International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-8, Issue-1, May 2019

A Cat Swarm Optimization Based Test Case Prioritization Technique to Perform Regression Testing

Ms. Manaswini B, Rama Mohan Reddy A

Abstract:Developing the software is a very essential in now a day. Software development lifecycle has phases like requirements, designing, coding, testing and deployment. 50 % of the time will be consumed by the testing phase. Software testing is a cost effective phase. Testing phase will be done at various levels with different kinds of the tests for identifying the defects based on the user requirements.one of the major and important test was the regression test. This test will be performed after adding the additional functionalities to the existing software. The different techniques to perform the regression test are selection of testcases, prioritizing the existing testcases, reset all etc. This paper proposed a novel algorithm to perform the regression testbased on the prioritizing the test case technique by using the catswarm optimization algorithm. The results that performed on the open source applications like jtopas, jmeter was shown the effectiveness of the proposed algorithm in the parameters of execution time and false detect rate.

Index Terms: Software, Testing, cat swarm, optimization, Regression test, and Test case prioritization.

I. INTRODUCTION

The process of developing the software was done by using the lifecycle of SDLC. 50% of the software development life cycle cost will beexpensed by the testing phase[1][24].In many organizations the software is very essential. To deliver the good and efficient software to the customers is tremendous task. So, before delivering the product to the customer, for identification of the hidden defects/false the software testing will be performed.At different levels various tests will be performed with the different kind of tests.Few of the tests like unit test, integration test, system test, acceptance test, regression test, alpha test, beta test, performance test, load test, stress test, usability test, security test, portability test. The major and essential test was the regression testing[24].The regression test was important and cost effective testing. This was performed after adding the extra/new facilities to the existing software.

Revised Manuscript Received on 22 May 2019.

Manaswini B*, Research Scholar, Department of Computer Science and Engineering, SVUCE, Sri Venkateswara University, Tirupati, India.

Rama Mohan Reddy A, Professor, Department of Computer Science and Engineering, SVUCE, Sri Venkateswara University, Tirupati, India.

© The Authors. Published by Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP). This is an open access article under the CC-BY-NC-ND license http://creativecommons.org/licenses/by-nc-nd/4.0/

To the new facilities /new advantages to the existing software may leads to various faults due to adding the extra functionalities to overcome this issue the regression testing will be performed.Already many researchers has proposed various approaches for decreasing the regression test cost, the frequently used approaches are “Regression test case prioritization “, “Selection of Test cases”.
Few of the selection of testcase methods are in the papers [3][4][5][6][7][8][9][10][11][12] and few of prioritizing test case approaches are used to increase the fault detect rate are in the papers [3][13][14][15][16][17][18][19][20][21][22][23].

Prioritizing the test cases will depend on the permutations of the different test suites which has maximum value of the objective function.Regression testing is used to perform the testing and to ensure that there are no any errors in the software after adding the extra functionalities. It is a type of an integration testing which requires more time to perform testing. Regression testing can be performed in various methodologies like retest all [25][26], test selection[25][27[28], test case reduction [25][29][30] and test case prioritization [25][31][32]. Due to the regression testing software quality was enhanced, the test coverage will be increased checks the impact of added code etc. [35] The general practice for the regression test is to re-execute all the type of test suite that which consumes more cost[36]. To perform the cost effective and high detection rate the current paper used an approach based on the cat swarm based optimization algorithm.

A. CatSwarm Approach :

Based on the various behaviors of various cats chu et. al. proposed a technique called as a catswarm optimization technique. The cats generally have two phases seeking phase and tracking/tracing phase to move the cats in the solution criteria. In the seeking phase the cats will find the factors like seeking memory pool, seeking range for selected dimensions, and number of coordinates will be changed and the self-coordinates for the estimation of movement of the cats where as in the tracing mode it checks for the best value by that calculate in seeking mode . The seeking and tracking repeatedly performed until we get the good solution[37].In this paper proposed and implemented an cat swarm approachto perform the regression test by prioritizing the test suites technique using the cat swarm optimization algorithm.The remainder of the paper is, in the next section related work was discussed, in section III the proposed work was discussed, in section IV the results and discussion was discussed and finally in section V conclusion was discussed.

II. RELATED WORK:

In paper, [1] authors proposed an algorithm for regression testing based on the genetic algorithm along with the multi-fitness value. It consider the metrics like flow of control with various conditions, statements which covered by the test case. The proposed was compared with the other existing techniques and results proved that the proposed was superior compared to the existing once.In paper[2] proposed a method for arranging the tests in the prioritized manner. This method is the combination of the programming method and mahalanobistaguchi method. Here considered the parameters like runtime of various tests and prioritizing manner of tests, the experimental results was performed on 300 industrial based software test cases and proved that it was more effective when compared to conventional methods.

In paper [3] authors proposed a heuristic technique by using the lexicographical ordering notation to increase the fault detect rate by arranging the test case in an order and it proved that good in terms of fault detect rate when compared to the existence once.

In paper[24] authors proposed a method to perform the regression test, based on the Ant colony optimization algorithm for the distribution environment which was done on the Hadoop framework.

In paper [25] authors proposed an algorithm for arranging the test cases in an order based on the importance of the test case by using the ant colony optimization technique. By considering the factors like detection of faults, time of execution and severity of faults and the effectiveness was proved in terms of the APFD value.

In paper [33] author proposed anapproach to improve the effectiveness of the regression test by using the metric fault severity.

In paper [34] author proposed a techniqueto prioritize the test cases based on the test process by identifying the faults and by considering the metric APFD.

In this paper [35] author proposed an algorithm based on the grouping technique to perform the regression test by prioritizing the test cases with the metric scope of the code.

In this paper [36] proposed an approach that enhance the effectiveness and efficiency of regression technique and this technique will decrease the cost of the regression test.

III. PROPOSED SYSTEM:

In the proposed system catswarm optimization algorithm was used for arranging the testcases in the prioritizing priority based on the fitness value that which calculated by using the metrics like coverage scope, fault detect rate and execution time for the various tests. The proposed algorithm was shown in Algorithm 1. The flow of the algorithm was shown in the Figure 1, Figure2 and Figure 3.

The proposed algorithm will work in two stages Seeking stage and Tracking/Tracing stage. In the seeking stage it calculates the Fault detection rate, coverage scope, and execution time and Fitness value by using the formulas (1) and (2) for every test case as shown in the Table 1. The flow of the seeking phase was shown in the Figure 2. Whereas in the Tracing phase based on the fitness values of individual test cases all the tests were arranged in the decreasing order. The flow of the tracing phase was shown in the figure 3. The sorted test case will be in the priority order which is the output of the Algorithm 1. The overall flow of the proposed algorithm was shown in the Figure 1.

Algorithm 1: Cat swarm optimization based regression testing
Input: T_i $[T_1,T_2,T_3,\text{----------},T_n]$
Output: T_p $[T_3,T_2,T_4,\text{----------},T_n]$
1: Start 2: Repeat 3: For every Test case T_i 4: Seeking phase 5: Fault detection Rate FD_i 6: Execution Time of TE_i 7: Coverage Scope TC_i 8: Calculate Fitness value FV_i 9: Tracing Phase 10: Based on fitness value arrange the test cases in order. 11: End for 12: Until all the test case finish. 13: End cat swarm optimization based regression testing

Where

T_i	: Test cases
T_p	: Test cases in prioritizing order.
n	: Total Number of test cases.
FD_i	: Fault Detection rate for individual test case
TE_i	: Execution time of the test case for individual test case.
TC_i	: Coverage Scope
FV_i	:Fitness Value.
HFD_i	: Higher Fault detection rate
LTE_i	: Less Execution time
HTC_i	: More coverage Scope

Fault detection Rate

$$FD_i = \frac{Number\ of\ faults\ detected\ by\ Test\ case\ Ti}{Execution\ time\ of\ Testcase\ Ti} \quad (1)$$

Fitness Value

$$FV_i = HFD_i \cap LTEi \cap HTCi \quad (2)$$

The Table 1 consists of the fifteen test cases with the faults for every individual test case that obtained from the open source application Jtopas and Jmeter.

Table 1: Test cases with the Faults

Faults / Test Cases	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15
T1	✓				✓			✓		✓					
T2		✓	✓												
T3							✓								
T4	✓			✓											
T5	✓							✓	✓						
T6		✓		✓		✓									
T7	✓														
T8	✓			✓		✓		✓							
T9		✓			✓				✓						
T10	✓		✓		✓		✓								
T11	✓											✓			
T12															✓
T13													✓		
T14										✓				✓	
T15												✓			✓

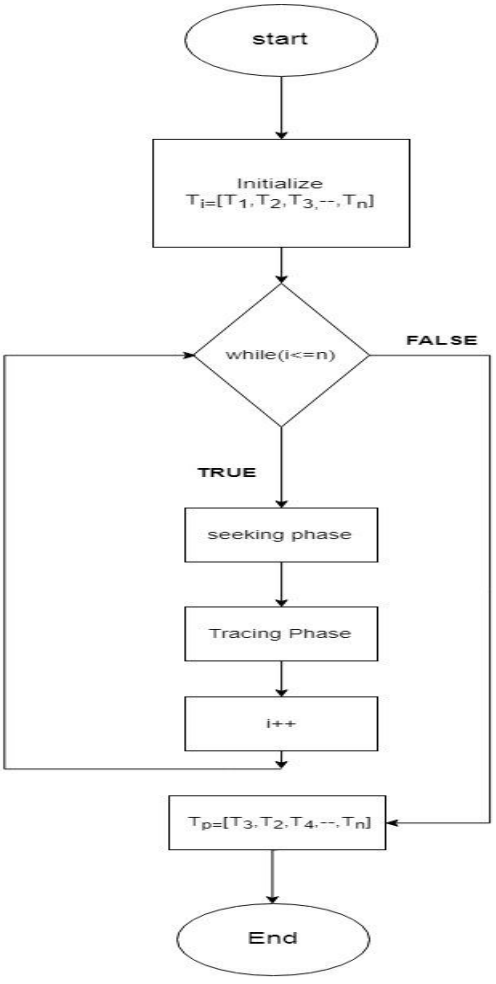

Figure 1: Flowchart for proposed algorithm.

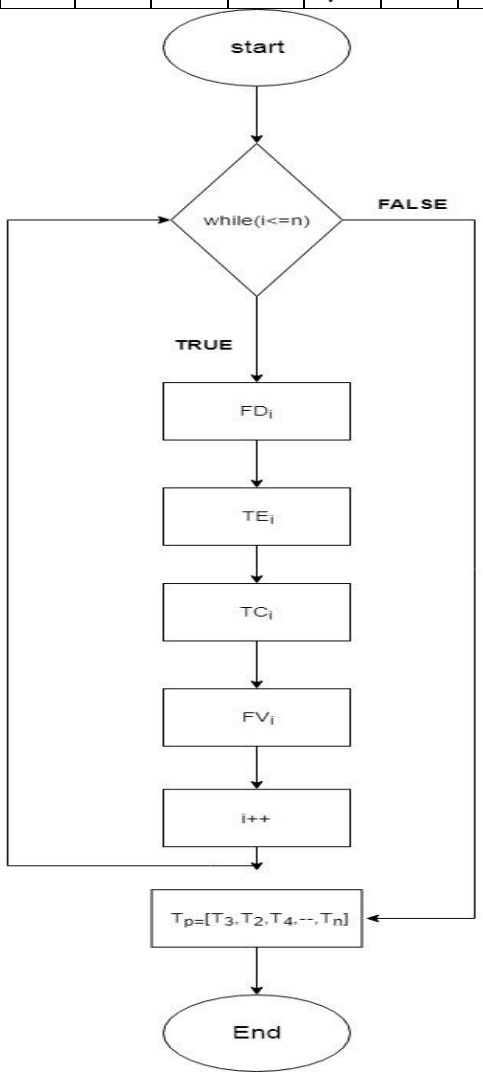

Figure 2: Flowchart of Seeking Phase

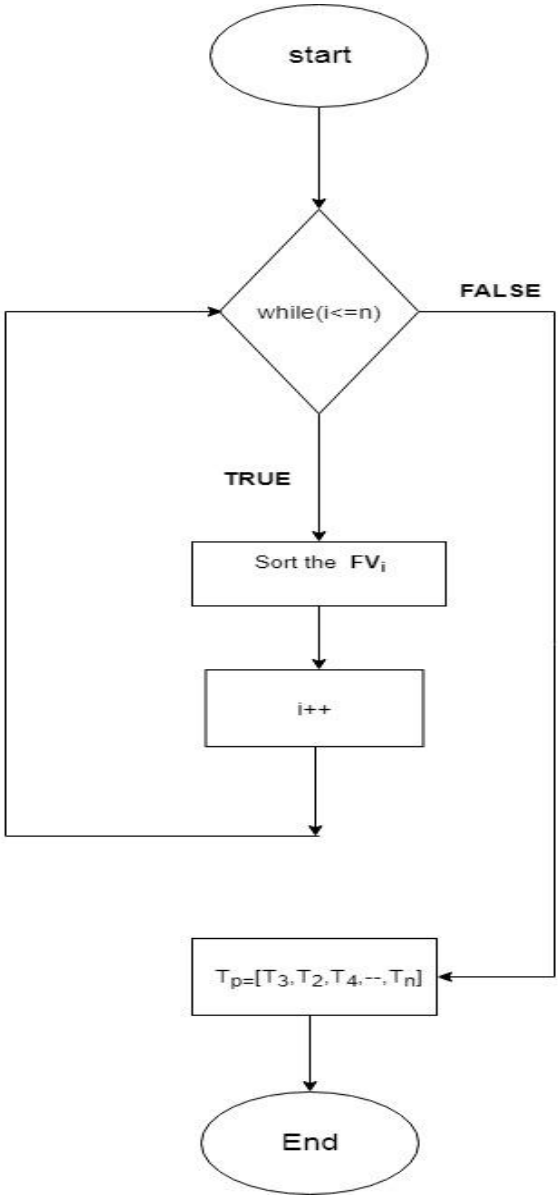

Figure 3: Flowchart of Tracing Phase

IV. RESULTS AND ANALYSIS:

The proposed Algorithm was applied on the opensource applications like jtopas and jmeter. The results was shown in the Figure 4, Figure 5 and Figure 6 in terms of execution time and fault detect.

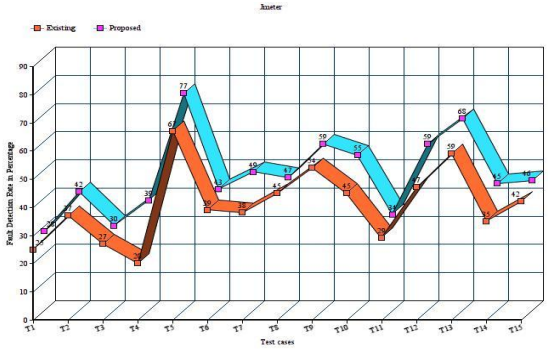

Figure 4: comparison of the existing and proposed systems on the Jmeter application in terms of fault detection rate.

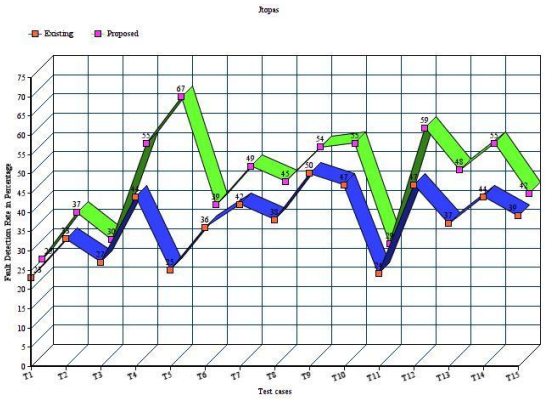

Figure 5: comparison of the existing and proposed systems on the Jtopas application in terms of fault detection rate.

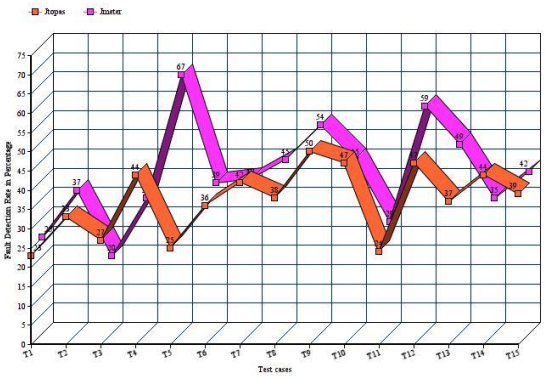

Figure 6: Fault detection rate of the proposed system on both the Jtopas and Jmetre applications

Table 2: shows the order of the test case before and after the proposed system

Table 3: Faults identified and execution Time for the individual Test case

Test case	Faults Detected	Run Time
T1	4	7
T2	2	2
T3	1	6
T4	2	4
T5	3	5
T6	3	9
T7	1	5
T8	4	6
T9	3	6
T10	4	8
T11	2	3
T12	1	8
T13	1	7
T14	2	4
T15	2	9

Original Order	Prior Order
T1	T14
T2	T9
T3	T3
T4	T12
T5	T6
T6	T4
T7	T7
T8	T15
T9	T2
T10	T1
T11	T10
T12	T5
T13	T13
T14	T8
T15	T11

V. CONCLUSION:

The work in this paper was based on the software testing particularly to perform the regression testing based on test case prioritization technique. To accomplish that here proposed algorithm based on the normal behavior of the cats called as a cat swarm optimization algorithm. This algorithm was performed in two phases called as seeking and tracing phase which will help to arrange the testcases in the prioritized order based on the fitness value. The proposed algorithm was implemented on the open source applications like jtopas and jmeter. The Results had shown the effectiveness of proposed algorithm in terms of the execution time and the fault detection rate.

REFERENCES:

1. Ahmed, AmrAbdelFatah, Mohamed Shaheen, and EssamKosba. "Software testing suite prioritization using multi-criteria fitness function." 2012 22nd International Conference on Computer Theory and Applications (ICCTA). IEEE, 2012.
2. Aman, Hirohisa, et al. "Application of Mahalanobis-Taguchi method and 0-1 programming method to cost-effective regression testing." 2016 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA). IEEE, 2016.
3. Eghbali, Sepehr, and LadanTahvildari. "Test case prioritization using lexicographical ordering." IEEE Transactions on Software Engineering 42.12 (2016): 1178-1195.
4. Briand, Lionel C., YvanLabiche, and George Soccar. "Automating impact analysis and regression test selection based on UML designs." International Conference on Software Maintenance, 2002. Proceedings.. IEEE, 2002.
5. Cartaxo, EmanuelaGadelha, Francisco G. Oliveira Neto, and Patrícia DL Machado. "Automated Test Case Selection Based on a Similarity Function." GI Jahrestagung (2) 7 (2007): 399-404.
6. Chen, TsongYueh, and Man Fai Lau. "Dividing strategies for the optimization of a test suite." Information Processing Letters 60.3 (1996): 135-141.
7. Chen, TsongYueh, and Man Fai Lau. "Test case selection strategies based on boolean specifications." Software Testing, Verification and Reliability 11.3 (2001): 165-180.
8. Harrold, Mary Jean, and M. L. Souffa. "An incremental approach to unit testing during maintenance." Proceedings. Conference on Software Maintenance, 1988.. IEEE, 1988.
9. Huang, Yu-Chi, Kuan-Li Peng, and Chin-Yu Huang. "A history-based cost-cognizant test case prioritization technique in regression testing." Journal of Systems and Software 85.3 (2012): 626-637.
10. Mirarab, Siavash, SoroushAkhlaghi, and LadanTahvildari. "Size-constrained regression test case selection using multicriteria optimization." IEEE transactions on Software Engineering 38.4 (2012): 936-956.
11. Sprenkle, Sara, et al. "An empirical comparison of test suite reduction techniques for user-session-based testing of web applications." 21st IEEE International Conference on Software Maintenance (ICSM'05). IEEE, 2005.
12. Yoo, Shin, and Mark Harman. "Pareto efficient multi-objective test case selection." Proceedings of the 2007 international symposium on Software testing and analysis. ACM, 2007.
13. Dickinson, William, David Leon, and Andy Podgurski. "Finding failures by cluster analysis of execution profiles." Proceedings of the 23rd international conference on Software engineering. IEEE Computer Society, 2001.
14. Elbaum, Sebastian, Alexey G. Malishevsky, and Gregg Rothermel. "Test case prioritization: A family of empirical studies." IEEE transactions on software engineering 28.2 (2002): 159-182.
15. Hao, Dan, Xu Zhao, and Lu Zhang. "Adaptive test-case prioritization guided by output inspection." 2013 IEEE 37th Annual Computer Software and Applications Conference. IEEE, 2013.
16. Kim, Jung-Min, and Adam Porter. "A history-based test prioritization technique for regression testing in resource constrained environments." Proceedings of the 24th international conference on software engineering. ACM, 2002.
17. Leon, David, and Andy Podgurski. "A comparison of coverage-based and distribution-based techniques for filtering and prioritizing test cases." 14th International Symposium on Software Reliability Engineering, 2003. ISSRE 2003.. IEEE, 2003.
18. Li, Zheng, Mark Harman, and Robert M. Hierons. "Search algorithms for regression test case prioritization." IEEE Transactions on software engineering 33.4 (2007): 225-237.
19. Rothermel, Gregg, et al. "Prioritizing test cases for regression testing." IEEE Transactions on software engineering 27.10 (2001): 929-948.
20. Smith, Adam M., and Gregory M. Kapfhammer. "An empirical study of incorporating cost into test suite reduction and prioritization." Proceedings of the 2009 ACM symposium on Applied Computing. ACM, 2009.
21. Srikanth, Hema, Laurie Williams, and Jason Osborne. "System test case prioritization of new and regression test cases." 2005 International Symposium on Empirical Software Engineering, 2005.. IEEE, 2005.
22. Zhang, Lingming, et al. "Bridging the gap between the total and additional test-case prioritization strategies." Proceedings of the 2013 International Conference on Software Engineering. IEEE Press, 2013.
23. Zhang, Lu, et al. "Time-aware test-case prioritization using integer linear programming." Proceedings of the eighteenth international symposium on Software testing and analysis. ACM, 2009.
24. Elanthiraiyan, Nandhini, and ChamundeswariArumugam. "Parallelized ACO algorithm for regression testing prioritization in hadoop framework." 2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies. IEEE, 2014.
25. Gao, Dongdong, XiangyingGuo, and Lei Zhao. "Test case prioritization for regression testing based on ant colony optimization." 2015 6th IEEE International Conference on Software Engineering and Service Science (ICSESS). IEEE, 2015.
26. Duggal, Gaurav, and Bharti Suri. "Understanding regression testing techniques." Proceedings of 2nd National Conference on Challenges and Opportunities in Information Technology. 2008.
27. Rothermel, Gregg, and Mary Jean Harrold. "A safe, efficient regression test selection technique." ACM Transactions on Software Engineering and Methodology (TOSEM) 6.2 (1997): 173-210.
28. Graves, Todd L., et al. "An empirical study of regression test selection techniques." ACM Transactions on Software Engineering and Methodology (TOSEM) 10.2 (2001): 184-208.
29. Harrold, M. Jean, Rajiv Gupta, and Mary Lou Soffa. "A methodology for controlling the size of a test suite." ACM Transactions on Software Engineering and Methodology (TOSEM) 2.3 (1993): 270-285.
30. Jones, James A., and Mary Jean Harrold. "Test-suite reduction and prioritization for modified condition/decision coverage." IEEE Transactions on software Engineering 29.3 (2003): 195-209.
31. Elbaum, Sebastian, Alexey G. Malishevsky, and Gregg Rothermel. "Test case prioritization: A family of empirical studies." IEEE transactions on software engineering 28.2 (2002): 159-182.
32. Srivastava, Praveen Ranjan. "TEST CASE PRIORITIZATION." Journal of Theoretical & Applied Information Technology 4.3 (2008).

33. Wang, Yiting, Xiaomin Zhao, and Xiaoming Ding. "An effective test case prioritization method based on fault severity." 2015 6th IEEE International Conference on Software Engineering and Service Science (ICSESS). IEEE, 2015.
34. Nayak, Soumen, Chiranjeev Kumar, and SachinTripathi. "Effectiveness of prioritization of test cases based on Faults." 2016 3rd International Conference on Recent Advances in Information Technology (RAIT). IEEE, 2016.
35. Sun, Chang-ai, et al. "dμreg: A path-aware mutation analysis guided approach to regression testing." Proceedings of the 12th International Workshop on Automation of Software Testing. IEEE Press, 2017.
36. Tsai, Pei-Wei, and VaciIstanda. "Review on cat swarm optimization algorithms." 2013 3rd International Conference on Consumer Electronics, Communications and Networks. IEEE, 2013.