Volume 5, Issue 4, 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Evaluation of test cases using Ant Colony Optimization with Enhanced Heuristics:An Implementation

[1]Gulwatanpreet Singh, [2]Poonam Rana, [3]Parveen Kakkar
[1]Assistant Professor, Dr BR Ambedkar NIT, Jalandhar, India
[2]DAVInstitute of Engineering and Technology, Jalandhar, India
[3]Asstt. Professor, DAV Institute of engineering and Technology, Jalandhar, India

Abstract--A test case in software engineering is a set of conditions or variables under which a tester will determine whether an application or software system is working correctly or not. Before evaluation of test cases, the very first step of utmost importance is to select the appropriate test cases. This step is of great importance as it is difficult to cover all the test cases pertaining to a specific data. Evaluation of test cases is effort consuming and error-prone process. After selecting the appropriate test cases, next comes the various techniques used for evaluation of test cases. In my earlier research paper, I proposed a technique used for evaluation of test cases by Ant colony System with a new heuristic function. That paper shows us about the various advantages we can gain if we use that particular heuristic function in our technique. An algorithm was given for the technique using that heuristic function. This paper implements the algorithm which was proposed in earlier paper[15], in MATLAB. Also, we have made the comparison of the new and old technique which has proven to be very efficient.

Keywords: software testing, Ant colony system, test case, heuristic function. Travelling Salesman Problem.

I. INTRODUCTION

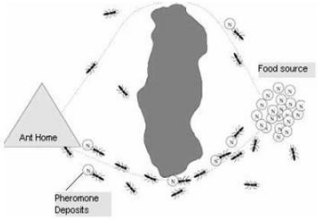
Software testing is an activity aimed at evaluating difference between the actual requirements and the requirements that are met in a software product or project. The outcome of the software testing phase is generally the errors that will be in the software. Testing conforms that the system or software works fulfilling the required needs .The main focus of software testing is to achieve the maximum level of a quality in the product. The whole process of software testing is generally divided into three different stages. a. Generation of test cases b. Execution of the test cases c. Evaluation of results of values of all the test cases in a test suite.

The process of testing any software system is a time consuming process and pays much more on cost factor [5].Lot of time and effort is required in this process. Software testing when gets combined with the artificial intelligence helps in making the work more efficient and error free. Combination of AI and software testing provide different advantages as mentioned in published papers [6] [7].

In this paper, we are going to implement the technique which helps in evaluation of test cases using ACO but with a different heuristics function. The heuristic is a way of defining how far we are from our goals. The proposed technique will overcome the problems that occur in the earlier evaluation of test cases with ACO. In my earlier studies, the heuristic function was chosen according to the mathematical operation (in the case of test cases pertaining to calculator). The whole structure of the paper is divided into 4 sections: **Section 1** contains the introduction about ACO and TSP problem and how the TSP is evaluated using ACO and it also describes the earlier technique used for evaluation of test cases by using ant colony systems. **Section 2** describes the technique which was proposed by us in the earlier paper and its implementation. **Section 3**reviews the results of this new technique and also compares the results with earlier techniques used. The comparisons are illustrated with graphs and tables.

SECTION I:

Ant Colony Optimization belongs to family of ant colony algorithms, in swarm intelligence methods. It consists of some meta heuristic optimizations or heuristic which is designed for finding, generating or selecting a lower-level procedure or heuristic (partial search algorithm) that may provide a sufficiently good solution to an optimization problem particularly with incomplete information or limited computation capacity. Meta-heuristics [13] may make few assumptions about the optimization problem being re-solved and thus they may be utilizable or usable for a range of problems. The following diagram describes the working of Ant colony optimization algorithm.



The framework of a basic ACO algorithm

While considering the theoretical properties of ACO metaheuristics, first thing is to generalize the ACO definition. Here, The ACO is introduced in the form of an algorithm that was theoretically studied. It works as follows. At each iteration, ants probabilistically construct solutions to the combinatorial optimization problem under consideration, exploiting a given pheromone model. Then, optionally, a local search procedure is applied to the constructed solutions. The following are the steps for this.

Initialize Pheromone Values (T). At the start of the algorithm the pheromone values are all initialized to a constantvalue c > 0.

Construct Solution (T). The basic ingredient of any ACO algorithm is a constructive heuristic for probabilisticallyconstructing solutions. A constructive heuristic assembles solutions as sequences of elements from the finite set of solution components C. A solution construction starts with an empty partial solution S= (). Then, at each construction step the current partial solution Sp is extended by adding a feasible solution component from the set N(S) C \ {Sp}.

SECTION II:

This section describes the technique used for evaluation of test cases using a different heuristic function which was proposed by us in the earlier paper. The algorithm for this technique is as following:

Algorithm for Evaluation of Test Cases using ACO with a new heuristic function

1. Start
2. Initialize the parameters
2.1. Initialize the different test cases i.e T = {T_1; T_2; T_3…..T_n}.
3. Initialize the trail value of pheromone i.et_i= 0
4. Each Ant is placed individually on starting state with empty memory M_k.
5. Do while (cycle loop gets completed)
{

5.1. Do upto (tour loop gets completed)
5.2. Update the value of t_i//(Local trail update)
}

6. Evaluate the tours.
6.1. Construct solution for every ant.
6.2. Check the best tour in terms of higher density of pheromone.
6.3. Display the results.
6.4. (Global Trail update)

7. Apply new heuristic function
7.1. For each path in the best tour do step 1 to 2
7.2. If path i (i = 1, 2, n) is not updated before do step 2
7.3$\eta_i = \eta_i + (\delta/$ best-so-far tour) // δ is parameter ranging from (0-

10) 8. Finish

By using this technique, It was proposed that we can overcome the difficulties that were arising while doing evaluation of test cases using ACO with earlier techniques, such as Pheromone stagnation, etc.
This enhanced ACS technique was integrated with a new heuristic function [15] that will update the heuristic value each time the ants find an enhanced solution in the iteration. After the ant constructs its solution, a global update process will be applied for updation of the best-so-far solution. This event will change the environment for the next coming iteration. A function would be generated at this moment to reflect this change and hence a new heuristic value will be attained. The new information will be applied to the best-so-far edge or tour.After applying this function, the values of heuristics will change according to the quality of best optimal solution. As we found the best solution, it will increase the heuristic value and vice versa. The parameter δ will reflect the influence of updating value that should be applied to heuristic value. This whole algorithm is implemented in MATLAB.

SECTION III

The following figure shows the tour path generated by the ants for 30 nodes (nodes being the values of test cases). i.e. the best possible path for evaluation of test cases using Ant colony optimization with a new heuristic function. The first node will always be chosen randomly and hence the shape of curve will variate every time we run the program but the variation of the curve between the values of X-axis and Y-axis(boundary values) will remain the same(for particular set of nodes).

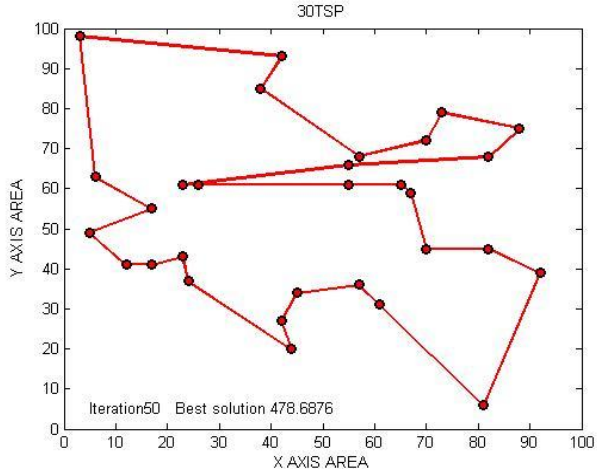


Fig.3.1 Tour path generated by ants for 30 test cases

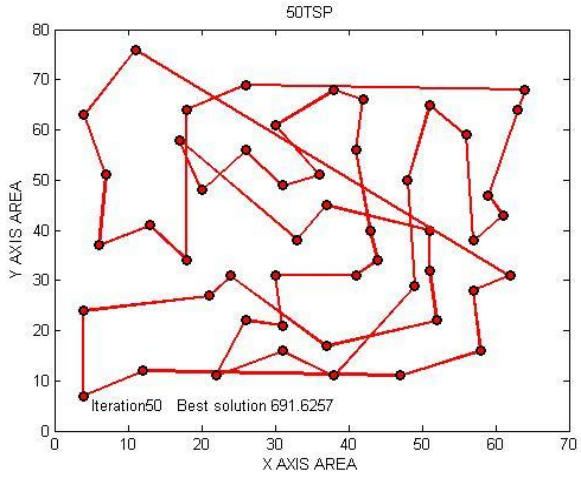
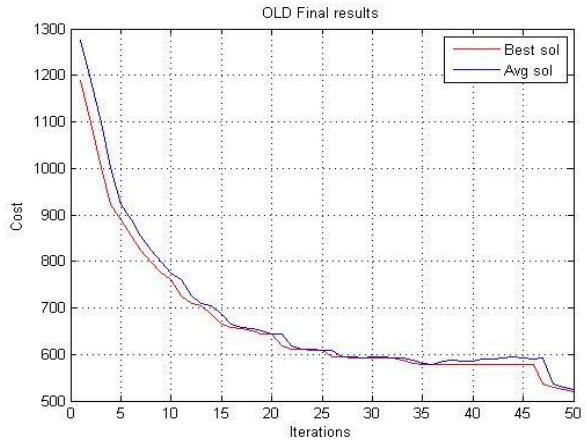


Fig.3.2 Tour path generated by ants for 50 nodes

II. BEST SOLUTION AND AVERAGE SOLUTION

The final result obtained by the ants will be the global best solution. Hence this program avoid being gets trapped in local optimal. In order to find the globally optimal solution, a global scope search effort is needed, which is indeed a basic part of the program. The essential idea is to apply a preliminary grid search or random search based global phase, followed by applying a local (convex programming) method.For instance, random multi start performs a local search from several points selected randomly from the search domain **D.** A common (standard) model form is the minimization of one real-valued function f in the parameter-space $\vec{x} \in P$, or its specified subset **.** On the other hand, average solution, as its name suggests, is considered as numerically the average of all the local optimum solutions. Although that is a good practice, but it does not ensure us about the global best solution. We can easily distinguish from the following figure that it is as close to the best solution.

The following curve shows the best solution and average solution generated by the program for evaluation of 30 test cases with Ant Colony Optimization.



Comparisons:

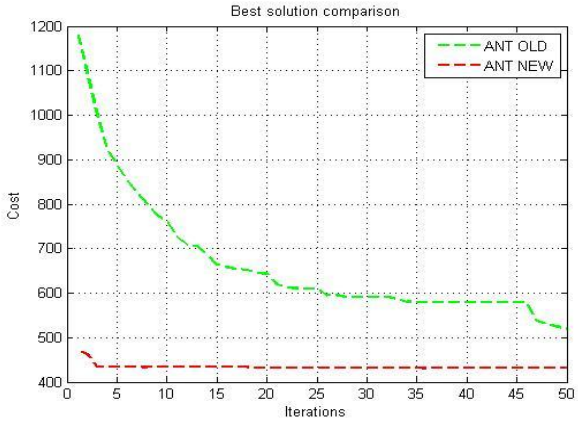


Fig.Comparison of best solution for between the new and old technique (30 Test Cases)

This above figure tells us about the cost incurred during different iteration. The cost per iteration is very less as compared to the old technique. Also, the total cost is comparatively reduced.

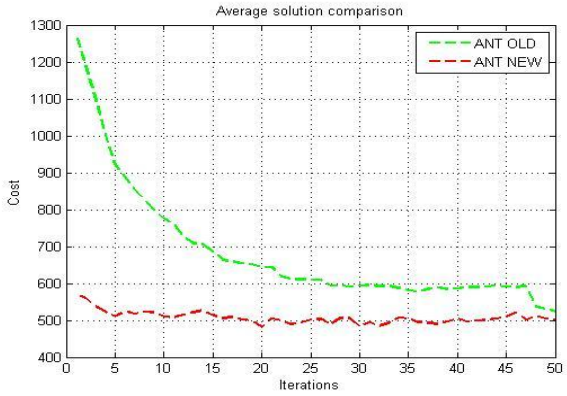


Fig. Comparison of Average solution for New and Old technique (30 Test Cases)

The exact values of best solution and best path values generated by the program are listed in the table below. As clearly seen, there is a significant difference in the best solution generated by the old technique and new technique (with a different heuristic function).

	OLD ACO	NEW ACO
Best Solution	478.6876	431.0884
Best Path Values	29,28,27,16,17,22,23,30,12,5,6, 2,18,19,20,21,10,3,13,4, 9,11, 8,14,15,24,25,26	10,21,20,19,11,7,8,14,15,24,25,26 29,28,27,16,17,22,23,30,12,13, 4, 6, 2, 1,18, 3, 9

Fig. Comparison of values of Best Solution and Best Path values (for 30 Test Case values)

Average Length and **Best Length:**

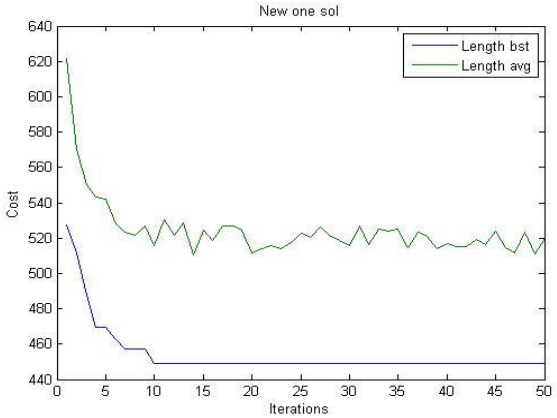


Fig. Values of Average Length and Best length generated by the new technique

Comparison of **Iterative Best CostAverage Node Branching**

Iterative best cost is a process of generating best cost path by cyclic process of prototyping, testing and refining a process. This process is intended to ultimately improve the quality and functionality of a design. In iterative design, interaction with the designed system is used as a form of research for informing and evolving a project, as successive versions, or iterations of a design are implemented.

In graph data structures, Node branching is given by the number of nodes connected to each node. If this value is not uniform, average node branching is used. Lower the **average node branching** factor more efficient is the algorithm.Higher average node branching factor results in more computational expensiveness and leading to combinatorial explosion. Combinatorial explosion is the effect of functions that grow very rapidly as a result of combinatorial considerations and results to excessive computation.

Comparison of factors using Efficiency and effectiveness of algorithms

Evaluation of test cases using old technique

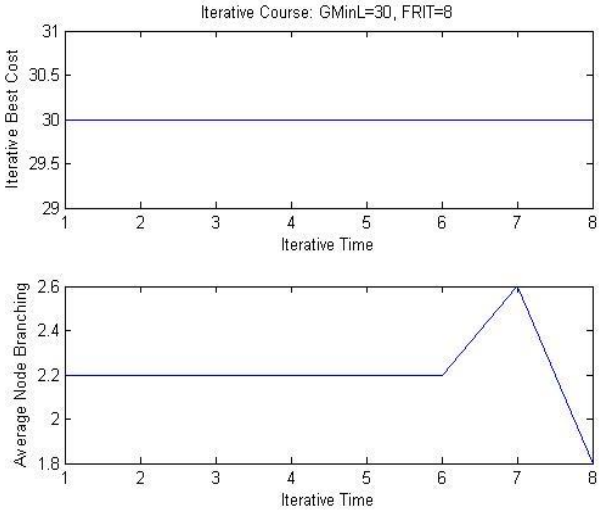


Fig: Iterative best cost and average node branching curve during evaluation of 30 test cases (old technique)

Evaluation of test cases using new technique

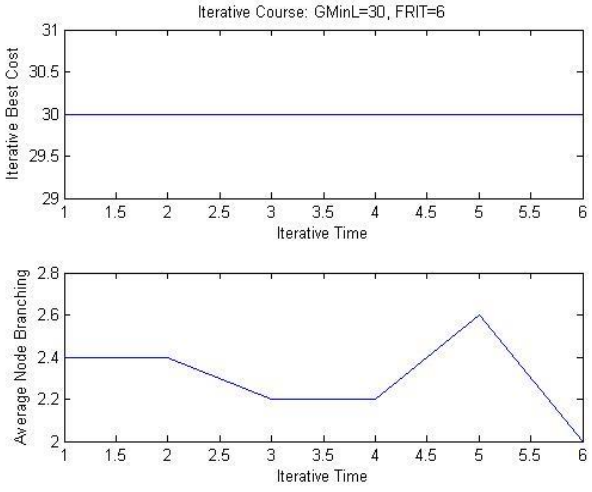


Fig: Iterative Best Cost and Average Node Branching curve during evaluation of 30 test cases (new technique)

It is clearly evident from the figures above that value of average node branching is pretty much reduced. The variation of curve with old technique was from factor of 2 to 2.6, but now the lower bound is decreased to 1.8 and hence the curve now vary from 1.8 to 2.6.Lesser the branching factor more is the efficiency. So, we can say that evaluating test suite by ant colony optimization with this new heuristic function is much efficient using this algorithm.

III. CONCLUSION

It is clearly proved now that this technique has a very high accuracy and efficiency for evaluating the test cases. We have analyzed the accuracy of algorithm using two different parameters that are precision and recall. This algorithm precision and recall value equals to 1.0, which is the maximum value for any algorithm. It means that this algorithm will generate 100% accurate results.

For analyzing efficiency of the algorithm we have considered the different parameters that are iterative best cost and average node branching. Iterative best cost and average node branching of this algorithm is much lower than the traditional approaches. So we can say that the proposed algorithm is 100% accurate and highly efficient for evaluation of test cases in test suite.

ACKNOWLEDGEMENT

We would like to acknowledge the authors ―Mustafa Muwafak Alobaedy‖ and ―Ku ruhana Ku-mahamud‘for their significant contribution in designing of this new heuristic function which we have applied in our technique.

REFERENCES

[1] Alobaedy, Mustafa Muwafak, and Ku Ruhana Ku-Mahamud, ―*New Heuristic Function in Ant Colony System forthe Travelling Salesman Problem ",* IEEE,Computing and Convergence Technology (ICCCT), 2012.

[2] MarcoDorigo, Mauro Birattari, and Thomas Stutzle. " Ant colony optimization ", IEEE, Computational Intelligence Magazine ,pp.28-39, 2013

[3] EricBonabeau, Marco Dorigo, Guy Theraulaz *"Swarm intelligence "*, MIT Press, 1999.

[4] J.L. Deneubourg, S. Aron, S. Goss, J.M. Pasteels , *" The self-organizing exploratory patternof the argentine ant"*, Journal of Insect Behavior, Vol. 3, No. 2, pp.159-168, 2000.

[5] Binder, R. V., Testing Object-oriented Systems: Models, Patterns, and Tools, Addison Wesley. 2000**.**

[6] Briand, L. C*., "On the many ways Software Engineering can benefit from Knowledge Engineering"*, Proc. 14th SEKE, Italy, pp. 3-6, 2002.

[7] AldeidaAleti, Lars Grunske, IndikaMeedeniya, Irene Moser*," Software Test SequenceOptimization Using Graph Based Intelligent Search Agent",* November 2009 pp. 505-509

[8] X.Zhang and T.Hoshino. *"A trail on model based test cases extraction and test data generation"*, in proceedings of third workshop on model based testing in practice, 2010.

[9] G.Singh, S.Gupta and B.Singh" *ACObased solution for tsp model forevaluation of software test suite"* in IAEME Volume 3, Issue 3, October - December (2012), pp. 75-82.

[10] McMinn, P., ― *Search-based Software Test Data Generation: A Survey"*, Software Testing, Verification and Reliability, Vol.14, No. 2, pp. 105-156, 2004.

[11] Ron Patton ―Software Testing‖ 2006.

[12] Joecolelifes' ―Software engineering‖ 2007.

[13] Blum C., Roli A.(2003) ―*Metaheuristics in combinatorial optimization: overview and conceptual comparison"*, ACM Computing Surveys 35 (3), pp.268–30

[14] A. Aljanaby,K.R Ku-Mahamud and N.M Norwawi. *"Optimizing Large scale combinatorial problems usingmultiple ant colony algorithms based on pheromone evaluation techniques",* International Journal of computerscience an network security, vol 8(10), p 54-58, 2008

[15] "Evaluation of Test Cases Using Ant Colony Optimization with a New Heuristic Function: A Proposed Approach" Published in "International Journal of advanced research in computer science and software engineering "Gulwatanpreet Singh**,** Poonam Sharma, Parveen Kakkar. Volume 4, Issue 8, August 2014 ISSN: 2277 128X

