Sigma J Eng & Nat Sci 38 (1), 2020, 405-420

Sigma Journal of Engineering and Natural Sciences
Sigma Mühendislik ve Fen Bilimleri Dergisi

Research Article

MULTI-OBJECTIVE SCHEDULING BY MAXIMIZING MACHINE PREFERENCES FOR UNRELATED PARALLEL MACHINES

İnci SARIÇİÇEK*[1]

[1]*Eskişehir Osmangazi University , Department of Industrial Engineering, ESKIŞEHIR;*
ORCID: 0000-0002-3528-7342

Received: 14.03.2019 Revised: 14.11.2019 Accepted: 08.01.2020

ABSTRACT

This work proposes to use the fitness scores of jobs to machines in unrelated parallel machine scheduling to maximize machine preferences by using the fitness scores of jobs. A bi-objective mathematical model for the unrelated parallel machine problem with sequence dependent setup times is designed to minimize makespan and maximize machine preferences of jobs. Bi-objective Simulated Annealing Algorithm is proposed for solving large sized problems. A Decision Support System designed for solving problems with objective function of the maximizing machine preferences in combination with other common scheduling objective functions for unrelated parallel machine scheduling problems. By using the proposed system, non-dominated solutions are compared and one solution is selected by considering trade-offs among performance measures of the solutions.
Keywords: Unrelated parallel machine scheduling, sequence dependent setup times, machine preferences, simulated annealing, tabu search.

1. INTRODUCTION

Scheduling is a decision making process with the goal of optimizing one or more objectives as well as allocating scarce resources to tasks over time. It has an important role in manufacturing and service industries [1]. Parallel Machine Scheduling (PMS) allocates jobs to machines in shop environments which have parallel machines. Each job can be performed by any of the machines. Optimal scheduling of critical resources such as machinery and manpower is essential to increase the efficiency, utilization, and profitability of the process to a particular performance criterion. In the shop environments classification, there are three parallel machine environments: identical machines, machines with different speeds, and unrelated machines in parallel. There are m different machines that can process the same job in an Unrelated Parallel Machine Scheduling Problem (UPMSP).

In the reviews of Allahverdi et al. [2, 3], the importance of setup times for real world problems and the classification of the parallel machine scheduling problems (PMSP) are given. Scheduling problems are classified in terms of batching versus non-batching setup times and sequence-independent versus sequence-dependent set-up times. There are resource alternatives

[*] Corresponding Author: e-mail: incisa@gmail.com, tel: (222) 239 37 50 / 3623

and sequence-dependent setup times in the plastic, glass, paper, and textile industries where setup times are significantly lengthy. A job may be assigned to one of the set of resources and consecutive jobs on the same resource must have a minimum setup time between them [4]. A considerable amount of theoretical research has been done in the field of PMS with sequence-dependent setup times. Howev er, the problem of unrelated parallel machine scheduling with sequence-dependent setup (UPMSSDS) times is understudied compared to the identical machine scheduling problem. Furthermore, although the problem of PMS with identical machines has been studied extensively, the unrelated parallel machine scheduling problem with non-batch sequence-dependent setup times has received less attention in the literature [5-10].

Unrelated parallel machine scheduling, which there are multiple machines with different speeds and specifications for different jobs in real-world manufacturing environments, has started to attract wide interest recently. Ravetti et al. [11] addressed the UPMSSDS problem with an objective of minimizing the sum of the makespan and weighted delays. Chen and Chen [12] proposed hybrid metaheuristics for the UPMSSDS problem. Tavakkoli-Moghaddam et al.[13] presented a two-level mixed-integer programming model that minimized bi-objectives that are the number of tardy jobs and the total completion time of all jobs where there are some precedence relationship between the jobs. They proposed an efficient genetic algorithm to solve the bi-objective parallel machine scheduling problem. Ant Colony Optimization algorithm was used for the non-preemptive UPMS problem with machine-dependent and sequence-dependent setup times by Arnaout et al. [14]. All jobs were available at time zero, all times were deterministic, and the objective was to minimize the makespan. In [15], the UPMS problem was solved by using machine and job-sequence dependent setup times with the objective of minimizing the total weighted earliness and tardiness. Lin et al. [16] studied the UPMS problem with sequence and machine-dependent setup times under due date constraints by using an artificial bee colony algorithm to minimize total tardiness. Ying et al. [17] presented makespan minimization for scheduling unrelated parallel machines with setup times. In [18], the problems concerning UPMS with aging effects and deteriorating maintenance activities were studied. For the addressed problems, three types of aging effect model were conducted. In the study of Nadari-Beni et al. [19], a fuzzy bi-objective mixed-integer linear programming model was proposed to minimize workload imbalance and total tardiness simultaneously as a bi-objective formulation for UPMSSDS problem, machine eligibility restrictions, and release dates. Their model was solved by two meta-heuristic algorithms, namely fuzzy multi-objective particle swarm optimization and fuzzy non-dominated sorting genetic algorithm for solving large-scale instances. In the study of Eroglu et al. [20], a genetic algorithm with local search was proposed for the UPMS problem with the objective of minimizing makespan. Afzalirad and Rezaeian [21] proposed hybrid meta-heuristics for unrelated parallel machine scheduling with machine eligibility and precedence constraints. Arroyo and Leung [22] addressed the problem of scheduling unrelated parallel batch processing machines with non-identical job sizes and unequal ready times. Ezugwu and Akutsah [23] proposed an improved firefly algorithm for the UPMSSDS problem. Bektur and Saraç [24], solved a generalized problem of scheduling with a common server, which is the unrelated parallel machine scheduling problem with sequence-dependent setup times and machine eligibility restrictions in the plastic part production industry. A mixed integer linear programming model and two metaheuristics were proposed in the study. Fanjul-Peyro et al. [25] proposed new mixed integer linear programs and a mathematical programming based algorithm for the unrelated parallel machine scheduling problem with machine and job sequence setup times with makespan minimization criterion.

There are various parallel machine scheduling problems with different constraints and specifications so that there are many proposed models for UPMS problems in the literature. Nevertheless, there is a need to take into account machine preferences of the jobs for parallel machines that have different processing capabilities in workshops. Chuang et al. [26] and Huang and Liao [27] tackled the problem of PMS with machine preference in manufacturing of electroetching aluminium foil. Machine prefere nce, in another words the fitness of jobs to machines, is a critical matter for many PMS environments. In the scheduling of different tonnage injection machines, it is important to assign moulds to the machines. The speeds of the machines that can process the job are included as the processing times of the machines in the models. However, for instance a machine with a high speed may not be appropriate for a specific job in terms of tolerances. Other processing specifications of the machines are also important for decision makers. For example, let us assume that a job can be handled on a milling machine or a turning lathe. The literature on machine scheduling suggests that machine-assignment and scheduling by taking machine speeds into account with respect to a particular objective function. On the other hand, in many practical scheduling problems, the decision of using a milling machine or a turning lathe for a certain job needs to be made by taking many criteria such as appropriateness of the machine, scrap rates, tolerances and cost into account other than the machine speed only.

This study focuses on solving the problem of UPMSSDS with machine preference *($Rm\,|\,ST_{sd}$, $M_j\,|\,C_{max}$, MP)*. By using the ranking of the machines with conflicting specifications that can handle an operation, the scores in the range of [0, 1] are used to solve the bi-objective scheduling problem. For this purpose, a multi-objective mathematical model for the problem of UPMSSDS is designed. The objective of the proposed model is to minimizing the makespan and to maximizing the machine preferences of the jobs which they are assigned to. Moreover, bi-objective simulated annealing algorithm is proposed to solve large sized problems.

The remainder of the paper is organized as follows. Problem description and the mathematical model are given in section 2. Proposed model, bi-objective simulated annealing algorithm, is explained in section 3 and experimental results are presented in section 4. Section 5 offers discussions and introduces decision support system and section 6 concludes the paper.

2. THE PROBLEM FORMULATION

2.1. Problem description

In most production environments, there are various machines that can process the same job. Scheduling activities normally consider these machines as parallel machines and scheduling is performed by using the machines' speeds to meet the objective or performance criteria such as the makespan and tardiness. However, there are many machines with different models and types. In a manufacturing company, the machines have numerous conflicting specifications, i.e. cutting tool requirements, table size, power, spindle speed, axis travel, positioning accuracy, and repeatability. Specifications and geometric features of the work-piece material are also important in the selection process. The shape, dimensions, thickness, dimensional tolerances, and surface finish requirements of the parts greatly affect the selection process in determining the suitability of a machine for the job. Moreover, different materials demonstrate unlike reactions when they are exposed to deformation. As a decision-making problem, the selection of the most appropriate machine is of great importance for planning and scheduling. Wide range of existing types and models makes the selection process a complex and difficult task. The fitness of jobs to machines can be determined by using multi-criteria methods based on measures such as tolerance limits, surface finishes tolerances, amortization and scrap rates.

Real world applications need allocation and scheduling of jobs by considering all processing elements. For this purpose, the UPMS problem should be solved by taking into consideration of the fitness of jobs to machines. There are machine preferences considering machine capability for a particular job to parallel machines. Machine preferences are described in the range of [0, 1] and symbolized as A_{ik}. The values in the range of [0, 1] are denotes machine preferences of a particular job to parallel machines. The objective is maximizing machine preferences and minimizing makespan. ($Rm\,|\,ST_{sd}$, $M_j\,|\,C_{max}$, *MP)* denotes the problem of UPMSSDS with machine preference. There are *n* jobs *($J_j = \{J_1, J_2, \dots, J_n\}$)* to be processed on *m* unrelated parallel machines *($M_i = \{M_1, M_2, \ldots, M_m\}$)*. The subscript *j* refers to a job, where the subscript *i* refers to machine [1]. Each job can only be processed on one machine and preemption of jobs is not allowed. Each machine can process only one job at a time. Let p_{ij} denote the processing time of J_j ($j \in \{1, 2, \ldots, n\}$) when assigned to machine M_i ($i \in \{1, 2, \ldots, m\}$). Since machines are unrelated, the processing time p_{ij} of job J_j on machine M_i depends on both job and machine type. Let S_{ijk} denote the machine based sequence-dependent setup time incurred when the job J_j switches to job J_k on machine M_i. Then $S_{ijk} = S_{jk}, \forall i$ and $\forall j \neq k$. M_j symbol denotes the set of machines that can process job *j*, not all machines capable of processing job *j*. A_{ik} denotes the machine preferences of job J_k to machine M_i. It is defined between $0 \leq A_{ik} \leq 1$. The values in the range of [0, 1] denote machine preferences of a particular job to unrelated parallel machines. Decision variable C_{ij} represents the completion time of job J_j at machine M_i. C_{max} is defined as the maximum completion time. Decision variables x_{ijk} and x_{ik} are defined as below.

$$x_{ijk} = \begin{cases} 1, if\ job\ j\ precedes\ job\ k\ on\ machine\ i \\ 0, otherwise \end{cases}$$

$$x_{ik} = \begin{cases} 1, if\ job\ k\ assigned\ to\ machine\ i \\ 0, otherwise \end{cases}$$

2.2. The Mathematical Model

There are general mathematical models for the UPMSSDS problem in the literature [15, 25]. (*Rm*│*ST_{sd}*, *M_j*│*C_{max}*, *MP*) denotes the UPMSSDS with machine preference scheduling problem in this study. A bi-objective mixed integer multi-objective mathematical model is developed by taking into account the machine preferences, which shows the capability of a machine to handle a job.

Objective function 1: (1)
Maximizing machine preferences

$$\boldsymbol{Max} \sum_{i} \sum_{k} A_{ik}\, x_{ik}$$

Objective function 2: Minimizing makespan (2)

$$\boldsymbol{Min\ C_{max}}$$

$$\sum_{i \in M} \sum_{\substack{j \in \{0\} \cup \{N\} \\ j \neq k}} x_{ijk} = 1 \qquad \forall k \in N \qquad (3)$$

$$\sum_{i \in M} \sum_{\substack{k \in N \\ j \neq k}} x_{ijk} \leq 1 \qquad \forall j \in N \qquad (4)$$

$$\sum_{k \in N} x_{i0k} \leq 1 \qquad \forall i \in M \qquad (5)$$

$$\sum_{\substack{h \in \{0\} \cup \{N\} \\ h \neq k, h \neq j}} x_{ihj} \geq x_{ijk} \qquad \forall j, k \in N, j \neq k, \forall i \in M \qquad (6)$$

$$\sum_{j \in \{0\} \cup \{N\}} x_{ijk} = x_{ik} \qquad \forall i \in M, j \neq k, \forall k \in N \qquad (7)$$

$$\boldsymbol{C_{ik} + V(1 - x_{ijk}) \geq C_{ij} + S_{ijk} + p_{ik}} \qquad \forall \boldsymbol{j} \in \{\boldsymbol{0}\} \cup \{\boldsymbol{N}\}, \quad \forall \boldsymbol{k} \in \boldsymbol{N}, \quad \boldsymbol{j} \neq \boldsymbol{k}, \quad \forall \boldsymbol{i} \in \boldsymbol{M} \tag{8}$$

$$\boldsymbol{C_{i0} = 0} \qquad \forall \boldsymbol{i} \in \boldsymbol{M} \tag{9}$$

$$\boldsymbol{C_{ij} \geq 0} \qquad \forall \boldsymbol{j} \in \boldsymbol{N}, \forall \boldsymbol{i} \in \boldsymbol{M} \tag{10}$$

$$C_{max} \geq C_{ij} \qquad \forall j \in N, \forall i \in M \tag{11}$$

$$\boldsymbol{x_{ijk}} \in \{\boldsymbol{0}, \boldsymbol{1}\}, \boldsymbol{x_{ik}} \in \{\boldsymbol{0}, \boldsymbol{1}\}, \forall \boldsymbol{j} \in \{\boldsymbol{0}\} \cup \{\boldsymbol{N}\}, \forall \boldsymbol{k} \in \boldsymbol{N}, \boldsymbol{j} \neq \boldsymbol{k}, \forall \boldsymbol{i} \in \boldsymbol{M} \tag{12}$$

V is a large positive number chosen to be larger than the workshop time horizon.

The objective function given by Eq. (1) represents the maximization of total machine preferences of jobs. The objective function in Eq. (2) is used to minimize the maximum completion time (makespan). Constraint set (3) ensures that every job is assigned to exactly one machine. With constraint set (4) the maximum number of successors of every job set to one. Set (5) limits the number of successors of the dummy jobs to a maximum of one on each machine. With Set (6), jobs are properly linked in machine: if a given job j is processed on a given machine i, a predecessor h must exist on the same machine [15]. In addition to constraints (3-6) proposed by Vallada and Ruiz [15], constraint (7) is used to determine whether job J_k is processed on machine M_i or not. It relates the decision variables *xijk* and *xik*. The first objective needs the decision variable *xik* in order to obtain the total job-machine preferences of jobs depend on the machine to which they are assigned. Constraint set (8) is to control the completion times of the jobs at the machines. Sets (9) and (10) define completion times as 0 for dummy jobs and non-negative for regular jobs, respectively. Set (11) defines the maximum completion time. Finally, set (12) defines the binary variables.

The proposed mixed integer mathematical model for the problem is coded with GAMS 23.3, CPLEX 12.1 solver for each objective function and it is used to test the performance of the metaheuristics in small problems. All experiments are performed on a PC with an Intel(R) Core (TM) 2 Quad CPU Q8400 processor, 2.66 GHz and 4 GB RAM.

3. PROPOSED MODEL

3.1. Simulated Annealing and Tabu Search Algorithms

In this work, simulated annealing (SA) and tabu search (TS) algorithms are employed to solve large-scale UPMS problems. The performance of the algorithms is tested on small test problems.

A solution is represented by a string of numbers comprising a permutation of n jobs for m machines as denoted by the set (0, 1, 2, . . ., n-1). The number (-1) is used to separate machines. For example, the solution representation shown in Figure 1 can be decoded as follows. Ten jobs are to be processed on five machines. The job operation sequences are: 4 and 3 on machine 1; 6, 7, 9, 8, and 5 on machine 3; 2 and 1 on machine 5. Job 0 is assigned to machine 2. No jobs are assigned to machine 4.

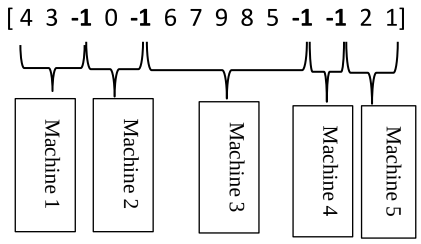


Figure 1. Solution representation for 5 machines 10 jobs.

The (m, n) shows the number of machines and number of jobs respectively. To evaluate the performance of the proposed model, two small test problems are generated for (5, 10) and (10, 15) pairs. Processing times are derived by using the uniform distribution (20, 50). Setup times are derived from the U (5, 20), U (20, 40), and U (40, 80) distributions for the short (SST), medium (MST), and long (LST) times, respectively. Machine preferences are in the range of [0, 1] random numbers. Due dates are derived by taking the average setup times and processing times for each job into account and by using alpha and beta coefficients:

$$\left[\propto\left(\frac{\sum_j s_{jk}}{n-1}+\frac{\sum_i p_{ij}}{m}\right), \beta\left(\frac{\sum_j s_{jk}}{n-1}+\frac{\sum_i p_{ij}}{m}\right)\right]$$

The width of the intervals is adjusted by the alpha and beta coefficients. The proposed TS and SA algorithms are coded in Microsoft Visual Studio 11.0. Since the coded SA and TS algorithms would be used in the stage of the multi-objective metaheuristic, CPLEX solver results are compared to the SA and TS results in order to test the algorithm's performance. In Table 1 and Table 2, the performance of the SA and TS is tested for the objectives of *min* C_{max} and *max MP*.

Table 1. Comparisons of the optimal solutions with SA and TS results for the objective of *min* C_{max}

Test Problems	*Min* C_{max} Objective Function Value			CPU(seconds)		
	CPLEX	SA	TS	CPLEX	SA	TS
Dataset1 (5, 10) SST	25	27	25	0.016	0.012	0.712
Dataset2 (5, 10) MST	41	41	41	0.015	0.141	0.733
Dataset3 (5, 10) LST	66	73	66	0.000	0.134	0.741
Dataset4 (10, 15) SST	15	19	20	0.032	0.185	1.159
Dataset5 (10, 15) MST	30	35	35	0.047	0.193	1.135
Dataset6 (10, 15) LST	53	57	59	0.031	0.211	1.142
Dataset7 (10, 20) SST	<45*	36	33	>10800*	0.572	1.221

* The GAMS/CPLEX solver did not reach the optimal solution within 10,800 seconds.

In Table 1, both SA and TS algorithms give the optimal or near-optimal solutions compared to the optimal solutions obtained by using CPLEX solver. The computational results of the data sets are presented for the same number of iterations (5000 iterations). The GAMS/CPLEX solver did not reach the optimal solution within 10800 seconds for the dataset7. CPLEX solver requires long CPU times to find an optimal solution for large-scale problems. When the number of job increases, the computational time dramatically increases. This issue confirms the need for metaheuristics for large-scale NP-hard problems. Table 1 shows that SA and TS algorithms perform similarly for minimizing the C_{max} objective function. Both algorithms found the optimal solution for the dataset2. TS algorithm found the optimal solutions for the dataset1, dataset2, and dataset3.

Table 2. Comparisons of the optimal solutions with SA and TS results for the objective of *max MP*.

Test Problems	*Max MP* Objective Function Value			CPU (seconds)		
	CPLEX	SA	TS	CPLEX	SA	TS
Dataset1 (5, 10) SST	7.7520	7.15	7.75	0.032	0.022	0.331
Dataset2 (5, 10) MST	7.7520	6.51	7.75	0.063	0.019	0.328
Dataset3 (5, 10) LST	7.7520	6.59	7.75	0.109	0.015	0.331
Dataset4 (10, 15) SST	13.1872	10.51	13.11	0.156	0.014	0.526
Dataset5 (10, 15) MST	13.1872	10.82	13.13	0.125	0.014	0.510
Dataset6 (10, 15) LST	13.1872	10.37	13.11	0.218	0.015	0.506
Dataset7 (10, 20) SST	17.8172	13.58	17.38	1.377	0.016	0.461

In Table 2, it is observed that TS algorithm gives better results than SA regarding the Max MP objective function for the same number of iterations. The solution time of SA algorithm is less than that of TS. SA algorithm has only one neighbourhood solution for each iteration while the size of neighbourhood is more than one neighbourhood solution for TS algorithm. This is an expected result when the solution times of two metaheuristics are compared.

In Table 1 and 2, it can be seen that the CPU times of SA are less than the CPU times of TS. On the other hand, the results of TS algorithm are better than SA algorithm results. There are trade-offs between solution times and solutions. Both are used in terms of variability in the solution space to find non-dominated solutions by using bi-objective SA algorithm.

3.2. The Proposed Bi-Objective Simulated Annealing Algorithm

The metaheuristics have emerged as a major advantage to solve large-scale NP-hard scheduling problems and have been started to use in multi-objective decision-making problems. Lin and Ying [28] proposed a multi-objective multi-point simulated annealing (MOMSA) algorithm for solving a multi-objective scheduling problem. In this paper, a MOMSA based algorithm is proposed for solving bi-objective UPMSSDS problems. The objectives of the algorithm contain the maximization of the machine preferences (*max MP*) and the minimization of the maximum completion time (*min* C_{max}). The proposed algorithm can be presented step-by step as follows.

Step 1: Generate solutions by using SA and TS algorithms for the *max MP* objective (K_1). Obtain *List1* from the solutions of the two algorithms.
Step 2: Generate solutions by using SA and TS algorithms for the *min* C_{max} objective (K_2). Obtain *List2* from the solutions of the two algorithms.
Step 3: Select relatively good 50 solutions from the *List* 1 for the *max MP* objective and select relatively good 50 solutions from the *List2* for the *min* C_{max} objective. Combine solutions in '*the good solutions list*", by giving the solution, the value of the objective function 1 of the solution (F1) and the value of the objective function 2 of the solution (F2), respectively. Delete duplicate solution if there is a same solution in *the good solutions list* from the *List1 and List2*. Compute the other objective function (1 or 2) values for the solutions in *the good solutions list*. Compute the number of good solutions, N_{sol}, in the list.
Step 4: Select non-dominated solutions from *the good solutions list* by using the algorithm of Mishra and Harit [29] for finding *the non-dominated set, S1*.

Step 5: Find *new non-dominated solutions* by using MOMSA based Bi-objective SA Algorithm.

Bi-objective SA algorithm
($I_{iter} - Iteration\ number,\ T_0 - Initial\ temperature,$ α − Cooling ratio)

```
1. Select $K_1$ as maximum $MP$ and $K_2$ as minimum $C_{max}$ by using the good solutions list for each objective $MP$ and $C_{max}$, respectively
2. Select a random solution ($currentState$) from $non\ dominated\ set\ S1$
3. Set $T \leftarrow T_0, Iter \leftarrow 0, N_{Sol} \leftarrow number\ of\ good\ solutions, I_{iter} \leftarrow 2000$, $\alpha = 0.9$
4. While($Iter < I_{iter}$)
4.1. Randomly generate $neighborList$ of $currentState$
4.2. For each $neighborState$ in $neighborList$
4.2.1. $\Delta_{MP} = MP(neighborState) - MP(currentState)$
4.2.2. $\Delta_{C_{max}} = C_{max}(neighborState) - C_{max}(currentState)$
4.2.3. $P_{Accept} \leftarrow 1$
    IF($\Delta_{\boldsymbol{MP}} < 0$) Then $P_{Accept} = P_{Accept} \times Exp(-\Delta_{\boldsymbol{MP}}/(K_1T))$
    IF($\Delta_{C_{max}} > 0$) Then $P_{Accept} = P_{Accept} \times Exp(-\Delta_{C_{max}}/(K_2T))$
    Generate $r \sim U(0,1)$
    IF( $r \leq PAccept$) Then $currentState = neighborState$
  Update $non\ dominated\ set\ S1$
     $N_{Sol} \leftarrow N_{Sol} + 1$
4.2.4. Update the value of $K_1$ and $K_2$
          4.3. $T = \alpha * T$
          4.4. $Iter \leftarrow I_{iter} + 1$
          4.5. Select a random solution ($currentState$) from the $non\ dominated\ set\ S1$
5. Print the non-dominated set S1
```

T_0 denotes the initial temperature and cooling ratio is α for the algorithm. The algorithm uses a neighbourhood search technique by progressing iteratively from one solution to another until a stopping condition (iteration number) is satisfied. Non-dominated set is obtained by using the bi-objective simulated annealing algorithm. A non-dominated solution where an objective function can not be improved without deteriorating the other objective function.

3.3. Experimental results

The functioning of the algorithm is shown on dataset1 (given in Appendix), the UPMSSDS problem with 5 machines and 10 jobs for '*max. MP*' and '*min. C_{max}*' objective functions. The good solutions list obtained by running SA and TS algorithms for the *max. MP* and *min. C_{max}* objective functions for data sets of 5 machines and 10 jobs are given in Table 3.

Table 3. The good solutions list by using SA and TS algorithms.

Solutions	**Max. MP**	**Min. C_{max}**
[4 0 -1 -1 5 7 3 6 8 -1 2 9 -1 1]	7.75	149
[4 0 -1 -1 5 3 7 2 6 8 -1 9 -1 1]	7.71	179
[4 0 -1 -1 7 3 5 2 6 8 -1 9 -1 1]	7.71	171
[0 4 -1 1 -1 2 5 7 3 6 8 -1 9 -1]	7.71	174
[0 1 4 -1 -1 6 5 8 7 3 -1 9 2 -1]	7.59	151
[0 1 4 -1 -1 6 5 8 7 3 9 -1 2 -1]	7.43	183
[0 1 4 -1 -1 3 5 8 7 6 9 -1 2 -1]	7.43	171
[0 3 4 -1 -1 1 5 8 7 6 9 -1 2 -1]	7.07	153
[0 3 4 1 -1 -1 5 9 7 6 8 -1 2 -1]	7.02	125
[0 1 -1 5 -1 7 8 2 9 3 4 6 -1 -1]	6.84	203
[4 3 -1 0 -1 6 7 9 8 5 -1 -1 2 1]	6.5	138
[8 0 4 -1 -1 3 9 5 2 1 -1 7 -1 6]	6.38	147
[-1 6 9 -1 0 3 4 5 2 1 -1 7 -1 8]	5.53	152
[0 -1 -1 8 3 6 2 -1 4 -1 9 5 1 7]	5.46	94
[8 -1 6 4 -1 7 3 -1 2 -1 1 9 0 5]	4.65	103
…..		
[5 9 -1 4 3 -1 8 1 2 -1 0 6 -1 7]	3.79	28
[7 5 -1 4 3 -1 6 1 -1 2 0 8 -1 9]	3.7	43
[5 9 -1 4 6 -1 8 1 -1 0 7 -1 3 2]	3.69	25
[7 9 -1 4 3 -1 8 1 -1 0 6 2 -1 5]	3.48	35
[2 -1 5 6 -1 1 -1 0 7 8 -1 4 3 9]	3.47	51
[7 5 -1 4 3 -1 8 1 2 -1 0 6 -1 9]	3.41	28
[1 7 5 -1 6 3 -1 8 0 -1 2 -1 9 4]	3.29	35
[9 -1 8 -1 2 1 4 -1 7 0 6 -1 5 3]	3.09	55
[7 5 -1 4 3 -1 8 1 -1 6 0 -1 2 9]	3	27
[7 9 -1 4 3 -1 2 1 -1 0 6 8 -1 5]	2.88	35
[7 9 -1 4 3 -1 0 1 -1 2 6 8 -1 5]	2.76	40
[7 5 -1 4 3 -1 0 1 -1 2 6 8 -1 9]	2.73	40
[1 5 7 -1 3 2 4 -1 9 -1 0 8 -1 6]	2.52	48
[1 5 7 -1 3 2 4 -1 9 -1 0 8 6 -1]	2.5	48
[7 5 1 -1 3 2 4 -1 9 -1 6 8 -1 0]	2.39	43
[1 5 7 -1 3 2 4 -1 9 -1 6 8 -1 0]	2.39	48

Non-dominated solutions given in Table 4 are selected by using the algorithm of Mishra and Harit [29] from the good solutions list.

Table 4. Non-dominated solutions (S1)

Solutions	**Max. MP**	**Min. C_{max}**
[4 0 -1 -1 5 7 3 6 8 -1 2 9 -1 1]	7.75	149
[0 3 4 1 -1 -1 5 9 7 6 8 -1 2 -1]	7.02	125
[0 3 4 8 -1 5 -1 9 2 6 7 -1 1 -1]	6.88	117
[0 3 4 8 -1 5 -1 -1 2 6 7 -1 1 9]	5.54	97
[0 9 -1 7 4 -1 6 1 -1 2 5 -1 3 8]	5.34	46
[1 0 -1 3 7 -1 4 6 -1 5 2 -1 8 9]	5.27	43
[6 0 -1 3 7 -1 4 1 -1 9 2 -1 5 8]	4.91	41
[6 1 -1 9 4 -1 8 5 -1 2 7 -1 3 0]	4.88	37
[5 1 -1 0 4 -1 8 6 -1 3 2 -1 7 9]	4.63	34
[0 8 -1 3 6 -1 9 1 -1 2 7 -1 4 5]	4.59	33
[0 8 1 -1 5 -1 4 9 -1 7 6 -1 3 2]	4.55	27
[0 8 -1 4 3 -1 1 2 -1 7 6 -1 5 9]	3.96	26
[8 0 -1 4 3 -1 1 2 -1 7 6 -1 5 9]	3.96	26
[5 9 -1 4 6 -1 8 1 -1 0 7 -1 3 2]	3.69	25

New non-dominated solutions given in Table 5 are obtained by using MOMSA based bi-objective SA algorithm. Dominated and non-dominated solutions are demonstrated in Figure 2.

Table 5. New non-dominated set by using bi-objective SA algorithm

Solutions	**Max. MP**	**Min. C_{max}**
[0 4 -1 -1 3 8 6 7 5 -1 2 9 -1 1]	7.75	147
[4 -1 5 0 1 7 -1 8 2 6 3 -1 9 -1]	7.38	105
[0 3 4 8 -1 5 -1 -1 2 6 7 -1 1 9]	5.54	97
[0 1 4 8 -1 5 -1 -1 2 6 7 -1 3 9]	5	90
[1 0 8 -1 7 5 -1 6 -1 3 -1 2 9 4]	4.93	46
[1 5 0 -1 7 6 -1 2 9 -1 4 -1 3 8]	4.79	37
[0 8 1 -1 5 -1 4 9 -1 7 6 -1 3 2]	4.55	27
[0 8 -1 4 3 -1 1 2 -1 7 6 -1 5 9]	3.96	26
[8 0 -1 4 3 -1 1 2 -1 7 6 -1 5 9]	3.96	26
[5 9 -1 4 6 -1 8 1 -1 0 7 -1 3 2]	3.69	25

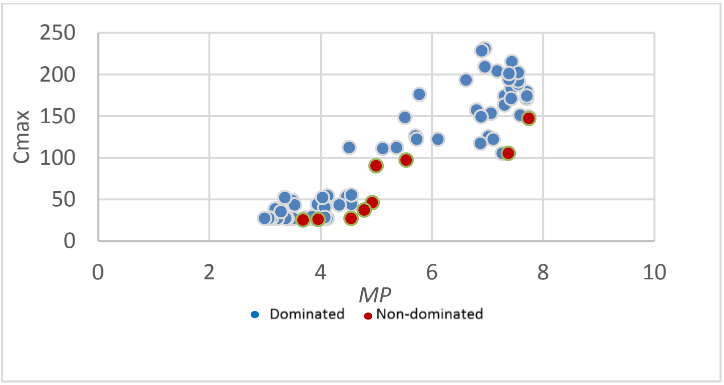


Figure 2. Dominated and non-dominated solutions for the illustrated problem.

Additional non-dominated solutions can be obtained by using bi-objective SA algorithm. In real-life problems, it is desirable that jobs are finished as soon as possible and that jobs are assigned to machines with high preference value. The proposed model provides alternatives to scheduler/decision maker. Decision-maker can select the appropriate solution from the non-dominated solutions list and carry out production planning activities.

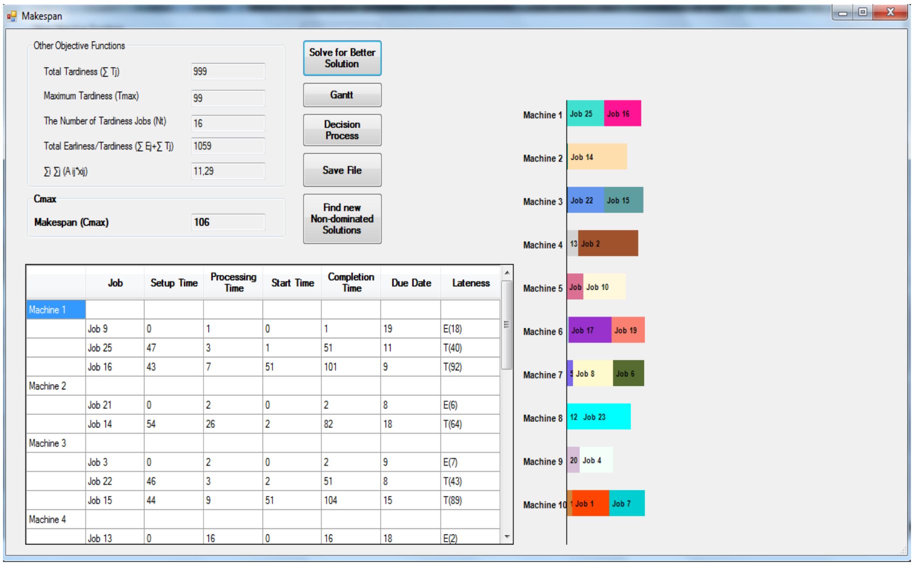
Decision-makers can select one of the non-dominated solutions to the work-shop according to consider trade-offs among schedules. Since an objective function value improves while another worse in multi-objective problems, if a decision-maker selects the last solution in Table 5 with the lowest value (25) in terms of the minimizing C_{max}, that solution is also the one with the worst value in the table in terms of the maximizing *MP* (3. 69). The C_{max} value in the first solution of the table with the greatest value in terms of the maximizing *MP* is 147. Therefore, it is the solution with the worst value in terms of the C_{max} objective value. However, it also has the greatest *MP* value with 7.75. Decision-makers need to consider trade-offs and select the appropriate schedule from the non-dominated solutions list for their workshops.

4. A DECISION SUPPORT SYSTEM TO SOLVE UPMSSDS WITH MACHINE PREFERENCES

Decision maker may want to consider other objective functions besides the makespan while the machine preferences of jobs are maximized. It is desired to maximize the machine preferences of jobs and minimize total tardiness, to minimize the number of tardy jobs, or to minimize maximum tardiness.

The objective to be minimized is always a function of the completion times of jobs. C_j denotes the completion time of job j. D_j represents the due date of job j. The lateness of job j is defined as $L_j = C_j - D_j$, which is positive when job j is completed late and negative when it is completed early. The tardiness of job j is defined as $T_j = max\ (L_j, 0)$ and the earliness of job j is defined as $E_j = max\ (-L_j, 0)$. *The* total tardiness ($\sum T_j$), maximum tardiness (T_{max}), total earliness and tardiness $\sum E_j + \sum T_j$ and the number of tardy jobs (n_t) are not only measures of academic interest, but are often objectives in practice.

In this study, a Decision Support System (DSS) model is designed to solve UPMSSDS with machine preference. The proposed model makes it possible to solve problem for common objective functions while maximizing machine preferences. A solution of the UPMSSDS problem with 10 machines and 25 jobs regarding the *Min.* C_{max} objective function is shown in Figure 3.



Other Objective Functions

Total Tardiness (∑ Tj)	999
Maximum Tardiness (Tmax)	99
The Number of Tardiness Jobs (Nt)	16
Total Earliness/Tardiness (∑ Ej+∑ Tj)	1059
∑ ∑ (A ij*xij)	11,29

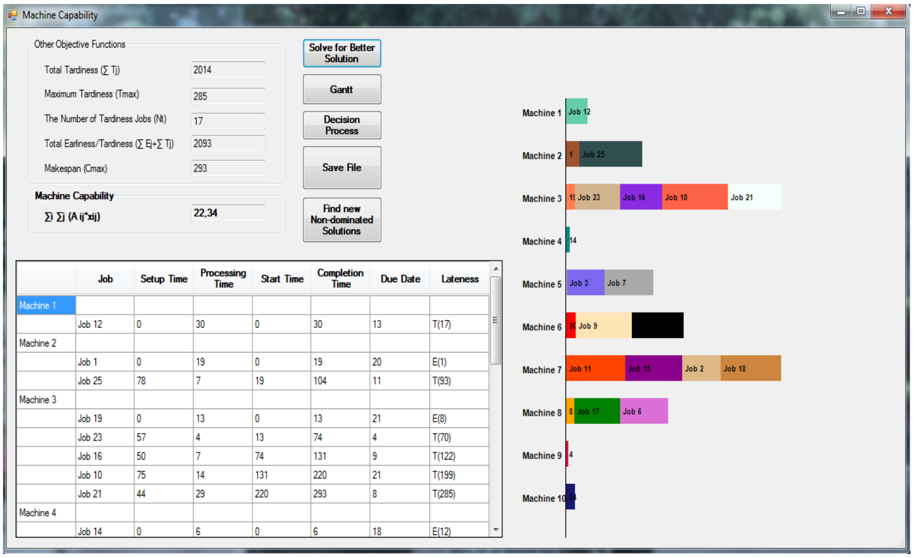
Cmax

Makespan (Cmax)	**106**

	Job	Setup Time	Processing Time	Start Time	Completion Time	Due Date	Lateness
Machine 1							
	Job 9	0	1	0	1	19	E(18)
	Job 25	47	3	1	51	11	T(40)
	Job 16	43	7	51	101	9	T(92)
Machine 2							
	Job 21	0	2	0	2	8	E(6)
	Job 14	54	26	2	82	18	T(64)
Machine 3							
	Job 3	0	2	0	2	9	E(7)
	Job 22	46	3	2	51	8	T(43)
	Job 15	44	9	51	104	15	T(89)
Machine 4							
	Job 13	0	16	0	16	18	E(2)

Figure 3. A solution of the UPMSSDS problem with 10 machines and 25 jobs regarding the *Min.* C_{max} objective function.

The solution for minimizing C_{max} is shown as a bar chart. While makespan is calculated as 106, total tardiness is 999, maximum tardiness is 99, the number of tardiness job is 16, total earliness and tardiness is 1059, and machine preference is 11.29 for related problem. On the other hand, when the problem is solved for maximizing *MP*, makespan increases to 293 from 106 (Figure 4).



Other Objective Functions

Total Tardiness (∑ Tj)	2014
Maximum Tardiness (Tmax)	285
The Number of Tardiness Jobs (Nt)	17
Total Earliness/Tardiness (∑ Ej+∑ Tj)	2093
Makespan (Cmax)	293

Machine Capability

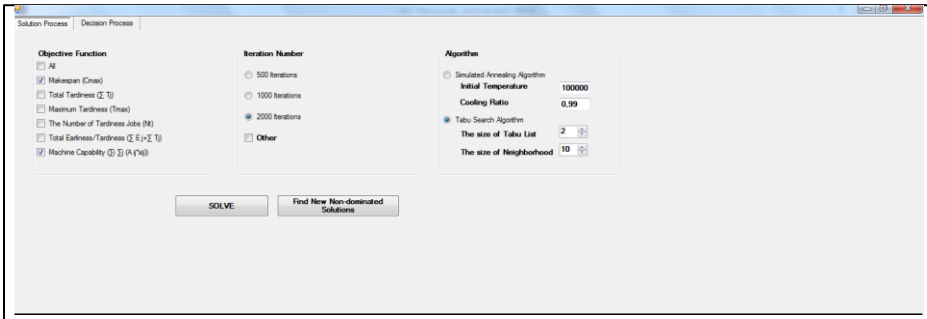
∑i ∑j (A ij*xij)	**22,34**

	Job	Setup Time	Processing Time	Start Time	Completion Time	Due Date	Lateness
Machine 1							
	Job 12	0	30	0	30	13	T(17)
Machine 2							
	Job 1	0	19	0	19	20	E(1)
	Job 25	78	7	19	104	11	T(93)
Machine 3							
	Job 19	0	13	0	13	21	E(8)
	Job 23	57	4	13	74	4	T(70)
	Job 16	50	7	74	131	9	T(122)
	Job 10	75	14	131	220	21	T(199)
	Job 21	44	29	220	293	8	T(285)
Machine 4							
	Job 14	0	6	0	6	18	E(12)

Figure 4. A solution of the UPMSSDS problem with 10 machines and 25 jobs regarding the *Max. MP* objective function

The machine preference increases to 22.34 from 11.29 in the solution of the UPMSSDS problem with 10 machines and 25 jobs regarding the *Max. MP* objective function (Figure 4). Other performance measures, total tardiness, maximum tardiness, the number of tardiness job, and total earliness and tardiness are obtained as 2014, 285, 17, and 2093, respectively.

There are two non-dominated solutions in Figure 3 and Figure 4. According to the solution in Figure 3, makespan is 106 and machine preference is 11.29 while makespan is 293 and machine preference is 22.34 in Figure 4. The proposed model makes it possible to find and compare new non-dominated solutions (Figure 5) and select one solution by considering trade-offs among the performance measures of solutions.



Objective Function
- [ ] All
- [✓] Makespan (Cmax)
- [ ] Total Tardiness (∑ Tj)
- [ ] Maximum Tardiness (Tmax)
- [ ] The Number of Tardiness Jobs (Nt)
- [ ] Total Earliness/Tardiness (∑ Ej+∑ Tj)
- [✓] Machine Capability (∑ ∑ (A ij*xij))

Iteration Number
- ○ 500 Iterations
- ○ 1000 Iterations
- ● 2000 Iterations
- [ ] Other

Algorithm
- ○ Simulated Annealing Algorithm
  - Initial Temperature: 100000
  - Cooling Ratio: 0,99
- ● Tabu Search Algorithm
  - The size of Tabu List: 2
  - The size of Neighborhood: 10

Solutions	Max. MP	Min. Cmax
[20 5 -1 11 9 0 24 -1 22 15 18 -1 13 17 -1 6 4 2 -1 12 19 -1 1 8 14 10 21 -1 7 -1 3 -1 16 23]	22.2	290
[11 5 -1 14 24 0 -1 15 22 9 18 -1 -1 4 2 -1 12 19 -1 17 21 10 8 -1 16 1 20 7 -1 -1 23 6 13 3]	21.88	247
[5 -1 24 11 0 14 -1 20 15 22 18 -1 6 -1 4 9 2 10 -1 12 19 -1 8 17 -1 7 1 -1 3 -1 23 13 21 16]	21.69	235
[5 6 -1 12 15 0 -1 2 10 -1 23 3 20 -1 4 16 18 -1 24 19 -1 11 1 21 -1 14 7 -1 17 22 -1 9 13 8]	15.66	133
[8 2 -1 23 17 9 -1 21 1 10 -1 6 16 14 -1 19 7 22 -1 18 20 24 -1 15 0 -1 5 4 -1 11 3 -1 12 13]	14.09	107

Figure 5. Finding new non-dominated solutions

Decision maker can obtain new non-dominated solutions by using an interface of DSS in Figure 5. He/she can select an appropriate one by considering trade-offs between two objective function values for shop-floor from the non-dominated solutions list. Production planners need to see trade-offs schedules generated for different objective functions. The designed DSS makes it possible to use appropriate schedule for shop-floor in planning horizon.

5. CONCLUSION

This paper studies unrelated parallel machine scheduling by considering the maximizing machine preferences. The machine preference scores taking values in the range of [0, 1] are obtained by taking different specifications such as scrap rates, quality, and cost into account. The maximizing machine preference objective function is employed together with the minimizing makespan. A bi-objective mathematical model for the unrelated parallel machine problem with sequence dependent setup times is designed and evaluated. The model aims to minimizing makespan and maximizing machine preferences of jobs depend on machine to which it is assigned. Bi-objective simulated annealing algorithm is proposed for solving the problem by making it possible to consider trade-offs and select the appropriate schedule to decision makers for their workshops.

The DSS is designed for solving UPMSSDS problems with objective function of the maximizing machine preferences in combination with other common scheduling objective functions for unrelated parallel machine scheduling problems. By using the proposed model, non-dominated solutions are compared and one solution is selected by considering trade-offs among performance measures of the solutions in real-world applications. The designed DSS can be used for the unrelated parallel machine scheduling problems. For future work, the proposed model is able to be modified for the other machine scheduling problems. All machine environments can be added to the system.

Acknowledgements

This work is supported by the Comity of Eskişehir Osmangazi University Scientific Research Projects under titled "A Decision Support System for Parallel Machine Scheduling with Sequence Dependent Setup Times and machine capability".

REFERENCES

[1] Pinedo, M. 2002. *Scheduling: theory, algorithms and systems.* Third ed. NewJersey: Prentice all.

[2] Allahverdi, A., J. N. D. Gupta, and T. Aldowaisan. 1999. "A review of scheduling research involving setup considerations." *Omega,* no.27:219-239.

[3] Allahverdi, A., C. T. Ng, T. C. E. Cheng, and M. Kovalyov. 2008. "A survey of scheduling problems with setup times or costs." *European Journal of Operational Research* no.187: 985-1032.

[4] França, P., M. Gendreau, G. Laporte, and F. Muller. 1996. "A tabu search heuristic for the multiprocessor scheduling problem with sequence dependent setup times." *International Journal of Production Economics* 43(2-3):79–89.

[5] Marsh, J.D. and Montgomery, D.C. (1973) 'Optimal procedures for scheduling jobs with sequence-dependent changeover times on parallel processors', *AIIE Technical Papers,* pp.279-286.

[6] Guinet, A. 1991. "Textile production systems: A succession of non-identical parallel processor shops." *Journal of the Operational Research Society* 42(8):655-671.

[7] Elmaghraby, S.E, A. Guinet, and K.W. Schellenberger. 1993 "Sequencing on parallel processors: An alternate approach." *OR Technical Report* no. 273, Raleigh, NC: North Carolina State University.

[8] Zhu, Z. and R.B. Heady. 2000. "Minimizing the sum of earliness/ tardiness in multi-machine scheduling: A mixed integer programming approach." *Computers and Industrial Engineering* no.38: 297–305.

[9] Weng, M.X., J. Lu, and H. Ren. 2001. "Unrelated parallel machine scheduling with setup consideration and a total weighted completion time objective." *International Journal of Production Economics* no.70: 215–226.

[10] Kim, C.O. and Shin, H.J. (2003) 'Scheduling jobs on parallel machines: A restricted tabu search approach', *International Journal of Advanced Manufacturing Technology,* Vol 22 No.3, pp.278–287.

[11] Ravetti, M. G., R.M. Geraldo, L.R. Pedro, and M.P. Panos. 2007. "A scheduling problem with unrelated parallel machines and sequence dependent setups." *International Journal of Operational Research,* 2(4):380-399.

[12] Chen, C-L., and C-L. Chen. 2009. "Hybrid metaheuristics for unrelated parallel machine scheduling with sequence-dependent setup times." *International Journal of Advanced Manufacturing Technology* 43(1-2):161-169.

[13] Tavakkoli Moghaddam, R., F. Taheri, M. Bazzazi, and F.Sassani. 2009. "Design of a genetic algorithm for bi-objective unrelated parallel machines scheduling with sequence-dependent setup times and precedence constraints." *Computers and Operations Research* 36(12), 3224-3230.

[14] Arnaout, J-P., G. Rabadi, and R. Musa. 2010. "A two-stage ant colony optimization algorithm to minimize the makespan on unrelated parallel machines with sequence-dependent setup times." *Journal of Intelligent Manufacturing* 21 (6):693-701.

[15] Vallada, E. and R. Ruiz. 2011. "A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times." *European Journal of Operational Research* 211(3): 612-622.

[16] Lin, S-W., C-C. Lu, and K-C. Ying. 2011. "Minimization of total tardiness on unrelated parallel machines with sequence- and machine-dependent setup times under due date constraints." *International Journal of Advanced Manufacturing Technology,* 53(1-4):353-361.

[17] Ying, K-C., Z-J. Lee, and S-W. Lin. 2012. "Makespan minimization for scheduling unrelated parallel machines with setup times." *Journal of Intelligent Manufacturing* 23(5): 1795-1803.

[18] Hsu, C-J., M. Ji, J-Y. Guo, and D-L. Yang. 2013. "Unrelated parallel-machine scheduling problems with aging effects and deteriorating maintenance activities." *Information Sciences*, no.253:163-169.

[19] Naderi-Beni, M., E. Ghobadian, S. Ebrahimnejad, and R. Tavakkoli-Moghaddam. 2014. "Fuzzy bi-objective formulation for a parallel machine scheduling problem with machine eligibility restrictions and sequence-dependent setup times dependent setup times." *International Journal of Production Research* 52(19):5799-5822.

[20] Eroglu, D.Y., H.C. Ozmutlu, and S. Ozmutlu. 2014. "Genetic algorithm with local search for the unrelated parallel machine scheduling problem with sequence-dependent set-up times." *International Journal of Production Research* 52(19):5841–5856.

[21] Afzalirad, M., and J. Rezaeian. 2016. "Design of high-performing hybrid meta-heuristics for unrelated parallel machine scheduling with machine eligibility and precedence constraints." *Engineering Optimization* 48(4):706-726.

[22] Arroyo, J.E.C. and J.Y.T. Leung. 2017. "Scheduling unrelated parallel batch processing machines with non-identical job sizes and unequal ready times." *Computers and Industrial Engineering* no.78:117-128.

[23] Ezugwu, A.E., and F. Akutsah. 2018. "An improved firefly algorithm for the unrelated parallel machines scheduling problem with sequence-dependent setup times", *IEEE Access,* Vol.6: 54459- 54478.

[24] Bektur, G. and T. Saraç. 2019. "A mathematical model and heuristic algorithms for an unrelated parallel machine scheduling problem with sequence-dependent setup times, machine eligibility restrictions and a common server". *Computers and Operations Research,* no.103:46–63.

[25] Fanjul-Peyro, L., R. Ruiz, and F. Perea. 2019. "Reformulations and an exact algorithm for unrelated parallel machine scheduling problems with setup times". *Computers and Operations Research,* no.101: 173–182.

[26] Chuang, M-C., C-J. Liao, and C-W. Chao. 2010. "Parallel machine scheduling with preference of machines". *International Journal of Production Research,* 48(14):4139–4152.

[27] Huang, C-J., and L-M. Liao. 2014. "Parallel machines scheduling with machine preference via agent-based approach", *Applied Mathematics and Computation,* no.233:298–309.

[28] Lin, S-W. and K-C. Ying. 2015. “A multi-point simulated annealing heuristic for solving multiple objective unrelated parallel machine scheduling problems’, *International Journal of Production Research,* Vol.53 No.4, pp.1065–1076.

[29] Mishra, K.K. and S. Harit. 2010. “A fast algorithm for finding the non-dominated set in multi objective optimization.” *International Journal of Computer Applications,* 1(25):35-39.

Appendix

Dataset for 5 machines, 10 jobs, unrelated parallel machine scheduling problem

Processing times of job J_j to machine M_i.

P_{ij}	1	2	3	4	5	6	7	8	9	10
1	3	2	16	24	29	4	18	15	1	11
2	5	24	10	16	5	19	8	20	21	23
3	14	3	7	28	5	25	17	30	3	14
4	4	29	1	24	25	27	3	12	8	25
5	13	28	6	8	5	5	27	18	17	5

Sequence dependent setup times from job J_j to job J_k

S_{jk}	1	2	3	4	5	6	7	8	9	10
1	-	18	19	7	19	15	6	9	13	20
2	20	-	7	20	20	12	17	7	11	19
3	17	20	-	15	5	18	19	15	17	16
4	11	15	7	-	16	5	9	5	6	18
5	16	10	20	5	-	12	11	17	17	7
6	12	12	15	16	17	-	9	15	15	7
7	6	12	20	10	14	8	-	17	9	13
8	16	19	20	13	7	7	9	-	18	9
9	18	8	19	10	8	9	14	12	-	10
10	18	14	13	19	9	17	17	11	14	-

Machine preference of job J_k to machine M_i.

A_{ik}	1	2	3	4	5	6	7	8	9	10
1	0.85303	0.07597	0.41727	0.48925	0.78025	0.13197	0.23478	0.16899	0.54701	0.18351
2	0.62206	0.23992	0.04965	0.33772	0.38974	0.94205	0.35316	0.64912	0.29632	0.36849
3	0.35095	0.12332	0.90272	0.90005	0.24169	0.95614	0.82119	0.73172	0.74469	0.62562
4	0.51325	0.18391	0.94479	0.36925	0.40391	0.57521	0.01540	0.64775	0.18896	0.78023
5	0.40181	0.23995	0.49087	0.11120	0.09646	0.05978	0.04302	0.45092	0.68678	0.08113