Iranian Journal of Operations Research
Vol. 14, No. 1, 2023, pp. 1-21
DOI:

Scheduling of incompatible families of jobs on a single batch processing machine to minimize the weighted number of tardy jobs and consumption of electrical energy

Ladan al-Sadat Mousavi [1], Fariborz Jolai[2,*]

This paper considers the optimization problem of scheduling jobs with identical sizes on a single batch processing machine. The jobs are divided into some incompatible families where each family contains the jobs with the same processing times and the jobs from different families could not proceed in the same batch. Our optimization problem has two objectives. The first objective is minimizing the weighted number of tardy jobs regarding the due dates of jobs given by customers. The second one aims to minimize operations costs by finding the schedules with the minimum electrical cost consummation under the Time-of-Use tariff policy. A two-objective mixed integer programming mathematical programming is proposed to find optimal solutions for small-size instances of the problem. To solve the medium and large-scale size of the problem, two meta-heuristic algorithms NSGA-II and MOPSO are proposed. The Computational experiments results show that two solution algorithms are capable to find near-optimal solutions at a reasonable computational time. The MOPSO algorithm generate more diverse solutions in less computational time comparing with NSGA II algorithm. But the quality of the solutions obtained by NSGA II algorithm are superior to the ones obtained by MOPSO.

***Keywords**: TOU tariff electrical policy, Batch processing machine, Number of tardy jobs, Incompatible job families, NSGA-II, MOPSO.*

Manuscript was received on 07/30/2023, revised on 08/11/2023 and accepted for publication on 09/03/2023.

1. Introduction

In recent years, job shop scheduling problem with energy considerations has been the focus of many researchers [1-2]. A high percentage of the world's energy is consumed in the manufacturing and production industries and the growing trend of energy prices has made it important to meet the economic needs of the society with energy considerations [3- 6] see also [46] and [47]. On the other hand, management tools such as production planning, whose implementation does not require new investment, have always been the first priority of managers to use in order to achieve goals [7-10]. One of the successful management strategies to save electricity consumption is the pricing policy based on the time of electricity consumption, which is called Time-of-Use policy, TOU for short [11].

Batch processing machines are found in many manufacturing environments [48]. Aircraft manufacturing industries [12], shoe manufacturing industries [13], automobile gearbox manufacturing [14], health systems [15] and also in chemical industries [16]. Other applications in the furniture, pharmaceutical, casting, metal, aerospace and air logistics industries are mentioned in reference [17]. But the most important application is in semiconductor manufacturing industries.

* Corresponding Author.
[1] Ph.D. candidate, Industrial Engineering department, Kish international Campus, University of Tehran, Kish Island, Iran.
[2] Professor, School of Industrial Engineering, College of Engineering, University of Tehran, Tehran, Iran.

Semiconductor industries produce integrated circuits from raw wafers and very thin discs from silicon or gallium arsenide raw materials and are known as the industry with the most complex production process in the world [18]. The main characteristic of these industries is the large number of production steps, multiple returns of the product to the previous steps, the large number of breakdowns and the simultaneous presence of normal production machines and batch processing in the production line. Batch processing machines are used in the diffusion and oxidation sections as well as the last stage of production, which is the final product test.

In job shop environments with batch processing machines, the jobs must be placed in a batch for processing. Each batch has a limited capacity depending on the capacity of the machine. The capacity of the machine is equal to the total size of the jobs that the machine is capable to processing simultaneously. If the size of the jobs is the same, the capacity is defined based on the maximum number jobs in a batch. The processing time of each batch can be fixed or depends on the jobs inside it. A family of jobs consists of several jobs that have the same processing time. Sometimes, due to technical limitations, it is not possible to put jobs belonging to different families together in one batch, called incompatible job families. In this article, we consider a job shop environment with a single batch processing machine with a limited capacity and incompatible families of jobs with the identical job sizes.

One of our objectives for determining the optimal schedule is the energy consumption of the batch processing machine. Most of the time, batch processing machines have high energy consumption. In some applications of batch processing machines, electricity consumption constitutes 10 to 50 percent of the final production costs [19]. A clear example of this high consumption can be found in the aluminum industry [20-21]. In these industries, many thermal furnaces are used, which are modeled as batch processing machines.

Another considered objective is minimization the total weighted number of tardy jobs. It is assumed that the importance (weight) of each job and its due date are given in advance. Minimizing the number of tardy jobs is one of the important criteria of scheduling theory. If we have n jobs and only one processing machine with the capacity of one job (classical scheduling problem), then the optimal schedule of the jobs is obtained by the Moore-Johnson algorithm in polynomial time [22]. But if each job has a different weight, the problem of minimizing the total weighted number of tardy jobs becomes NP-hard [23]. Jolai [24] showed that the problem of minimizing the number of tardy jobs on a batch processing machine with limited capacity and incompatible jobs families is also an NP-hard problem. Therefore, the problem investigated in this article, which considers the objective of minimization of electrical energy consumption along with minimization of the total weighted number of tardy jobs, is a difficult problem.

According to the authors' knowledge, this issue has not been investigated so far. The main contributions of our study are:

- The problem of minimizing the weighted number of tardy jobs on a single batch processing machine with energy consumption consideration is studied for the first time.
- The problem is modeled as s mufti-objective mixed integer mathematical programming model.
- Two multi-objective metaheuristic algorithms are adopted for solving the considered problem.

.

In the second section of this paper, the relevant researches have been reviewed. The third section presents the mathematical model of the considered problem. In the fourth section, the developed

solution methods are described. The fifth section shows the results of numerical experiments. In the sixth section managerial insights are provided, In the section seven the research results are discussed and future research fields are suggested.

2. Literature review

Recently, a comprehensive review article has been published for batch processing machine scheduling problems under certainty conditions [25]. Our study shows that in all previous researches on batch processing machine scheduling problems with energy consideration a TOU policy is implied. Also, most of the conducted researches have considered the minimization of the completion time of all jobs (makespan) as the evaluation criteria of the schedule. The summary of research related to makespan criterion is presented in Table 1. As can be seen, a large number of researchers have considered only one batch processing machine in the assumptions of the problem [26-33]. Among these researches, references [31] and [32] have assumed the variable machine speed where the amount of electricity consumption will differ with the machine speed. Also, reference [33] is the only single machine problem that does not take the same arrival time of jobs. In these three articles [31-33], the job sizes are non-identical.

Due to the many applications of production environments with parallel machines in the real world, the extension of single batch processing machine models to identical parallel batch processing machines has been considered [34-37]. Authors of [34] have considered a number of parallel batch processing machines with different speeds and the same size of jobs. Reference [35] considers unequal size jobs with unequal arrival time, [36] takes into account the jobs with different sizes but machines with different speeds. And finally [37] has investigated a problem with similar assumptions as [36] but presents different heuristic solution methods. There are only two studies with incompatible families of jobs, both of which have an objective function other than makespan. The first one is [38], which considers a batch processing machines problem with incompatible jobs families and the criteria of energy cost minimization under the TOU policy and minimization of the total weighted sum of jobs completion time. The planning horizon is broken into several equal periods of time and the processing of a batch can continue in several periods. The electricity tariff is different in each period. Identical parallel machines, jobs with different sizes, different arrival times of jobs are other assumptions of that article. For this problem, a mathematical model has been presented and optimal properties have been proven for special cases of same-size jobs, jobs with zero arrival time, and a special case where the maximum batch size is an integer multiple of the job size. These properties have been used to improve the basic mathematical model. The same authors study the criterion of minimizing the total tardiness in [39]. They are the first researchers who have studied a due date related criterion together with the criterion of energy consumption in the environment of batch processing machines.

Among the most recent relevant researches to scheduling problem with energy consideration we could mention: [43] study TOU policy in flow shop environment, [44] provide metaheuristic solution for energy-efficient scheduling of a two-stage flexible printed circuit board and [45] develop a multi-objective scheduling model for a flexible manufacturing system to reduce peak load using an energy storage system

The assumptions of our research are close to [39], with the main difference that we consider the minimization of the weighted number of tardy jobs along with the minimization of the cost of electricity consumption. Also, the development of new solution methods to solve the problem has been done here.

Table 1: BPM scheduling problem with makespan criteria and energy consumption consideration

Authors	Year of publish	Number of machines		Speed of machine		Size of jobs		Non zero Ready time Of jobs
		one	parallel	Fix	Variable	identical	Non identical	
Cheng et al	**2014**	*			*		*	
Cheng Et al	**2016**	*			*		*	
Wu Et al	**2019**	*			*		*	
Cheng, Et al	**2017**	*			*		*	
Cheng Et al	**2016**	*			*		*	
Wang, Et al	**2016**	*		*		*		
Zhang, Et al	**2017**	*		*		*		
Zhou et al	**2020**	*	*	*			*	
Zhou Et al	**2018**		*		*		*	*
Jia Et al	**2017**		*	*			*	*
Jia Et al	**2019**		*	*			*	*
Qian Et al	**2020**		*	*			*	*

3. Description of the problem and its mathematical model

In this research, we study the problem of minimizing the total weighted of the number of tardy jobs on a batch processing machine with incompatible jobs families with energy considerations. The assumptions of the problem are:

1- We have a batch processing machine that is able to process a number of jobs simultaneously.

2- The capacity of the machine is limited.

3- *n* jobs with given processing time and due dates are present in the job shop at zero time. Jobs are divided into *f* incompatible job families. Jobs of each family have the same processing time.

4- The size of the jobs is identical.

5- The jobs inside a batch are of the same type of job family, and the processing time of the batch is equal to the processing time of the job family it contains.

6- It is not allowed to interrupt the process of a batch.

7- The planning horizon is broken into a number of identical time periods.

8- The cost of electricity consumption in each period is the average rate of electricity cost in that period. Minimizing the cost of electricity consumption during the planning horizon is one of the considered objectives.

9- In order to satisfy customers, minimizing the total weighted number of tardy jobs is considered as another objective.

The decision variables include the optimal number of batches of each family of jobs, the sequence of processing of the batches on the machine, and the determination of the start and end of processing of each batch so that a compromise between the two objective functions is made.

3.1. Mathematical model

To explain the mathematical model, we need to define the indices, parameters and decision variables as follows.

Index

j	Indices of set of all jobs	j=1,2,…n
s	Indices of set of families	s=1,2,3,…f
k	Indices of set of batches	k=1,2,…n
t	Indices of time period	t=1,2,3…T
$\zeta(s)$	Set of jobs belong to family s	
$s(j)$	The family of job j	

Parameters

d_j	Due date of job j	
p_s	Processing time of jobs belong to family s	
Ec_t	Cost of a unit energy in time period t	
E_s	Energy consuming for processing a job of family j	
W_j	Weight of job j	
B	Capacity of batch processing machine	
M	A big number	

Decision variables

Ts_k^t	=1 if the processing of batch k is started at period t, otherwise is zero
U_j	=1 if the job j is tardy, otherwise is zero
u_k^j	=1 if job j is assigned to batch k, otherwise is zero
v_k^s	=1 if batch k contains the jobs of family s, otherwise is zero
x_{skt}	is defined for linearization of the model, where we have a multiplication of $Ts_k^t and\ v_k^s$

Now, we could present our mathematical model as below.

$$Z_1 = \sum_{j=1}^{n} W_j . U_j \qquad (1)$$

$$Z_2 = \sum_{s=1}^{f}\sum_{k=1}^{n}\sum_{t=1}^{T} E_s.x_{skt} \sum_{t'=t}^{t+P_s-1} Ec_{t'} \quad (2)$$

ST:

$$\sum_{k=1}^{n} u_k^j = 1 \qquad \forall\, j = 1,2,\dots,n \quad (3)$$

$$\sum_{j\in\zeta(s)}^{f} u_k^j \le B.v_k^s \qquad \forall\, k = 1,2,\dots,n,\quad s = 1,2,\dots,f \quad (۴)$$

$$\sum_{s=1}^{f} v_k^s \le 1 \qquad \forall\, k = 1,2,\dots,n \quad (۵)$$

$$\sum_{t=1}^{T} Ts_k^t = \sum_{s=1}^{f} v_k^s \qquad \forall\, k = 1,2,\dots,n \quad (6)$$

$$\sum_{t=1}^{T} t.Ts_k^t + \sum_{s=1}^{f} p_s.v_k^s \le \sum_{t=1}^{T} t.Ts_{k+1}^t \qquad \forall\, k = 1,2,\dots,n-1 \quad (7)$$

$$\sum_{t=1}^{T} t.Ts_k^t \le \left(d_j - p_{s(j)}\right) + M.(1 + U_j - u_k^j) \qquad \forall\, j = 1,2,\dots,n,\quad k = 1,2,\dots,n \quad (8)$$

$$t.Ts_k^t + \sum_{s=1}^{f} p_s.v_k^s \le T \qquad \forall\, k = 1,2,\dots,n,\quad t = 1,2,\dots,T \quad (9)$$

$$x_{skt} \le Ts_k^t \qquad \forall\, k = 1,2,\dots,n,\quad s = 1,2,\dots,f,\quad t = 1,2,\dots,T \quad (10)$$

$$x_{skt} \le v_k^s \qquad \forall\, k = 1,2,\dots,n,\quad s = 1,2,\dots,f,\quad t = 1,2,\dots,T \quad (11)$$

$$x_{skt} \ge Ts_k^t + v_k^s - 1 \qquad \forall\, k = 1,2,\dots,n,\quad s = 1,2,\dots,f,\quad t = 1,2,\dots,T \quad (12)$$

$$Ts_k^t, U_j, u_k^j, v_k^s, x_{skt} \in \{0,1\} \qquad \forall\, j = 1,2,\dots,n,\quad k = 1,2,\dots,n,\quad s = 1,2,\dots,f,\quad t = 1,2,\dots,T \quad (۱۳)$$

In equation (1), as the first objective function of the model, the sum of the weighted number of tardy jobs is minimized, in equation (2), as the second objective function of the model, the cost of energy consuming is minimized. Constraint (3) guarantees that each job is assigned to exactly one batch. Inequality (4) is the capacity constraint and guarantees that each batch contains at most B jobs. On the other hand, both constraints (4) and (5) simultaneously guarantee that jobs inside a batch

belong to the same family. Constraint (6) guarantees that if a batch is formed from a family, then a specific time must be allocated to start the completion of the jobs in that batch. Constraint (7) shows that the start time of the jobs in the k+1 *th* batch is at least equal to the start time of the jobs in the k *th* batch plus the time required to complete the jobs in the k *th* batch. Constraint (8) guarantees that if job *j* is not completed on time, then the decision variable U_j is equal to one. Constraint (9) impose that all processing of batches finish at most in period T. Constraints (10-12) are linearization constraints and state that x_{skt} is equal to one if and only if both decision variables Ts_k^t and v_k^s are equal to one. At the end, constraint (13) shows the sign constraints.

As we will see in the computational experiments section, the above mathematical model is only able to solve the small instances of the problem. Therefore, in the next section, two evolutionary algorithms are proposed to solve the problem.

4. Proposed meta-heuristic solution algorithms

In this section two meta-heuristic algorithms, named MOSPO and NSGA II, are proposed to solve the considered problem. Both of these algorithms are among the most effective and practical multi objective optimization algorithms that have been used by researchers for years.

4.1. MOPSO algorithm

The MOPSO algorithm is a multi-objective version of the PSO (particle swarm optimization) algorithm [41]. In PSO algorithm, each solution vector of the problem is called a particle, and the set of particles together form a population. Then in each iteration of the algorithm, it is tried to update the solutions based on the results of the previous velocityv_k^i, the best individual experience $pbest_k^i$and the best group experience ($gbest_k$). The best individual experience is the best solution that a particle has ever experienced, and the best group experience is the best solution that all particles have obtained until the last iteration. In the PSO algorithm, two relations (14) and (15) are used to update the solution vectors or particles:

$$v_{k+1}^i = w \times v_k^i + c_1 \times rand \times \left(pbest_k^i - x_k^i\right) + c_2 \times rand \times \left(gbest_k - x_k^i\right) \quad (14)$$

$$x_{k+1}^i = x_k^i + v_{k+1}^i \quad (15)$$

In the above two relations, x_k^i represents the position (location) of the i *th* particle in the kth iteration, v_k^i the velocity of the i *th* particle in the kth iteration, w is the inertia coefficient, and c1 and c2 represent the individual and collective learning (acceleration) coefficients, respectively, and rand is a random value between 0 and 1. The basis of the MOPSO algorithm is almost similar to the PSO algorithm, with the difference that due to the multi-objective nature of this algorithm, it is no longer possible to talk about the existence of a best group experience. In MOPSO, we have a set of non-dominant solutions. The non-dominant solutions obtained in each repetition of this algorithm are stored in a memory with a limited capacity and based on the criterion of diversity; they will have a

chance to be selected as the best solution or the best group experience. If the memory of the Pareto solutions reservoir is full, some non-dominated solutions are removed from the memory based on their diversity index.

One of the advantages of the MOPSO algorithm is its high search speed compared to other multi-objective algorithms. MOPSO uses a simple method to classify Pareto solutions based on diversity criteria. It first divides the solution space into square and cube-shaped grids. The number of these grids is a control parameter that is determined by the decision maker or expert or with the help of test setup methods.

Now, we will describe the steps of the MOPSO algorithm.

Step 1: Create the initial population of particles.

Step 2: Identify the non-dominated solutions and update the Pareto solution reservoir. Note that by gridding the solution space, the solutions with best diversity index are stored in the reservoir.

Step 3: Using the roulette wheel, choose one of the available non-dominated solutions in the reservoir by chance as the best group experience. In this research, if the best individual experience of each particle does not dominate the best experiences of that particle in the previous iterations, we randomly select one of the Pareto values from among the previous solutions as the best individual experience of that particle.

Step 4: Using equations (14) and (15), update the position of the particles.

Step 5: Compare the new non-dominated solutions with the existing solutions in the reservoir and if any of the solutions are dominated, remove it from the reservoir. In the case that the number of non-dominated solutions remaining in the reservoir is greater than the capacity of the reservoir, then we eliminate the solutions that have less diversity.

Step 6: If the stop condition is met, stop the algorithm, otherwise return to the step 2.

4.2. NSGA II algorithm

The multi-objective genetic algorithm with non-dominated sorting or NSGA II is a multi-objective version of the genetic algorithm [42]. The basis of NSGA II algorithm is based on two concepts of Pareto fronts and crowding distance. The solutions in the Pareto fronts are ranked based on the crowding distance criterion. Congestion distance actually measures the amount of density of solutions around a particular solution. The more a solution is located at a distance from its neighboring solutions, the more unique that solution is and is better than other solutions. The lower the solutions are placed in the front and the greater the crowding distance, then that solution will have a greater chance for the cross over operation and the production of a new generation. The steps to implement the NSGA II algorithm are as follows:

Step 1: Construct the population of the initial solutions.

Step 2: Classify the solutions based on the two criteria of the Pareto front and the crowding distance and assign a chance to each solution based on these two criteria.

Step 3: Using the roulette cycle, select the solutions to apply the cross over operation and produce a new generation.

Step 4: Among the solutions with the same parents, select a number of solutions randomly with equal chance and apply the mutation operator to create a new generation of solutions.

Step 5: Consider the solutions obtained from cross over and mutation operator with the previous solutions and rank them again based on the criteria of Pareto front and crowding distance.

Step 6: If the stop condition is met, stop the algorithm, otherwise return to the step 2.

In the following, the method of defining the solution vector in the two MOPSO and NSGA II algorithms as well as the cross over and mutation operators used in the NSGA II algorithm are introduced.

4.3. Solution vector in NSGA II and MOPSO algorithms

The appropriate definition of the solution vector, while increasing the efficiency of the search operators, prevents the algorithms from getting stuck in the local optimal points. In this research, the Random Key structure has been used to display the solution vectors. Let n_s as the number of jobs in the family s, then the number of optimal batches will be equal to $\sum_{s=1}^{f}\lceil n_s/B\rceil$ [40]. The structure of the solution vector contains two parts, the first part consists of $\sum_{s=1}^{f}\lceil n_s/B\rceil$ cells and the second part have j cells. A continuous random value in the range from 0 to 1 is assigned to each cell. The first part of the solution vector shows the sequence of batches on the machine, and the second part depicts how to assign different jobs to the batches. In order to make it clear how to extract the values of the decision variables from the proposed solution vector, assume that there are a total of 11 jobs in 3 different families, so that jobs 1 to 4 belong to the first family, jobs 5 to 8 to the second family and the rest belongs to the third family. Also, assume that the maximum size of each batch (B) is equal to 2 jobs. The number of batches in the optimal solution will be equal to ⌈4/2⌉+⌈4/2⌉+⌈3/2⌉= 6, and the length of the solution vector in this example will be equal to 11+6=17. Figure 1 depicts the solution vector of this example.

Part one of solution vector						Part 2 of solution vector										
Batch 1	Batch 2	Batch 3	Batch 4	Batch 5	Batch 6	Job1	Job2	Job3	Job4	Job5	Job6	Job7	Job8	Job9	Job10	Job11
0.05	0.17	0.93	0.11	083	0.15	086	0.6	0.2	0.8	0.78	0.10	0.17	0.63	0.33	0.27	0.79

Figure 1: An example of a solution vector

We specify the sequence of batches by sorting the numerical values of the cells in the first part of the solution vector in ascending order. Thus, we will have the sequence of batches 1-4-6-2-5-3. The smallest value in the second part of the solution vector is the value 0.10 corresponding to the sixth job belonging to the second family, therefore we assign this job to the first batch, and the next smallest value is the number 0.17 corresponding to the seventh job. This job is also belonging to the second family, and due to the fact that the first batch is not completely filled, therefore, we assign this job to the first batch as well. In the next step, according to the value of 0.20, it is the turn to allocate job number three; this job is assigned to the second batch. According to the value of 0.27, the tenth job from the third family will be selected. This job is not in the same family as job number three thus is assigned to a new batch, the third batch. With the continuation of this process, all jobs are assigned to different batches.

Having sequence of batches and assignment of jobs to batches, the start and completion times of the jobs as well as the number of tardy jobs and energy consumption of the machine can be easily calculated.

4.4. Cross over operator

In this research, the continuous cross over operator is used to generate new offspring. Let X1 and X2 are the solution vectors of the first and second parent, respectively, and Y1 and Y2 as the solution vectors of the first and second offspring. Parameter θ is a continuous random value between 0 and 1. The two equations (16) and (17) explain the way new children are generated in continuous cross over.

$$Y_1 = \theta \times X_1 + (1 - \theta) \times X_2 \quad (16)$$

$$Y_2 = \theta \times X_2 + (1 - \theta) \times X_1 \quad (17)$$

4.5. Mutation operator

We have used the swap operator in order to make mutations in the solution vectors. First a solution vector is selected randomly. Then two cells of this solution are chosen randomly and their values are replaced with each other.

5. Computational experiments

The 15 instances of the problem are generated randomly as [40] and [26]. These instances are divided in three classes of small, medium and large problems as Table 2.

Table 2: Randomly generated instances of the problem				
# of jobs in a family	**Capacity of machine**	**# of families**	**#problem**	
2	2	2	1	**small**
4	2	2	2	
6	2	3	3	
6	3	3	4	
9	3	3	5	
8	4	5	6	**medium**
12	4	5	7	
16	4	7	8	
15	5	7	9	
20	5	7	10	
14	7	8	11	**large**
21	7	8	12	
28	7	10	13	
24	8	10	14	
32	8	10	15	

The batch processing time of a family is generated according to a discrete distribution with values 2, 4, 10, 16, 20 with probabilities 0.2, 0.2, 0.2, 0.3, 0,2 and 0.1, respectively. The cost of each unit of energy is generated based on a discrete uniform distribution in the interval 5 to 20. The amount of energy required by the machine to process a batch of a family is produced based on a uniform distribution in the interval of 20 to 50. The weight (importance) of jobs is generated following a continuous uniform distribution in the interval from 0 to 1. The number of planning periods is equal to the sum of the times required to complete the operation of all batches. It has been assumed that the due dates of the jobs have a discrete uniform probability distribution in the interval from 0.3t to t.

5.1. Performance evaluation measures of multi-objective algorithms

In this research, six measures have been used to compare the solutions of the algorithms:

1. The number of Pareto solutions.
2. The quality measures. It measures how many percent of the solutions of each algorithm remain non-dominated.
3. The distance from the ideal point. The ideal point is the point where each of the objective functions has its best possible value. In order to calculate the distance measures from the ideal we use the equation (18):

$$MID = \frac{\sum_{i=1}^{n} \sqrt{\sum_{j=1}^{m} \left(\frac{f_{ij} - f_j^*}{f_j^{max} - f_j^{min}}\right)^2}}{n} \qquad (18)$$

In this equation, n is the number of Pareto solutions of an algorithm, m is the number of objective functionsf_{ij} is the value of the jth objective function of the ith Pareto solution, f_j^* is the ideal value for the jth objective function, and f_j^{min} and f_j^{max} are the minimum and maximum values, respectively. They display the jth objective function in the set of Pareto solutions.

4. Spacing index: it measures the dispersion of non-dominant solutions by measuring the standard deviation of the distance of consecutive Pareto solutions on the Pareto frontier.
5. Index of the greatest expansion: the length of the diameter of a set of Pareto solutions (Pareto frontier) is measured. In fact, this index measures the distance of the farthest Pareto solutions from each other. If we assume that f_{ij} is the value of the jth objective function of the i-th Pareto solution and m is the number of objective functions, then the maximum expansion index is calculated using equation (19):

$$D = \sqrt{\sum_{j=1}^{m} (max_i(f_{ij}) - max_i(f_{ij}))^2} \quad (19)$$

6. Time: the time index is the last measure used in this research to compare the proposed multi-objective algorithms. Obviously, the shorter the execution time of an algorithm, the better the algorithm will be.

5.2. Setting the parameters of the algorithms

In this research, Taguchi's method has been used to adjust the parameters of two multi-objective algorithms, NSGA II and MOPSO. The seven control parameters of MOPSO algorithm are:

- The number of iterations of the algorithm as a stopping condition (MaxIt)
- Number of members of the population (nPop)
- The capacity of the non-dominated solution reservoir (nRep)
- Coefficient of inertia (w)
- Individual learning coefficient (C_1)
- Coefficient of collective learning (C_2)
- The number of grids in each dimension of the solution space (nGrid)

By study the previous researches and with the help of trial-and-error technique, the different levels of the parameters are selected as Table 3.

By performing the 27 tests of the Taguchi method, the values for each of the parameters of the MOPSO algorithm in different problem sizes is obtained as Table 4.

Table 3: MOPSO different levels of the parameters

level			parameters
high	**medium**	**low**	
700	400	200	$MaxIt$
۵۰۰	۳۰۰	۲۰۰	$nPop$
۱۰۰	۷۵	۵۰	$nRep$
۱/۵	۱	۰/۵	w
۲	۱/۵	۱	C_1
۲	۱/۵	۱	C_2
۷	۵	۳	$nGrid$

Table 4: Parameters values of MOPSO

Proposed parameters values			
Large size instance	**Medium Size**	**Small size**	**parameter**
400	400	200	$MaxIt$
۵۰۰	۵۰۰	۲۰۰	$nPop$
۱۰۰	۷۵	۵۰	$nRep$
۱	۱	۱	w
۲	۲	۲	C_1
۲	۲	۲	C_2
۷	۷	۵	$nGrid$

It should be noted that the two objective functions investigated in this research are not of the same type. Therefore, first, both objective functions are normalized between 0 and 1, and then the index values are calculated. On the other hand, due to the fact that Taguchi analysis receives only one value as the result of the test, the suggested indices are normalized again between 0 and 1 after calculation. The way of normalization is that positive indices such as the index of the number of Pareto solutions, the largest value is mapped to 0 and the smallest value to 1, and the rest of the values are also normalized according to these two values in the range of 0 to 1. But for negative indicators, like time, the smallest value is mapped to 0 and the largest value to 1. After the normalization of each of the indicators, the indicators are multiplied by specific weights according to their importance from the point of view of the decision maker and are added together until at the end one number remains as the final score to be submitted to the Taguchi algorithm. The weights used for the indicators of the number of Pareto solutions, quality, distance from the ideal, spacing, maximum expansion and time are assumed to be 0.1, 0.3, 0.2, 0.2, 0.1 and 0.1, respectively.

The NSGA II algorithm has four parameters to adjust. These four parameters are:

- The number of iterations of the algorithm as a stopping condition (MaxIt)
- Number of members of the population (nPop)
- Crossover rate (pCrossover)

Table 5: NSGA II different levels of the parameters

Level			Parameters
High	**Medium**	**Low**	
۳	۲	۱	
500	300	200	$MaxIt$
۵۰۰	۳۰۰	۲۰۰	$nPop$
0.90	0.85	0.80	$pCrossover$
0.2	0.15	0.1	$pMutation$

Table 6: Parameters values of NSGA II

Proposed values			
Large instance	Medium instance	Small instance	Parameters
500	500	200	$MaxIt$
300	300	۲۰۰	$nPop$
0.9	0.9	0.9	$pCrossover$
0.2	0.2	0.2	$pMutation$

- Mutation rate (pMutation)

Table 5 shows the suggested levels of each of the control parameters of this algorithm. The results of applying Taguchi's method for NSGA II are presented in Table 6.

5.3. Numerical results

To analyze the performance of the proposed algorithms, we will use the 15 random instances of the problem. Also, the six mentioned measures are used to compare the algorithms. The epsilon constraint method is used to find optimal solutions in small size instances. Gams software version 24.1.3 was used to implement the epsilon constraint method and MATLAB 2015b software was used to implement MOPSO and NSGA II algorithms. Also, all methods are coded on Laptop with CPU of 2.8 GHz.

First, we will examine the obtained Pareto frontier. Consider figures (2), (3), (4) and (5) that depict the Pareto front resulting from solving the third, sixth, thirteenth and fifteenth instances, respectively. Figure (2) shows that in the 3th instance, as one of the small size problems, all three algorithms had almost the same performance. The MOPSO algorithm has been able to outperform the other two algorithms by producing more Pareto solutions, although the solutions of the NSGA II algorithm and the epsilon constraint are better than the MOPSO algorithm in terms of quality and dispersion measure. But with the growth of the problem size, as can be seen from the three figures (3), (4) and (5), it is the NSGA II algorithm that overcomes the MOPSO algorithm, in fact, by increasing the size of the instances, the solutions obtained by NSGA II algorithm perform better than MOPSO algorithm in almost all measures.

It should be noted that because the behavior of the developed algorithms in small sample problems is similar to the third problem, medium size problems are similar to the sixth problem and large size problems are similar to the 13th problem, their solution diagrams are not shown.

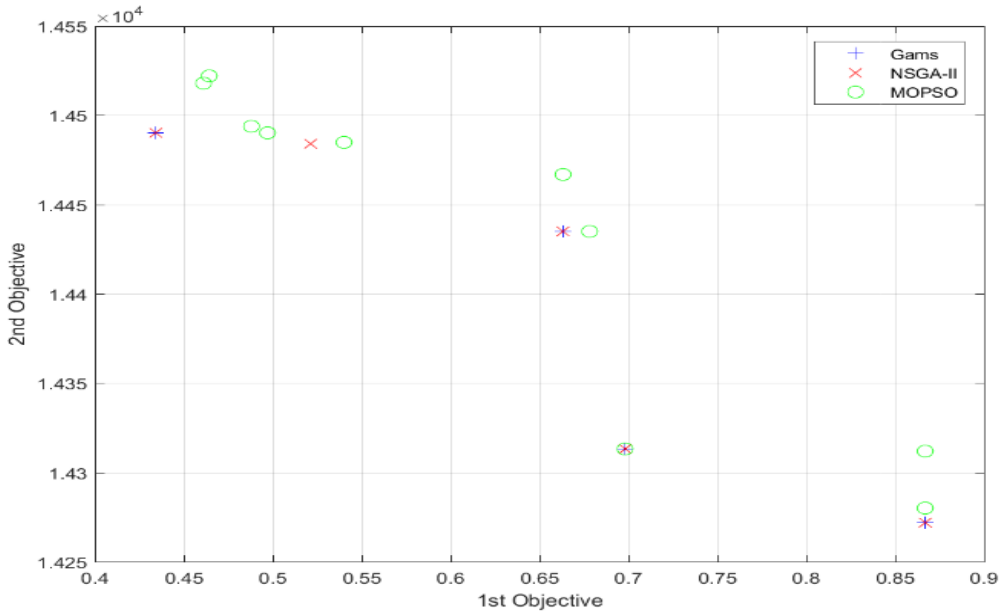

Figure 2: Pareto frontier for instance 3

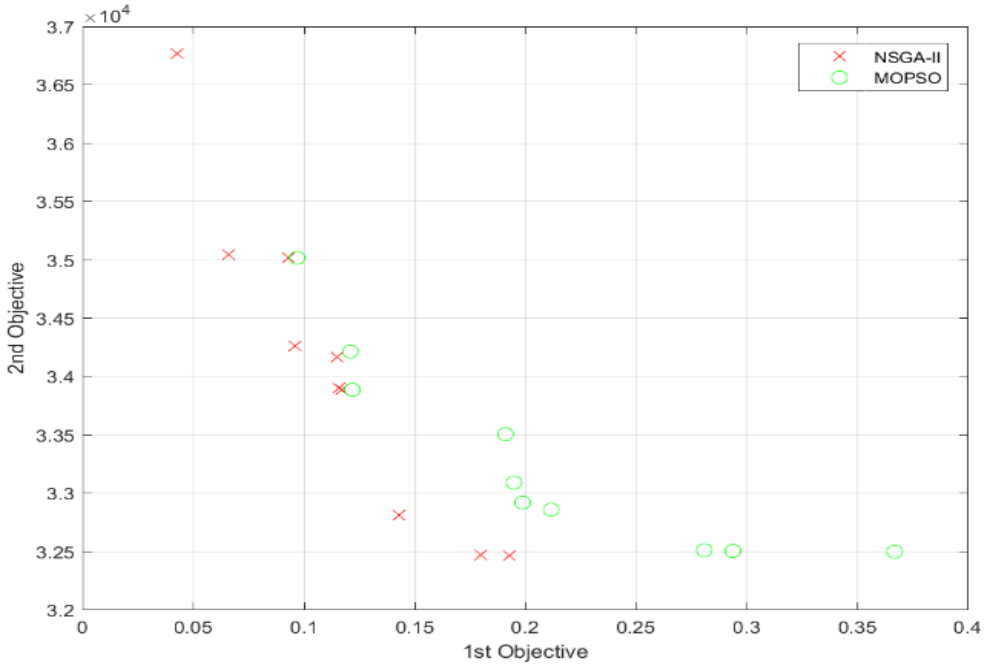

Figure 3: Pareto frontier for instance 6

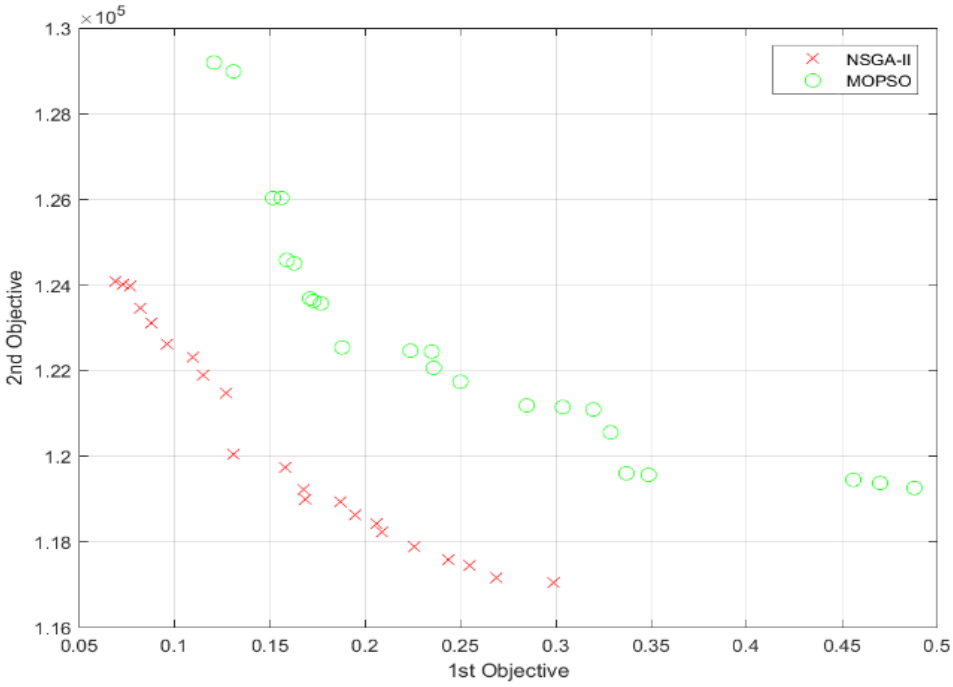

Figure 4: Pareto frontier for instance 13

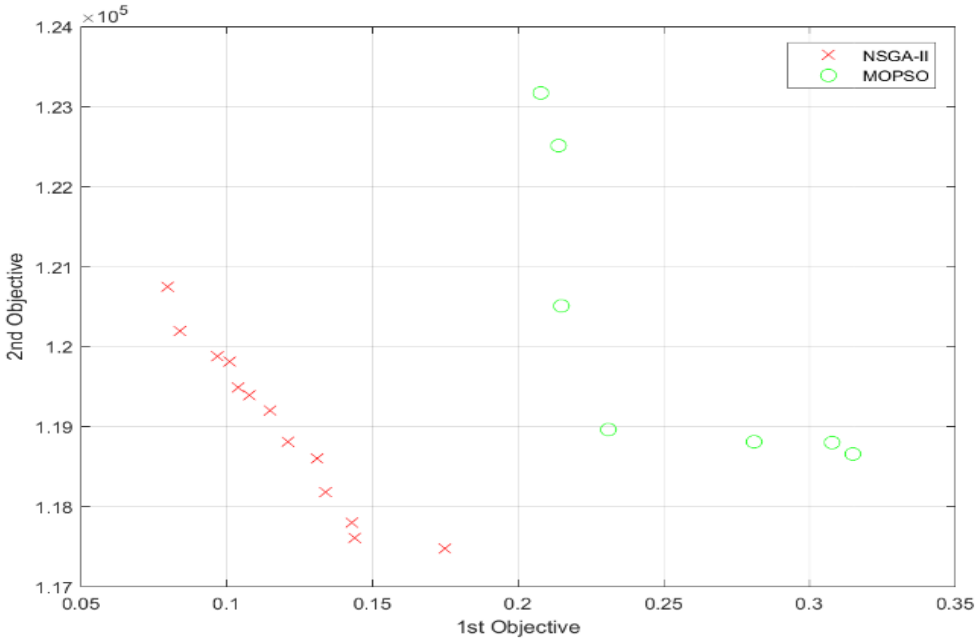

Figure 5: Pareto frontier for instance 15

Table 7 shows the values of all six measures. In this table, the highlighted numbers (in full color) actually depict the best results obtained by different algorithms in different measures.

As can be seen from Table 7, it was the MOPSO algorithm that was able to obtain more Pareto solutions and recorded a better performance than the other two algorithms. But as it is clear from the quality index, the solutions obtained by the MOPSO algorithm, despite being more numerous, are dominated by the solutions of the epsilon constraint and NSGA II algorithms. From the point of view of the distance from the ideal, the epsilon algorithm has dominated the other two algorithms in small size of instances, but in medium and large sizes that the epsilon algorithm was not able to solve, it is the NSGA II algorithm that has dominated the other algorithm. In the spacing criterion, the two proposed meta-heuristic algorithms have similar performance and none of them has won over the other, but in the last two measures, especially the time criterion, the MOPSO algorithm can be considered the absolute winner of the competition with the NSGA II algorithm.

Table 8 show the solution method with the best and the worst performance of the solution methods regarding the six measures.

To summarize, it can be said that despite the higher speed of the MOPSO algorithm and the generation of more diverse solutions by this algorithm, the solutions obtained by this algorithm did not have the necessary quality and the Pareto frontier produced by the NSGA II algorithm was able to dominate the competitor's algorithm absolutely, especially in the large size problem experience.

Table 7: The values of performance measures

Time			Greatest expansion			Spacing index			Distance from ideal point			Quality			# pareto solutions			
NSGA II	MOPSO	Epsilon constraint	NSGA II	MOPSO	Epsilon constraint	NSGA II	MOPSO	Epsilon constraint	NSGA II	MOPSO	Epsilon constraint	NSGA II	MOPSO	Epsilon constraint	NSGA II	MOPSO	Epsilon constraint	# instances
46.5	40.5	**2.2**	**1.41**	**1.41**	**1.41**	**0**	**0**	**0**	**1**	**1**	**1**	**100**	**100**	**100**	**2**	**2**	**2**	**1**
54.3	32.1	**3.7**	**1.41**	**1.41**	**1.41**	**0.66**	**0.66**	**0.66**	**0.94**	**0.94**	**0.94**	**100**	**100**	**100**	**3**	**3**	**3**	**2**
55.1	**25.4**	15891	1.32	**1.34**	1.32	0.12	0.16	**0.07**	0.84	0.91	**0.83**	**100**	10	**100**	5	**10**	4	**3**
54.6	21.8	**223.3**	**1.41**	**1.41**	**1.41**	0.34	**0.32**	0.34	0.84	**0.78**	0.84	**100**	60	**100**	4	**5**	4	**4**
107.8	**93.6**	25601	**1.41**	1.22	0.97	0.09	**0.09**	0.10	0.71	0.71	**0.69**	88.8	55.5	**100**	**9**	**9**	5	**5**
542.4	**146.3**	---	**1.10**	1.01	---	0.12	**0.08**	---	**0.51**	0.60	---	**100**	0	---	**10**	**10**	--	**6**
583.5	**194.7**	---	1.00	**1.07**	---	0.14	**0**	---	**0.51**	0.92	---	**100**	0	---	**5**	2	--	**7**
590.4	**217.3**	---	0.75	**1.13**	---	**0.05**	0.07	---	**0.35**	0.80	---	**100**	0	---	**15**	**15**	--	**8**
548.8	**324.8**	---	0.50	**1.15**	---	**0.10**	0.44	---	**0.28**	0.85	---	**100**	0	---	4	**7**	--	**9**
591.4	**315.0**	---	**1.33**	0.66	---	0.14	**0.10**	---	**0.62**	0.84	---	**100**	0	---	**14**	8	--	**10**
581.2	**408.7**	---	**1.19**	0.86	---	**0.14**	0.21	---	**0.67**	0.99	---	**100**	0	---	**8**	6	--	**11**
600.2	**487.7**	---	0.66	**0.89**	---	**0.07**	**0.07**	---	**0.37**	0.94	---	**100**	0	---	9	**11**	--	**12**
595.3	**459.9**	---	0.79	**1.19**	---	**0.02**	0.06	---	**0.42**	0.71	---	**100**	0	---	22	**23**	--	**13**
592.3	**456.5**	---	0.86	**1.05**	---	0.09	**0.07**	---	**0.42**	0.85	---	**100**	0	---	15	**16**	--	**14**
620.1	**458.6**	---	0.70	**0.91**	---	**0.03**	0.11	---	**0.36**	0.94	---	**100**	0	---	**13**	7	--	**15**

Table 8: The best and the worst proposed solution methods		
Measure of performance	The best method	The worst method
Time	MOPSO	Epsilon Constraint
Greatest expansion	MSPSO , NSGA II	Epsilon Constraint
Spacing index	MSPSO , NSGA II	Epsilon Constraint
Distance from ideal point	NSGA II	Epsilon Constraint
Quality	NSGA II	MOPSO
Number of pareto solutions	MOPSO	Epsilon Constraint

6. Managerial insights

This research well showed for managers the importance of considering the reduction of energy consumption in production planning. Numerical experiments showed that we may implement a schedule that has a minimum weighted of tardy jobs, but imposes a high energy consumption on the workshop and vice versa.

Often the workshop managers' values are in conflict with the customers' values, and we must to choose a schedule based on a compromise between these values. This research presented a mathematical model and two meta-heurists to achieve this compromise.

Another point is that without any new investment, the managers could improve the performance of their workshop.

7. Conclusion and future research direction

In this research, a bi-objective scheduling problem of incompatible jobs families with the same size on a batch processing machine with limited capacity was investigated. The first objective function pays attention to the customer's interests and is equal to the minimization of the total weighted number of tardy jobs. The second objective function considers the interests of the job shop owner and is equal to the minimization of the cost of electricity consumed by the batch processing machine under the time-dependent electricity pricing policy.

To find the optimal solutions, a bi-objective integer mathematical model was developed and the epsilon constraint method was used to reach optimal Pareto solutions of the problem. Also, two meta-heuristic multi-objective methods NSGA-II and MOPSO were presented to solve problems with medium and large sizes and find Pareto solutions close to optimal.

The performance of the developed model and methods was evaluated based on six conventional criteria in the literature on multi-objective optimization on 15 sample instances of the problem that were randomly generated. The computational results showed that the mathematical model is only able

to solve small-sized problems in a reasonable time, and the NSGA-II algorithm has a better performance than MOPSO in most of the evaluation measures, despite the high computational time in most of the evaluation samples.

There are many areas for further research. Including considering the job shop environments of parallel machines and flow shop, considering jobs with different sizes and dynamic arrival time, and modeling the problem in uncertain conditions.

References

[1] Gao, K., Huang, Y., Sadollah, A., & Wang, L. (2020). A review of energy-efficient scheduling in intelligent production systems. *Complex & Intelligent Systems*, *6*(2), 237-249.
[2] Terbrack, H., Claus, T., & Herrmann, F. (2021). Energy-oriented production planning in industry: a systematic literature review and classification scheme. *Sustainability*, *13*(23), 13317.
[3] Ding J.Y., Song S.J., Wu C. (2016) Carbon-efficient scheduling of flow shops by multi-objective optimization. European Journal of Operational Research, 48(3): 758-771.
[4] Mouzon, G. (2008) Operational Methods and Models for Minimization of Economic Literature of Energy Consumption in a Manufacturing Environment. Wichita State University. 46, 871-909.
[5] Luo H., Du B., Huang G.Q., Chen H.P., Li X.L. (2013) Hybrid flow shop scheduling considering machine electricity consumption cost. International Journal of Production Economics, 146: 423-439.
[6] Liu Y., Dong H.B., Lohse N., Petrovic S., Gindy N. (2014a) An investigation into minimising total energy consumption and total weighted tardiness in job shops. Journal of Cleaner Production, 65: 87-96.
[7] Mouzon G, Yildirim MB, Twomey J (2007) Operational methods for minimization of energy consumption of manufacturing equipment. Int J Prod Res 45(18–19):4247–4271
[8] Li L, Sun ZY, Yao XF, Wang DH (2016) Optimal production scheduling for energy efficiency improvement in biofuel feedstock preprocessing considering work-in-process particle separation. Energy 96:474–481
[9] Lora AT, Riquelme JC, Ramos JLM, Santos JMR, Exposito AG (2003) Application of evolutionary computation techniques to the optimal short-term scheduling of the electrical energy production. Curr Top Artif Intell 3040:656–665
[10] He Y, Liu F, Cao HJ, Li CB (2005) A bi-objective model for jobs hop scheduling problem to minimize both energy consumption and makespan. J Cent S Univ Technol 12:167–171
[11] Junheng CHENG, Feng CHU, Weili XIA, Jianxun DING, Xiang LIN Bi-objective optimization for single-machine batch scheduling considering energy cost*
[12] Van der Zee, D. J., Van Harten, A., & Schuur, P. C. (1997). Dynamic Job Assignment Heuristics for Multi-Server Batch Operations - A Cost based Approach. International Journal of Production Research, 35(11), 3063–3094.
[13] Fanti, M. P., Maione, B., Piscitelli, G., & Turchiano, B. (2007). Heuristic Scheduling of Jobs on a Multi-Product Batch Processing Machine. International Journal of Production Research, 34 (8), 2163–2186.
[14] Gokhale, R., & Mathirajan, M. (2011). Heuristic Algorithms for Scheduling of a Batch Processor in Automobile Gear Manufacturing. International Journal of Production Research, 49, pp. 2705–2728.
[15] Ozturk, O., Espinouse, M. L., Mascolo, M. D., & Gouin, A. (2012). Makespan Minimization on Parallel Batch Processing Machines with Non-Identical Job Sizes and Release Dates. International Journal of Production Research, 50 (20), 6022–6035.

[16] Potts, C. N., & Kovalyov, M. Y. (2000). Scheduling with Batching: A Review. European Journal of Operational Research, 120 (2), 228–249
[17] M. Mathirajan, A.I. Sivakumar, A literature review, classification and simple meta-analysis on scheduling of batch processors in semiconductor, Int. J. Adv.Manuf. Technol. 29 (9–10) (2006) 990–1001
[18] Mönch, L., Fowler, J. W. & Mason, S. J. (2013). Production planning and control semiconductor wafer fabrication facilities: Modeling, analysis and systems. Operations research/computer science interfaces. New York: Springer.
[19] H. Hadera, and I. Harjunkoski, (2013)"Continuous-time Batch Scheduling Approach for Optimizing Electricity Consumption Cost," Proceedings of the 23rd European Symposium on Computer Aided Process Engineering-ESCAPE 23, pp. 403-408.
[20] H. Luo, G.Q. Huang, Y.F. Zhang, Q.Y. Dai, X. Chen, (2009) Two-stage hybrid batching flowshop scheduling with blocking and machine availability constraints using genetic algorithm, Robot. Comput. Integr. Manuf. 25 (6) 962–971.
[21] H. Luo, G.Q. Huang, Y.F. Zhang, Q.Y. Dai, (2011) Hybrid flowshop scheduling withbatch-discrete processors and machine maintenance in time windows, Int. J.Prod. Res. 49 (6) 1575–1603.
[22] Kenneth R. Baker, Dan Trietsch, Principles of Sequencing and Scheduling, 2019 John Wiley and Sons Int.
[23] Brucker, P., Knust, S. (2012). Algorithms and Complexity. In: Complex Scheduling. GOR-Publications. Springer, Berlin, Heidelberg.
[24] Jolai, Fariborz. (2005): Minimizing number of tardy jobs on a batch processing machine with incompatible job families." European Journal of Operational Research 162.1,184-190.
[25] Fowler, J. W., & Mönch, L. (2022). A survey of scheduling with parallel batch (p-batch) processing. *European Journal of Operational Research*, *298*(1), 1-24.
[26] Cheng J, Chu F, Xia W, Ding J, Ling X (2014) Bi-objective optimization for single-machine batch scheduling considering energy cost. In: Proceedings of the 2014 International Conference on Control, Decision and Information Technologies (CoDIT), Metz, pp 236–241
[27] Cheng J, Chu F, Chu C, Xia W (2016a) Bi-objective optimization of single-machine batch scheduling under time-of-use electricity prices. RAIRO Oper Res 50(4–5):715–732
[28] Peng Wu1 · Junheng Cheng2,3 · Feng Chu1,3 (2019) Large-scale energy-conscious bi-objective single-machine batch scheduling under time-of-use electricity tariffs via effective iterative heuristics Annals of Operations Research https://doi.org/10.1007/s10479-019-03494-7
[29] Cheng J, Chu F, Liu M, Wu P, Xia W (2017) Bi-criteria single-machine batch scheduling with machine on/off switching under time-of-use tariffs. Comput Ind Eng 112:721–734
[30] Cheng J, Chu F, Liu M, Xia W (2016b) Single-machine batch scheduling under time-of-use tariffs: new mixed-integer programming approaches. In: Proceedings of the 2016 IEEE international conference on systems, man, and cybernetics (SMC), pp 3498–3503
[31] Wang S, Liu M, Chu F, Chu C (2016) Bi-objective optimization of a single machine batch scheduling problem with energy cost consideration. J Clean Prod 137:1205–1215
[32] Shibohua Zhang, Ada Che, Xueqi Wu & Chengbin Chu (2017): Improved mixed-integer linear programming model and heuristics for bi-objective single machine batch scheduling with energy cost consideration, Engineering Optimization
[33] Shengchao Zhou, Mingzhou Jin , Ni Du, (2020) Energy-efficient scheduling of a single batch processing machine with dynamic job arrival times. Energy 209 118420
[34] Shengchao Zhou, Xiaolin Li, Ni Du, Yan Pang, Huaping Chen, (2018) A multi-objective differential evolution algorithm for parallel batch processing machine scheduling considering electricity consumption cost, *Computers and Operations Research*
[35] Jia Z-H, Zhang Y-I, Leung JY-T, Li K (2017) Bi-criteria ant colony optimization algorithm for minimizing makespan and energy consumption on parallel batch machines. Appl Soft Comput 55:226–237

[36] Jia Z-H, Wang Y, Wu C, Yang Y, Zhang X-Y, Chen H-P (2019) Multi-objective energy-aware batch scheduling using ant colony optimization algorithm. Comput Ind Eng 131:41–56
[37] S.-y. Qian, Z.-h. Jia and K. Li, (2020A multi-objective evolutionary algorithm based on adaptive clustering for energy-aware batch scheduling problem, Future Generation Computer Systems
[38] Rocholl J, Mönch L, Fowler JW (2018) Electricity power cost-aware scheduling of jobs on parallel batch processing machines. In: Proceedings of the 2018 Winter Simulation Conference, Gothenburg, pp 3420–3431
[39] Jens Rocholl · Lars Mönch · John Fowler (2020) Bi-criteria parallel batch machine scheduling to minimize total weighted tardiness and electricity cost Journal of Business Economics 90:1345–1381
[40] Dauzère-Pérès, S., & Mönch, L. (2013). Scheduling jobs on a single batch processing machine with incompatible job families and weighted number of tardy jobs objective. Computers & Operations Research, 40(5), 1224-1233.
[41] Margarita Reyes-Sierra and Carlos A. Coello Coello (2006) Multi-Objective Particle Swarm Optimizers: A Survey of the State-of-the-Art. International Journal of Computational Intelligence Research. Vol.2, No.3, pp. 287–308
[42] Shanu Verma, Millie Pant and Vaclav Snasel (2021) A Comprehensive Review on NSGA-II for Multi-Objective Combinatorial Optimization Problems. IEEE Access. VOL 9, 57757-57791.
[43] Ling Xue, Xiuli Wang (2023) A multi-objective discrete differential evolution algorithm for energy-efficient two-stage flow shop scheduling under time-of-use electricity tariffs. Applied Soft Computing Vol 133, 109946.
[44] Lei Yue, Hao Wang, Jabir Mumtaz, Mudassar Rauf, Zhifu Li (2023). Energy-efficient scheduling of a two-stage flexible printed circuit board flow shop using a hybrid Pareto spider monkey optimisation algorithm. Journal of Industrial Information Integration Vol. 31, 100412.
[45] Kiran V. Sagar, J. Jerald & Muhammed Anaz Khan (2023) A multi-objective scheduling model for a flexible manufacturing system to reduce peak load using an energy storage system. International Journal on Interactive Design and Manufacturing (IJIDeM) . DOI: https://doi.org/10.1007/s12008-023-01334-4
[46] Alireza Malekijavan , Hamidreza Zafarani, Mehdi Aslinejad (2022) Stochastic Optimal Operation of a Multi carrier Energy System with Electric Vehicles and Combined Heat and Power Units**.** Iranian Journal of Operations Research. Vol 13, No. 1, pp. 83-102
[47] Saiedeh Gholami, Mahdi Jalalian, Reza Ramezanian (2016) Exploring Energy Efficiency and Service Quality of Airlines with Cruise Speed Control**.** Iranian Journal of Operations Research. Vol 7, No. 1, pp. 43-68.
[48] Hamidreza Haddad (2021) Implementation of clustering on a multi objective single machine batch scheduling problem (A case study in Iran)**.** I Iranian Journal of Operations Research. Vol 12, No. 1, pp. 109-126.