ICIC Express Letters
Part B: Applications
Volume **8**, Number **11**, November **2017**

ICIC International ©2017 ISSN 2185-2766
pp. 1465–1471

AN IMPROVED ANT COLONY OPTIMIZATION FOR JOB-SHOP SCHEDULING PROBLEM

Yang Jiang[1] and Qiulei Ding[2,*]

[1]School of Light Industry & Chemical Engineering
Dalian Polytechnic University
No. 1, Qinggongyuan, Ganjingzi District, Dalian 116034, P. R. China
jiangyang@dlpu.edu.cn

[2]School of Business Administration
Dongbei University of Finance and Economics
No. 217, Jianshan Street, Shahekou District, Dalian 116025, P. R. China
*Corresponding author: qiuleid@dufe.edu.cn

Received May 2017; accepted August 2017

Abstract. *The job-shop scheduling problem (JSP), proven to be an NP-hard problem, is difficult to obtain the optimal or near-optimal solution. Ant colony optimization (ACO) is a metaheuristic that takes inspiration from the foraging behavior of a real ant colony to solve the optimization problem. This paper proposes an improved ant colony optimization (IACO) to solve JSP. At the beginning, by adjusting pheromone approach and introducing the approach of crossover and mutation, the IACO is able to prevent the search process from getting trapped in the local optimal solution. Then, by combining the ACO with other heuristics, the IACO improves the convergence speed. Furthermore, the IACO is tested using the classical sets. The effectiveness of IACO on solving JSP is validated by comparing the computational results with those previously presented in the literature.*
Keywords: Job-shop scheduling problem, Ant colony optimization, Combinatorial optimization problem

1. Introduction.

JSP belongs to a class of problems that are known to be strongly NP-hard problems [1]. For every finite size problem, the optimal solutions can be obtained in bounded time via exact algorithm such as a branch-and-bound method and mathematical programming. However, it was found that such algorithms may take a prohibitive amount of time, even for a moderate size problem. For pragmatic purposes there is a need to look for more efficient approaches that can find good solutions within the reasonable time.

In the past decade, lots of metaheuristics have been widely studied to solve the JSP. These approaches include simulated annealing [2], tabu search [3], genetic algorithm [4], and other heuristics [5,6]. A comprehensive survey of job shop scheduling techniques can be founded in [7].

In recent years a metaheuristic called ACO has been receiving extensive attention due to its successful applications for many combinatorial optimization problems [8-10]. ACO is presented by Colorni et al. [11,12] in the 1990s, which is a swarm intelligence algorithm by imitating the behavior of real ants searching for the food. Real ants are able to communicate information concerning food sources via an aromatic essence called pheromone. While they move along, real ants lay down pheromone in a quantity that depends on the quality of the food source discovered. Other ants, observing the pheromone trail, are attracted to follow it. Thus, the path will be enhanced and will therefore attract more ants. From the behavioral mechanism described above, the ACO is proposed by simulation with artificial ants searching the solution space instead of real ants searching their environment.

Colorni et al. [13] were the first researchers that applied ACO to solving JSP. They presented a hybrid ant system algorithm that added the 2-opt heuristic. Udomsakdigool and Kachitvichyanukul [1] proposed a multiple colony ant algorithm to solve JSP with the objective that minimizes the makespan. The computational results demonstrated that the algorithm performed well on the benchmark problems. Huang and Liao [14] put forward a hybrid algorithm combining ACO with the taboo search algorithm for the classical JSP. The algorithm employed a novel decomposition method inspired by the shifting bottleneck procedure, and a mechanism of occasional reoptimizations of partial schedules. Xing et al. [15] proposed a knowledge-based ant colony optimization (KBACO) algorithm. In KBACO algorithm, knowledge model learned some available knowledge from the optimization of ACO, and then applied the existing knowledge to guiding the current heuristic searching. Korytkowski et al. [16] provided a heuristic method based on ACO to determine the suboptimal allocation of dynamic multi-attribute dispatching rules to maximize system performance.

It is noted that great efforts have been made by existing researches for improving the ACO to solve JSP, but the premature convergence and low search speed still need further study. Therefore, this paper tries to present IACO to overcome those mentioned difficulties. The remainder of this paper is organized as follows. Section 2 constructs the mathematic model of JSP and Section 3 demonstrates the principle of IACO. Section 4 compares the computational results with existing literature and shows the good performance of the proposed algorithm. Finally, Section 5 provides conclusions and directions for the future research.

2. Mathematic Model of JSP.

2.1. Problem definition.

The problem is confined to the following conditions. Given n jobs, each job has m operations and must be processed on m machines. The objective is to find a schedule of minimal time to complete all jobs, where

(1) the machine can only process one job at a time;

(2) every job is available for processing at time 0; and

(3) once processing is initiated, the operation must be completed on the machine without interruption.

2.2. Notations.

n: the number of jobs;
m: the number of machines;
V: the set of jobs, $V = \{v_1, v_2, \dots, v_n\}$;
U: the set of machines, $U = \{u_1, u_2, \dots, u_m\}$;
c_{ik}: the completion time of job i in machine k;
p_{ik}: the processing time of job i in machine k;
d_i: the due date of v_i;
M: a large positive number;

$$a_{ijk} = \begin{cases} 1, & v_i \text{ is processed in } u_j \text{ before } u_k \\ 0, & \text{otherwise} \end{cases};$$

$$x_{ijk} = \begin{cases} 1, & v_i \text{ is processed before } v_j \text{ in } u_k \\ 0, & \text{otherwise} \end{cases}.$$

2.3. Mathematic model.

The model is constructed as follows:

$$\min \max_{1\le k\le m} \left\{ \max_{1\le i\le n} c_{ik} \right\} \tag{1}$$

$$c_{jk} - p_{jk} + M(1 - x_{ijk}) \ge c_{ik} \quad i, j = 1, 2, \dots, n;\ k = 1, 2, \dots, m \tag{2}$$

$$c_{ik} - p_{ik} + M(1 - a_{ijk}) \ge c_{ij} \quad i = 1, 2, \dots, n;\ j, k = 1, 2, \dots, m \tag{3}$$

$$d_i \leq \max_{1 \leq k \leq m} c_{ik} \quad i = 1, 2, \ldots, n \tag{4}$$

In this model, the objective function (1) is to minimize the makespan, which is the minimal time to complete all jobs. Formulas (2) and (3) are the technological and processing constraints. Formula (4) ensures that all jobs are completed before their due date.

3. The Principle of IACO.

In order to prevent the search process from getting trapped in local optimal solution, the IACO is presented by adjusting the pheromone and introducing the approach of crossover and mutation. In order to improve the convergence speed, the IACO is proposed by combining the ACO with other heuristics. The brief description is shown in Figure 1. The details are demonstrated in the next subsection.

```
Input: An instance of JSP
       Initialize pheromone value and parameter values
while (termination condition not met) do
  for ant $i = 1$ to $n$ do
    Construct solution
    Apply crossover and mutation operations
    Apply neighborhood exchange
  next ant
  Update pheromone value and parameter values
end while
Output: Optimal solution
```

FIGURE 1. Brief description of IACO

3.1. Introduction of adjusting the pheromone.

Considering the importance of the information interchange between ant colonies by pheromones, this section focuses on three aspects for adjusting pheromone to avoid the IACO becoming trapped in local optimal solution. The details are shown as follows.

(1) In the ACO algorithm, the optimal direction of ant colonies is always the path where the pheromone is given by the most ant colonies. Therefore, it is difficult to find the best solution because the pheromone which is far from the global optimal solution may be enhanced. Although the ACO improves the selection probability randomly, it is still subject to becoming trapped in local optimal solution.

Therefore, the definite and random selections are adopted to improve the global optimization capability in IACO. When the search is becoming trapped in local optimal solution, the solution space can be further searched by adjusting the pheromone and increasing the random selection probabilities.

(2) At every edge, the maximum or minimum pheromone trails may lead to premature convergence of the search. Therefore, the IACO puts forward the $\tau_{\min}$ and $\tau_{\max}$ as the minimum and maximum pheromone trails and limits the pheromone trail τ_{ij} in the interval $(\tau_{\min}, \tau_{\max})$. Meanwhile, the pheromone trails are deliberately initialized to $\tau_{\max}$, which helps to find a better solution at the beginning of the search. Additionally, in cases where the pheromone trails differ greatly, the idea of computing average pheromone trails between τ_{ij} and $\tau_{\max}$ is absorbed, which will play a significant role in obtaining a new solution.

(3) It is difficult for the ACO algorithm to solve large-scale problems because of the existence of the trail evaporation $1 - \rho$. The larger $1 - \rho$ is, the better the global optimization capability will be. However, if so, the convergence speed of the algorithm will be slowed down. Therefore, in order to overcome the difficulty, this paper adopts a dynamic $1 - \rho$ value rather than a constant value.

3.2. **Approach of the crossover and mutation.** The crossover and mutation operations can increase the variance of the population and search the solution space completely in GA. Thus, these two operations are adopted in IACO to avoid premature convergence.

(1) Crossover operation

When the search gets trapped in local optimal solution, the crossover operation will be carried out between the encodings of optimal solution and sub-optimal solution. The rules of crossover operation are shown as follows.

• Assume that encoding I is I1|I2|I3 and encoding J is J1|J2|J3, respectively. Then apply two point crossover operation to the above two encodings.

• Assume that I2 and J2 are the crossover sections, respectively. Then insert J2 into I and generate a new encoding K that is I1|J2|I2|I3.

• Delete the nodes among I1, I2, and I3 which are the same as J2. Then update the encoding K.

• Apply the same way to J and generate a new encoding L.

• Select the encoding of optimal solution among I, K and L.

(2) Mutation operation

Mutation plays a significant part in improving the diversity of the population. In a similar way to GA, IACO is designed to avoid premature convergence and obtain a better solution by decreasing pheromone trails greatly in any path of the local optimization routing. Moreover, these paths will be selected by small random probabilities since the distribution of the pheromone trails in the previous solutions would be destroyed by too many times of mutation, which lead the search to the bad direction.

3.3. **Combination with other heuristics.** The ACO is a strong coupling algorithm, where the convergence speed can be improved by combining with other heuristics. Exchange mechanism is originally applied in the vehicle routing problem with time windows. It can obtain a new solution from the current solution by exchanging the nodes. We demonstrate that a neighborhood exchange mechanism based on the exchange mechanism can significantly improve the convergence speed of ACO. A number of neighborhood operators (among those listed below) are included in the paper.

• **Random swaps.** It randomly selects positions i and j with $i \neq j$ and swaps the customers located in positions i and j in the sequence of solution. See Figure 2, where $i = 3$ and $j = 7$.

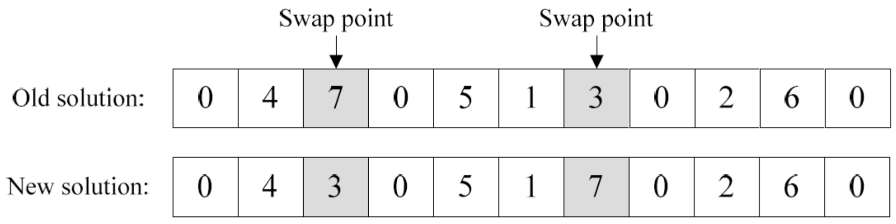
Swap point
Swap point
Old solution: 0 4 7 0 5 1 3 0 2 6 0
New solution: 0 4 3 0 5 1 7 0 2 6 0

Figure 2. Random swaps

• **Random swaps of subsequences.** It is an extension of the previous one, where two subsequences of old solution with random lengths are selected and swapped. Figure 3 gives an example of this operator.

• **Random insertions.** It randomly selects positions i and j with $i \neq j$ and relocated the customer from position i to position j. See Figure 4, where customer 2 is relocated from position 9 to position 5.

• **Random insertions of subsequences.** It is an extension of the operator of random insertions. This operator randomly selects positions i and j with $i \neq j$ and relocates a subsequence of old solution with random length starting from position i to position j. An example is shown in Figure 5.

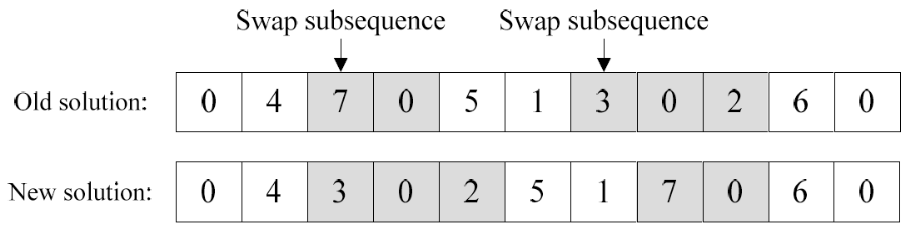

FIGURE 3. Random swaps of subsequences

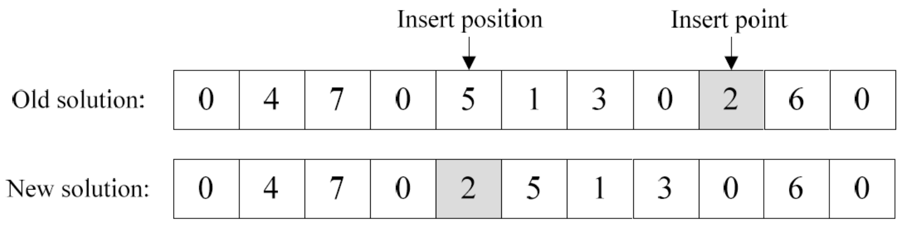

FIGURE 4. Random insertions

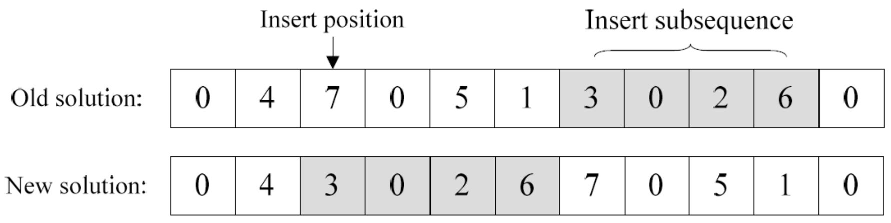

FIGURE 5. Random insertions of subsequences

4. **Numerical Analysis.** In this section, we present computational experiments to illustrate the theoretical results obtained in the previous sections.

4.1. **Computational results.** The IACO is tested using the classical sets of JSP, which are TA [17] and DMU [18]. For each instance, the proposed algorithm is independently executed 10 times to compute the average value. We then conduct a performance comparison between IACO and other heuristics, including *i*-TSAB [3], GES [5], and AlgFix [6]. The results are illustrated in Table 1.

TABLE 1. Computational results of solving different instances

Algorithm	TA05	TA11	TA20	TA37	TA43	DMU12	DMU26	DMU45	DMU61	DMU77
i-TSAB	1224	1361	1351	1778	1859	3519	4679	3321	5294	6930
GES	1224	1357	1348	1779	1870	3518	4667	3273	5293	7006
AlgFix	1224	1358	1348	1784	1869	3522	4688	3273	5310	6949
IACO	1224	1357	1348	1779	1858	3520	4665	3275	5277	6908

4.2. **Comparison among different heuristics.** The comparison obtained from the above results can be summarized as follows.

- Compared with *i*-TSAB, IACO obtains better or close solutions in 8 out of 10 problems (80%).
- Compared with GES, IACO obtains better or close solutions in 8 out of 10 problems (80%).
- Compared with AlgFix, IACO obtains better or close solutions in 9 out of 10 problems (90%).

The comparison indicates that the IACO is competitive with the existing heuristics and has been able to improve the best solutions known for a number of instances, especially for large size problems, such as DMU61 and DMU77. It is noted that those parameters affect the overall performance of IACO, so the solution obtained by our algorithm would be improved further if every parameter is set more reasonably.

5. Conclusions.

An improved algorithm named IACO is presented to solve JSP in this paper. At the beginning, the premature convergence is effectively avoided by adjusting pheromone dynamically. Then, the introduction of the crossover and mutation operations also prevents the IACO from getting trapped in local optimal solution. Meanwhile, by combining with the neighborhood exchange mechanism, the search speed is also greatly improved. The computational analysis between this paper and the existing literature indicates that the IACO is competitive with the best existing heuristics.

Possible future research may focus on how to set every parameter reasonably by extensive testing and/or theoretical analysis. Another promising direction is to enhance the ants learning ability and endow them with more intelligence by combining with agent technology to promote the coordination between ant colonies according to the characteristics of autonomy, proactivity, reactivity and intelligence.

Acknowledgment. This work has been partially supported by the grants from the Natural Science Foundation of China (No. 71301020), Program for Liaoning Excellent Talents in University (No. WJQ2014030), and Program for Liaoning Scientific Research in University (No. L2014216). The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

REFERENCES

[1] A. Udomsakdigool and V. Kachitvichyanukul, Multiple colony ant algorithm for job-shop scheduling problem, *International Journal of Production Research*, vol.46, no.15, pp.4155-4175, 2008.
[2] M. E. Aydin and T. C. Fogarty, A simulated annealing algorithm for multi-agent systems: A job-shop scheduling application, *Journal of Intelligent Manufacturing*, vol.15, no.6, pp.805-814, 2004.
[3] E. Nowicki and C. Smutnicki, An advanced tabu search algorithm for the job shop problem, *Journal of Scheduling*, vol.8, pp.145-159, 2005.
[4] L. Asadzadeh, A local search genetic algorithm for the job shop scheduling problem with intelligent agents, *Computers & Industrial Engineering*, vol.85, pp.376-383, 2015.
[5] P. M. Pardalos and O. V. Shylo, An algorithm for the job shop scheduling problem based on global equilibrium search techniques, *Computational Management Science*, vol.3, pp.331-348, 2006.
[6] P. M. Pardalos, O. V. Shylo and A. Vazacopoulos, Solving job shop scheduling problems utilizing the properties of backbone and "big valley", *Computational Optimization and Applications*, vol.47, pp.61-76, 2010.
[7] I. A. Chaudhry and A. A. Khan, A research survey: Review of flexible job shop scheduling techniques, *International Transactions in Operational Research*, vol.23, no.3, pp.551-591, 2016.
[8] X. J. Liu, H. Yi and Z. H. Ni, Application of ant colony optimization algorithm in process planning optimization, *Journal of Intelligent Manufacturing*, vol.24, no.1, pp.1-13, 2013.
[9] T. Liao, T. Stützle, M. A. M. D. Oca et al., A unified ant colony optimization algorithm for continuous optimization, *European Journal of Operational Research*, vol.234, no.2, pp.597-609, 2014.
[10] D. Thiruvady, A. T. Ernst and G. Singh, Parallel ant colony optimization for resource constrained job scheduling, *Annals of Operations Research*, vol.242, no.2, pp.355-372, 2016.
[11] A. Colorni, M. Dorigo and V. Maniezzo, Distributed optimization by ant colonies, *Proc. of European Conference on Artificial Life*, Paris, France, pp.134-142, 1991.
[12] A. Colorni, M. Dorigo and V. Maniezzo, An investigation of some properties of an "Ant Algorithm", *Proc. of the Parallel Problem Solving from Nature Conference*, Brussels, Belgium, pp.509-520, 1992.
[13] A. Colorni, M. Dorigo, V. Maniezzo et al., Ant system for job-shop scheduling, *Operations Research Statistics & Computer Science*, vol.34, no.1, p.114, 1994.
[14] K. L. Huang and C. J. Liao, Ant colony optimization combined with taboo search for the job shop scheduling problem, *Computers & Operations Research*, vol.35, no.4, pp.1030-1046, 2008.

[15] L. N. Xing, Y. W. Chen, P. Wang et al., A knowledge-based ant colony optimization for flexible job shop scheduling problems, *Applied Soft Computing*, vol.10, no.3, pp.888-896, 2010.

[16] P. Korytkowski, S. Rymaszewski and T. Wiśniewski, Ant colony optimization for job shop scheduling using multi-attribute dispatching rules, *The International Journal of Advanced Manufacturing Technology*, vol.67, no.1, pp.231-241, 2013.

[17] E. D. Taillard, Parallel taboo search techniques for the job shop scheduling problem, *ORSA Journal on Computing*, vol.6, pp.108-117, 1994.

[18] E. Demirkol, S. Mehta and R. Uzsoy, A computational study of shifting bottleneck procedures for shop scheduling problems, *Journal of Heuristics*, vol.3, pp.111-137, 1997.