ISSN 1750-9653, England, UK
International Journal of Management Science and Engineering Management, 6(2): 117-123, 2011
http://www.ijmsem.org/

Ant colony algorithm for multi-criteria job shop scheduling to minimize makespan, mean flow time and mean tardiness*

Apinanthana Udomsakdigool[1]†, Voratas Khachitvichyanukul[2]

[1] Department of Industrial Technology, College of Industrial Technology King Mongkut's University of Technology North Bangkok, Bangkok 10800, Thailand

[2] Field of Industrial Engineering & Management, School of Advanced Technologies Asian Institute of Technology, Pathumthani 12120, Thailand

(*Received 21 February 2010, Revised 21 July 2010, Accepted 13 October 2010*)

Abstract. In the real world situation many scheduling problems faced by decision maker are involved more than one aspect and therefore multiple criteria analysis is required. This paper presents ant colony algorithm for solving the multi-objective Job Shop Scheduling Problem (JSP). The objectives considered in this study include the minimization of makespan, mean flow time, and mean tardiness. The proposed algorithm is tested on many benchmark problems up to 15 jobs × 10 machine. The results obtained have shown that the proposed algorithm is a feasible and effective approach for the multiple-objective problem.

Keywords: ant colony algorithm, multi-objective scheduling, job shop scheduling problem

1 Introduction

The JSP is generally defined as problems with the aim of optimizing one or more scheduling objectives. Most research in the JSP is concerned with the optimization of a single criterion. In the real-life scheduling problems, the decision maker is often faced with situations in which the appropriateness of a schedule is measured against multiple objectives. Therefore the multiple criteria analysis is required. The JSP is known to be a strong NP-hard problem. Hence the JSP included m objectives must also be an NP-hard problem. Mathematical programming approaches for solving multi-objective scheduling problem are computationally intractable for practical problems. Due to the high complexity of computational of the DFLP; Thus in the past decade there are many efficient methods which can find good solutions in an acceptable time have been widely studies such as simulated annealing, tabu search, genetic algorithm and ant colony optimization. Baykasoğlu et al. (2002) [2] developed metaheuristic based on a multiple dispatching rule and tabu search for a multi-objective job shop scheduling problem. Loukil et al. (2003) [9] proposed multi-objective simulated annealing for three models of problems: one machine, parallel machine and permutation flow shop. Rajendran and Ziegler (2004) [11] developed ant algorithm for solving the permutation flow shop scheduling to minimize makespan and total flow time of jobs. Xia and Wu (2005) [13] hybridize the particle swarm and simulated annealing to solve the multi-objective flexible job shop scheduling. Choobineh et al. (2006) [4] introduced tabu search algorithm for three objectives one machine scheduling with sequence-dependent setup times. Varadharajan and Rajendran (2006) [12] proposed a multi-objective simulated-annealing algorithm for solving permutation flow shop scheduling with the objective of makespan and total flow time of jobs. Gao et al. (2007) [7] hybridize the genetic algorithm and bottleneck shifting procedure. Eren and Güner (2009) [6] analyze the single machines bi-criteria problem with learning effect. The technique they propose called discrete differential evaluation algorithm. Recently Xing et al. (2009) [14] create a simulation model for the Multi-objective flexible job shop schedule. Behnamian et al. (2009) [3] developed the solution approach for the Parallel-machine scheduling problems with sequence-dependent setup times. The proposed algorithm comprises three components based on an ant colony optimization (ACO), a simulated annealing (SA), and a variable neighborhood search (VNS). A good review of multi-criteria scheduling problems can be found in Hoogeven (2004) [8].

In this paper the ant colony algorithm is proposed for solving the multi-criteria JSP that minimizes the weighted sum of makespan, mean flow time and mean tardiness. The makespan and mean flow time criteria are focused on improving resource utilization and productivity. When the customers are concerned with the due date of jobs, the tardiness criterion is perceived as measure of conformity with the due date. The remainder of this paper is organized as follows. The problem description is introduced in Section 2.

* The joint activities received funding from the National Research Council of Thailand, the Engineering Management Program of Kasetsart University, and the Asian Institute of Technology. This project is supported by the College of Industrial Technology, King Mongkut's University of Technology North Bangkok Grant for Research Projects 2006.

† *Correspondence to*: Tel.: +66-2913-2500, fax: +66-2913-2500 ext 6523. *E-mail address*: aug@kmutnb.ac.th.

Published by World Academic Press, World Academic Union

The detail of proposed algorithm is described in Section 3. In Section 4, the computational experiments are performed on JSP benchmark and the conclusions are given in Section 5.

2 Problem description

Let J be a set of n jobs $\{J_i\}_{i=1}^{n}$ to be processed on a set M of m machines $\{M_j\}_{j=1}^{m}$. O_{ij} is the operation of job J_i which has to be processed on machine M_j for a processing period P_{ij}. Each job J_i consists of a chain of operations $\{O_{ij}\}_{j=1}^{m}$ which represents the predetermined order of job J_i through the machines. The objective is to find a schedule that minimizes the makespan, mean flow time and mean tardiness as Eq. (1).

$$\min\ f(x) = \{f_1(x), f_2(x), f_3(x)\} \quad x \in \mathbf{X}, \tag{1}$$

where x is the schedule, X is a set of feasible schedules. f_1, f_2, f_3 are the objective functions of makespan, mean flow time and mean tardiness respectively.

Assumptions made in this paper are:

(1) Machine setup times are negligible.
(2) Machine preemption is not allowed.
(3) Machines are available any time.
(4) No machine may process more than one job at a time.
(5) No job may be processed on more than one machine simultaneously.

3 Ant colony algorithm

Ant colony algorithms are becoming popular approaches for solving combinatorial optimization problems in the literature including the assignment problem, traveling salesman and scheduling. A comprehensive review on ant algorithms can be found in Dorigo and Stützle (2004) [5]. Generally in ant algorithm a finite-size colony of artificial ants searches for good-quality solutions of the JSP. The concept of the proposed algorithm is to have a population of artificial ants that iteratively constructs solution to the JSP, starting at an initial node; every ant selects the next node to move according to the 2-step proportional transition rule. When all the ants complete the solution, the sequence of the nodes visited represents the solution to the JSP. The weighted sum of objective function with variable weights is computed and a pheromone update rule is performed. When ants repeat the solution procedure for a number of iteration the solution will be emerge. The brief description is shown in Figs. 1 and 2 with the details described as follows.

3.1 Initialize pheromone and parameter setting

The pheromone value on each path is initialized with random values drawn from the interval (0.1, 0.25) in order to enforce the diversification at the start of the algorithm. The lower bound of pheromone value is set to a small positive constant (0.001) to prevent the algorithm from prematurely converging to a solution. The algorithm is terminated when reach 2000 iterations. The important weight of the pheromone trail, α is 1 and the important weight of heuristic information, β is 5. The relative importance between exploitation and exploration, q_0 is 0.5. The random number, q is uniformly distributed in [0, 1]. The number of ants is set equal to the number of operations and the ants are divided into three subcolony of equal size. Each group use different heuristic information to guide their search. The weight of three objectives of makespan, w_1, mean flow time, w_2, mean tardiness, w_3 are set to 0.5, 0.3, and 0.2 respectively.

3.2 Construct solution

At a construction step the ants applied the probability transitional rule to construct their solution detailed as follows. In each step of operation i in iteration t, the ant k selects an operation by taking a random number q. If $q \le q_0$, the operation is chosen according to Eq. (2).Otherwise an operation is selected according to Eq. (3).

$$j_i^k(t) = \begin{cases} \arg\{\max_{0 \in C_i^k} [\tau_{i0}^k(t)]^\alpha [\eta_{i0}^k]^\beta\}, & \text{if } q < q_0 \\ J, & \text{otherwise} \end{cases} \tag{2}$$

where j is the selected operation at the present step of operation i, J is an operation selected from the random proportional transition rule defined as Eq. (3).

$$p_{i0}^k(t) = \begin{cases} \frac{[\tau_{i0}^k(t)]^\alpha [\eta_{i0}^k]^\beta}{\sum_{0 \in C_i^k} [\tau_{i0}^k(t)]^\alpha [\eta_{i0}^k]^\beta}, \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

where i is the operation at the current step, o is the operation in candidate list, p_{io} is a probability to select operation o for the next step, τ_{io} is the pheromone trail between operation i and o, η_{io} is the heuristic information between operation i and o. Each group of ant use one of three heuristic as Eq. (4).

$$\eta_0 = \begin{cases} \frac{wr(0)}{\sum_{0 \in C} wr(0)}, & \text{if ant } k \in \text{subcolony}^{1st} \\ \frac{1/p(0)}{\sum_{0 \in C} 1/p(0)}, & \text{if ant } k \in \text{subcolony}^{2nd} \\ \frac{1/d(0)}{\sum_{0 \in C} 1/d(0)}, & \text{if ant } k \in \text{subcolony}^{3rd} \end{cases} \tag{4}$$

where p is the processing time of operation, wr and d are the work remaining and due date of job that operation belong to. C_i is the set of operations in candidate list which generated from non-delay schedule and is a subset of allowable operations, A. After ant k selects one operation this operation will be kept in the visit list, V. Ant repeats the construction step until the unscheduled operation list, U is empty. A sequence, S of all the operations in V is representing a solution for the JSP. Then the objective function, Z of all ants are calculated by synthesizing the three objectives into a weighted sum as Eq. (5). The objective values on the three criteria have to be normalized before they are summed because they are of different scales. The normalized value of $C_{\max}$, $\bar{F}$ and $\bar{T}$, ($C'_{\max}$, $\bar{F}'$, $\bar{T}'$) are calculated as Eqs. (7), (8) and (9) respectively.

$$Z = w_1 C'_{\max} + w_2 \bar{F}' + w_3 \bar{T}', \tag{5}$$

where w_1, w_2 and w_3 denote the weight of three objectives of makespan, mean flow time, and mean tardiness respectively.

$$C'_{\max} = \begin{cases} \frac{C_{\max} - \text{best}C_{\max}}{\text{worst}C_{\max} - \text{best}C_{\max}}, & \text{if worst } C_{\max} \neq \text{best}C_{\max} \\ 1.0, & \text{otherwise} \end{cases} \tag{6}$$

where $C_{\max} = \max(C_i)$, C_i is the completion time of job i. The best $C_{\max}$ and the worst $C_{\max}$ are the best and the worst makespan since start algorithm.

$$\bar{F}' = \begin{cases} \frac{\bar{F} - \text{best}\bar{F}}{\text{worst}\bar{F} - \text{best}\bar{F}}, & \text{if worst } \bar{F} \neq \text{best}\bar{F} \\ 1.0, & \text{otherwise} \end{cases} \tag{7}$$

where $\bar{F} = \frac{1}{n}\sum_{i=1}^{n} F_i, F_i = C_i - r_i$.

F_i is the flow time of job i and r_i is the release time of job i. In the study r_i of all jobs are set to 0. The best $\bar{F}$ and the worst $\bar{F}$ are the best and the worst mean flow time since start algorithm.

Input: A problem instance of the JSP
/* Step 1: Initialization: Set Parameters value*/
Set number of iteration, $N_t = t_{\max}$
Set number of SubColony $N_{sc} = 3$
Set number of ants $N_a = a$
Set Pheromone information weight, $\alpha = \alpha_c$
Set Heuristic information weight, $\beta = \beta_c$
Set Intensify rate, $\rho = \rho_c$
Set Exploration/Exploitation weight, $q_0 = q_{0c}$
Set objective1 weight, $w_1 = w_{c1}$
Set objective2 weight, $w_2 = w_{c2}$
Set objective3 weight, $w_3 = w_{c3}$
For each edge (i,j) **do**
  Set an initial pheromone value $\tau_{ij}(t_0) = \tau_0$
End for
*/*Main loop*/*
/* Step 2: Solution Construction and multi-objective evaluation*/
Set Best solution since start algorithm, $S_s(t_0) = \phi$
Set Best solution in iteration, $S_i(t_0) = \phi$
For $t = 1$ to t_{max} **do**
  For SC $=1$ to 3
    For $k = 1$ to a **do**
    Set Unvisit list, $U^k = \forall O$ /*all operations*/
      Set Visit list, $V^k = \phi$
      Set Allowable list, $A^k = \phi$
      Set Candidate list, $C^k = \phi$
    End for
    For $k = 1$ to a **do**
    / Starting node */*
      Place ant k on the starting node
      Store this information in V^k
      Delete this operation from U^k
   / Build the solution for each ant */*
    End for
    For ant $k = 1$ to a **do**
      Ant builds a tour step by step until $U^k = \phi$ by apply the following steps:
        Ant randomly choose q number, $q = \text{rand}(0,1)$
        Choose the next operation j from C^k according to equation (2) If $q \le q_0$
        Otherwise an operation is selected according to equation (3) using heuristic
          information according to equation (4)
        Keep operation j in V^k and delete operation j from U^k
    End for
    For ant $k = 1$ **to** a **do**
        Compute the function Z^k according to equation (5)
    End for
  End for
/* Step 3: Local Improvement */
  For SC $=1$ to 3
    For $k = 1$ to a **do**
        Ant in iteration perform local improvement
        Select best solution of ants of iteration t
        Update the $S_i(t)$
      Update the $S_s(t)$
        Update *best/worst* C_{max}, *best/worst mean flow time*, *best/worst mean tardiness and best Z*
    End for
  End for

Fig. 1 Procedure of multi-criteria ant colony algorithm

/* Step 4: Update pheromone trial */
  For SC $= 1$ to 3
    For $k = 1$ to a **do**
      For each edge (i,j) in V^k of $S_s(t)$ **do**
        Update pheromone trials according to the equation (11)
      End for
    End for
  End for
/* Step 5: restart process, option */
  If ants can not find better schedule in r iteration the restart process is applied
    Keep the best solution in previous search as the S_s
    Randomly initialize new pheromone value and start step 2.
  End if
End for
Output: Best solution

Fig. 2 Procedure of multi-criteria ant colony algorithm (cont.)

$$\bar{T}' = \begin{cases} \frac{\bar{T}-\text{best}\bar{T}}{\text{worst}\bar{T}-\text{best}\bar{T}}, & \text{if worst } \bar{T} \neq \text{best}\bar{T} \\ 1.0, & \text{otherwise} \end{cases} \tag{8}$$

where

$$\bar{T} = \frac{1}{n}\sum_{i=1}^{n} T_i,$$

$$T_i = \max\{0, L_i\}, L_i = C_i - d_i.$$

T_i is the tardiness of job i, Li is the lateness of job i and d_i is the due date of job i. The best $\bar{T}$ and the worst $\bar{T}$ are the best and the worst mean tardiness since start algorithm. The due date of each job is estimated using TWK method (see Baker, 1984 [1]) as Eq. (9).

$$d_i = r_i + c\sum_{j=1}^{n} p_{ij}, \tag{9}$$

where c denotes the tightness factor of due date. In the study c is set to 1.2, 1.5 and 2 for tight, medium and loose due date.

3.3 Local improvement

The local improvement procedure explores the best solution from a certain neighborhood of a given schedule and keeps it as the solution. The neighborhood solution is defined as follows (see Nowicki and Smutinicki, 1996 [10]).

Denote the critical path C_p in the sequence S by $C_p = (o_1, \cdots, o_w)$, where $o_i \in O$, $1 \leq i \leq w$ and w is the number of operations in this path. The C_p depends on S but for simplicity in notation it will not be expressed explicitly. The critical path is naturally decomposed into subsequences $B_1, \cdots, B_r$ called blocks in S on C_p, where
(1) $B_j = (o_{aj}, o_{aj+1}, \cdots, o_{bj})$, $j = 1, \cdots, r$ and $1 = a_1 \leq b_1 < b_1 + 1 = a_2 \leq b_2 < b_2 + 1 = a_3 \leq \cdots a_r \leq b_r = w$;
(2) B_j contains operations processed on the same machine $M(B_j)$;
(3) Two consecutive blocks contain operations processed on difference machine, $M(B_j) \neq M(B_{j+1})$, $j = 1, \cdots, r-1$.

The neighborhood of solution, $N(S)$ is defined as processing orders obtained from S by applying the move of pair of operation near the border line of blocks on a single C_p. In each critical path the set of moves is defined as Eq. (10).

$$V(C_p) = \bigcup_{j=1}^{r} V_j(C_p), \tag{10}$$

where

$$V_1(C_p) = \begin{cases} \{(o_{b_1-1}, o_{b_1})\}, & \text{if } a_1 < b_1 \text{ and } r > 1 \\ \emptyset, & \text{otherwise} \end{cases}$$

$$V_j(C_p) = \begin{cases} \{(o_{a_j}, o_{a_j+1}), (o_{b_1-1}, o_{b_1})\}, & \text{if } a_j < b_j \text{ and } r > 1 \\ & j = 2, \cdots, r-1 \\ \emptyset, & \text{otherwise} \end{cases}$$

$$V_r(C_p) = \begin{cases} \{(o_{a_r}, o_{a_{r+1}})\}, & \text{if } a_1 < b_1 \text{ and } r > 1 \\ \emptyset, & \text{otherwise} \end{cases}$$

The $V_1(C_p)$ move and $V_r(C_p)$ move express that in the first block the last two operations are swapped, and in the last block the first two operations are swapped respectively. The $V_j(C_p)$ express that the first two (and the last two) operations in every blocks $B_2, \cdots, B_{r-1}$, each of which contains at least two operations are swapped.

Table 1 The best values for each objective

Problem	Size	Optimal $C_{\max}$		$C_{\max}$	$\bar{F}$	$\bar{T}$
LA01	10×5	666	Best $C_{\max}$	666	596.100	254.600
			Best $\bar{F}$	773	503.700	164.500
			Best $\bar{T}$	773	503.700	164.500
			Best Z	666	580.300	239.100
LA06	15×5	926	Best $C_{\max}$	926	805.733	491.000
			Best $\bar{F}$	1107	623.200	304.133
			Best $\bar{T}$	1107	623.200	304.133
			Best Z	1065	670.200	351.133
LA11	20×5	1222	Best $C_{\max}$	1222	1081.550	760.850
			Best $\bar{F}$	1583	760.750	440.050
			Best $\bar{T}$	1583	760.750	440.050
			Best Z	1379	786.600	466.550
LA16	10×10	945	Best $C_{\max}$	988	917.700	277.300
			Best $\bar{F}$	1092	764.000	136.700
			Best $\bar{T}$	1092	764.000	136.700
			Best Z	1075	843.600	201.900
LA21	15×10	1046	Best $C_{\max}$	1185	1082.667	116.133
			Best $\bar{F}$	1485	934.867	18.933
			Best $\bar{T}$	1355	953.000	4.467
			Best Z	1319	999.600	43.733

Remark: $c = 1.2$.

3.4 Pheromone updating

After all the ants complete their solutions, the best solution in iteration, S_i is compared with the best solution found since the start of algorithm, S_s and the best one is used to update its local pheromone matrix. The rule of pheromone updating is defined as Eq. (11).

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \rho\Delta\tau_{ij}(t), \tag{11}$$

where

$$\tau_{ij}(t) = \begin{cases} 1, & \text{if } (i,j) \in \text{path in } S_s \\ 0, & \text{otherwise} \end{cases}$$

ρ is the pheromone evaporating parameter. The minimum pheromone value is set to 0.001.

3.5 Restart process

When the system is trapped in an area of the search space a restart process is performed to diversify to find new, possibly better solution in other search space. In this step if ants can not find better schedule in 100 iterations the restart process is applied. The best solution in previous search is kept as the S_s then the algorithm is started at step 2 with new random value of pheromone trail matrix.

4 Computional results

The proposed algorithm is coded in C and run on an Intel (R) Core (TM) 2 Duo 2 GHz 3GB RAM Windows platform. To investigate the performance of the proposed algorithm, 5 benchmark problems are tested and compared with their optimal solutions first. The results are shown in Tab. 1. Then 25 benchmark problems with three level of tightness are tested. The number of jobs and machines ($n \times m$) are ranging from 10×5 to 15×10. The value of parameters are set as the same numbers for all problem as detailed in Subsection 3.1. Each of problem instances was repeated for ten trials. The best solution is obtained from ten trails of this tested algorithm.

Table 2 Experimental results on benchmark problems

Problem	Size	Tightness	$C_{\max}$	$\bar{F}$	$\bar{T}$	Z_{best}	$\bar{Z}$	σ_z	Time (sec)
LA01	10×5	T	666	580.300	239.100	0.0095	0.0101	0.0009	22
		M	666	591.200	172.400	0.0098	0.0104	0.0017	23
		L	666	572.800	89.100	0.0089	0.0101	0.0009	22
LA02	10×5	T	752	490.000	174.900	0.0071	0.0074	0.0002	21
		M	732	481.600	95.500	0.0059	0.0067	0.0015	22
		L	730	491.100	38.800	0.0041	0.0055	0.0017	21
LA03	10×5	T	703	441.500	156.000	0.0043	0.0048	0.0006	23
		M	693	478.700	132.100	0.0061	0.0064	0.0001	24
		L	703	436.200	23.600	0.0055	0.0059	0.0004	21
LA04	10×5	T	689	483.300	183.500	0.0030	0.0035	0.0005	22
		M	674	480.000	118.000	0.0012	0.0021	0.0011	23
		L	673	466.500	50.600	0.0029	0.0036	0.0009	23
LA05	10×5	T	605	453.800	180.200	0.0029	0.0032	0.0001	25
		M	607	460.300	121.500	0.0036	0.0038	0.0001	21
		L	593	461.600	46.400	0.0025	0.0035	0.0007	20
LA06	15×5	T	1065	670.200	351.133	0.0103	0.0114	0.0018	55
		M	1009	710.267	313.267	0.0125	0.0136	0.0014	56
		L	1017	709.200	194.400	0.0112	0.0122	0.0009	54
LA07	15×5	T	982	637.200	338.000	0.0072	0.0073	0.0001	57
		M	984	628.533	262.000	0.0078	0.0089	0.0015	56
		L	967	656.600	174.533	0.0107	0.0112	0.0005	58
LA08	15×5	T	989	656.533	352.467	0.0055	0.0064	0.0004	55
		M	965	622.933	248.333	0.0039	0.0049	0.0012	56
		L	989	625.867	150.333	0.0072	0.0074	0.0001	57
LA09	15×5	T	1036	762.067	425.067	0.0118	0.0123	0.0019	58
		M	974	802.533	379.200	0.0116	0.0125	0.0007	54
		L	1039	702.667	167.867	0.0069	0.0073	0.0001	55
LA010	15×5	T	1054	681.733	360.600	0.0110	0.0114	0.0005	55
		M	1042	657.867	261.467	0.0073	0.0088	0.0015	59
		L	1043	660.067	140.933	0.0072	0.0074	0.0001	56
LA011	20×5	T	1379	786.600	466.550	0.0098	0.0137	0.0020	141
		M	1366	816.300	416.800	0.0114	0.0136	0.0012	149
		L	1387	856.150	340.400	0.0153	0.0160	0.0015	143
LA012	20×5	T	1177	727.650	447.850	0.0119	0.0131	0.0015	139
		M	1189	686.950	337.950	0.0108	0.0111	0.0002	143
		L	1103	756.900	407.650	0.0110	0.0123	0.0009	142

The results in Tab. 1 show that the best $C_{\max}$ obtained from the proposed algorithm are equal to the optimal $C_{\max}$ especially the small size problem. For the large size problems the deviations of the optimal $C_{\max}$ are less than 15%. The primary results are encouraging. The algorithm is able to find the good solution. For extensive computational study 25 benchmark problems are tested with the tightness factor of due date is set to 1.2 (tight), 1.5 (moderate), and 2 (loose). The results are shown in Tabs. 2 and 3.

5 Conclusion

In this paper the ant algorithm is proposed for the multi-objective JSP. The objective functions used are makespan, mean flow time and mean tardiness. To diversify the search ants use different heuristic information based on priority dispatching rule and intensify the search using local search. The algorithm is tested on several benchmark problems. The results show that the proposed algorithm is able to find the competitive solutions. The extensive study especially for the large size problem should be done in order to investigate the behavior of the proposed algorithm. However from the primary results this algorithm can be considered an alternative for solving the multi-objective JSP. There are many possibilities to improve the solution. First, the values of parameters that control the search should be fine-tuned for each instance. Second, in the primary experiment when the search is time-limited it is found that the solution obtained from non-delay schedule is better than the solu-

Table 3 Experimental results on benchmark problems (cont.)

Problem	Size	Tightness	$C_{\max}$	$\bar{F}$	$\bar{T}$	Z_{best}	$\bar{Z}$	σ_z	Time (sec)
LA013	20×5	T	1314	801.200	491.100	0.0135	0.0143	0.0016	151
		M	1307	758.250	374.200	0.0093	0.0097	0.0001	145
		L	1313	794.400	287.250	0.0128	0.0135	0.0018	141
LA014	20×5	T	1408	910.550	590.250	0.0172	0.0186	0.0023	143
		M	1374	883.100	486.800	0.0123	0.0132	0.0012	143
		L	1344	872.050	341.450	0.0093	0.0104	0.0011	142
LA015	20×5	T	1362	877.200	550.850	0.0111	0.118	0.0004	147
		M	1408	883.750	544.300	0.0129	0.138	0.0015	142
		L	1397	918.800	384.450	0.0155	0.0169	0.0025	146
LA016	10×10	T	1075	843.600	201.900	0.0100	0.0109	0.0009	115
		M	1065	852.400	72.400	0.0099	0.0108	0.0020	112
		L	1076	821.500	0.000	0.0065	0.0075	0.0007	116
LA017	10×10	T	838	791.600	235.400	0.0106	0.0115	0.0010	120
		M	841	789.400	117.800	0.0100	0.0109	0.0013	118
		L	840	803.800	23.100	0.0098	0.0112	0.0015	121
LA018	10×10	T	982	749.800	132.900	0.0044	0.0048	0.0003	115
		M	972	755.300	43.900	0.0049	0.0061	0.0019	117
		L	970	797.100	0.000	0.0043	0.0049	0.0004	118
LA019	10×10	T	903	814.300	173.200	0.0043	0.0048	0.0007	111
		M	942	852.900	64.100	0.0073	0.0099	0.0022	120
		L	927	801.900	0.000	0.0021	0.0033	0.0009	114
LA020	10×10	T	1039	828.400	183.400	0.0076	0.0077	0.0002	117
		M	1009	847.300	87.100	0.0063	0.0075	0.0010	111
		L	1006	850.900	0.000	0.0046	0.0051	0.0003	116
LA021	15×10	T	1319	999.600	43.733	0.0092	0.0105	0.0009	745
		M	1333	995.667	48.333	0.0109	0.0129	0.0011	735
		L	1333	995.667	48.333	0.0109	0.0113	0.0001	738
LA022	15×10	T	1058	971.800	386.400	0.0086	0.0098	0.0003	741
		M	1060	975.867	249.333	0.0089	0.0101	0.0007	752
		L	1060	994.667	92.600	0.0101	0.0122	0.0013	744
LA023	15×10	T	1150	1059.067	412.733	0.0073	0.0077	0.0002	740
		M	1131	1042.000	236.400	0.0052	0.0068	0.0010	745
		L	1153	1044.533	46.733	0.0063	0.0081	0.0007	750
LA024	15×10	T	1086	985.733	368.000	0.0083	0.0097	0.0010	742
		M	1095	997.533	244.133	0.0116	0.0128	0.0013	740
		L	1087	1009.533	76.000	0.0108	0.0114	0.0002	743
LA25	15×10	T	1119	1033.333	433.000	0.0115	0.0125	0.0009	745
		M	1151	1037.067	297.333	0.0125	0.0138	0.0012	748
		L	1135	1020.600	119.333	0.0117	0.0119	0.0001	752

tion obtained from active schedule. If time is not limited searching solution in active schedules should be better for large size problems. Finally, a faster local search method can be added to improve the speed of the algorithm.

References

[1] Baker, K. (1984). Sequencing rules and due date assignments in job shop. *Management Science*, 30(9):1093–1104.

[2] Baykasoğlu, A., Özbakur, L., and Dereli, T. (2002). Multiple dispatching rule based heuristic for multi-objective scheduling of job shop using tabu search. In *Proceedings of MIM: 5th International Conference on Managing Innovations in Manufacturing (MIM), Milwaukee, Wisconsin, USA.*

[3] Behnamian, J., Zandieh, M., and Ghomi, S. F. (2009). A parallel-machine scheduling problems with sequence-dependent setup times using an ACO, SA and VNS hybrid algorithm. *Expert Systems with Applications*, 36:9637–9644.

[4] Choobineh, F., Mohebbi, E., and Khoo, H. (2006). A multi-objective tabu seach for a single-machine scheduling problem with sequence-dependent setup times. *European Journal of*

Operational Research, 175(1):318–337.

[5] Dorigo, M. and Stützle, T. (2004). *Ant colony optimization.* The MIT Press, Massa-chusetts, Cambridge, MA.

[6] Eren, T. and Güner, E. (2008). A bicriteria flowshop scheduling with a learning effect. *Applied Mathematical Modelling*, 32(9):1719–1733.

[7] Gao, J., Gen, M., and et al. (2007). A hybrid of genetic algorithm and bottleneck shifting for multiobjective flexible job shop scheduling problems. *Computer & Industrial Engineering*, 53(1):149–162.

[8] Hoogeven, H. (2004). Multicriteria scheduling. *European Journal of Operational Research*, 167(3):592–623.

[9] Loukil, T., Teghem, J., and Tuyttens, D. (2003). Solving multi-objective production scheduling problem using metaheuristic. *European Journal of Operational Research*, 161(1):42–61.

[10] Nowicki, E. and Smutinicki, C. (1996). A fast tabu search algorithm for the job-shop problem. *Management Science*, 42(6):797–813.

[11] Rajendran, C. and Ziegler, H. (2004). Ant-colony algorithms for permutation flowshop scheduling to minimize makespan/total flowtime of jobs. *European Journal of Operational Research*, 155:426–438.

[12] Varadharajan, T. and Rajendran, C. (2006). A multi-objective simulated-annealing algorithm for scheduling in flowshops to minimize the makespan and total flowtime of jobs. *European Journal of Operational Research*, 167(3):772–795.

[13] Xia, W. and Wu, Z. (2005). An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problem. *Computer & Industrial Engineering*, 48(2):409–425.

[14] Xing, L., Chen, Y., and Yang, K. (2009). Multi-objective flexible job shop schedule: design and evaluation by simulation modeling. *Applied Soft Computing*, 9(1):362–376.