International Journal of Control Theory and Applications

ISSN : 0974-5572

Volume 9 • Number 46 • 2016

A Hybrid Cuckoo-ACO Algorithm for Test Case Prioritization

Mohammed Husnain Siddique[a], Deepak Panwar[b] and Pradeep Tomar[c]

[a]*Amity School of Engineering & Technology, Amity University Rajasthan, India. Email: husil4772@gmail.com*
[b,c]*School of Information and Communication Technology, Gautam Buddha University, India*

Abstract: Regression testing is one of the types of maintenance testing and it is performed in case of bug-fixing or whenever there is any new functionality incorporate in the dynamic environment of software development process. Due to which the cost of development process increases because it directly affects the validation process. The addition of the new requirements with limited resources of development process in a time constrained environment definitely increase the cost. Efficient and effective test case selection from the available test suite becomes very critical problem in this scenario and this situation makes it important to applying of some techniques which are prioritization of test cases and selection of test cases for re-arrangement of test cases in a particular schedule and these test cases are selected in a particular sequence order to fulfillment of some selected criteria. This paper suggests Cuckoo Search (CS) algorithm followed by Modified Ant Colony Optimization (M-ACO) algorithm to conclude the test cases in an optimized order in time constrained environment. CS algorithm is inspired by some species of cuckoos having constrained brood lethargy and laying of their eggs in other host bird's nest. Due to dependency of this algorithm on one single parameter, cuckoo search unlike other optimization algorithms, is more efficient and very easy to implement. But the hybrid Cuckoo-ACO optimization technique is more appropriate for the test case selection and prioritization according to the proposed empirical study.

Keywords: Cuckoo Search, Ant Colony Optimization, Software Quality, Test Case Prioritization, Hybrid Cuckoo-ACO Algorithm.

1. INTRODUCTION

Software testability is an important aspect of SDLC. Testing of the software is done with the intent to find faults by executing the program[1]. Software testing is one of the most promising areas of research in Artificial Intelligence. Cuckoo search is one of the meta-heuristic techniques of AI which consists of imitating animal behavior which was proposed by Yang and Deb in 2009[4]. The Cuckoo Search algorithm was inspired by some cuckoo species having obligate brood parasitism and laying their eggs in the host bird's nests. Some of the female parasitic cuckoos of these species are able to imitate the patterns and colors of the eggs of some specific species of hosts. This increases the re-productivity of cuckoo by reducing the probability of the eggs being abandoned by the hosts. These parasitic cuckoos lay their eggs in the nests where the host bird just laid its own eggs. Cuckoo's eggs are hatched slightly earlier than host's eggs. After the hatching of first cuckoo chick, its first action is to blindly propelling the host's eggs out of the nest intent to evict the host's eggs. This action increases the share of food provided to the cuckoo's chick by the host bird. This paper proposes a hybrid algorithm composing Cuckoo Search and Standard ACO algorithm for optimization of solutions for NP-Hard Problems like Travelling Salesman Problem by reducing execution time cost and obtain best path cost in minimum time.

2. SOFTWARE QUALITY

Software quality measures and ensures the quality of the software, how well software is designed and also ensures the quality of conformance. Software has been considered a product and in quality sense, all the attributes are expected to be improved for stakeholder satisfaction. The main purpose of Quality Assurance is to verify that applicable procedures and standards are being followed[2]. The focus of quality control is on finding and removing defects. Testability is a software quality metric that is of relevance for software dependability as well as test costs. It helps the software testers to keep the test efforts under control. Due to cost and time constraints, the entire test suites are difficult to run. Regression testing ensures that the modifications in the software are correct and have not affected the unmodified modules of the software[10]. Instead of running entire test suite, selected test cases are run. Thus test case prioritization is done in order to cover maximum faults in minimum execution time. In order to reduce cost related to regression testing, many techniques are proposed including regression test prioritization[3, 5, 7] and regression test selection[6]. Cuckoo search algorithm is a meta-heuristic optimization algorithm which is recently developed and is suitable for optimization problem solving. After the proposal of cuckoo search algorithm, it has been used to optimize the problems more effectively and efficiently than other Particle Swarm Optimization (PSO) techniques. This paper presents the optimal solution for one of the most famous examples of NP – Hard problems, i.e. Travelling Salesman Problem. For obtaining the optimal solution, researchers have used Cuckoo search algorithm along with Ant Colony Optimization algorithm.

3. PROPOSED ALGORITHM

The proposed algorithm is the combination of Standard Ant Colony optimization Algorithm and Cuckoo search Algorithm to produce a hybrid algorithm to obtain highly optimized solution for NP-Hard problems like TSP. The proposed algorithm operates in two steps to optimize solution.

$$x_1^{(t+1)} = x_1^{(t)} + \alpha \oplus \text{L\'evy}(\lambda) \tag{1}$$

$$\text{L\'evy} \sim u = t - \lambda,\ (1 < \lambda \leq 3) \tag{2}$$

where, $\alpha > 0$ is the step size which is related to the scale of the problems' interest and Equation 2 provides L´evy distribution to draw random step length while L´evy Flight provides random walk[4].

A. Hybrid Cuckoo-ACO algorithm

Begin

Initialize population of nests of host x_i (i=1, 2, 3.., n)

While(1<MaxGen) or (criterion stop)

Get random solution by Le`vy Flight.

Evaluate solution fitness f_i.

Choose new solution randomly among the solutions

having solution fitness f_j.

if($f_j < f_i$)

Replace previous solution *j* by new solution.

end

probability function (P_a) of the nest having

worse solution decides that the nest has to be abandoned by host and

new ones to be built.

Store and keep the best solution (S_b).

end while

Initialize τ, η and visibility for every edge for traversal by Ant

for(*i* < Maximum Iteration)

for every ant **do**

Obtain the probability to choose the next node by adding best nest solution (**S_b**, obtained by Cuckoo search) with heuristic information of Ant.

Choose the next node to move on by the Ant probabilistically.

Add that move as heuristic information for each of the Ant.

Repeat above steps each and every ant traverses all the paths and completes one solution.

end

for each of the ant who completes traversal of one new solution **do**

Update τ for the edges where traversal by ant has completed

end

if (local best solution is much better than the global slоution)

save the local solution as the global solution

best cost = global solution

end

end

end

end // end of algorithm

In step 1, Cuckoo search algorithm is performed to obtain solution vector corresponding to the best optimum value for Mantegna's algorithm[8]. The result for the solution vector showing values for best nest for optimization of solution is used. The nest having minimum value is the best solution for Cuckoo search which is denoted by S_b, is used as additional heuristic information to perform next step of the proposed algorithm.

$$p_{ij}(\kappa) = \begin{cases} \dfrac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta+S_b}}{\sum_{j \in C}\tau_{ik}^{\alpha}\eta_{ik}^{\beta+S_b}} & \text{If } j \in C \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

In step 2, standard ACO algorithm is modified by adding the best nest value with the heuristic information which increases the overall heuristic information and thus increase the probability to find best node to traverse by the Ant. The increased probability to find the best node reduces the number of iterations. Reduction in number of iterations will lead to reducing the consumption of time to obtain best cost of path.

Table 1 shows the comparison between standard ACO and Hybrid Algorithm describing better optimization of the solution for NP-Hard Problems. The parameters for the proposed algorithm are set as No. of nest = 25, P_a = 0.25, No. of Ants = 40, No. of iterations = 320 and Pheromone Evaporation rate = 5%. Figure 1 shows the plot solution and graph for TSP performed by Hybrid Algorithm. The result shows that the proposed algorithm gives best cost in minimum iterations and reduces execution time.

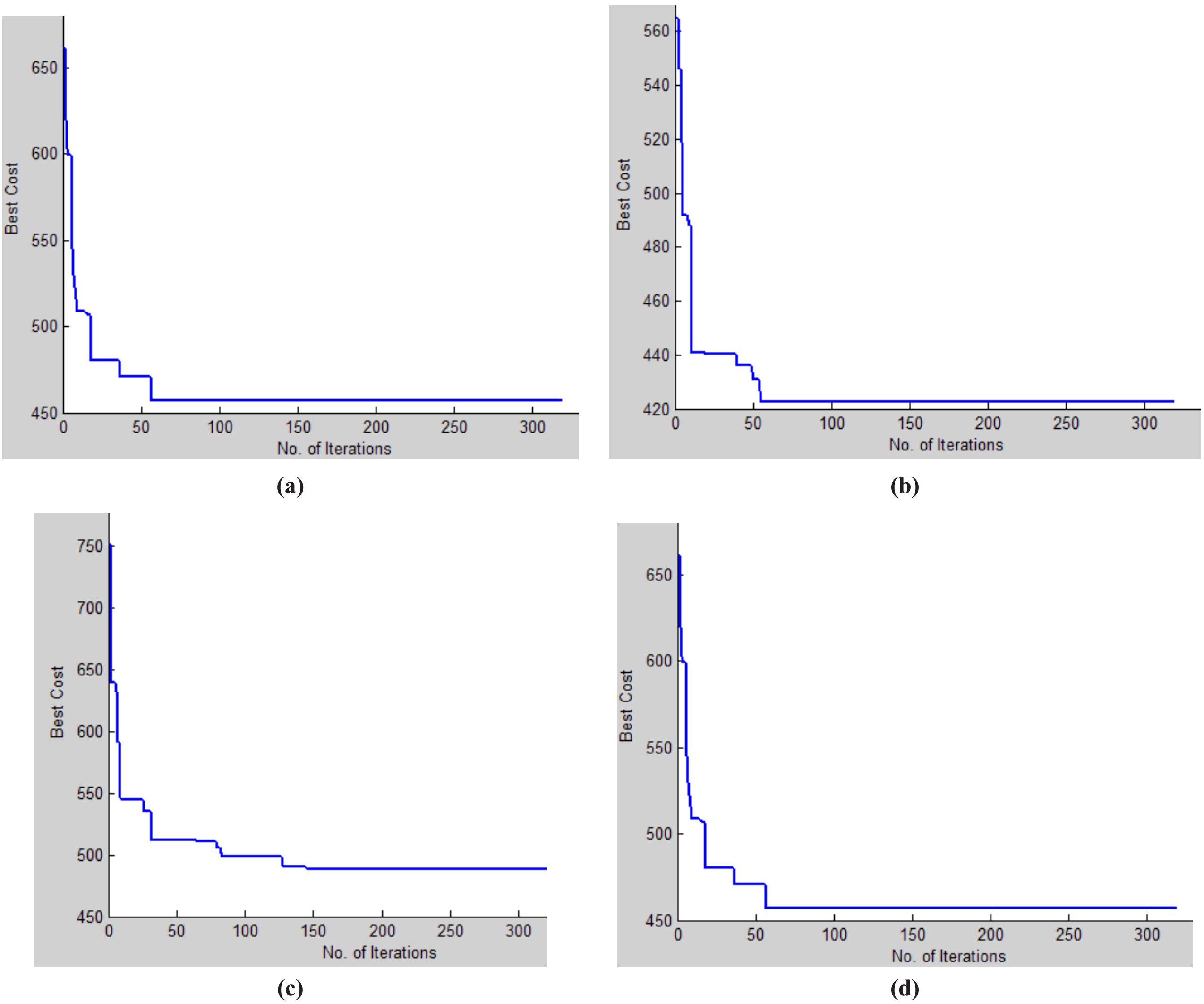

Figure 1: Plot solutions for TSP using Hybrid Algorithm at (a) No. of cities = 25, (b) No. of cities = 30, (c) No. of cities = 35 and (d) No. of cities = 40

Table 1
Comparison between Standard ACO and Hybrid Algorithm for TSP

Number of Cities	*Best Cost for Path*	*Standard ACO*		*Proposed Algorithm*	
		Best Cost	*Accuracy*	*Best Cost*	*Accuracy*
20	391.8958	393.8225	99%	391.8958	100%
25	422.7331	430.832	98%	422.7331	100%
30	457.0836	470.33	97%	457.0836	100%
35	468.5679	486.0842	96%	468.5679	100%
40	488.8696	502.7137	97%	488.8696	100%

4. EXAMPLE VALIDATION

In this section, this paper has applied the hybrid algorithm on previous triangle categorization problem and prioritizes the test cases generated using DD-Path graph for the given problem where number of independent paths is equal to the cyclometric complexity of the DD-Path graph[9]. The regression test suite T consists of seven test cases i.e., {T1, T2, … T7} covering with execution time 3, 5, 4, 4, 5, 4 and 2 respectively. The test suite covers all the 10 faults shown in the Table 2 in minimum execution time. The priority of test cases is provided according to the weight of the path traversed by the Ant from Source node S to destination node D. Weight of the test case is obtained by using formula given below.

$$\text{weight (test case)} = \sum_{i=S}^{D} \text{weight(edge)}_i \text{ /Tour length} \tag{4}$$

$$\text{weight (edge)} = \text{pheromone} \times \text{new heuristic value} \tag{5}$$

where the new heuristic value is obtained by addition of best solution S_b of cuckoo search and previous heuristic information of the Ants. Table 3 shows four orderings of test cases for comparison with proposed algorithm test case ordering. Original ordering, Random ordering, optimal ordering and proposed algorithm ordering are shown in Table 3. The table shows the priority ordering of the test cases by proposed algorithm is superior to other orderings.

Optimal ordering of test cases require 14 time units to cover total faults whereas Proposed algorithm requires execution of first three test cases covering all the defined faults in 12 units of time whereas random ordering and original ordering require much execution time the optimal order and proposed algorithm order. TABLE 3 describes the proposed algorithm ordering of test cases according to the weight of the path to be traversed from source node to destination node.

Table 2
Faults detected by Test cases and their Execution time

Fault \ *Test Case*	*T1*	*T2*	*T3*	*T4*	*T5*	*T6*	*T7*
F1		∅					
F2	∅				∅		∅
F3			∅	∅		∅	
F4			∅		∅		
F5		∅					
F6	∅			∅		∅	
F7		∅			∅		
F8	∅	∅				∅	
F9			∅		∅		
F10	∅			∅			∅
No. of faults covered	4	4	3	3	4	3	2
Execution Time	3	5	4	4	5	4	2

5. CONCLUSION

This paper focused on nature inspired optimization algorithm and the ultimate goal of the research was to find an appropriate optimization algorithm for the test case prioritization. Because the optimization of test cases is directly proportional to the profit of software development industry and by selection the optimal test cases, the developer team can reduce the cost and increase the profit. So, the proposed hybrid cuckoo-ACO algorithm is very beneficial for optimal test case prioritization and it would be very beneficial for software development industry.

Table 3
Various Orderings of test cases for Prioritization

Original Ordering	*Random Ordering*	*Optimal ordering*	*Proposed Algorithm Ordering*
T1	T6	T2	T2
T2	T2	T5	T1
T3	T3	T4	T3
T4	T1	T3	T5
T5	T5	T1	T4
T6	T7	T6	T7
T7	T4	T7	T6

REFERENCES

[1] Mayers GJ. The Art of Software testing, John Wiley and Sons, USA. 1997.

[2] Jolate P, CMM in Practice, Person Education Pte. Ltd., Delhi, 2004.

[3] Elbaum S, Rothermel G, Kanduri S, Malishevsky AG. Selecting a cost-effective test case prioritization technique. Software Quality Journal. 2004; 12(3):185-210.

[4] Yang XS, Deb S, “Cuckoo search via lévy flights”, Processing of World Congress on Nature & Biologically Inspired Computing”, IEEE Publications, USA. 2009 Dec; 210-214.

[5] Krishnamoorthi R, Sahaaya SA, Mary A. Regression Test Suite Prioritization using Genetic Algorithms. International Journal of Hybrid Information Technology. 2009; 2(3):35.

[6] Graves, TL, Harrold MJ, Kim MJ, Porter A, Rothermel G. An empirical study of regression test selection techniques. ACM Trans. on Softw. Eng. and Meth. 2001; 10(2).

[7] Rothermel G, Harrold MJ, Dedhia J. Regression test selection for C++ programs, Software Testing, Verification and Reliability. 2000; 10(2):77-109.

[8] Rothermel G, Untch RH, Chu C., Harrold MJ. Test Case Prioritization: An Empirical Study, in: Proceedings of the International Conference on Software Maintenance, Oxford, UK, 1999 Sept; 179-188.

[9] Rothermel G, Untch RJ, and Chu C. Prioritizing test cases for regression testing. IEEE Transaction on Software Engineering. 2001; 27(10):929-948.

[10] Walcott KR, Lou SM, Kapfhammer GM, Robert SR. Time Aware Test Suite Prioritization. 2006; 1-2.

[11] Mantegna RN. Fast and Accurate algorithm for Numerical Simulation of L`evy Stable Stochastic Processes. 1994; 49(4):4677-4683.