



## Interacted Multiple Ant Colonies Optimization Approach to Enhance the Performance of Ant Colony Optimization Algorithms

Alaa Aljanaby (corresponding author)

Graduate Department of Computer Science

College of Arts and Sciences, Universiti Utara Malaysia

06010 Sintok, Kedah, Malaysia

E-mail: alaa.aljanaby@gmail.com

Ku Ruhana Ku-Mahamud

Graduate Department of Computer Science

College of Arts and Sciences, Universiti Utara Malaysia

06010 Sintok, Kedah, Malaysia

E-mail: ruhana@uum.edu.my

Norita Md. Norwawi

Faculty of Science and Technology

Universiti Sains Islam Malaysia

71800 Nilai, Bandar Baru Nilai, Malaysia

E-mail: norita@usim.edu.my

### Abstract

One direction of ant colony optimization researches is dividing the ants' population into several colonies. These colonies work together to collectively solve an optimization problem. This approach offers good opportunity to explore a large area of the search space. This paper proposes a new generic algorithmic approach that utilized multiple ant colonies with several new interaction techniques. Computational test shows promising results of the new approach. The proposed approach outperforms the single colony ant algorithms in term of solution quality with the same computational effort.

**Keywords:** Ant colony optimization, Combinatorial optimization problems, Stagnation problem

### 1. Introduction

One of the successful applications of swarm intelligence is the application of Ant Colony Optimization (ACO). Swarm intelligence is a field of artificial intelligence that studies the intelligent behavior of groups such as the behavior of natural systems of social insects like ants, bees, wasps, and termites. ACO is inspired from the foraging behavior of real ants. Using a combination of priori information (heuristics) about the candidate solutions quality of and posteriori information (pheromone) about the goodness of the previously obtained solutions are the key elements of ACO success (Dorigo & Stützle, 2002; Blum, & Dorigo, 2005; Dorigo Birattari, & Stützle, 2006).

The problems tackled by ACO are called combinatorial optimization problems. These complex problems appear when the task is to get the best solution out of many possible solutions to a given problem. Traveling salesman problem (TSP), vehicle routing problem, quadratic assignment problem (QAP), sequential ordering problem and network routing problem are some well known examples of these problems (Dorigo & Stützle, 2002; Blum & Roli, 2003).

Ant System (Dorigo, Maniezzo, & Colorni, 1996), Ant Colony System - ACS (Dorigo & Gambardella, 1997), Max-Min Ant System - MMAS (Stützle & Hoos, 2000), Ranked Ant System - RAS (Bullnheimer, Hartl, & Strauss, 1999) and Best Worst Ant System - BWAS (Cordon, Fernandez, Herrera, & Moreno, 2000) are several well known ACO

algorithms. These algorithms show interesting performance but are still far from being ideal. These algorithms can get a good solution at the early stages of the algorithm execution. However all ants speedily converged to a solution and the algorithm is unable to improve that solution. This is a common problem that all ACO algorithms suffer from regardless of the application domain. This is called search stagnation problem and the chance of stagnation proportionally increases with the increase of the problem size.

Multiple Ant Colonies Optimization (MACO) is an ongoing direction of ACO researches that aims to improve the performance of ACO algorithms. In this approach several colonies of ants are cooperatively working to solve a combinatorial optimization problem (Jong & Wiering, 2001; Kawamura, Yamamoto, Suzuki, & Ohuchi, 2000). MACO increases the chance of ACO algorithms to explore a large area of the search space and hopefully find (near-) optimal solution.

This paper considers the enhancement of ACO solution by proposing a new algorithmic approach that utilizes multiple ant colonies with certain techniques to organize the activities of these colonies. The rest of this paper is organized as follows. Section 2 reviews the related literature while section 3 proposes the new algorithmic approach. The computational results are presented in section 4. Concluding remarks and suggested future work are presented in the final section.

## 2. Related work

Jong and Wiering (2001) proposed a multiple ant colonies system for bus-stop allocation problem. In this problem there are  $n$  bus-stops and  $m$  bus-lines. A solution is to build  $m$  bus-lines, each one consisting of a sequence of  $n$  bus-stops that minimizes the average travel time. The results of the new algorithm were better than the results obtained from other meta-heuristic algorithms like greedy algorithm and simulated annealing.

Kawamura et al. (2000) proposed a MACO algorithm based on colony level interaction. A colony effects other colonies and these effects are different from one colony to another. These effects are determined by using an array  $M \times M$  parameters, where  $M$  is the number of colonies. No exact way of choosing this large number of parameters was shown. The algorithm was tested on some TSP instances and the results were comparable with AS results but cannot outperform the results of the best known ant algorithms like ACS and MMAS.

Middendorf, Reischle, & Schmeck (2002) proposed the idea of using several ant colonies parallelized over several processors. The parallelized multiple colonies have to exchange information after completing a certain number of iterations. They tested their algorithm on one TSP instance and one QAP instance. The results showed that the algorithm with moderate number of colonies gave better results than AS for the TSP while the results for the QAP instance was worst than the results of ACS. Just like Kuwamura et al. (2000) the results of Middendorf et al. (2002) were also not compared with the results of ACS and MMAS. In fact the results presented in both papers were worst than those of ACS and MMAS.

## 3. The proposed interacted multiple ant colonies approach

The proposed algorithmic approach contains three mechanisms that are used to organize the work of the individuals in each colony and the work of all colonies. In other words there are two levels of interaction. The first is the colony level and the second is the population level.

The colony level interaction can be achieved through the pheromone depositing process within the same colony. The pheromone updating mechanism is responsible for the implementation of this kind of interaction. The population level interaction is achieved by evaluating the pheromones of different colonies using some evaluation function. The important aspect here is the pheromone evaluating mechanism. The degree of interaction of the different colonies is the role of the exploration / exploitation mechanism. This algorithmic approach will be called hereafter Interacted Multiple Ant Colonies Optimization (IMACO).

The work activities of a single colony in the proposed IMACO algorithm are based on ACS. Each colony has its own pheromone that is used as the interaction mechanism between the ants of the same colony. The interaction between ant colonies using pheromone can be organized in different terms. The IMACO algorithm is described as follows.  $M$  colonies of  $m$  ants each are working together to solve some combinatorial problem. The probabilistic decision of the ant  $k$  belongs to the colony  $v$  to move from node  $i$  to node  $j$  is defined as:

$$j = \begin{cases} \arg \max_{l \in N_i^{kv}} \{f(P_{il}) H_{il}^\beta\} & \text{if } q \leq q_0 \\ S & \text{otherwise} \end{cases} \quad (1)$$

where  $f(P_{ij})$  is the evaluation function of pheromone on the edge  $(i,j)$  and  $H_{ij}$  is the problem dependent heuristic.  $N_i^{kv}$  is the set of remaining nodes to be visited by the  $k^{\text{th}}$  ant of colony  $v$ .  $\beta$  is a parameter that determines the relative

importance of pheromone versus heuristic,  $q$  is a random variable distributed in  $[0, 1]$  and  $q_0$  is a parameter and  $0 \leq q_0 \leq 1$ .  $S$  is a random variable selected according to the following probabilistic rule.

$$S = \begin{cases} \frac{f(P_{ij}) H_{ij}^\beta}{\sum_{l \in N_i^k} f(P_{il}) H_{il}^\beta} & \text{if } j \in N_i^{kv} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

### 3.1 Pheromone evaluation mechanism

The proposed pheromone evaluation mechanism evaluates the pheromone as an average of the pheromone values of all colonies on some edge. This means that an ant will make its decision to choose an edge based on the average of the available experiences of ants of all colonies that visited this edge in the past. This variant of IMACO is referred hereafter as IMACO-AVG.

Given that for each edge there are  $M$  pheromone values each belongs to a single colony. Average pheromone evaluation function evaluates the pheromone on any edge as an average of the available  $M$  values. The pheromone evaluation function for IMACO-AVG is defined as:

$$f(P_{ij}) = \frac{\sum_{v=1}^M P_{ij}^v}{M} \quad (3)$$

where  $P_{ij}^v$  is the pheromone of colony  $v$  on the edge  $(i, j)$ .

### 3.2 Exploration / exploitation control mechanism

Each ant makes a probabilistic decision when it needs to move to a new node. The probabilistic decision is based on heuristic information (cost) and pheromone information. Pheromone represents information about previous experiences of the ant's own colony and of the other colonies. While heuristic represent a priori information about the goodness of a solution. The relative importance of heuristic and pheromone information is determined by using the weighting parameter  $\beta$ .

Another parameter  $q_0$  is usually used in ant's probabilistic decision as trade-off between exploitation (choosing the edge with the higher value of the multiplication of pheromone and heuristic values) and exploration (choosing the edge randomly according to some probability distribution).

In this paper  $\beta$  and  $q_0$  are set to 2 and 0.9 respectively, these values are commonly used in ACS single colony algorithm. This study conduct experiments on cases using different values for the ant colonies for these two parameters. The goal of these tests is to reach a balanced form of exploitation / exploration that yields to the best algorithm performance.

### 3.3 Pheromone updating mechanism

The proposed pheromone updating mechanism encourages a balanced form of exploitation of previous experiences and exploration of new or improved paths. Basically the mechanism incorporates global and local pheromone updating. Local pheromone update encourages the exploration of new areas of the search space by reducing the importance of the visited edges. The global pheromone update encourages the exploitation of previously good solutions by giving extra weight to the edges of global best solutions.

Global pheromone updating includes that best ant of each colony deposits an amount of pheromone on its own path. After all ants of all colonies complete their tours (i.e., one algorithm iteration), the ant that finds the so far best solution in its colony will be allowed to deposit an amount of the colony's pheromone on the edges of its tour according to the following rule:

$$P_{ij}^v = (1 - \sigma)P_{ij}^v + \sigma\Delta P_{ij}^{v.bs} \quad (4)$$

where  $\sigma$  is the trail evaporation such that  $(1 - \sigma)$  represents the pheromone persistence. This parameter is used to avoid unlimited accumulation of pheromone trails and allows the algorithm to forget previously done bad choices.  $\Delta P_{ij}^{v.bs}$  is the pheromone quantity added to the connection  $(i, j)$  belonging to the best solution of  $v^{\text{th}}$  colony  $L^{v.bs}$  and is given by:

$$\Delta P_{ij}^{v.bs} = \begin{cases} 1/L^{v.bs} & \text{if } (i, j) \text{ belongs to the} \\ & \text{best tour of colony } v \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

The best solution used in the updating mechanism in ACS is the global best solution. This work considers the idea of using a combination global and iteration best solution. Using the iteration best solution after a certain number of using the global best solution in the pheromone updating mechanism helps to make a search diversion by directing the search process to different good solutions which avoids the quick convergence to a single solution at the early stages of the algorithm run.

This mechanism was implemented individually for each colony. The general updating policy used in this research is  $xG$   $II$ , i.e., one update will be based on iterative best solution after  $x$  updating times of using global best solution. In this paper the results reported for  $x=25, 50$  and  $75$  along with the basic result without using this proposed updating policy.

Local pheromone update is then applied by each ant on the visited edges. It includes that each ants reduces the amount of pheromone on paths it uses in order to give a better chance to other paths to be chosen by future generations. It is a very important rule as it is performed during the solution construction which helps to yield different pheromone evaluation values for the same edge in the same iteration at different solution construction steps. The local pheromone update is given by:

$$P_{ij}^v = (1 - \gamma) P_{ij}^v + \gamma P_0 \quad (6)$$

where  $\gamma$  is another pheromone evaporation parameter and  $P_0$  is the initial pheromone value.

#### 4. Computational result

IMACO-AVG has been tested using lin318 TSP benchmark instances taken from TSP library (TSPLIB, 1995). The optimal solution for lin318 is 42029. Experiments were run using 1, 2, 3, 4, 5, 7, 9, 12, 15, 20 and 30 colonies where each colony has 10 ants. The results are averaged over 10 trials with 10000 iterations per trial. This is based on the assumption that the algorithm ran 3000 iterations for each 100 nodes of the problem instance. The parameter setting are  $\beta=2$ ,  $\sigma = \gamma = 0.1$  and  $q_0 = 0.9$ . The heuristic function used for TSP is the inverse of the distance, i.e.,  $H_{ij}=1/d_{ij}$ .

Figure 1 shows the results of applying ACS, and IMACO-AVG lin318. For comparison purposes, ACS was run using 10, 20, 30, 40, 50, 70, 90, 120, 150, 200 and 300 ants. Note that since ACS has single ant colony, so in Figure 1 the number of ants used by ACS is equal to the number of colonies multiplied by 10. As shown in this figure, the results of ACS deteriorate as the number of utilized ants increases. This means that ACS cannot benefit from the increase in the number of utilized ants because the algorithm always get trapped in stagnation situation and cannot improve the solution quality. Better results are obtained when the number of ants is in the range of 20 to 30.

It is obvious that IMACO-AVG algorithm obtained better results in term of the overall average solution. The superior of IMACO-AVG is clear as this algorithm shows a stabilizing performance using the increased number of colonies. The average pheromone evaluation technique was a successful organizing technique of the ants' activities up to 30 utilized colonies.

Figure 2 shows the result of using IMACO-AVG with different pheromone updating techniques. A better result is obtained when updating the pheromone based on iterative-best solution once after each 50 times of pheromone updating based on global best solution.

Figure 3 shows a comparison between the trial average time of IMACO-AVG and ACS with 11 runs for each algorithm with a specified number of ants/colonies. The computational average time of both versions of IMACO was close to that of ACS when the number of colonies utilized is 10 or less. However, the difference between ACS and IMACO average time starts increasing when the number of utilized colonies was beyond 10. Based on these results, the rest of the experiments focus on testing the different versions of IMACO with no more than 10 colonies.

Experiment was also conducted using IMACO-AVG 50G 11 with 9 colonies on different symmetric and asymmetric instances of TSP. The results are reported in Table 1 which depicts the comparison of ACS and MMAS. In this table, the optimal solution for each TSP instance is shown in each column header under the instance name and all reported results of all algorithms represent the overall average solution. The results of ACS and MMAS are taken from the literature (Stützle & Hoos, 2000). The number of iteration IMACO-AVG ran on each instance was according to the assumption made by Stützle and Hoos (2000) which is equal to  $k*10000*$ no of nodes/no of ants where  $k=1$  for symmetric instances and  $k=2$  for asymmetric instances.

The results show that IMACO-AVG outperformed ACS for all instances. In comparing IMACO-AVG with MMAS (which is the best known ant algorithm) for the first small instances, MMAS perform better. However, the results of IMACO-AVG were very close. For other bigger instances where the chance of stagnation increases IMACO-AVG outperformed MMAS for giving the best average solution.

#### 5. Conclusion and future work

IMACO, the new algorithmic approach divides the ants' population into several colonies and effectively coordinates

their works. The results show that IMACO-AVG can outperform the ACS algorithm with similar number of ants. IMACO-AVG has been further tested on different TSP and compared with ACS and MMAS and its superior performance was obvious. Even though IMACO is computationally more expensive than other ACS and MMAS, it is still within the same computational class when number of colonies is ten or less.

Testing new pheromone evaluation mechanisms is a possible future direction. Another interesting future work is testing different values for the parameters involved in the exploration / exploitation control mechanism and investigating the best range for each parameter.

## References

- Blum, C. & Dorigo, M. (2005). Search bias in ant colony optimization: On the role of competition-balanced systems. *IEEE Trans. on Evolutionary Computation*, 9(2), 159-174.
- Bullnheimer, B., Hartl, R. F., & Strauss, C. (1999). A new ranked-based version of the ant system: A computational study. *Central European Journal for Operations Research and Economics*, 7(1), 25-38.
- Cordon, O., Fernandez, I., Herrera, F., & Moreno, L. (2000). A new ACO model integrating evolutionary computation concepts: The Best-Worst Ant System. In M. Dorigo, M. Middendorf, & T. Stützle (Eds.), *proceedings of ANTS2000-From Ant Colonies to Artificial Ants: A Series of International Workshops on Ant Algorithms* (pp. 22-29). IRIDIA, Université Libre de Bruxelles, Brussels, Belgium.
- Dorigo, M. & Stützle, T. (2002). The Ant Colony Optimization Metaheuristic: Algorithms, Applications, and Advances. In: F. Glover and G. Kochenberger (Eds.), *Handbook of Metaheuristics*. Kluwer Academic Publishers.
- Dorigo, M., & Gambardella, L. (1997). Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Trans on Evolutionary Computation*, 1(1), 53-66.
- Dorigo, M., Birattari M. & Stützle T. (2006). Ant Colony optimization: Artificial Ants as Computational Intelligence Technique. *IEEE Computational Intelligence Magazine*, Nov. 2006.
- Dorigo, M., Maniezzo, V., & Colormi, A. (1996). The ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 26(1), 29-41.
- Jong, J., & Wiering, M. (2001). Multiple ant colony system for the bus-stop allocation problem. In *Proceedings of the Thirteenth Belgium-Netherlands Conference on Artificial Intelligence (BNAIC'01)*, 141-148.
- Kawamura, H., Yamamoto, M., Suzuki, K. & Ohuchi, A. (2000). Multiple ant colonies algorithm based on colony level interactions. *IEICE Trans. Fundamentals*, E83-A(2).
- Middendorf, M., Reischle, F., & Schmeck, H. (2002). Multi Colony Ant Algorithms. *Journal of Heuristics* (special issue on Parallel Meta-heuristics), 8(3): 305-320.
- Stützle T., & Hoos H. (2000). Max-Min Ant System. *Journal of Future Generation Computer Systems*, 16, 889-914.
- TSPLIB(1995). <http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/tsp/>.

Table 1. Results Comparison for Symmetric and Asymmetric TSP Instances

Algorithm	TSP instance						ATSP instance			
	<i>kroA100</i>	<i>d198</i>	<i>lin318</i>	<i>att532</i>	<i>rat783</i>	<i>fl1577</i>	<i>ry48p</i>	<i>ft70</i>	<i>kro124</i>	<i>ftv170</i>
	<i>21282</i>	<i>15780</i>	<i>42029</i>	<i>27686</i>	<i>8806</i>	<i>22137</i>	<i>14422</i>	<i>38673</i>	<i>36230</i>	<i>2755</i>
ACS	21420.0	16045	43296.85	28522.8	9066.0	23136.0	14565.4	39099.0	36857.0	2826.5
MMAS	21291.6	15956.8	42346.60	28112.6	8951.5	NA	14523.4	38922.7	36573.6	2817.7
IMACO-AVG	21290.6	15953.5	42341.6	28075.7	8932.8	22520.3	14491.6	38871.4	36489.2	2791.7

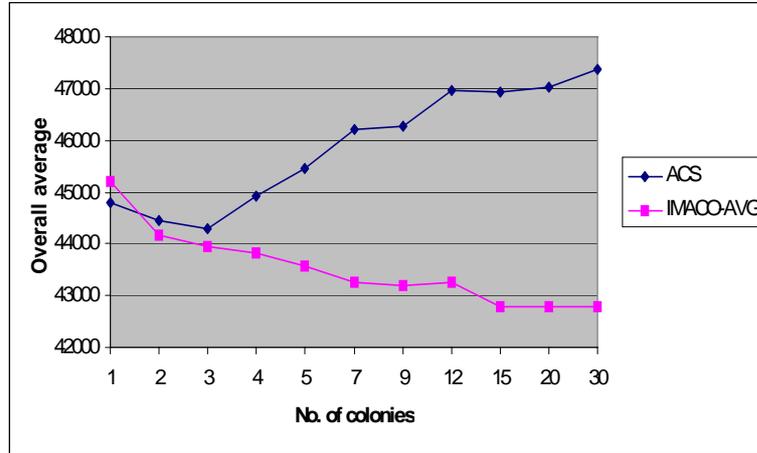


Figure 1. ACS and IMACO-AVG performance comparison

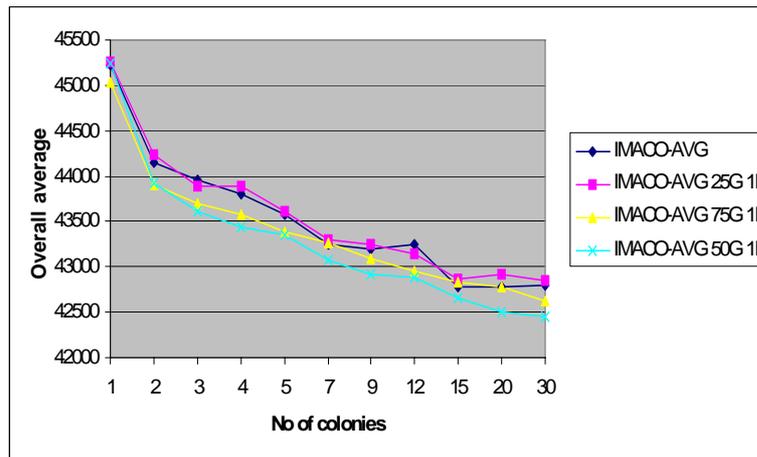


Figure 2. Different IMACO-AVG variants performance comparison

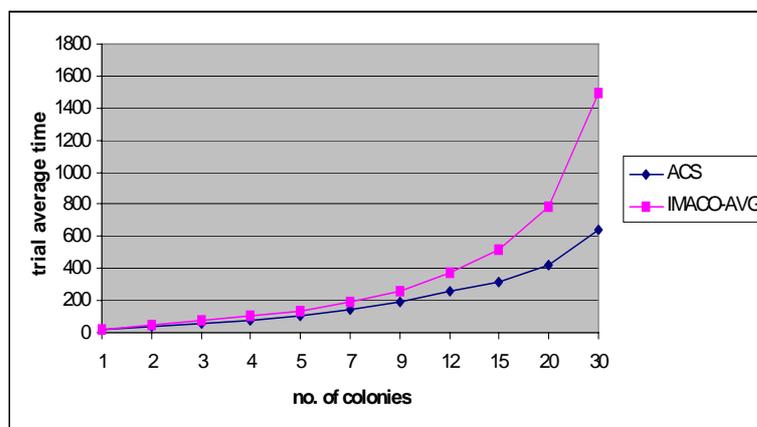


Figure 3. ACS and IMACO-AVG trial average time Comparison