Contents lists available at Science-Gate

International Journal of Advanced and Applied Sciences

Journal homepage: http://www.science-gate.com/IJAAS.html

Multicriteria scheduling problem: a hybrid ant colony algorithm integrating the decision-maker's preferences

Mohamed Anis Allouche [*], Tahar Jouili, Mohamed Ali Omri

College of Business Administration, Northern Border University, Arar, Saudi Arabia

ARTICLE INFO

Article history:
Received 18 May 2017
Received in revised form
8 August 2017
Accepted 8 August 2017

Keywords:
Multicriteria scheduling problem
Ant colony algorithm
Variable neighborhood search
Satisfaction functions
Decision-maker's preferences

ABSTRACT

The aim of this paper is to present a new hybrid metaheuristic approach based on Ant colony algorithm and the variable neighborhood search noted by ACS_VNS to solve a permutation flowshop scheduling problem. In this context, several criteria are considered which are: the makespan, the total flowtime and the total tardiness of jobs. The proposed approach uses the compromise programming model and the concept of satisfaction function taking into account, explicitly, the decision-maker preferences (DMP). It has been tested through a computational experiment and the obtained results are compared to others for all criteria and for the makespan criterion. The obtained results show the performance of the proposed approach which can be considered as a good tool for multicriteria scheduling problem, especially since it does not necessitate a long computational time.

1. Introduction

As defined by T'kindt and Billaut (2002), a scheduling problem consists of finding the sequence of a set of jobs (tasks) to be processed on different machines, so that technological constraints are satisfied and one or several performance criteria are optimized. Generally, the objectives of scheduling problem are conflicting in nature. Indeed, it is generally impossible to find a sequence in which all conflicting scheduling criteria are optimized simultaneously. Hence, the decision maker needs to make some trade-offs between the scheduling criteria in order to obtain the most satisfactory sequence. Several researchers have studied the problem of multi-criteria scheduling flow shop type. Gangadharan and Rajendran (1994) have considered the minimization of the makespan and the total completion time criteria. They applied a simulated annealing technique to develop their own heuristics. Rajendran and Ziegler (2004) proposed two ant colonies algorithms to solve a scheduling problem in order to minimize simultaneously the makespan and total completion time. Lin et al. (2008) have introduced new features in an ant colony algorithm inspired by the behavior of ants to develop a new algorithm and produce better solutions. Loukil et al. (2005), Varadharajan and Rajendran (2005) have adapted a multi-objective simulated annealing (MOSA) method to solve a multicriteria scheduling problem.

There are several other studies that have been developed in the area of multi-criteria scheduling problem such as: Tavakkoli-Moghaddam et al. (2007), Eren (2007), Javadi et al. (2008), Chang et al. (2008), Ruiz and Allahverdi (2009) and Qian et al. (2009). T'kindt and Billaut (2002) presented a quite complete literature review regarding multi-criteria scheduling theory. These entire works do not take into account, explicitly, the decision maker's preferences in the decisional process. Nevertheless, small numbers of proposed approaches cited in the literature, consider the decision maker's preferences in the multicriteria scheduling problems. Gagné et al. (2004) treated a problem of industrial multi-objective scheduling a single machine with dependent setup times. They proposed a generic procedure to search a compromise solutions based on ant colonies algorithm which takes into account the preferences set by the decision maker. This procedure based on the following three steps: a) determining the ideal point, b) determining the compromise solution and c) presenting effective solutions to the decision maker. For multi-objective scheduling problems, Gagné et al. (2005) have used metaheuristics, based on hybrid Tabu Search/Variable Neighborhood Search (Tabu-VNS), to finding compromise solutions. In the same context, Allouche et al. (2009) have presented an exact method to solve a multi-criteria permutation

* Corresponding Author.
Email Address: medanis.allouche@yahoo.fr (M. A. Allouche)
https://doi.org/10.21833/ijaas.2017.09.023

flowshop problem. This approach based on compromise programming model and the concept of satisfaction functions and that, by incorporating explicitly the preferences of the decision maker. The satisfaction functions measure the intensity of preference regarding the deviations between the achievement and the aspiration levels of the following criteria: makespan, total flow time and total tardiness. Allouche (2010) has proposed a Tabu Search metaheuristic to solve permutation flow shop scheduling problem where several criteria are to be considered, i.e., the makespan, total flowtime and total tardiness of jobs. The Compromise Programming model and the concept of satisfaction functions are used to integrate explicitly the manager's preferences. In this paper, we propose a new metaheuristic approach based on ant colony optimization and variable neighborhood search to solve the permutation flowshop scheduling problem by considering the same criteria used in Allouche (2010). The proposed metaheuristic approach incorporates explicitly the manager's preferences by using the concept of satisfaction functions. It will be tested through a computational experiment and compared to results obtained by Allouche (2010).

This paper is organized as follows. Components and steps of the proposed metaheuristic are given in section 2 and 3 respectively. In section 4, Numerical experiments on several multicriteria scheduling problems are presented and computational results and discussions are presented in section 5. Section 6 concludes the paper with making the focus to the advantages and limits of our work.

2. Metaheuristic components

In this section, we present the different components of the proposed metaheuristic. In fact, it is based on the three main elements: the compromise programming model, the satisfaction functions concept and the hybrid ant colony algorithm.

2.1. The compromise programming model

The Compromise Programming model (CP) was developed first by Zeleny (1973). It is based on minimizing the distance between the achievement level of the objective $q(f_q(x)$ and the ideal value (g_q^*) associated with this objective. The value can be obtained according to the objective of optimization, and the mathematical model form of the CP model is as follows:

Aim to maximise	Aim to minimise
$\text{Min} \sum_{q=1}^{Q} W_q^- \delta_q^-$	$\text{Min} \sum_{q=1}^{Q} W_q^+ \delta_q^+$
Subject to:	Subject to:
$f_q(x) + \delta_q^- = g_q^*$	$f_q(x) - \delta_q^+ = g_q^*$
$(\forall\, q \in Q)$	$(\forall\, q \in Q)$
$x \in F$	$x \in F$
$\delta_q^- \geq 0 \; (\forall\, q \in Q)$	$\delta_q^+ \geq 0 \; (\forall\, q \in Q)$.

where:

W_q^+ *and* W_q^- are, respectively, the weights of positive and negative deviations of the objective q.
F: the set of feasible solutions

2.2. The concept of satisfaction functions

The concept of satisfaction functions was introduced in the "Goal Programming model" by Martel and Aouni (1990). These functions enable the integration of the decision maker's preferences, and this, in connection with the deviations between the levels of achievement and aspiration goals (these funds are set by the decision maker). These functions have the following general form (Fig. 1):

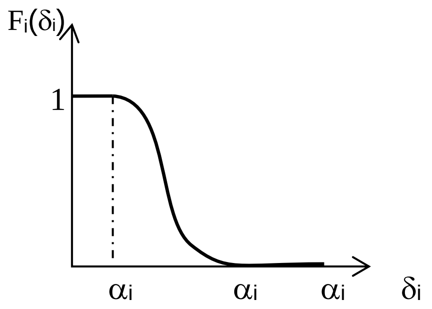

Fig. 1: General form of a satisfaction function

where:
$F_q(\,\delta_q)$: Value of the satisfaction function associated with deviations δ_q,
α_{qd}: Indifference threshold,
α_{qO}: Nil satisfaction threshold,
α_{qv}: Veto threshold.
$\delta_q \in [0, \alpha_{qd}]$: Within this interval, the decision maker's satisfaction level reaches its maximum value of 1.
$\delta_q \in [\alpha_{qd}, \alpha_{qO}]$: Within this interval, the decision maker's satisfaction function is monotonously decreasing.
$\delta_q \in [\alpha_{qO}, \alpha_{qv}]$: Within this interval, the decision maker's satisfaction function is equal to zero. When δ_q exceeds the veto threshold, the solution will be rejected by the decision maker.
Note that many other shapes of satisfaction functions can be found in Martel and Aouni (1990).

2.3. The hybrid ant colony algorithm

The proposed approach is a hybridation of two metaheuristics. The first one is based on ant colony algorithm and the second one is based on the variable neighborhood search (VNS). Note that the Ant colony Optimization (ACO) heuristic was first introduced by Dorigo et al. (1996). It is inspired by the real-life behavior of ants. Several kinds of ant colony algorithm are cited in the literature such as the "Ant System (AS)", proposed by Dorigo et al. (1996), the "Ant Colony System (ACS)" proposed by Gambardella and Dorigo (1996), the "Ant-Q" proposed by Gambardella and Dorigo (1995). In this paper, we used ACS algorithm which is detailed in Fig. 2.

```
(1) Initialization: the pheromone trails, the heuristic information
    and parameters.
(2) Iterative loop:
(3) A colony of ant determines starting jobs.
(4) 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
(5) Apply the global updating rule
(6) Cycle: If the maximum number of iterations is reached, then
    STOP;
  Else go to step 2.
```

Fig. 2: The ACS algorithm

On the other hand, variable neighborhood search (VNS) is one of the recently proposed metaheuristics in the field of combinatorial optimization. It was first introduced by Hansen and Mladenović (1997). It is based on the systematic change of the neighborhood within a possibly randomized local search algorithm. Contrary to other metaheuristics based on local search methods, VNS does not follow a trajectory but explores increasingly distant neighborhoods of a current solution, and jumps from it to a new one if and only if an improvement has been made.

A major advantage of VNS is that it avoids any blockage in a local optimum. Using VNS needs to set some parameters such as: a) the adopted neighborhood structures and b) the order of implementation of these structures. The VNS algorithm is represented in Fig. 3.

```
1. Initialization: find an initial solution x.
2. Repeat the following steps:
a) Randomly generate a solution x' ∈ V(x);
b) Apply to x' procedure k local search until a maximum number
of iterations
To a local optimum x'';
If x' is better than x'' then k = 1 otherwise k = k + 1.
```

Fig. 3: The VNS algorithm

The structure of the proposed algorithm which is a hybridation of ant colony algorithm and variable neighborhood search is presented in Fig. 4.

```
[1] Initialization: the pheromone trails, the heuristic
information and parameters
[2] Iterative loop:
[3] A colony of ant determines starting jobs.
[4] 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
[5] Apply VNS
[6] Apply the global updating rule
[7] Cycle: If the maximum number of iterations is reached, then
STOP;
Else go to step 2.
```

Fig. 4: ACS-VNS algorithm

3. Metaheuristic steps

In this section, we present the different steps of the proposed metaheuristic approach. The first step determines the lower bounds of each criterion (ideal points) and the second step consists of determining the best compromise solutions (sequence).

3.1. Determination of ideal points

In this step, we optimize each criterion separately by using the hybrid ant colonies. The obtained solutions represent the ideal points g_q*.

The basic concepts of the proposed metaheuristic are presented as follows:

- ***The pheromone information***: is presented by a matrix P (*nxn*) where n is the number of tasks. It is initialized at τ_0, a real value fixed by the users and will be modified by ants during the execution process. In this paper, τ_0 is fixed at 0.1. Note that τ_{ij} is considered as the intensity of the pheromone trail for all tasks pairs (i,j) where *i* is the scheduled task and *j* the new task to be scheduled which having the maximum value of τ_{ij}.

- ***The Heuristic information (η_{ij}):*** the choice of the next task (job) to be scheduled is influenced by heuristic information (visibility) related to the characteristics of tasks such as operating (processing) time. This information can be static or dynamic. Note that in our case, it was considered as static. We initialize the elements of matrix H with different values which give priority to tasks having the shortest processing time. The value of an item η_{ij} is calculated as follows:

$$\eta_{ij} = \frac{1}{1+\sum_{r=1}^{m} p_{jr}},$$

where:
p_{jr}: is the operating time of the task *j* on the machine *r*.

We note that the heuristic information represents, for an ant positioned on task *j*, the quality of other unscheduled tasks. Any tasks having the maximum value of η_{ij} will be qualified as a best quality.

- ***The state transition rule***: The choice of the next task scheduling is done by applying the state transition rule that defines how an ant *k* in the position *i* choose the position *j* at time *t*. The algorithm generates a random number *q* belonging to [0, 1]. This number will be compared to q_0. Note that q_0 is a parameter that determines the relative importance of the exploitation of existing information (intensification) and the exploration of new solutions (diversification).

If ($q \le q_0$) Then $j = argmax\{[\tau(i,u)]^{\alpha}[\eta(i,u)]^{\beta}\}$ $u \in S_k(i)$, otherwise *j* = *J*; *J* is chosen according to probability:

$$P_{ij}^{k}(t) = \frac{[\tau(i,j)]^{\alpha}[\eta(i,j)]^{\beta}}{\sum_{u\in S_{k(i)}}[\tau(i,u)]^{\alpha}[\eta(i,u)]^{\beta}}$$

$j \in S_k(i)$

where:
S$_k$ (i): set of feasible tasks that an ant k found in the task i can select. The ant chooses the task j that has the greatest value of P$_k$.
(τ_{ij}): represents the intensity of the pheromone trail on the arc (i, j).
α and β parameters used to vary the relative importance of the trail intensity and the visibility.

We note that, at time t, from an existing partial jobs sequence, each ant k chooses the next job to add using the probabilistic rule $P_{ij}^{k}(t)$ based on visibility (η_{ij}) and the intensity of the pheromone rule (τ_{ij}).

- ***The local updating rule***: when an ant moves from task i to task j, it updates the value of pheromone information τ_{ij} (updated step by step). This update is done in order to reduce the value of τ_{ij} and to make the visited task less attractive for further research to other components (diversification), until a complete scheduling is built. The rule of the local update is given as follows:

$$\tau(i,j) = (1 - \rho l).\tau(i,j) + \rho l.\tau_0$$

where:
ρl: a parameter that determines the amount of the reduction of the pheromone level

- ***The VNS method:*** In this step, the best-obtained solution (schedule) provided by ant colony algorithm will be considered as the initial solution. The VNS method looks for the best solution using the neighborhood exploration. In fact, three neighborhood procedures (structures) are used in the following order:

1. *Insertion procedure*: insert the obtained task in position *a* to a position *b*.
2. *Random permutation procedure*: swapping two tasks in position *a* and *b*.
3. *Two adjacent-tasks permutation procedure:* swapping two adjacent tasks in position *a* and *a+1.*

- ***The global updating rule:*** The purpose of this update is to increase the rate of pheromone between tasks belonging to the best sequence obtained in the previous step until this iteration (intensification). The rule of the global update is given as follows:

$$\tau(i,j) = (1 - \rho g).\tau(i,j) + \rho g.\Delta\tau(i,j)$$

$$\Delta\tau(i,j) = \begin{cases} (L_b)^{-1} \; if \; (i,j) \in best \; sequence \\ 0 \qquad\qquad otherwise \end{cases}$$

where:

L_b: the value of the objective function.
ρg: the global evaporation rate of pheromone,
$0 < \rho g < 1$

3.2. Determination of the best compromise solution

The proposed approach takes into account the decision maker's preferences for each criterion. These preferences are expressed using the satisfaction functions. The retained satisfaction function is presented in Fig. 5.

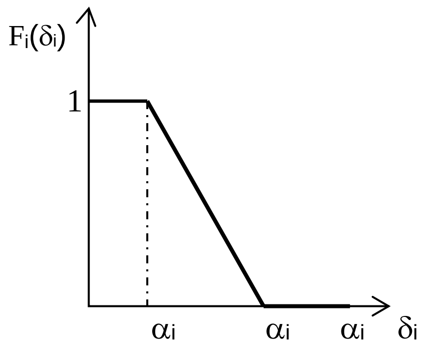

Fig. 5: Retained satisfaction function

The analytical form of this function is:

$$F_1(\delta_1) = \begin{cases} 1 & if \; 0 \le \delta_1 \le \alpha_{qd} \\ \frac{\alpha_{qo} - \delta_1}{\alpha_{qo} - \alpha_{qd}} & if \; \alpha_{qd} \le \delta_1 \le \alpha_{qo} \\ 0 & if \; \delta_1 > \alpha_{qo} \end{cases}$$

For each criterion, these thresholds, α_{qd}, α_{qo} and α_{qv} are fixed by the decision maker. The retained satisfaction function will be applied to the three criteria that we are considering in our computational experiment. In addition to these concepts, the set of Pareto optimal solutions is used. It contains all non-dominated sequences. In fact, the principle of dominance concerns only the value of the optimized criteria and not the value of the objective function.

4. Numerical experiments

The proposed metaheuristic was tested through a set of benchmark problems presented by Taillard (1993) and the obtained results are compared with those of Allouche (2010). These problems have different sizes such as 20 tasks (5-10-20 machines), 50 tasks (5-10 machines) and 100 tasks (5 machines). For each size of problem, we have tested 10 different instances. Taillard's benchmarks were served only for the makespan criterion. A series of numerical experiments is done in two steps. The first one determines the ideal point of each criterion. As to the second step, it incorporates explicitly the decision maker's preferences to determine the best compromise solution. After several experiments, the value of the following parameters is fixed at; α=β=2, q$_0$=0.9, $\rho g = \rho l$ =0.2, R=0.6, T=0.4 Note that all the computations are performed using a Pentium Centrino Duo machine, 1,73- GHz and 1-G- RAM. The proposed approach is coded in C language.

5. Computational results

As we mentioned above, the first step consists of optimizing each criterion separately in order to obtain their best values (ideal points). Table 1 summarizes the best obtained values of the makespan (C^*_{max}), Total flowtime ($\sum C_i^*$) and Total tardiness ($\sum T_i^*$) of a permutation flowshop scheduling problem characterized by 20 tasks and 10 machines (10PF/20/Criterion).

Table 1: Best founded values of 10PF/20/C_{max}, 10PF/20/$\sum C_i$ and 10PF/20/ $\sum T_i$

	Best founded value					
	C^*_{max}	CPU Time	$\sum C_i^*$	CPU Time	$\sum T_i^*$	CPU Time
Tai 20_10_1*	1582	0.703	20911	0.875	4043	0.296
Tai 20_10_2	1659	1	22440	1.609	5411	0.281
Tai 20_10_3	1496	7.39	19877	0.875	3835	0.234
Tai 20_10_4	1378	10	18710	4.656	3466	0.297
Tai 20_10_5	1419	1.266	18679	0.89	2515	0.328
Tai 20_10_6	1397	1.25	19245	0.812	3345	0.625
Tai 20_10_7	1484	1.031	18448	0.813	3247	0.281
Tai 20_10_8	1538	5.86	20241	0.765	4599	0.281
Tai 20_10_9	1593	1.843	20330	0.812	4124	0.313
Tai 20_10_10	1591	1.312	21320	0.828	4403	1.765
Average CPU Time (seconds)		2.265		1.293		0.47

*Tai20_10_1: Data founded in Taillard benchmark for 20 tasks, 10 machines and problem number 1

Regarding the total tardiness criterion, we considered the Daniels and chambers (1990) technique to generate due date of tasks (jobs) of different problems. The due date (d_j) is randomly generated within the following interval:

$$d_j \in \left[ABP\left(1 - T - \frac{R}{2}\right), ABP\left(1 - T + \frac{R}{2}\right)\right]$$
$$\text{ABP} = (n + m - 1)\,\bar{P}$$

and

$$\bar{P} = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} P_{ij}}{n*m}$$

where:

ABP: Average Busy Period that serves as an approximation of the achievement time of the task in the sequence.
T: delay factor or average percentage of overdue tasks, T ∈ (0.4, 0.6, 0.8),
R: control factor of the extent of due dates, R∈(0.2, 0.6, 1),
$\bar{P}$: Mean processing time,
n: number of tasks,
m: number of machines.

It should be noted that the due date for each task is computed by taking T equal to 0.4 and R equal to 0.6 as done by Allouche (2010). The best-founded value of makespan, total flowtime and total tardiness illustrated in Table 1, are the same as obtained by Allouche (2010). Therefore, the ACS-VNS needs less computational time to display solutions than TS method. As mentioned in Table 1, for 20 tasks, 10 machines scheduling problem, the average computational time is 2.265 seconds, 1.293 seconds and 0.47 seconds for the makespan, total flowtime and total tardiness respectively.

The obtained results are compared with those given by Allouche (2010) only for the total flowtime and the total tardiness criteria. Note that, in their approach, Allouche (2010) were used a Tabu search method. Table 2 resumes the obtained results for scheduling problems characterized by 20 tasks and 10 machines.

Table 2: Obtained results with two different methods

	$\sum C_i$	$\sum T_i$	Deviation	ACS-VNS	Tabu Search TS	Deviation
	ACS-VNS	Tabu Search TS				
Tai 20_10_1	20911	20911	0	4043	4450	-407
Tai 20_10_2	22440	22559	-119	5411	5775	-364
Tai 20_10_3	19877	19915	-38	3835	4144	-309
Tai 20_10_4	18710	18710	0	3466	3729	-263
Tai 20_10_5	18679	18713	-34	2515	2730	-215
Tai 20_10_6	19245	19245	0	3345	3854	-509
Tai 20_10_7	18448	18481	-33	3247	3687	-440
Tai 20_10_8	20241	20254	-13	4599	4811	-212
Tai 20_10_9	20330	20353	-23	4124	4145	-21
Tai 20_10_10	21320	21323	-3	4403	4704	-301

Based on the results illustrated in Table 2, we easily note the performance of ACS-VNS approach compared to the Tabu search approach (TS) proposed by Allouche (2010). 70% of results have a negative deviation versus 30% of nil deviation for the total flowtime criterion. For the total tardiness, 100% of results have a negative deviation. Two other parameters that perform ACS-VNS approach are the average of computation time and the average of obtaining satisfaction level of problems which are referred by Tables 3 and 4 respectively.

According to the results displayed in Table 3, it is clear that our approach is more efficient than TS method. In a large cases of scheduling problem (20 tasks-5machines, 20 tasks-10 machines, 20 tasks-20 machines, 50 tasks-5 machines) the average of computation time needed by ACS_VNS to display the compromise solution is very low compared to TS.

Table 3: comparison of average computation time

	Computational Time (seconds)	
Problem Size	ACS-VNS	TS
20_5	0.5389	0.79
20_10	2.2655	11.528
20_20	5.5369	25.229
50-5	12.5296	19.087
50_10	3171.2	47.301

Indeed, for scheduling problem characterized by 50 tasks and 10 machines, the average of computational time is lower than presented by ACO-VNS. This can be explained by the fact that the performance of our approach tends to deteriorate when the problem size increases.

Table 4: comparison of average satisfaction level

	Satisfaction Level	
Problem Size	ACS-VNS	TS
20_5	0.99	1
20_10	0.98	0.95
20_20	0.92	0.91
50-5	1	1
50_10	0.96	0.95

On the other hand, the average of the satisfaction level of the decision maker's for most of the problems are closed to 1 which mean that the decision maker is fully satisfied and the obtained sequences meet his preferences and aspiration levels.

6. Conclusion

In this study, we have presented a new hybrid metaheuristic approach to solve permutation flowshop scheduling problems which incorporates explicitly the decision maker's preferences. The proposed approach based on Ant colony algorithm and the variable neighborhood search noted by ACS_VNS is compared to Tabu search method presented by Allouche (2010). Indeed, it has performed competitively with the best results obtained by Allouche (2010).

The numerical experiments were done on a different set of problems with different sizes. The quality of the obtained solutions confirms the efficiency of the ACS_VNS compared to the TS. Using ACS_VNS, we can produce much better results than TS. It can be used for dependent as well as independent criteria. The obtained solutions minimize the deviation between the achievement level of each scheduling objective and its ideal points.

On the other hand, the proposed approach ACS_VNS is flexible, friendly use, it can support more than three criteria and the decision makers can provide easily the required information for establishing the satisfaction functions. Finally, it can be considered as a good tool for multicriteria scheduling problem, especially since it does not necessitate a long computational time.

Acknowledgment

This paper has been supported by Deanship of Scientific Research of the Northern Border University under the reference number 6-14-1436-5

References

Allouche M, Aouni B, Martel J, Loukil T, and Rebaï A (2009). Solving multi-criteria scheduling flow shop problem through compromise programming and satisfaction functions. European Journal of Operational Research, 192(2): 460-467.

Allouche MA (2010). Manager's preferences modeling within multi-criteria flowshop scheduling problem: A metaheuristic approach. International Journal of Business Research and Management ISSN, 1(2): 33-45.

Chang P, Chen S, Fan C, and Chan C (2008). Genetic algorithm integrated with artificial chromosomes for multi-objective flowshop scheduling problems. Applied Mathematics and Computation, 205(2): 550-561.

Daniels RL and Chambers RJ (1990). Multiobjective flow-shop scheduling. Naval Research Logistics (NRL), 37(6): 981-995.

Dorigo M, Maniezzo V, and Colorni A (1996). Ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 26(1): 29-41.

Eren T (2007). A multicriteria flowshop scheduling problem with setup times. Journal of Materials Processing Technology, 186(1): 60-65.

Gagné C, Gravel M, and Price W (2004). Optimisation multi-objectifs à l'aide d'un algorithme de colonie de fourmis. INFOR: Information Systems and Operational Research, 42(1): 23-42.

Gagné C, Gravel M, and Price W (2005). Using metaheuristic compromise programming for the solution of multiple-objective scheduling problems. Journal of the Operational Research Society, 56(6): 687-698.

Gambardella LM and Dorigo M (1995). Ant-Q: A reinforcement learning approach to the traveling salesman problem. In the 12th International Conference on Machine Learning, Morgan Kaufmann, Burlington, USA: 252–260.

Gambardella LM and Dorigo M (1996). Solving symmetric and asymmetric TSPs by ant colonies. In the IEEE International Conference on Evolutionary Computation, IEEE, Nagoya, Japan: 622-627. https://doi.org/10.1109/ICEC.1996.542672

Gangadharan R and Rajendran C (1994). A simulated annealing heuristic for scheduling in a flowshop with bicriteria. Computers and Industrial Engineering, 27(1-4): 473-476.

Hansen P and Mladenović N (1997). Variable neighborhood search for the p-median. Location Science, 5(4): 207-226.

Javadi B, Saidi-Mehrabad M, Haji A, Mahdavi I, Jolai F, and Mahdavi-Amiri N (2008). No-wait flow shop scheduling using fuzzy multi-objective linear programming. Journal of the Franklin Institute, 345(5): 452-467.

Lin B, Lu C, Shyu S, and Tsai C (2008). Development of new features of ant colony optimization for flowshop scheduling. International Journal of Production Economics, 112(2): 742-755.

Loukil T, Teghem J, and Tuyttens D (2005). Solving multi-objective production scheduling problems using metaheuristics. European Journal of Operational Research, 161(1): 42-61.

Martel J and Aouni B (1990). Incorporating the decision-maker's preferences in the goal-programming model. Journal of the Operational Research Society, 41(12): 1121-1132.

Qian B, Wang L, Huang D, Wang W, and Wang X (2009). An effective hybrid DE-based algorithm for multi-objective flow shop scheduling with limited buffers. Computers and Operations Research, 36(1): 209-233.

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(2): 426-438.

Ruiz R and Allahverdi A (2009). Minimizing the bicriteria of makespan and maximum tardiness with an upper bound on maximum tardiness. Computers and Operations Research, 36(4): 1268-1283.

Taillard E (1993). Benchmarks for basic scheduling problems. European Journal of Operational Research, 64(2): 278-285.

Tavakkoli-Moghaddam R, Rahimi-Vahed AR, and Mirzaei AH (2007). Solving a bi-criteria permutation flow shop problem using immune algorithm. In the IEEE Conference on Computational Intelligence in Scheduling, IEEE, Honolulu, USA: 49-56. https://doi.org/10.1109/SCIS.2007.367669

T'kindt V and Billaut JC (2002). Multicriteria Scheduling: Theory, Models and algorithms. Springer-Verlag, Berlin, Germany.

Varadharajan T and Rajendran C (2005). 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.

Zeleny M (1973). Compromise programming. In: Zeleny M and Cochrane JL (Eds.), Multiple criteria decision making: 262–301. University of South Carolina Press, South Carolina, Columbia.