An Efficient Ant Colony System For Coverage Based Test Case Prioritization

Chengyu Lu, Jinghui Zhong(Corresponding Author)
School of Computer Science and Engineering
Guangzhou, China
jinghuizhong@gmail.com

ABSTRACT

Test case prioritization which aims to improve the efficiency of regression testing by ordering test cases is an active research topic that has attracted increasing attention recently. This paper proposes an efficient Ant Colony System(ACS) framework to solve the coverage based test case prioritization problem. A tour based heuristic function and a tree shape pheromone updating rule are designed. The underlying rationale is to make full use of the ants' partially built solutions in their later traveling and to vary their search at the same time. The proposed method is tested on six benchmark problems with different scales, and the experimental results have demonstrated that the proposed method can outperform several state-of-the-art methods in terms of the Average Percentage of Statement Coverage(APSC).

KEYWORDS

Ant Colony System, Regression Test, Test Case Prioritization, Statement Coverage

ACM Reference Format:
Chengyu Lu, Jinghui Zhong(Corresponding Author). 2018. An Efficient Ant Colony System For Coverage Based Test Case Prioritization. In *GECCO '18 Companion: Genetic and EvolutionaryComputation Conference Companion, July 15–19, 2018, Kyoto,Japan.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3205651.3205680

1 INTRODUCTION

Regression Testing is a type of software testing which aims to ensure that any modifications should neither introduce new bugs in the inserted code nor affect the software's formerly normal functions. It is performed throughout the entire life cycle of the software and new test cases generated in a single run will be stored and reused in the subsequent testing. For this reason, the cost of performing regression testing could be very expensive[3].

Various methods have been proposed to reduce the cost of regression testing, among which test case prioritization(TCP) is one of the most widely used [3]. TCP techniques sort test cases in the test suite in order to have those cases with higher priority executed earlier. In this way the performance goal of regression testing are hopefully met faster and the previous execution time would be spent the most cost-worthy in case of unexpected termination. Existing methods on TCP problems include greedy strategies[4], Genetic Algorithm(GA) [4, 5] and Swarm Intelligence(SI) [2, 6].

GECCO '18 Companion, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-1-4503-5764-7/18/07.
https://doi.org/10.1145/3205651.3205680

Maximizing the rate of statement coverage is one of the most common objectives of TCP techniques[4], which aims to cover as many executable lines of code as possible at a quicker speed. The TCP problem with the objective of maximizing the rate of statement coverage is known to be NP-hard and intractable for traditional deterministic methods[4].

Ant Colony Optimization(ACO) is a powerful swarm intelligence algorithm that has been successfully applied to a wide range of NP-hard optimization problems [1]. In the literature, several ACOs have been proposed to solve TCP [2, 6]. However, existing ACOs focus only on designing pheromone updating strategies but ignore the heuristic function, which makes them not effective enough to solve TCP mentioned above. In this paper, an efficient ACO framework with both a highly efficient heuristic function and a tree-based pheromone updating rule is proposed to solve TCP.

2 PROBLEM DEFINITION

Rothermel et al. [3] formally defined the TCP problem in regression testing as follows:

Definition 2.1. The Test Case Prioritization Problem:
Given: T, a test suite, PT, the set of permutations of T, and f, a function from PT to real numbers.
Problem: Find $T^{'} \in PT$ such that $(\forall T^{''})\ (T^{''} \in PT)\ (T^{''} \neq T^{'})\ [f(T^{'}) \geq f(T^{''})]$

PT denotes the set of all possible ordering of T and f is a function that, applied to any such ordering, yields an award value. Proposed by Li et al. [4], the Average Percentage of Statement Coverage(APSC) metric is employed as f:

$$APSC\left(T^{'}\right) = 1 - \frac{TS_1 + TS_2 + \cdots + TS_m}{mn} + \frac{1}{2n} \tag{1}$$

where m is the number of statements in the tested program, n is the number of test cases in T, and TS_i is the first test case in $T^{'}$ that covers statement i.

2.1 Proposed Algorithm

The proposed algorithm is designed based on the Ant Colony System (ACS) [1] and is named the Coverage Based ACS(CB-ACS). Specifically, a test suite $T = \{t_1, t_2, ..., t_n\}$ containing n test cases that covers a total of m statements in a tested program is coverted to a graph $G = \{V, E\}$. V is the set of vertices containing $n + 1$ elements in total, with n real vertices mapping to the n test cases and one virtual vertex v_{-1} serves as a virtual starting vertex for all ants. Ants are allowed to stop wherever full statement coverage is achieved, unnecessary to traverse all the vertices.

An artificial ant k maintains its own covered statement set Sa_k, in which statements that have already been covered by itself are recorded. Denoted by η the heuristic function, the heuristic value

for ant k from vertex v_r to v_s is defined as:

$$\eta_k(r,s) = |(S \setminus Sa_k) \cap Sv_s| \tag{2}$$

where Sv_s is the covered statement sets of vertex v_s, S is the set of all statements, $S \backslash Sa_k$ obtains the complementary set of Sa_k within S, and $|\cdot|$ counts the number of elements in the input set.

Every artificial ant sees G based on which and how many statements it has already covered. Ant k located at r considers any other candidate vertex s to be connected if and only if there are some additional statements in s. The more additional statements a vertex contains, a larger heuristic value it yields and more preferable it is by k. In essence, the topology of the graph in an ant's eyes changes whenever it makes a transition and one graph usually has different topology in two ants' eyes.

Different solutions with identical *APSC* values are regarded as equivalent solutions. In CB-ACS, all the equivalently optimal solutions form a global best tour tree T_g. During every iteration, the iteration-best tours that are equivalent to the global-best tours are added to T_g. If they yield an even larger *APSC*, the original tree will be deleted and they become the new T_g.

Additional Greedy strategy always selects test cases with the largest number of additional statements. In this paper a Tree-Based Layer-Wise Additional Greedy(TBLW-ADG) strategy is proposed both to find the first T_g (i.e., T_{g_0}) and to guide the ants' searches in the initialization, ensuring that all nodes at the same layer of the tree yield the same layer-largest additional statement gain. Denoted by $Depth(T_{g_0})$ the depth of tree T_{g_0}, the original pheromone concentration τ_0 on graph G is initialized as the ratio of *APSC* of T_{g_0} to its depth.

The global pheromone updating(GPU) rule updates the pheromone level on G during every iteration, which is defined as follows:

$$\tau(r,s) \leftarrow (1-\alpha)\cdot\tau(r,s) + \begin{cases} \frac{2\alpha \cdot APSC(T_g)}{MaxChild_{T_g}(r)+MinChild_{T_g}(r)}, & if\ (r,s) \in T_g \\ 0, & otherwise \end{cases} \tag{3}$$

There may be several nodes in different branches in T_g that represent the same vertex r in G. The *MaxChild* function returns the maximal child number of tree nodes representing r while *MinChild* on the other hand returns the minimal. $\alpha \in (0,1)$ is the pheromone decay parameter indicating the rate of evaporation. By means of T_g, ants are encouraged to search on a broader neighborhood of the known optimal paths.

3 EXPERIMENTAL RESULTS

The proposed CB-ACS is experimentally studied and compared with three other algorithms, which are Genetic Algorithm(GA) [4], the additional greedy strategy(AG) [4] and the traditional ACO(T-ACO) [6]. The widely used Siemens programs [3] consisting of six tested programs was adopted. Test suites for each programs are assembled from their corresponding test pools into two granularities: the small size and the large size [4], with 100 suites for each granularity. The numeric parameters for the CB-ACS are set as follows: $\beta = 2$, $q_0 = 0.9$, $\alpha = \rho = 0.1$. The number of ants are 16 and the maximal repeating time is 150. The comparison results are listed in table 1, where +, -, and * means that the performance of the competitors are significantly better than, worse than and similar to CB-ACS under Wilcoxon test with $\alpha = 0.05$.

Table 1: Experimental Results: Average APSC (*100%)

P	test suite	AG	GA	T-ACO	CB-ACS
Rep.	S	93.9699-	93.9661-	93.1736-	**94.0439**
	L	99.7573-	99.3692-	99.3638-	**99.7621**
Pri.2	S	92.5324-	92.5639-	91.8178-	**92.5824**
	L	99.8199-	99.5439-	99.6499-	**99.8202**
Pri.	S	94.6878-	94.6726-	93.9962-	**94.7125**
	L	99.8375-	99.4253-	99.4820-	**99.8385**
Tot.	S	90.8710-	**90.9055***	90.6269-	**90.9067**
	L	98.1340-	98.0420-	97.8944-	**98.1432**
Sch.	S	92.3179-	**92.3229***	92.1507-	**92.3229**
	L	**99.8012***	99.6425-	99.7245-	**99.8012**
Sch.2	S	92.3896-	92.4193-	92.1329-	**92.4219**
	L	99.7587-	99.6564-	99.7261-	**99.8092**

It can be observed that CB-ACS can achieved significantly better or at least competitive performance on all tested programs, in terms of APSC. With the help of TBLW-ADG, the solutions produced by CB-ACS are guaranteed to be no worse than AG. Compared with GA and T-ACO, the proposed GPU-rule helps individuals search on a broader neighborhood of the known best paths while the new heuristic function makes them stick to the coverage objective when selecting test cases. The above results indicate the high search efficiency of the proposed CB-ACS.

4 CONCLUSIONS

This paper proposes a new ACS framework to solve the TCP problem with an objective to maximize the rate of statement coverage. Experiments on benchmark TCP problems have shown that the proposed method can offer very promising performance in comparison with several state-of-the-art methods. In future we will try to apply the proposed work to multiobjective TCP problems, e.g., to maximize the rate of statement coverage while minimize the effective execution time of the regression testing simultaneously.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (Grant No. 61602181) and the Fundamental Research Funds for the Central Universities (Grant No. 2017ZD053).

REFERENCES

[1] M. Dorigo; L. M. Gambardella. 1997. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE Trans. on Evol. Compt.* 1, 1 (1997), 53–66.
[2] Yi Bian; Zheng Li; Ruilian Zhao; Dunwei Gong. 2017. Epistasis Based ACO for Regression Test Case Prioritization. *IEEE Trans. on Emerging Topics in Computational Intelligence* 1, 3 (2017), 213–223.
[3] G. Rothermel; R. H. Untch; Chengyun Chu; M. J. Harrold. 2001. Prioritizing Test Cases for Regression Testing. *IEEE Trans. on Soft. Eng.* 27, 10 (2001), 929–948.
[4] Zheng Li; Mark Harman; Robert M. Hierons. 2007. Search Algorithms for Regression Test Case Prioritization. *IEEE Trans. on Soft. Eng.* 33, 4 (2007), 225–237.
[5] P. Konsaard; L. Ramingwong. 2015. Total Coverage Based Regression Test Case Prioritization Using Genetic Algorithm. In *Procedings of the 2015 International Conference on Electrical Engineering/electronics, Computer, Telecommunications and Information Technology*. IEEE, 1–6.
[6] Y. Singh; A. Kaur; B. Suri. 2010. Test case prioritization using ant colony optimization. *ACM SIGSOFT Software Engineering Notes* 35, 4 (2010), 1–7.