Optimizing Large Scale Combinatorial Problems Using Multiple Ant Colonies Algorithm Based on Pheromone Evaluation Technique

Alaa Aljanaby[*], Ku Ruhana Ku Mahamud[**], and Norita Md. Norwawi[**]

[*]Computer Science Dept., College of Art and Science, University of Nizwa, Oman
[**]Faculty of Information Technology, University Utara Malaysia (UUM), Malaysia

Abstract
The approach of using multiple ant colonies is an extension of the Ant Colony Optimization framework. It offers a good chance to improve the performance of the ant algorithms by encouraging the exploration of a wide area of the search space without losing the chance of exploiting the history of the search. In this paper a new multiple ant colonies optimization algorithm is proposed. The new algorithm is based on the ant colony system and utilizes average and maximum pheromone evaluation mechanisms. The new algorithm can effectively be used to tackle large scale optimization problems. Computational tests show promises of the new algorithm.

Keywords:
Ant Colony Optimization, Meta-heuristic Algorithms, Combinatorial optimization problems.

1. Introduction

Ant colony optimization (ACO) is a recent family member of the meta-heuristic algorithms and can be used to solve complex optimization problems with few modifications by adding problem-dependent heuristics [8]. ACO is a biological inspiration simulating the ability of the real ant colony of finding the shortest path between the nest and the food source. The main element of ACO success is the use of a combination of priori information (heuristics) about the quality of candidate solutions (also called greedy strategy) and posteriori information (pheromone) about the goodness of the previously obtained solutions.

The class of complex optimization problems that can be tackled by ACO called combinatorial optimization problems. Traveling salesman problem (TSP), quadratic assignment problem, vehicle routing problem, job secluding problem and network routing problem are some examples of these problems [1]. These problems arise when the task is to find the best out of finite but huge number of possible solutions to a given problem. Because of the large number of candidate solutions, using exact algorithm to enumerate all solutions is practically impossible [2]. The only way to tackle these problems is to use a heuristic search algorithm that tries to find an optimal or near-optimal solution in a suitable time amount.

Ant System [6], Ant Colony System [6], Max-Min Ant System [13], Ranked Ant System [3] and Best Worst Ant System [4] are well known ACO algorithms. These algorithms show interesting performance and are competitive with other state of the art optimization methods. However, more research work is needed to enhance the ACO algorithms performance especially on large scale optimization problems. Increasing the number of ants used to tackle a large problem almost yield to a worse algorithm performance.

In this paper, a new ACO meta-heuristic algorithm is proposed. The new algorithm utilizes multiple ant colonies and can efficiently be used to tackle large scale optimization problems. The rest of this paper is organized as follows. Section 2 reviews the available research works that considered the use of multiple ant colonies optimization. Section 3 proposes the new algorithm. The computational results of applying the proposed algorithm on two TSP benchmark instances are presented in section 4. Section 5 concludes and suggests the future work.

2. Multiple Ant Colonies Optimization Approach

Multiple ant colonies optimization (MACO) is an extension of the ACO framework where a number of ant colonies working together to solve some combinatorial optimization problem. Gambardella et al. [9] proposed multiple ant colonies system for the vehicle routing problem with time windows. The algorithm has been designed to solve vehicle routing problems with two objective functions which are the minimization of the number of tours (or vehicles) and the minimization of the total travel time, where number of tours minimization takes precedence over travel time minimization. The basic idea is to coordinate the activity of different ant colonies, each of them optimizing a different objective.

Manuscript received October 5, 2008
Manuscript revised October 20, 2008

The results of this approach have shown to be competitive with the existing methods.

A similar approach has been used by Jong and Wiering [11] to tackle bus-stop allocation problem with n bus-stops and m bus-lines. A solution is to construct 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 outperformed the results obtained from greedy algorithm and simulated annealing.

A MACO algorithm based on colony level interaction has been proposed by Kawamura et al. [10]. The algorithm used large number of parameters that must be set in advance. These parameters determine the effect of each colony to all other colonies and they organized as an array of size M×M, where M is the number of colonies. No specific way of choosing this large number of parameters was shown. The effect of a colony towards another colony may be positive or negative. Different colony structures were tested with some parameter setting. The algorithm tested on some TSP instances and the results were better than AS results but did not compare with the best performing ACO algorithms like ant colony system.

Sim and Sun [12] propose some conceptual ideas of MACO approach as a new ACO framework for network routing problem. The authors believe that using multiple ant colonies to explore the network offers the opportunity to find new and better paths and reduces the chance of stagnation. However, the authors think that this approach offers a new direction for ant-based optimization in general and for network routing problem in specific.

3. The Proposed Algorithm

The proposed algorithm is based on the ant colony system [6] which considered being one of the best performing ant algorithms. Number of ant colonies is used by the new algorithm; each colony has its own pheromone that is used as an interaction between the ants of the same colony. The interaction between ant colonies using pheromone can be organized in different terms. Two kinds of pheromone interaction (evaluation) are proposed in this paper. The first one is evaluating 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 some edge based on the average of the available experiences of ants of all colonies that visited this edge in the past. This variant of MACO is referred hereafter as MACO-AVG. The other mechanism evaluates the pheromone as the maximum value of the pheromone values of all colonies on some edge. In this variant, referred as MACO-MAX, an ant's decision to choose some edge will be based on the best available experience of ants of all colonies that previously visited this edge.

The MACO 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:

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

The random variable S is selected according to the following probabilistic rule:

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

Where N_i^{kv} is the set of remaining nodes to be visited by the k^{th} ant of colony *v* located at node *i* and P_{ij}^{v} is the pheromone of colony *v* on the edge (*i*,*j*).

The pheromone evaluation function $f(P_{ij})$ on the edge (*i*, *j*) for MACO-AVG will be defined as:

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

Whereas for MACO-MAX, the pheromone evaluation function is defined as:

$$f(P_{ij}) = \underset{v=1}{\overset{M}{Max}} \ P_{ij}^{v} \qquad (4)$$

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 *global pheromone update*:

$$P_{ij}^{v} = (1-\sigma) P_{ij}^{\ v} + \sigma \Delta p_{ij}^{\ v.bs} \qquad (5)$$

Where $\Delta P_{ij}^{v.bs}$ is the pheromone quantity added to the connection (i, j) belonging to the best solution of v^{th} colony $L^{v.bs}$ and is given by:

$$\Delta p_{ij}^{v.bs} = \begin{cases} 1/L^{v.bs} & \textit{if } (i, j) \text{ belongs to the best tour of colony } v \\ 0 & \text{otherwise} \end{cases} \qquad (6)$$

Local pheromone update is applied by each ant on the visited edges. It is very important rule as it is performed during the solution construction this helps to yield different pheromone evaluation values for the same edge in the same iteration at different solution construction steps and it is given by:

$$P_{ij}^{v} = (1-\gamma)P_{ij}^{v} + \gamma p_0 \qquad (7)$$

4. Computational Results

Full implementation of the MACO-AVG, MACO-MAX and the original ACS was developed along with this research work using visual C++ under windows XP. The problem to be tackled by MACO-AVG and MACO-MAX is the TSP. TSP is a well known traditional routing problem; it is almost used as starting test bed for new ant algorithms before applying them on other types of combinatorial problems.

Given n cities and distances between them, TSP is a problem of finding minimal length of closed tour that visit each city only once. The heuristic function is the inverse of the distance, i.e., $H_{ij}=1/d_{ij}$. Number of experiments was run using 2, 3, 4 and 5 colonies. For each experiment different setting of β is used to demonstrate the algorithm robustness. MACO-AVG and MACO-MAX have been tested using two TSP benchmark instances which are kroA100 (optimal solution is 21282) and lin318 (optimal solution is 42029) taken from TSP library [14].

In our implementation, ACS and MACO incorporated with a candidate list which is usually used when tackling large optimization problem. The candidate list contains for each node a number of the closest neighboring nodes. An ant first tries to choose the next node to move to from the candidate list; if all nodes in candidate list are already visited by the ant then it chooses the next node to move to from all other nodes not included in the candidate list. Candidate list size is considered to be 15 in our experimental work.

The results are averaged over 20 trials with 3000 and 10000 iterations per trial for kroA100 and lin318 respectively. Table 1 and Table 2 show the testing results of the experiments run with kroA100 and lin318 respectively. The symbol s next to the number of colonies refers that all colonies have the same value of β which is equal to 2. Different values of β are also tested and referred to by the symbol d next to the number of colonies. In this case β=2 is set for the first colony and β=3 for the second one and so on. Other parameter setting are $\sigma = \gamma = 0.1$ and $q_0 = 0.9$. The results reported in these two tables are the best solution obtained from all algorithm runs, the average of the best solutions, the standard deviation from the average solution and the average of the execution time of all runs.

MACO-AVG and MACO-MAX outperforms the original ACS one colony algorithm with the same number of ants. The performance of ACS declines as the number of ants increases. The problem was in the coordination of ants' population work to make use of the increment in the ants' number. This drawback has been overcome by the use of more than one ant colony and the use of an appropriate pheromone evaluation mechanism. The two MACO variants are almost showing similar performance with better results of MACO-AVG especially for the bigger problem instance.

Previous studies have shown that ACS gives the best result when using ten ants with a large number of iterations. Table 3 shows an additional experiment ran with ACS on Lin318 using 10 ants for 20 trials and 250000 iterations per trial. The overall average was 43420.90. ACS requires 472.25 seconds to reach this result. A similar result, which is 43429.50, was obtained by MACO-AVG with 5 colonies of 10 ants each working for 20 trials of 10000 iterations each, MACO-AVG requires 135.90 seconds. The superior of MACO-AVG in term of the time required to reach the same solution is obvious as MACO-AVG was 3.5 times faster than ACS reaching a similar result.

5. Conclusion and Future Work

The proposed algorithm divides the ants' population into multiple colonies and effectively coordinates their works. An average and maximum pheromone evaluation functions are used in the process of the ant's decision making. The results show that the proposed algorithm outperforms the ACS algorithm with similar number of ants.

The future work is the use of the proposed algorithmic framework on some other combinatorial optimization problems. New pheromone evaluation mechanism is another possible future direction. Another interesting future work is in the global pheromone update mechanism. In this paper the global best solution of each colony is considered. It is interesting to test the case where some colonies consider global best solution while others consider iteration best solution in the global pheromone update mechanism.

References

[1] Blum, C. & Roli, A. Metaheuristics in combinatorial optimization: Overview and conceptual comparisons. *ACM Computing Surveys*, 35(3), 268-308, 2003.

[2] Blum, C. & Dorigo, M. Search bias in ant colony optimization: On the role of competition-balanced systems. *IEEE Trans. on Evolutionary Computation*, 9(2), 159-174, 2005.

[3] Bullnheimer, B., Hartl, R. F., & Strauss, C. A new ranked-based version of the ant system: a computational study. *Central European Journal for Operations Research and Economics*, 7(1), 25-38, 1999.

[4] Cordon, O., Fernandez, I., Herrera, F., & Moreno, L. A new ACO model integrating evolutionary computation concepts: The Best-Worst Ant System. In: M. Dorigo, M. Middendorf, and T. Stützle, editors, *Abstract Proceedings of ANTS2000- From Ant Colonies to Artificial Ants: A Series of International Workshops on Ant Algorithms*, 22-29, 2000.

[5] Dorigo, M., Maniezzo, V. & Colorni, A. The ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics,* Part B, 26(1), 29-41, 1996.

[6] Dorigo, M. & Gambardella, L. Ant colony system: A cooperative learning approach to the traveling salesman problem. IEEE Trans. on Evolutionary Computation, 1(1), 53-66, 1997.

[7] Dorigo, M. & Stützle, T. The Ant Colony Optimization Metaheuristic: Algorithms, Applications, and Advances. In: F. Glover and G. Kochenberger (Eds.), *Handbook of Metaheuristics*. Kluwer Academic Publishers, 2002.

[8] Dorigo M., Birattari M. & Stützle T. Ant Colony optimization: Artificial Ants as Computational Intelligence Technique. *IEEE Computational Intelligence Magazine*, Nov. 2006.

[9] Gambardella, L., Taillard E., & Agazzi, G. MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows. In D. Corne, M. Dorigo, & F. Glover (Eds.), *New Ideas in Optimization*, London: McGraw Hill, 1999.

[10] Kawamura, H., Yamamoto, M., Suzuki, K. & Ohuchi, A. Multiple ant colonies algorithm based on colony level interactions. In *IEICE Trans. Fundamentals*, E83-A(2), 2000.

[11] Jong, J., & Wiering, M. 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, 2001.

[12] Sim, K. M., & Sun, W. H. Multiple ant colony optimization for network routing. In Proceeding of the First International Symposium on Cyber Worlds (CW’02), 153-164, 2002.

[13] Stützle T., & Hoos H. Max-Min Ant System. Journal of Future Generation Computer Systems, Vol. 16, pp 889 – 914, 2000.

[14] http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/tsp/

Table1: KroA100 Results

Algorithm	Colony / Ant	Best Solution	Overall Average	Std Dev	Time Avg.
ACS	20	21315	**21441.30**	112.13	3.45
MACO-AVG	2s/10	21282	21460.35	177.21	4.30
	2d/10	21282	21491.40	178.75	4.30
MACO-MAX	2s/10	21292	21445.45	211.54	4.15
	2d/10	21282	21495.00	287.70	4.15
ACS	30	21292	21448.35	186.42	5.25
MACO-AVG	3s/10	21292	21434.90	141.00	6.90
	3d/10	21282	21466.70	230.70	6.85
MACO-MAX	3s/10	21282	**21389.90**	162.54	6.60
	3d/10	21282	21408.25	124.97	6.65
ACS	40	21296	21476.60	138.66	7.05
MACO-AVG	4s/10	21282	**21373.85**	66.10	9.30
	4d/10	21282	21402.16	76.95	9.30
MACO-MAX	4s/10	21282	21398.15	122.29	9.15
	4d/10	21292	21411.55	79.92	9.20
ACS	50	21368	21629.70	194.69	8.85
MACO-AVG	5s/10	21282	21405.75	128.96	12.00
	5d/10	21282	21407.39	62.22	12.15
MACO-MAX	5s/10	21292	**21370.15**	110.14	11.75
	5d/10	21282	21478.05	126.14	11.80

Table 2: Lin318 Results

Algorithm	Colony / Ant	Best Solution	Overall Average	Std Dev	Time Avg.
ACS	20	43459	44459.00	918.33	38.80
MACO-AVG	2s/10	43440	**44157.12**	516.19	48.55
	2d/10	42907	44173.80	579.62	48.90
MACO-MAX	2s/10	43447	44418.77	745.38	46.10
	2d/10	43393	44200.27	519.96	46.10
ACS	30	43339	44287.00	631.48	60.00
MACO-AVG	3s/10	42936	43958.55	516.63	77.35
	3d/10	42980	43805.00	636.42	77.70
MACO-MAX	3s/10	42914	44035.93	660.15	73.45
	3d/10	42638	**43767.78**	604.13	73.50
ACS	40	43876	44934.20	894.84	80.10
MACO-AVG	4s/10	42658	43809.86	587.59	105.00
	4d/10	42787	**43522.60**	469.58	105.15
MACO-MAX	4s/10	43012	43719.00	483.63	101.15
	4d/10	42741	43718.61	494.09	101.85
ACS	50	44699	45452.15	411.54	102.05
MACO-AVG	5s/10	42739	**43429.50**	380.20	135.90
	5d/10	42766	43651.93	529.96	140.00
MACO-MAX	5s/10	42819	43695.55	581.19	131.45
	5d/10	42999	43808.79	647.78	132.05

Table 3: Lin318 ACS and MACO Comparison

Algorithm	Colony/ant	Trial/Iteration	Overall Avg.	Time Avg.
ACS	1/10	20/250000	43420.90	472.25
MACO-AVG	5s/10	20/10000	43429.50	135.90

Alaa Aljanaby received his B.Sc. and M.Sc. degrees in Computer Science from University of Basra, Iraq in 1993 and 1997, respectively. Since 1997 he was working as a lecturer at a number of universities in Iraq, Jordan and Oman. Currently he is also doing his PhD at University Utara Malaysia. His research interests include Meta-heuristic optimization algorithms, ant colony optimization and distributed and parallel algorithms. He published a number of papers in national and international conferences and Journals.

Prof. Dr. Ku Ruhana Ku Mahamud holds a Bachelor in Mathematical Sciences and a Masters degree in Computing, both from Bradford University, UK in 1983 and 1986, respectively. Her PhD in Computer Science was obtained from University Pertanian Malaysia in 1993. As an academic, her research interests include computer systems performance modeling, ant colony optimization and intelligent agent. These and other works have been published in international and national journal papers, proceedings of international conferences and other publications. Her book on ‘C Programming’ has won her the best publication award in the academic book category in 1999 and in 2002; she published another academic book on ‘Mathematics for Business’.

Dr. Norita Md Norwawi is an Associate Prof. at University Utara Malaysia. She obtained her Bachelor in Computer Science in 1987 from the University of New South Wales, Australia. She received her Masters degree in Computer Science from National University of Malaysia in 1994. In 2004 she obtained her PhD specializing in Temporal Data Mining from Northern University of Malaysia. As an academician, her research interests include artificial intelligence, multi-agent system, temporal data mining, text mining and knowledge mining. Her works have been published in international conferences, journals and won awards on research and innovation competition in national and international level.