Techniques

Sr. No.	Technique	Pros	Cons
1.	Genetic Algorithm	• Easy to implement • Better solution after every iteration	• Slow Convergence • Problem of finding suitable Fitness Function • Applicable to small population
2.	Artificial Neural Network	• Adaptive Learning • Self-Organization • Fast Processing	• Requires large volume of training data • Black box view of hidden layers • Poor Interpretation
3.	Fuzzy Logic	• Ease of implementation • Interpretable	• Problems of finding suitable membership values
4.	Swarm Intelligence	• Self-Adaptive • Scalable • Parallelism	• Poor Interpretability • Premature Convergence