Available online at www.sciencedirect.com
ScienceDirect
Computers & Industrial Engineering 54 (2008) 411–420
computers & industrial engineering
www.elsevier.com/locate/dsw

Ant colony optimization for multi-objective flow shop scheduling problem

Betul Yagmahan [a], Mehmet Mutlu Yenisey [b,*]

[a] Department of Industrial Engineering, Faculty of Engineering and Architecture, Uludag University, Gorukle Campus, Bursa 16059, Turkey
[b] Department of Industrial Engineering, Istanbul Technical University, Macka 34367, Istanbul, Turkey

Received 16 January 2007; received in revised form 13 August 2007; accepted 18 August 2007
Available online 25 August 2007

Abstract

Flow shop scheduling problem consists of scheduling given jobs with same order at all machines. The job can be processed on at most one machine; meanwhile one machine can process at most one job. The most common objective for this problem is makespan. However, multi-objective approach for scheduling to reduce the total scheduling cost is important. Hence, in this study, we consider the flow shop scheduling problem with multi-objectives of makespan, total flow time and total machine idle time. Ant colony optimization (ACO) algorithm is proposed to solve this problem which is known as NP-hard type. The proposed algorithm is compared with solution performance obtained by the existing multi-objective heuristics. As a result, computational results show that proposed algorithm is more effective and better than other methods compared.

Keywords: Flow shop; Scheduling; Multi-objective; Ant colony optimization

1. Introduction

Recently, the flow shop production has been widely used in many industrial areas. For this reason, the flow shop scheduling problem become attentively studied problem over the last 50 years (see Gupta & Stafford, 2006). The objective of this problem mostly focuses to minimize the total completion time, i.e. makespan. Additionally, objectives such as total flow time, tardiness, idle time are also considered.

First research concerned to the flow shop scheduling problem has been done by Johnson (1954). Johnson described an exact algorithm to minimize makespan for the *n*-jobs and two-machines flow shop scheduling problem. Later, several algorithms such as branch and bound, beam search to yield the exact solution for this problem are proposed (e.g. Ashour, 1970; Baker, 1975; Ignall & Schrage, 1965; McMahon & Burton, 1967; Szwarc, 1973). The flow shop scheduling problem includes many jobs and machines. Hence it is classified as combinatorial optimization problem. Therefore, it is in NP-hard problem class and near optimum solution

* Corresponding author. Tel.: +90 212 293 1300x2052; fax: +90 212 240 7260.
E-mail addresses: betul@uludag.edu.tr (B. Yagmahan), yenisey@itu.edu.tr (M.M. Yenisey).

0360-8352/$ - see front matter
doi:10.1016/j.cie.2007.08.003

techniques are preferred to solve. Several heuristic approaches for the flow shop scheduling problem have been developed (e.g. Dannenbring, 1977; Gupta, 1971; Hundal & Rajgopal, 1988; Nawaz, Enscore, & Ham, 1983; Palmer, 1965; Smith & Dudek, 1967; Widmer & Hertz, 1989). In recent years, metaheuristic approaches such as simulated annealing (SA), tabu search (TS), genetic algorithms (GA) are very desirable to solve combinatorial optimization problems regarding to their computational performance. As considering the recent studies for the flow shop scheduling problem, it is obvious that the solution methods based on metaheuristic approach are frequently proposed. Osman and Potts (1989), Taillard (1990), Ogbu and Smith (1991), Ishibuchi, Misaki, and Tanaka (1995), Reeves (1995), and Nowicki and Smutnicki (1996, 2006) are well-known studies for metaheuristics.

Majority of studies for the flow shop scheduling problem focuses to minimize makespan. However, there exist several important objectives like total flow time, total machine idle time other than makespan. These objectives are very useful in minimizing total scheduling cost. That is the reason for considering the flow shop scheduling problem with multi-objectives of makespan, total flow time and total machine idle time in this study. Ho and Chang (1991) have been the first researchers to propose a heuristic procedure to solve this problem. Rajendran (1994, 1995) has presented a new heuristic approach. A recent heuristic based on genetic algorithm using same objective has been developed by Sridhar and Rajendran (1996).

Recently, ACO approach has become the mostly used technique to solve combinatorial optimization problems. The well-known examples for ACO algorithms to solve the flow shop scheduling problem with the objective of makespan are proposed by Stützle (1998), Ying and Liao (2004), and Rajendran and Ziegler (2004).

This study is the first application of ACO metaheuristic to multi-objective flow shop scheduling problem. Performance of proposed algorithm compares with the NEH heuristic (Nawaz et al., 1983) and the Ho and Chang (1991) (HC) heuristic that is the first algorithm to develop for this problem. NEH heuristic results in a solution to minimize makespan for the flow shop scheduling problem. The underlying principle of this heuristic method is that a job with higher total processing time should be given higher priority than a job with less total processing time. However, several modifications are necessary in order to use NEH heuristic for multi-objective flow shop scheduling problem. NEH heuristic basically yields a schedule for makespan. For the required modifications, algorithm is redesigned to compute the other objective values, i.e. total flow time and machine idle time. HC heuristic minimizes not only makespan, but also total flow time and total machine idle time. This heuristic algorithm improves the incumbent schedule minimizing gaps between successive operations in solutions. The pair of jobs with most negative gaps has to be placed at the end of the schedule and that with most positive gaps has to be placed at the beginning of the schedule. To verify the algorithms, computational experiments are conducted on the benchmark problems from Reeves (1995).

The remainder of the paper is organized as follows. In Section 2, we explain the basis of ACO. In Section 3, formulation of problem dealt with in this paper is presented. In Section 4, we present an ant colony system algorithm (ACS) for this problem. The computational experiments and results are given in Section 5. Finally, conclusions are discussed in Section 6.

2. Ant colony optimization

Ant colony optimization is proposed as a new metaheuristic approach for solving hard combinatorial optimization problems in literature (e.g. Stützle & Dorigo, 2003). The basic mechanism of ACO metaheuristic is that a colony of artificial ants cooperates in finding good solutions to combinatorial optimization problems. An important and interesting behavior of ant colonies appears their foraging behavior. In particular, ants are capable of finding the shortest paths between food sources and their nest without using visual cues. While walking from food sources to the nest and vice versa, ants exploit on their ground a substance called pheromone. Ants can smell pheromone substance and, when choosing their way, they tend to choose, in probability, paths marked by strong pheromone concentrations. It has been shown experimentally that a colony of ants to pheromone trail following behavior can find shortest path (see Dorigo & Di Caro, 1999).

The first example of such an algorithm is Ant System (AS) proposed by Dorigo, Maniezzo, and Colorni (1991a, 1991b, 1996), Colorni, Dorigo, and Maniezzo (1992a, 1992b) for the Traveling Salesman Problem (TSP). Then, studies tried to improve its performance and, consequently, various ACO algorithms are proposed. Some of them are ACS by Gambardella and Dorigo (1996), Ant-Q by Gambardella and Dorigo

(1995), MMAS by Stützle and Hoos (1996, 1997) and rank based ant system by Bullnheimer, Hartl, and Strauss (1999). An extensive book concerned with ACO algorithms has recently published by Dorigo and Stützle (2004).

3. Problem formulation

The multi-objective flow shop scheduling problem consists of scheduling of given n jobs with same order and given processing time on m machines. The flow shop scheduling problem has a main assumption, i.e. n jobs process on m machines in the same order (Baker, 1974). We use the following notations:

$t(i,j)$ processing time for job i on machine j $(i = 1, 2, \ldots, n)$, $(j = 1, 2, \ldots, m)$.
n total number of jobs to be scheduled.
m total number of machines in the process.
$\{\pi_1, \pi_2, \ldots, \pi_n\}$ permutation job set.

The makespan, total flow time and total machine idle time objectives considered in this paper can be formulated as follows:

Completion times $C(\pi_i,j)$:

$$C(\pi_1, 1) = t(\pi_1, 1)$$
$$C(\pi_i, 1) = C(\pi_{i-1}, 1) + t(\pi_i, 1) \quad i = 2, \ldots, n$$
$$C(\pi_1, j) = C(\pi_1, j-1) + t(\pi_1, j) \quad j = 2, \ldots, m$$
$$C(\pi_i, j) = \max\{C(\pi_{i-1}, j),\ C(\pi_i, j-1)\} + t(\pi_i, j)\} \quad i = 2, \ldots, n;\ j = 2, \ldots, m$$

Makespan, f_1:

$$f_1 = C(\pi_n, m)$$

Total flow time, f_2:

$$f_2 = \sum_{i=1}^{n} C(\pi_i, m)$$

Total machine idle time, f_3:

$$f_3 = \{C(\pi_1, j-1) + \sum_{i=2}^{n}\{\max\{C(\pi_i, j-1) - C(\pi_{i-1}, j), 0\}\} | j = 2, \ldots, m\}$$

4. Adapting ACS approach for problem

Ant colony system is proposed by Gambardella and Dorigo (1996) to solve traveling salesman problem. The structure of proposed methods in this study to solve the multi-objective flow shop scheduling problem is given in Fig. 1. Firstly, at the initialization step, the pheromone trails, the heuristic information and the parameters are initialized. Secondly, in the iterative step, a colony of ants determines starting jobs. Each ant repeatedly applies the state transition rule to select the next processing job until a complete schedule is constructed. When building a schedule, both the heuristic information and pheromone amount are used to determine the jobs to be chosen.

While constructing the schedule, an ant also decreases the amount of pheromone between selected jobs by applying the local updating rule to vary other ants schedule and avoid in leading to local optima. When all ants have completed their schedules, a local search is applied to the best schedule in order to obtain a better schedule. Then, the global updating rule is applied to increase pheromone between jobs of the best schedule up to the current iteration and decrease pheromone between other jobs. Thus, all the ants will focus on a better schedule. Thirdly, until reaching the maximum number of iterations as a stopping rule, the procedure is repeated.

1. Initialization: The pheromone trails, the heuristic information and the parameters are initialized
2. Iterative loop:
  2.1 A colony of ants determines starting jobs.
  2.2 Construct a complete schedule for each ant:
    Repeat
    Apply state transition rule to select the next processing job
    Apply the local updating rule
    Until a complete schedule is constructed
  2.3 Apply local search process
  2.4 Apply the global updating rule
3. Cycle: If the maximum number of iterations is realized, then STOP;
  Else go to step 2.

Fig. 1. ACS approach.

For proposed algorithm to solve the multi-objective flow shop scheduling problem, the exact position of jobs is important. Here, pheromone trail τ_{ij} is sequence desire of job j to follow job i.

While finding appropriate solution to problem, after an ant k chooses the next job to move to by applying the state transition rule, selected job is added tabu$_k$. Until the last job is selected, procedure is repeated.

4.1. State transition rule

In the process of schedule forming, an ant k in job i selects the job j to move by applying the following state transition rule:

$$j = \begin{cases} \arg\max\limits_{u \in S_k(i)} \{[\tau(i,u)]^{\alpha}[\eta(i,u)]^{\beta}\} & \text{if } q \leqslant q_0 \\ J & \text{otherwise} \end{cases} \tag{1}$$

where, $\tau(i,u)$ is the amount of pheromone trail on edge (i,u), $\eta(i,u) = 1/\delta(i,u)$ is the inverse of the distance $(\delta(i,u))$ between job i and job u. $\eta(i,u)$ denotes the reciprocal cost measure (e.g. distance) between nodes i and u in the original ant colony system. The distance concept (definition of heuristic information) used here will be clarified later. $S_k(i)$ is the set of feasible jobs to be selected by ant k in job i. It should be emphasized that the set of feasible jobs not contained in tabu$_k$. Additionally, α is a parameter that allows a user to control the relative importance of pheromone trail ($\alpha > 0$); β is a parameter that determines the relative importance of heuristic information ($\beta > 0$); q is a value chosen randomly with uniform probability in $[0,1]$ and q_0 is a parameter that determines the relative importance of exploitation versus exploration ($0 \leqslant q_0 \leqslant 1$). J is a random variable selected according to the following random-proportional rule probability distribution, which is the probability of that ant k chooses job j with larger $p_k(i,j)$ to move from job i:

$$p_k(i,j) = \begin{cases} \frac{[\tau(i,j)]^{\alpha}[\eta(i,j)]^{\beta}}{\sum_{u \in S_k(i)}[\tau(i,u)]^{\alpha}[\eta(i,u)]^{\beta}} & \text{if} \quad j \in S_k(i) \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

When an ant in job i chooses a job j to move to, it samples a random number q. If $q \leqslant q_0$, then the best job is chosen according to the Eq. (1), otherwise the best job is chosen according to Eq. (2).

4.2. Local updating rule

While constructing a schedule, an ant decreases the pheromone trail level between selected jobs by applying the following local updating rule:

$$\tau(i,j) = (1 - \rho l) \cdot \tau(i,j) + \rho l \cdot \tau_0 \tag{3}$$

where, τ_0 is the initial pheromone level and ρl $(0 < \rho l < 1)$ is the local pheromone evaporating parameter.

4.3. Global updating rule

Global updating rule is applied after all ants completed their schedules. Global updating provides a greater amount of pheromone trail between adjacent jobs of best schedule. The pheromone trail level is updated as follows:

$$\tau(i,j) = (1 - \rho g).\tau(i,j) + \rho g \cdot \Delta\tau(i,j) \tag{4}$$

where,

$$\Delta\tau(i,j) = \begin{cases} (L_{\mathrm{b}})^{-1} & \text{if } (i,j) \in \text{ best schedule} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

In Eq. (4), ρg $(0 < \rho g < 1)$ is the pheromone evaporating parameter of global updating and L_{b} is the objective function value of the best schedule until the current iteration.

4.4. Heuristic information

The use of heuristic information to manage the ants' probabilistic solution process is important since it provides problem specific knowledge. The heuristic information used in this study is distance between two jobs determined by SPIRIT (Sequencing Problem Involving a Resolution by Integrated Taboo Search Techniques) rule presented by Widmer and Hertz (1989). According to this rule, the distance between job i $(i = 1, 2, \ldots, n \cup N)$ and job u $(u = 1, 2, \ldots, n)$ is given by the following equation:

$$d_{ij} = t_{i1} + \sum_{k=2}^{m} (m - k) * |t_{ik} - t_{j,k-1}| + t_{jm} \tag{6}$$

SPIRIT is based on a weighting of the difference between the processing times of jobs. The distance d_{ij} between two jobs is a measure of increase in objective function value if job i scheduled after job j.

For job i $(i = 1, 2, \ldots, n \cup N)$ and job u $(u = 1, 2, \ldots, n)$ heuristic information is described as follows:

$$\eta(i,u) = \frac{1}{d_{iu}} \qquad (i \neq u) \tag{7}$$

4.5. Candidate list

While a schedule is constructed, one of the possible difficulties arising in ACO algorithms is the big-sized neighborhood. When an ant tries to schedule a job with a big-sized neighborhood, it faces a large number of alternatives to choose. This causes that the schedule construction is slowed down. Moreover, the possibility for ants to schedule the same job becomes very small. The use of candidate lists can reduce the above-mentioned problem.

Candidate list forms a data structure providing additional local heuristic information. Candidate list contains preferred jobs to be scheduled next from a given job. Instead of examining all the alternatives from a job, a job from the candidate list is scheduled. If all jobs in the candidate list are scheduled then other jobs can be examined.

For our study the candidate list contains the *cl* jobs ordered by increasing processing time. The list is scanned sequentially according to tabu list in order to prevent moving unscheduled jobs.

4.6. Local search

ACO algorithms perform the best when coupled with local search algorithms. As soon as global updating rule is completed, local search procedure is performed. Insert local search method is used for proposed ACS algorithm. This procedure is obtained by removing the job at position i and by inserting it at position j. Let (i,j) be a pair of positions and $\pi = (\pi_1, \ldots, \pi_{i-1}, \pi_i, \pi_{i+1}, \ldots, \pi_j, \pi_{j+1}, \ldots, \pi_n)$ the current permutation. The new permutation π^* is obtained by removing the job $\pi(i)$ at a position i and inserting it at position j. If $i < j$ a right insert is performed and we get $\pi^* = (\pi_1, \ldots, \pi_{i-1}, \pi_{i+1}, \ldots, \pi_j, \pi_i, \pi_{j+1}, \ldots, \pi_n)$ and if $i > j$ a left insert is performed and we get $\pi^* = (\pi_1, \ldots, \pi_{j-1}, \pi_i, \pi_j, \ldots, \pi_{i-1}, \pi_{i+1}, \ldots, \pi_n)$.

5. Computational experience

In this section, the results of computational experiments performed to evaluate performance of proposed algorithm for multi-objective flow shop scheduling problem are presented. Twenty-one test problems with the number of jobs varying from 20 to 75 and the number of machines varying from 5 to 20 given by Reeves (1995) are used for experiments. There are three instances for each problem size. The test problems can be downloaded from OR-library web site (http://people.brunel.ac.uk/~mastjjb/jeb/info.html). Performance of solutions to yield using test problem is compared with the NEH heuristic and the HC heuristic that is the first algorithm to develop for this problem. The codes of the proposed algorithm, NEH and HC heuristic were developed in VP and implemented on Intel Pentium M760 2.00 GHz system with 1024 MB DDR-2 RAM.

At this point, some information about parameter analysis would be useful. The details for parameter analysis can be found in Yagmahan (2005) and Yagmahan and Yenisey (2006). Since these publications are in Turkish a brief explanation is given here. Firstly, several experiments were conducted on test problems in order to determine the tendency for the values of parameters. Ten test problems were used for this purpose. Test problems were made of 20 machines and five jobs. Initially, parameters were set as $\alpha = 1$, $\beta = 2$, $\rho l = 0.1$, $\rho g = 0.1$, $q_0 = 0.9$. Other parameters were taken as $k_{\max} = 20$ (number of ants in the colony) and $t_{\max} = 1000$ (maximum number of iteration). In the each step, only one of the parameters was tested. Each test was repeated five times. The candidate values were found as $\alpha \in \{0, 0.5, 1, 2, 5\}$, $\beta \in \{0, 0.5, 1, 2, 5\}$, $\rho l \in \{0, 0.1, 0.2, 0.3, 0.5\}$, $\rho g \in \{0, 0.1, 0.2, 0.3, 0.5\}$ and $q_0 \in \{0, 0.25, 0.5, 0.75, 0.9\}$. Test results showed that these values were suitable for the problem. Later, additional tests were conducted in order to determine the best values. After completing tests, Taguchi Analysis was applied for the lowest two values of parameters.

The best values of computational experiments for flow shop scheduling problem with only makespan objective were obtained for $\alpha = 2$, $\beta = 0.5$, $\rho l = 0.2$, $\rho g = 0.2$, $q_0 = 0.9$, $k_{\max} = 20$, $t_{\max} = 1000$ and $cl = 15$. These values were set for the default value of parameters. Every test was repeated with 10 runs for each instance and the best solution was selected. Hence, there were 210 runs in total.

The relative percentage increase (Δ) in any objective value for schedule T generated by any algorithm is given as follows:

$$\Delta(f_i) = \left(\frac{f_i - \min(f_i, f'_i, f''_i)}{\min(f_i, f'_i, f''_i)}\right) * 100 \tag{8}$$

where, f_i, f'_i, and f''_i are any objective value yielded by HC heuristic, NEH heuristic and proposed algorithm, respectively. Equal relative weighting is chosen to the makespan, total flow time and total machine idle time for total objective value, i.e. $a_1 = a_2 = a_3 = 0.333$. The relative percentage increase in total objective value for schedule T is given as follows:

$$\Delta = a_1\Delta(f_1) + a_2\Delta(f_2) + a_3\Delta(f_3) \tag{9}$$

The relative percentage increase in makespan, total flow time, total machine idle time and total of all objectives based on problem size are presented in Tables 1–4, respectively. Δ_{best} is the relative percentage error of the best solution out of 10 runs, Δ_{avg} is the relative percentage error of average solution out of 10 runs and Δ_{worst} is the relative percentage error of the worst solution out of 10 runs for makespan, total flow time, total machine idle time and total of all objectives.

Table 1 shows that NEH heuristic has yield best performance for makespan objective. It can be seen that proposed algorithm is better than improved HC heuristic and worse than NEH heuristic for makespan objective. Proposed ACO algorithm has the best performance for total flow time, total machine idle time and multiple objectives. On average, an improvement of 2640.38%, 962.72% and 404.42% for, respectively, the best, average and the worst ACO solution according to the best of HC and NEH heuristics was achieved. The com-

Table 1
Performance solutions of algorithms for makespan objective

$n \times m$	Problem number	HC $\Delta(f_1)$	NEH $\Delta(f_1)$	ACO $\Delta(f_1)_{best}$	$\Delta(f_1)_{avg}$	$\Delta(f_1)_{worst}$
20 × 5	*ReC01*	2.99	0.00	3.22	5.63	7.83
20 × 5	*ReC03*	3.00	0.00	0.80	2.16	3.53
20 × 5	*ReC05*	0.00	0.00	0.86	1.23	2.34
20 × 10	*ReC07*	1.56	1.56	0.00	0.00	0.06
20 × 10	*ReC09*	0.06	0.06	0.00	0.03	0.06
20 × 10	*ReC11*	2.65	2.65	0.00	0.00	0.00
20 × 15	*ReC13*	0.00	0.00	0.25	0.40	1.20
20 × 15	*ReC15*	0.89	0.00	1.49	3.71	4.77
20 × 15	*ReC17*	2.26	1.36	0.00	0.00	0.00
30 × 10	*ReC19*	0.00	0.00	0.55	3.45	5.54
30 × 10	*ReC21*	0.66	0.90	0.00	1.73	4.05
30 × 10	*ReC23*	2.76	2.67	0.00	0.00	1.72
30 × 15	*ReC25*	1.46	1.46	0.00	1.27	3.21
30 × 15	*ReC27*	3.32	1.13	0.00	1.18	5.16
30 × 15	*ReC29*	0.00	0.00	1.34	2.34	3.51
50 × 10	*ReC31*	0.47	0.00	2.65	3.88	4.88
50 × 10	*ReC33*	0.63	0.00	2.69	4.26	5.85
50 × 10	*ReC35*	0.76	0.00	2.51	3.78	5.02
75 × 20	*ReC37*	0.93	0.00	0.86	2.65	4.23
75 × 20	*ReC39*	0.13	0.13	0.00	2.04	3.30
75 × 20	*ReC41*	2.32	0.00	0.77	2.05	4.06
Average		1.28	0.57	0.86	1.99	3.35

Table 2
Performance solutions of algorithms for total flow time objective

$n \times m$	Problem number	HC $\Delta(f_2)$	NEH $\Delta(f_2)$	ACO $\Delta(f_2)_{best}$	$\Delta(f_2)_{avg}$	$\Delta(f_2)_{worst}$
20 × 5	*ReC01*	11.46	13.84	0.00	0.00	0.00
20 × 5	*ReC03*	14.95	16.77	0.00	0.00	0.00
20 × 5	*ReC05*	6.87	7.63	0.00	0.00	0.00
20 × 10	*ReC07*	5.23	5.23	0.00	0.00	0.00
20 × 10	*ReC09*	10.11	10.11	0.00	0.00	0.00
20 × 10	*ReC11*	6.96	6.96	0.00	0.00	0.00
20 × 15	*ReC13*	4.47	4.47	0.00	0.00	0.00
20 × 15	*ReC15*	5.62	4.66	0.00	0.00	0.00
20 × 15	*ReC17*	6.63	7.81	0.00	0.00	0.00
30 × 10	*ReC19*	6.74	6.74	0.00	0.00	0.00
30 × 10	*ReC21*	7.20	7.29	0.00	0.00	0.00
30 × 10	*ReC23*	15.38	17.75	0.00	0.00	0.00
30 × 15	*ReC25*	7.43	7.43	0.00	0.00	0.00
30 × 15	*ReC27*	4.82	5.55	0.00	0.00	0.00
30 × 15	*ReC29*	5.73	5.73	0.00	0.00	0.00
50 × 10	*ReC31*	9.19	9.54	0.00	0.00	0.00
50 × 10	*ReC33*	11.44	11.76	0.00	0.00	0.00
50 × 10	*ReC35*	9.47	11.98	0.00	0.00	0.00
75 × 20	*ReC37*	7.11	7.28	0.00	0.00	0.00
75 × 20	*ReC39*	5.30	5.86	0.00	0.00	0.00
75 × 20	*ReC41*	5.87	7.86	0.00	0.00	0.00
Average		8.00	8.68	0.00	0.00	0.00

putation times for HC and NEH heuristics are less than 1 s. The proposed algorithm is very usable in point of view of having acceptable computational time. However, computation time spent for the proposed approach is longer than HC and NEH heuristics.

Table 3
Performance solutions of algorithms for total idle time objective

$n \times m$	Problem number	HC $\Delta(f_3)_{avg}$	NEH $\Delta(f_3)$	ACO $\Delta(f_3)_{best}$	ACO $\Delta(f_3)_{avg}$	ACO $\Delta(f_3)_{worst}$
20×5	*ReC01*	24.05	7.30	0.00	0.00	5.72
20×5	*ReC03*	32.62	15.25	0.00	0.00	0.00
20×5	*ReC05*	9.15	7.82	0.00	0.00	0.00
20×10	*ReC07*	16.06	16.06	0.00	0.00	0.00
20×10	*ReC09*	19.69	19.69	0.00	0.00	0.00
20×10	*ReC11*	19.54	19.54	0.00	0.00	0.00
20×15	*ReC13*	13.84	13.84	0.00	0.00	0.00
20×15	*ReC15*	16.89	27.69	0.00	0.00	0.00
20×15	*ReC17*	10.39	7.23	0.00	0.00	0.00
30×10	*ReC19*	7.78	7.78	0.00	0.00	4.30
30×10	*ReC21*	29.91	30.43	0.00	0.00	0.00
30×10	*ReC23*	21.10	19.28	0.00	0.00	0.00
30×15	*ReC25*	12.02	12.02	0.00	0.00	0.00
30×15	*ReC27*	13.93	10.69	0.00	0.00	0.00
30×15	*ReC29*	27.48	27.48	0.00	0.00	0.00
50×10	*ReC31*	2.97	4.50	0.00	4.57	14.51
50×10	*ReC33*	21.73	17.83	0.00	0.00	0.00
50×10	*ReC35*	16.64	10.06	0.00	0.00	0.21
75×20	*ReC37*	13.59	11.99	0.00	0.00	0.00
75×20	*ReC39*	8.75	6.74	0.00	0.00	2.58
75×20	*ReC41*	6.67	5.15	0.00	0.00	0.00
Average		16.42	14.21	0.00	0.22	1.30

Table 4
Performance solutions of algorithms for multiple objectives

$n \times m$	Problem number	HC Δ	NEH Δ	ACO Δ_{best}	ACO Δ_{avg}	ACO Δ_{worst}
20×5	*ReC01*	12.83	7.05	1.07	1.88	4.52
20×5	*ReC03*	16.86	10.67	0.27	0.72	1.18
20×5	*ReC05*	5.34	5.15	0.29	0.41	0.78
20×10	*ReC07*	7.62	7.62	0.00	0.00	0.02
20×10	*ReC09*	9.95	9.95	0.00	0.01	0.02
20×10	*ReC11*	9.72	9.72	0.00	0.00	0.00
20×15	*ReC13*	6.10	6.10	0.08	0.13	0.40
20×15	*ReC15*	7.80	10.78	0.50	1.24	1.59
20×15	*ReC17*	6.43	5.47	0.00	0.00	0.00
30×10	*ReC19*	4.84	4.84	0.18	1.15	3.28
30×10	*ReC21*	12.59	12.87	0.00	0.58	1.35
30×10	*ReC23*	13.08	13.23	0.00	0.00	0.57
30×15	*ReC25*	6.97	6.97	0.00	0.42	1.07
30×15	*ReC27*	7.36	5.79	0.00	0.39	1.72
30×15	*ReC29*	11.07	11.07	0.45	0.78	1.17
50×10	*ReC31*	4.21	4.68	0.88	2.81	6.46
50×10	*ReC33*	11.27	9.86	0.90	1.42	1.95
50×10	*ReC35*	8.96	7.35	0.84	1.26	1.74
75×20	*ReC37*	7.21	6.42	0.29	0.88	1.41
75×20	*ReC39*	4.73	4.24	0.00	0.68	1.96
75×20	*ReC41*	4.95	4.34	0.26	0.68	1.35
Average		8.57	7.82	0.29	0.74	1.55

6. Conclusions

Recently, the flow shop scheduling problem has been becoming attentively studied field, as using in many industrial areas. There have a lot of researches been done and various solution approaches have been proposed in literature. Most of these studies have been mainly focused on only single criterion. However, many real-world problems require solution approaches which take into account multiple criteria. For this reason, researches recently conducted tend to find solution approaches for multi-objective situation of systems. Hence, we consider the flow shop scheduling problem with multi-objectives of makespan, total flow time and total machine idle time in this study.

Nowadays, metaheuristic approaches have become very popular to solve combinatorial optimization problems in literature because of their performance. When considering researches for the flow shop scheduling problem, solution methods based on metaheuristic approaches are frequently proposed. In this paper, proposed algorithm is based on ACS metaheuristic. In order to verify the performance of proposed algorithm, computational experiments are conducted on the benchmark problems. Results show that the proposed ant colony algorithm performs better than HC and NEH heuristics for the multi-objective flow shop scheduling problem. In addition, by proposed algorithm proves its usability because of having appropriate computational time.

Proposed ACO algorithm can be applied for single or multiple objectives case considering other criteria like mean flow time, total tardiness, and maximum tardiness then discussed in this paper for flow shop problems. Furthermore, it is possible that proposed ant colony algorithm will be able to be used for scheduling problems in various production systems such as job shop, cellular manufacturing, flexible manufacturing, assembly lines.

References

Ashour, S. (1970). An experimental investigation and comparative evaluation of flowshop sequencing techniques. *Operations Research, 18*(3), 541–549.

Baker, K. R. (1974). *Introduction to sequencing and scheduling*. New York: Wiley.

Baker, K. R. (1975). A comparative study of flow shop algorithms. *Operations Research, 23*, 62–73.

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

Colorni, A., Dorigo, M., & Maniezzo, V. (1992a). Distributed optimization by ant colonies. In F. J. Varela & P. Bourgine (Eds.), *Proceedings of the first European conference on artificial life* (pp. 134–142). Cambridge: MIT Press.

Colorni, A., Dorigo, M., & Maniezzo, V. (1992b). An investigation of some properties of an ant algorithm. In R. Männer & B. Manderick (Eds.), *Proceedings of PPSN-II, second international conference on parallel problem solving from nature* (pp. 509–520). Amsterdam: Elsevier.

Dannenbring, D. G. (1977). An evaluation of flowshop sequencing heuristics. *Management Science, 23*(11), 1174–1182.

Dorigo, M., Maniezzo, V., & Colorni, A. (1991a). Positive feedback as a search strategy. Technical Report, 91-016, Italy: University of Milan.

Dorigo, M., Maniezzo, V., & Colorni, A. (1991b). The ant system: An autocatalytic optimizing process. Technical Report, 91-016 (Revised), Italy: University of Milan.

Dorigo, M., Maniezzo, V., & Colorni, 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.

Dorigo, M., & Di Caro, G. (1999). The ant colony optimization meta-heuristic. In D. Corne, M. Dorigo, & F. Glover (Eds.), *New ideas in optimization* (pp. 11–32). London: McGraw-Hill.

Dorigo, M., & Stützle, T. (2004). *Ant colony optimization*. Cambridge: MIT Press.

Gambardella, L. M., & Dorigo, M. (1995). Ant-Q: A reinforcement learning approach to the traveling salesman problem. In A. Prieditis & S. Russell (Eds.), *Proceedings of the Twelfth international conference on machine learning (ML-95)* (pp. 252–260). Palo Alto: Morgan Kaufman.

Gambardella, L. M., & Dorigo, M. (1996). Solving symmetric and asymmetric TSPs by ant colonies. In *Proceedings of the 1996 IEEE international conference on evolutionary computation* (pp. 622–627). Piscataway: IEEE Press.

Gupta, J. N. D. (1971). An improved combinatorial algorithm for the flowshop scheduling problem. *Operations Research, 19*(7), 1753–1758.

Gupta, J. N. D., & Stafford, E. F. (2006). Flow shop scheduling research after five decades. *European Journal of Operational Research, 169*, 699–711.

Ho, J. C., & Chang, Y. L. (1991). A new heuristic for the n-job, m-machine flowshop problem. *European Journal of Operational Research, 52*, 194–202.

Hundal, T. S., & Rajgopal, J. (1988). An extension of Palmer' s heuristic for flowshop scheduling problem. *International Journal of Production Research, 26*, 1119–1124.
Ignall, E., & Schrage, L. (1965). Application of the branch and bound technique to some flow shop scheduling problems. *Operations Research, 13*(3), 400–412.
Ishibuchi, H., Misaki, S., & Tanaka, H. (1995). Modified simulated annealing algorithms for the flow shop sequencing problem. *European Journal of Operational Research, 81*(2), 388–398.
Johnson, S. M. (1954). Optimal two- and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly, 1*(1), 61–68.
McMahon, G. B., & Burton, P. (1967). Flowshop scheduling with branch and bound method. *Operations Research, 15*, 473–481.
Nawaz, M., Enscore, E., & Ham, I. (1983). A heuristic algorithm for the m-machine, n-job flowshop sequencing problem. *OMEGA, 11*(1), 91–95.
Nowicki, E., & Smutnicki, C. (1996). A fast tabu search algorithm for the permutation flow shop problem. *European Journal of Operational Research, 91*, 160–175.
Nowicki, E., & Smutnicki, C. (2006). Some aspects of scatter search in the flow shop problem. *European Journal of Operational Research, 169*, 654–666.
Ogbu, F. A., & Smith, D. K. (1991). Simulated annealing for the permutation flowshop problem. *OMEGA, 19*, 64–67.
Osman, I. H., & Potts, C. N. (1989). Simulated annealing for permutation flow-shop scheduling. *OMEGA, 17*(6), 551–557.
Palmer, D. S. (1965). Sequencing jobs through a multi-stage process in the minimum total time – a quick method of obtaining near optimum. *Journal of the Operational Research Society, 16*(1), 101–107.
Rajendran, C. (1994). A heuristic algorithm for scheduling in flowshop and flowline-based manufacturing cell with multi-criteria. *International Journal of Production Research, 32*, 2541–2558.
Rajendran, C. (1995). Heuristics for scheduling in flowshop with multiple objectives. *European Journal of Operational Research, 82*, 540–555.
Rajendran, C., & Ziegler, H. (2004). Ant-colony algorithms for flowshop scheduling to minimize makespan/total flowtime of jobs. *European Journal of Operational Research, 155*(2), 426–438.
Reeves, C. R. (1995). A genetic algorithm for flow shop sequencing. *Computers and Operations Research, 22*(1), 5–13.
Smith, M. L., & Dudek, R. A. (1967). A general algorithm for solution of the n-job, m-machine sequencing problem of the flowshop. *Operations Research, 15*, 71–82.
Sridhar, C., & Rajendran, C. (1996). Scheduling in flowshop and cellular manufacturing systems with multiple objectives: A genetic algorithmic approach. *Production Planning and Control, 7*, 374–382.
Stützle, T., & Hoos, H. (1996). Improving the ant system: A detailed report on MAX-MIN ant system. Technical Report, AIDA-96-12 (Revised version), Darmstadt: Darmstadt University of Technology.
Stützle, T., & Hoos, H. (1997). The MAX–MIN ant system and local search for the traveling salesman problem. In T. Bäck, Z. Michalewicz, & X. Yao (Eds.), *Proceedings of the 1997 IEEE international conference on evolutionary computation (ICEC'97)* (pp. 308–313). Piscataway: IEEE Press.
Stützle, T. (1998). An ant approach to the flow shop problem. *Proceedings of the sixth European congress on intelligent techniques and soft computing (EUFIT' 98)* (Vol. 3, pp. 1560–1564). Aachen, Germany: Verlag Mainz.
Stützle, T., & Dorigo, M. (2003). The ant colony optimization metaheuristic: Algorithms, applications, and advances. In F. Glover & G. Kochenberger (Eds.), *Handbook of metaheuristics* (pp. 251–285). Norwell, MA: Kluwer Academic Publishers.
Szwarc, W. (1973). Optimal elimination methods in the $m \times n$ flowshop scheduling problem. *Operations Research, 21*, 1250–1259.
Taillard, E. (1990). Some efficient heuristic methods for the flowshop-sequencing problem. *European Journal of Operational Research, 47*(1), 65–74.
Widmer, M., & Hertz, A. (1989). A new heuristic method for the flowshop sequencing problem. *European Journal of Operational Research, 41*, 186–193.
Yagmahan, B. (2005). Ant colony optimization for multi-objective flow shop scheduling problem, Istanbul Technical University, Ph.D. Thesis, Istanbul, Turkey: Department of Industrial Engineering (in Turkish).
Yagmahan, B., & Yenisey, M. M. (2006). Ant colony optimization parameter tuning for flow shop scheduling problem. *ITU Journal Series D: Engineering, 5*(2), 133–141 (in Turkish).
Ying, K. C., & Liao, C. J. (2004). An ant colony system for permutation flow-shop sequencing. *Computers and Operations Research, 31*(5), 791–801.