An Experimental Study of the Search Stagnation in Ants Algorithms

Alaa Aljanaby
Faculty of IT
NTEC Tertiary Group
New Zealand

ABSTRACT

This paper conducts experimental tests to study the stagnation behavior the Interacted Multiple Ant Colonies Optimization (IMACO) framework. The idea of different ant colonies use different types of problem dependent heuristics has been proposed as well. The performance of IMACO was demonstrated by comparing it with the Ant Colony System (ACS) the best performing ant algorithm. The computational results show the dominance of IMACO and that IMACO suffers less from stagnation than ACS.

General Terms

Artificial Intelligence, Swarm Intelligence, Evolutionary Algorithms.

Keywords

Ant colony optimization, combinatorial optimization problems, search stagnation.

1. INTRODUCTION

Ant Colony Optimization (ACO) algorithms can get a worthy solution at the early stages of the search procedure but inappropriately all ants quickly converged to a single solution resulting in the algorithm being stagnant and unable to improve that solution [7, 14]. This is a common problem that all ACO algorithms suffer from regardless of the application domain; it is called search stagnation problem. The chance of stagnation correspondingly increases with the increase of the problem size.

IMACO framework attempts to improve the performance of ACO algorithms by using several ant colonies with certain methods to organize the work of these colonies. The proposed framework comprises necessary techniques that encourage the controlled exploration of the search space along with a good exploitation of previously obtained good solutions. Exploration lets ants to look at the search space for some new solutions. This should be done under certain control to avoid the exploration of a very wide area of the search space that might be far from the optimal solution. On the other hand a good exploitation of the search history is necessary to search the solution space in the neighborhood of previously found good solutions. However, a very strong exploitation is not required because it might increase the convergence speed of ants to the same solution obtained previously. [1, 2, 3].

Machine scheduling problems, traveling salesman problem, quadratic assignment problem, vehicle routing problem, and network routing problem are some well-known examples of Combinatorial Optimization Problems that have great importance in research and development. These problems have a discrete set of feasible solutions and the goal is to find the optimal solution (the best solution from the feasible solutions). These problems are theoretically proven as NP - hard problems [6, 12, 13, 15]. The only way to tackle these problems is to use approximate (heuristic) algorithms such as tabu search, evolutionary computation, simulated annealing, genetic algorithms and ACO. Single Machine Total Weighted Tardiness Problem (SMTWTP) is an important combinatorial optimization problem that considers the job scheduling for sequential processing on a single machine and the target is to minimize the total tardiness of all jobs. This is a key task in manufacturing and production planning. SMTWTP is used as the application domain of this work.

The rest of this paper is organized as follows. The framework of IMACO is described in section 2. A brief description of SMTWTP is given in Section 3. In section 4, an experimental study is conducted to analyze the stagnation behavior of IMACO and compare it with that of ACS. The result of applying IMACO with different problem dependent heuristics to solve all 375 available instances of SMTWTP is also presented. The final section concludes the paper.

2. INTERACTED MULTIPLE ANT COLONIES OPTIMIZATION

IMACO framework has been proposed in previous works of the author [1, 2, 3]. In this framework there are two levels of interaction the first one is the colony level and the second one 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 responsibility here is of the pheromone evaluating mechanism.

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 an interaction between the ants of the same colony. The interaction between ant colonies using pheromone can be organized in different terms.

Global and local pheromone updating are used in IMACO. Global pheromone updating includes that best ant of each colony deposits an amount of pheromone on its own path. The best ant refers to the ant that got the so far best (global) solution since the starting of the algorithm execution or the ant that got the best solution in the current iteration of the algorithm execution. In this work a combination of so far best and iteration best ants are allowed to update the pheromone.

To create a search diversification IMACO uses iteration best solution once in the pheromone updating after each 50 times of using the global best solution [8, 9]. Local pheromone updating includes that each ants reduces the amount of pheromone on paths it uses in order to give a more chance to other paths to be chosen by the future generations. 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.

2.1 Evaluation Technique

The pheromone of different colonies has been evaluated using two mechanisms. The first 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 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 IMACO is referred hereafter as IMACO-AVG.

The second mechanism evaluates the pheromone as the maximum value of the pheromone values of all colonies on some edge. This mechanism, referred as IMACO-MAX, chooses the max value among the available values. This mechanism lets an ant's decision to choose some new edge be based on the best available experience of ants of all colonies that previously visited this edge. This kind of cooperation using max pheromone evaluation is trying to make an early exploitation of the history of the search by choosing the max (best) available pheromone value. The result of this max pheromone evaluation function lets an ant to follow the best available information about the goodness of particular edge. However, since best pheromone comes from different colonies, this will provide necessary diversification that helps ants' to avoid the attraction to a one good solution.

The above two mechanism are pure average and max evaluation that depends 100% on the average evaluation function. The following rule is a more general which evaluates the pheromone as a composition between the pheromone values of the ant own colony and the value of the pheromone evaluation function based on some pheromone evaluation rate. Consider that the composition rate is 0.5; an ant will build 50% of its decision based on its own colony's experience and the other 50% based on the experiences of other colonies. This new variant will be called IMACO-AVG E λ and IMACO-MAX E λ where λ is the pheromone evaluation rate; its value is in the range [0, 1]. The pheromone evaluation function is then defined as:

$$f'(P_{ij}) = \lambda P_{ij}^{s} + (1-\lambda)\ \ f(P_{ij}) \qquad (1)$$

Where P_{ij}^{s} is the pheromone belongs to colony s on edge (*i*, *j*). Note that IMACO-AVG E0 and IMACO-MAX E0 represent the pure pheromone evaluation and IMACO-AVG and E1 IMACO-AVG represent no interaction between utilized ant colonies.

2.2 Exploration Technique

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. Exploration and exploitation is controlled by the parameter q_0 whose value is in [0, 1]. It 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). Setting q_0 to zero means that the algorithm uses a pure exploration while pure exploitation is reached by setting q_0 to one. However, the value used for q_0 in many research papers usually between 0.5 and 0.9 [1, 2]. Most of the work done using ACS in solving different problems was with q_0 =0.9 which gives the algorithm a high chance of exploitation without losing the chance of exploration.

IMACO considers the case where different ants' colonies have different values for the parameter q_0. The value 0.8 has been assigned to the centre colony whose number equal to int (no. of colonies / 2). This value is increased / decreased for the colonies after / before the centre colony by a changing factor called QCF. This technique enables the utilized ant colonies to work with different levels of exploration. Some will prefer high exploration of new areas of search space while other colonies will prefer high exploitation search history.

3. THE TEST PROBLEM

In SMTWTP *n* jobs have to be processed without pre-emption on a single machine that can handle no more than one job at a time. The processing and set-up requirement of any job are independent of its position in the sequence. The release time of all jobs is zero. Thus, jobs *j* (*j*=1, …, *n*) becomes available at time zero, requires uninterrupted positive processing time p_j, which includes set-up and knock-down times on the machine, has a positive weight w_j, and has a due time d_j by which it should ideally be finished. For a given processing order of the jobs, the completion time c_j and the tardiness T_j=max{0, c_j-d_j} of job *j* can be computed. The problem is to find a processing order of the jobs with minimum total weighted tardiness $\sum_{j=1}^{n} W_j T_j$ [4, 5].

The SMTWTP is an *NP*-hard scheduling problem for which instances with more than 50 jobs often cannot be solved to optimality with state of the art branch and bound algorithms [8]. The total number of available instances is 125 for values of *n*=40, *n*=50 and *n*=100. Optimal values of solutions are available for 124 and 115 of 40 and 50 job problem instances respectively. The values for unsolved problems are the best known solution to [8]. These solutions appear to be optimal since they have not been enhanced for a long time. The best known solutions to date of the 100-job instances are available and most of them are according to [8, 9].

Three types of problem specific heuristic are examined in this work. These problem specific heuristic are easily calculated and have been studied in the literature [8] and are as follows.

- Earliest Due Date (EDD): this heuristic puts the jobs in non-decreasing order of the due dates d_j and given by:

$$H_{ij} = \frac{1}{d_j} \qquad (2)$$

- Modified Due Date (MDD): this heuristic puts the jobs in non-decreasing order of the modified due dates mdd_j which given by mdd_j=*max*{*C*+p_j, d_j}, where *C* is the sum of the processing times of the already scheduled jobs. This heuristic is given by:

$$H_{ij} = \frac{1}{mdd_j} \qquad (3)$$

- Apparent Urgency (AU): this heuristic puts the jobs in non-decreasing order of the apparent urgency which given by:

$$au_j = \frac{w_j}{p_j}\exp(-\frac{\max\{d_j - C_j, 0\}}{k\overline{P}}) \qquad (4)$$

Where $\overline{P}$ is the average processing time of the remaining jobs, k is a scaling parameter which set to 2 [8]. The heuristic is given by:

$$H_{ij} = \frac{1}{au_j} \qquad (5)$$

4. SEARCH STAGNATION ANALYSIS

ACS, IMACO-AVG and IMACO-MAX for SMTWTP have been implemented using visual C++. Both versions of IMACO have been applied to all available 375 instances of SMTWTP. Based on past work of the authors the number of colonies utilized by IMACO-AVG and IMACO-MAX was 8 colonies, the evaluation rate was λ=0.4 and the exploration / exploitation control parameter was QCF=0.025 [2, 3]. In addition to use IMACO with EDD, MDD and AU, this section is developing the idea of using IMACO with different combination of the three heuristics. For instance, using EDD-MDD means that half of the utilized ant colonies will use EDD while the other half of these colonies will use MDD.

The global pheromone updating is performed by according to rules **3** and **4**. The value of best solution (global-best or iteration-best) mentioned in rule **4** represents the total weighted tardiness of the jobs sequence of the best solution. Local pheromone updating is performed using rule **5** and P_0 the initial value of pheromone trials that usually assigned a small value computed as $P_0 = \frac{1}{nT_{EDD}}$ where n is the number of jobs and T_{EDD} is the total weighted tardiness of job sequence obtained by EDD.

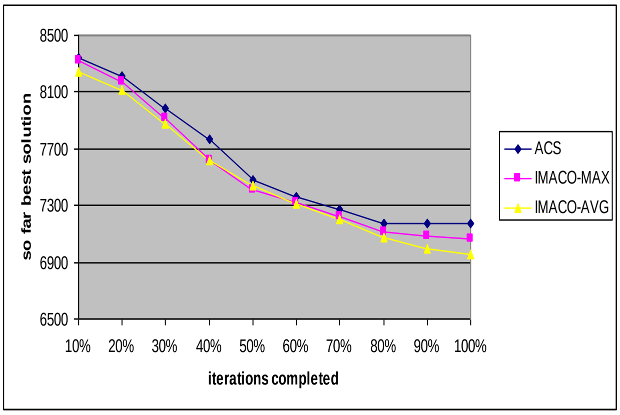

Figure 1: One trial typical run on 100-job SMTWTP instance

The stagnation behavior of IMACO is the main concern of this paper. This section tries to experimentally analyze the stagnation behavior by observing 100 typical runs of ACS, IMACO-MAX, and IMACO-AVG and calculate how many times that each algorithm get trapped into stagnation situation. Figure 1 shows the results of tracing one typical run of the three algorithms on SMTWTP 100-job. ACS ran with 10 ants, IMACO-AVG E.4 and IMACO-AVG E.4 with 8 colonies of 10 ants each, all algorithms ran with MDD. It obvious that ACS cannot improve its solution after 70% of algorithm iterations completed, the last 30% of the algorithm iterations ACS search process was stagnant. IMACO-AVG was the best among the other algorithms. It was able to direct its search away from stagnation situation and improve its solution.

Table 1 shows a result of 100 of typical runs like those described above. The numbers in the table shows how many times the algorithm cannot improve its solution from the previous step. The algorithm which suffers more from stagnation was ACS. The chance of stagnation increases after 50% of iterations completed because all ants converged to one solution or one dominant path with highest amount of pheromone and ability to get out from this situation decrease with the time. IMACO-AVG was the algorithm that suffers less from stagnation. This proves that the techniques incorporated in this algorithm did well to avoid such situations.

Table 1. Stagnation situations in 100 trials typical run on 100-job SMTWTP instance

Iterations completed	ACS	IMACO-MAX	IMACO-AVG
10%	0	0	0
20%	0	0	0
30%	0	0	0
40%	1	0	0
50%	10	1	0
60%	15	7	1
70%	20	4	0
80%	24	6	3
90%	28	3	2
100%	28	3	1

Table 2. Results for 40, 50 and 100 job instances

Algorithm	Heuristic	40-job	50-job	100-job
ACS	EDD	39	33	24
	MDD	44	37	27
	AU	36	30	21
IMACO-AVG	EDD	45	38	30
	MDD	53	45	37
	AU	41	34	26
	EDD-MDD	57	50	42
	EDD-AU	43	37	28
	MDD-AU	49	43	34
	EDD-MDD-AU	54	47	38
IMACO-MAX	EDD	43	37	30
	MDD	48	42	34
	AU	38	31	24
	EDD-MDD	53	46	39
	EDD-AU	43	36	30
	MDD-AU	47	40	32
	EDD-MDD-AU	49	43	34

Table 2 shows the results of experiments performed on 125 instances of 40, 50 and 100 job SMTWTP. The results presented in these tables are the number the optimal solution found (out of 125). The results of ACS presented in Table 2 are of the implementation developed with this research work. The reason is that the results of ACS presented in the literature usually with local search while all results presented here are without using local search. It is important to mention that all algorithms ran exactly the same number of computation steps.

Concerning the use of different combination of heuristics, EDD-MDD was the best combination as it always reaches the best results. EDD-MDD-AU heuristic was in the second rank followed by MDD. This seems normal as previous studies [8, 10] show the ranking of these heuristic according to the goodness of the results obtained was MDD, EDD and AU respectively. The results obtained from IMACO confirmed this getting the best results when using the best two heuristics, i.e., EDD-MDD combination. In fact, the use of a combination of heuristics increases the ability of different colonies to achieve high diversion in the search process and therefore increase the ability to improve the quality of the obtained solutions.

5. CONCLUSION

The computational results show that IMACO-AVG and IMACO-MAX outperform ACS in terms of the number of optimal solutions found. IMACO-AVG was the best algorithm that found the best results all the way. It is the capability of IMACO to avoid the stagnation situation and improves its solutions with the time. The improved performance of IMACO is based mainly on the interaction used between ant colonies and the type of information used by ants when making their decision. The proposed interaction plays on two directions which are cooperation and diversification. Pheromone evaluation mechanism plays the main role in cooperation. Pheromone evaluation was the mean to combine the pre-acquired information about the quality of the solutions represented as pheromone values.

Average pheromone evaluation was the best technique that places IMACO-AVG in front of other ant algorithms. Pheromone evaluation needs a big support from other mechanisms. On the other hand, letting different colonies works with different levels of exploration / exploitation was of a great aid in achieving diversification. Some colonies prefer a higher exploration while others prefer a higher exploitation. This provides the whole search process with a wide range of good solutions that ants of different colonies choose their best solution from.

6. REFERENCES

[1] Aljanaby A., K.R. Ku-Mahamud and N.M. Norwawi, 2010. Interacted multiple ant colonies to enhance the performance of ant colony optimization algorithms. Journal of Computer and Information Science (CIS), Canada, vol. 3, no. 1, pp. 29-34.

[2] Aljanaby A., K.R. Ku-Mahamud and N.M. Norwawi, 2010. Revisiting pheromone evaluation mechanism in the interacted multiple ant colonies framework. Proc. of 10th international conference on Artificial Intelligence and Applications (AIA2010), Austria, pp.12-15.

[3] Aljanaby A., K.R. Ku-Mahamud and N.M. Norwawi, 2010. An exploration Technique for the interacted multiple ant colonies framework. Proc. of 1st international conference on Intelligent Systems, Modelling, and Simulation (ISMS2010), Liverpool, UK, pp. 92-95.

[4] Baggio, G., J. Wainer and C. Ellis, 2004. Applying Scheduling Techniques to Minimize the Number of Late Jobs in Workflow Systems. Proc. of ACM symposium on Applied computing, Nicosia, Cyprus, pp. 1396-1403.

[5] Besten, M., T. Stützle and M. Dorigo, 2000. Ant Colony Optimization for the Total Weighted Tardiness Problem. Proc. of Parallel Problem Solving from Nature Conference, Paris, France, pp. 611-620.

[6] Blum, C. and A. Roli, 2003. Meta-heuristics in combinatorial optimization: Overview and conceptual comparisons. ACM Computing Surveys, vol. 35, no.3, pp. 268-308.

[7] Blum, C. and M. Dorigo, 2005. Search bias in ant colony optimization: On the role of competition-balanced systems. IEEE Trans. on Evolutionary Computation, vol. 9, no. 2, pp. 159-174.

[8] Congram, R., Potts, C., and van de Velde, S, 2002. An Iterated Dynasearch Algorithm for the Single-Machine Total Weighted Tardiness Scheduling Problem. INFORMS Journal on Computing, vol. 14, no. 1, pp. 52-67.

[9] Crauwels, H., C. Potts and L. van Wassenhove, 1998. Local Search Heuristics for the Single Machine Total Weighted Tardiness Scheduling Problem. INFORMS Journal on Computing, vol. 10, no. 3, pp. 341-350.

[10] Dorigo, M. and T. Stützle, 2002. The Ant Colony Optimization Meta-heuristic: Algorithms, Applications, and Advances. In: Handbook of Meta-heuristics (Eds. F. Glover and G. Kochenberger), pp. 250-285, Kluwer Academic Publishers.

[11] Dorigo, M. and T. Stützle, 2004. Ant colony optimization, London: The MIT Press.

[12] Guo J. E. and W. G. Diao, 2014. An improved ant colony optimization algorithm with crossover operator. Open Mechanical Engineering Journal, vol. 8, no. 1, pp. 96-100.

[13] Pang, S. C., T. M. Ma and T. Liu, 2015. An improved ant colony optimization with optimal search library for solving the traveling salesman problem. Journal of Computational and Theoretical Nanoscience, vol. 12, no. 7, pp. 1440-1444.

[14] Yan X. S., 2012. Efficiency analysis of swarm intelligence and randomization techniques. Journal of Computational and Theoretical Nanoscience, vol. 9, no. 2, pp. 189-198.

[15] Yue Y and X. Wang, 2015. An Improved Ant Colony Optimization Algorithm for Solving TSP. International Journal of Multimedia and Ubiquitous Engineering vol.10, no.12, pp.153-164.