Journal of Cleaner Production 193 (2018) 424–440

Contents lists available at ScienceDirect

Journal of Cleaner Production

journal homepage: www.elsevier.com/locate/jclepro

Bi-objective identical parallel machine scheduling to minimize total energy consumption and makespan

Shijin Wang [a], Xiaodong Wang [a], Jianbo Yu [b], Shuan Ma [a], Ming Liu [a, *]

[a] Department of Management Science & Engineering, School of Economics & Management, Tongji University, Shanghai 200029, PR China
[b] Department of Industrial Engineering, School of Mechanical Engineering, Tongji University, Shanghai 200029, PR China

ARTICLE INFO

Article history:
Received 5 December 2017
Received in revised form
6 May 2018
Accepted 7 May 2018
Available online 12 May 2018

Keywords:
Identical parallel machine scheduling
Makespan
Total energy consumption
Augmented ε
-constraint method
Constructive heuristic
NSGA-II

ABSTRACT

Currently, energy consumption reduction is playing a more and more important role in production and manufacturing, especially for energy-intensive industries. An optimal production scheduling can help reduce unnecessary energy consumption. This paper considers an identical parallel machine scheduling problem to minimize simultaneously two objectives: the total energy consumption (TEC) and the makespan. To tackle this NP-hard problem, an augmented ε-constraint method is applied to obtain an optimal Pareto front for small-scale instances. For medium- and large-scale instances, a constructive heuristic method with a local search strategy is proposed and the NSGA-II algorithm is applied to obtain good approximate Pareto fronts. Extensive computational experiments on randomly generated data and a real-world case study are conducted. The result shows the efficiency and effectiveness of the proposed methods.

1. Introduction

Nowadays, with the rapid economic growth, expanding populations and acceleration of globalization, energy demand keeps a rapid and ongoing growth at a rate of 30-year increase of 56% between 2010 and 2040 (EIA, 2013). Among all sectors, the industrial sector, including manufacturing, continues to contribute nearly 50% the world's total delivered energy. More specifically, the world's total energy consumption in industrial sector will grow from 58.9 PWh in 2010 to 90.4 PWh in 2040 (EIA, 2013). As for China, industrial sector consumes about 69.44% of the total energy consumption in 2014, among which 82.9% is due to manufacturing sector (CSY, 2016). In 2014, the total energy consumption in manufacturing in China is 295,686.4 units with each unit of 10,000 t of SCE (Standard Coal Equivalent).

The rapid growing energy consumption is one of the main reasons for the environment pollution due to the increasing amount of the green house gas emissions, in particular, the carbon dioxide (CO_2) (Ding et al., 2016a). For example, about 28% greenhouse gas emissions in U.S. (Mouzon, 2008), about 18–20% CO_2 emission in Germany (Luo et al., 2013), and at least 26% of the total CO_2 emission in China (Liu et al., 2014a), are from the manufacturing energy consumption, especially for energy-intensive industries including mould, chemical, glass and petroleum. For instance, the energy consumption of smelting and pressing of ferrous metals industry (i.e., mainly for injection moulds and stamping die) accounts for around 28.8% of the total energy consumption of manufacturing sector in 2014 (CSY, 2016). Therefore, it is highly urgent to improve energy usage efficiency in manufacturing, to save energy consumption and further to reduce greenhouse gas emissions (Gahm et al., 2016; Giret et al., 2015; Merkert et al., 2015).

In this paper, motivated by scheduling challenges of an automobile stamping die manufacturing shop floor and based on our preliminary work in Wang and Ma (2016), we study a bi-objective parallel machine scheduling problem with energy consideration, in which machines have different energy consumption rates. The time-of-use (TOU) electricity price, one of the most common demand-side management strategies, is considered. TOU means different electricity prices during different times of day, which is practical in real-world electricity market due to the peak, valley and normal periods. Two objectives are considered: the makespan and the total energy consumption (TEC). The reasons for considering

* Corresponding author.
E-mail addresses: shijinwang@tongji.edu.cn (S. Wang), mingliu@tongji.edu.cn (M. Liu).

https://doi.org/10.1016/j.jclepro.2018.05.056
0959-6526/

these two objectives are as follows: (i) a minimum makespan usually implies a high utilization of the machines in the shop floor, and (ii) energy takes a large portion of the total costs in the stamping die manufacturing company, among which electricity accounts for more than 80%. According to the financial report of the case study company in 2016, the total cost of electricity takes 82% of the total energy costs, water takes 6.3%, gasoline takes 7.3% and diesel oil takes 4.4%, respectively. While most electricity consumptions are attribute to the machining processes in the CNC shop floor.

The contributions of the paper can be summarized as follows:

(i) A bi-objective identical parallel machine scheduling problem is derived from a real-world automobile stamping die manufacturing. By introducing two variables and three constraints related with the completion time of jobs, the single-objective model in Che et al. (2017b) is extended to be a bi-objective mixed integer programming for minimizing TEC and makespan simultaneously. Different energy consumption rates of machines and the TOU electricity prices are considered.
(ii) An augmented ε-constraint method is adapted to obtain the exact Pareto front for small-scale instances and for medium- and large-scale instances, two methods: a constructive heuristic method with a local search strategy and an application of the NSGA-II algorithm are proposed to obtain good approximate Pareto fronts in a reasonable computation time.
(iii) Performance of the methods are compared and validated by extensive computational experiments with instances randomly generated and a real-world case study.

The remainder of the paper is organized as follows. Section 2 reviews the relevant literature. Section 3 describes the problem and extension of the model in Che et al. (2017b) to our problem. Section 4 introduces the augmented ε-constraint method and two methods for approximate Pareto fronts. Section 5 provides the computational experiments with the comparisons. Finally, Section 6 concludes the paper and gives out the future research directions.

2. Literature review

In recent years, production scheduling with considerations of energy efficiency has been receiving more and more research interests and attentions. Following the prior work of Mouzon et al. (2007), there have been many emerging researches on energy-efficient scheduling in various manufacturing settings including single machine (cf., Che et al., 2017a; Cheng et al., 2017; Fang et al., 2016; Che et al., 2016; Liu et al., 2014b; Shrouf et al., 2014; Yildirim and Mouzon, 2012), parallel machine (cf., Wu and Che, 2018; Zeng et al., 2018; Che et al., 2017b; Wang et al., 2017; Ding et al., 2016a; Li et al., 2016a; Li et al., 2016b; Fang and Lin, 2013; Ji et al., 2013; Li et al., 2011), flow shop (cf., Mansouri, et al., 2016; Ding et al., 2016b; Lin et al., 2015; Zhang et al., 2014), job shop (cf., Zhang and Chiong, 2016; Liu et al., 2014a), flexible flow shop (cf., Tang et al., 2016; He et al., 2015; Dai et al., 2013; Luo et al., 2013; Bruzzone et al., 2012) and parallel batch machine (cf., Jia et al., 2017). Readers are referred to Gahm et al. (2016) and Giret et al. (2015) for a comprehensive surveys of energy-efficient scheduling in manufacturing.

2.1. Parallel machine scheduling with energy considerations

In the following, we mainly discuss the papers that focus on the parallel machine scheduling with energy considerations in manufacturing setting.

Li et al. (2011) studied the identical parallel machine scheduling problem to minimize the makespan, in which the processing times are controllable with limited resource consumption, and critical and non-critical machines are considered. A simulated annealing algorithm is designed to solve the problem.

Fang and Lin (2013) addressed a parallel machine scheduling problem to minimize the total weighed job tardiness and power cost. The machines are heterogeneous with adjustable processing speeds. They proposed an integer linear programming formulation and two objectives are added up to form a single objective. Two constructive heuristic methods and a particle swarm optimization (PSO) algorithm are developed to tackle the problem.

Ji et al. (2013) studied a uniform parallel machine scheduling problem to minimize the total resource consumption (including carbon emission, water consumption and electricity usage) with a bound of makespan. A mixed integer linear programming model is formulated and the strong NP-hardness of the problem is proven. A heuristic and a PSO algorithm are developed to solve the problem.

Li et al. (2016a) studied an unrelated parallel machine scheduling problem to minimize the energy and tardiness cost. The energy consumption of machines for running, idle and warm-up are included. A mathematical model is developed and the two objectives are added together to come up with a single objective. Ten heuristic methods with priority rule, energy consumption, and combinational rules are proposed to solve the problem.

Li et al. (2016b) studied a parallel machine scheduling problem to minimize the makespan or the total completion time, with the constraints that the total cost (the sum of the energy cost and the clean up cost) is within a given range. They proposed a linear time algorithm for the preemptive case of makespan minimization, and they proposed efficient heuristics for non-preemptive cases of makespan minimization and the total completion time minimization, respectively.

Ding et al. (2016a) considered a parallel machine scheduling problem to minimize the total electricity cost with completion time limited to a predetermined production deadline. They constructed a time-interval-based mixed integer programming model (i.e., the time periods with the same electricity price are used as an interval) and reformulated that using Dantzig-Wolfe decomposition, based on which a column generation heuristic is developed.

Zeng et al. (2018) investigated a bi-objective scheduling on uniform parallel machines to minimize the total electricity cost and the number of machines used under TOU tariffs, and developed an iterative search framework to obtain the Pareto fronts.

Che et al. (2017b) studied an energy-conscious unrelated parallel machine scheduling problem under TOU electricity pricing to minimize the total electricity cost in bounded makespan. A two-stage algorithm with insertion is proposed to solve this problem.

Wang et al. (2017) investigated a parallel machine scheduling problem with a bounded electricity power demand peak. Jobs can be processed with different cutting modes, which corresponds to different processing times and power demands. The objective is to minimize the makespan. A genetic algorithm based two-stage heuristic method is developed.

Wu and Che (2018) considered an energy-efficient bi-objective unrelated parallel machine scheduling problem to minimize both makespan and total energy consumption. A memetic differential evolution (MDE) algorithm is proposed to tackle this problem. List scheduling heuristic and local search are developed to strengthen the algorithm.

2.2. Related researches on life cycle assessment perspective

In the perspective of a life cycle assessment (LCA), the CO_2 emissions generated in transportation and recycling procedure will

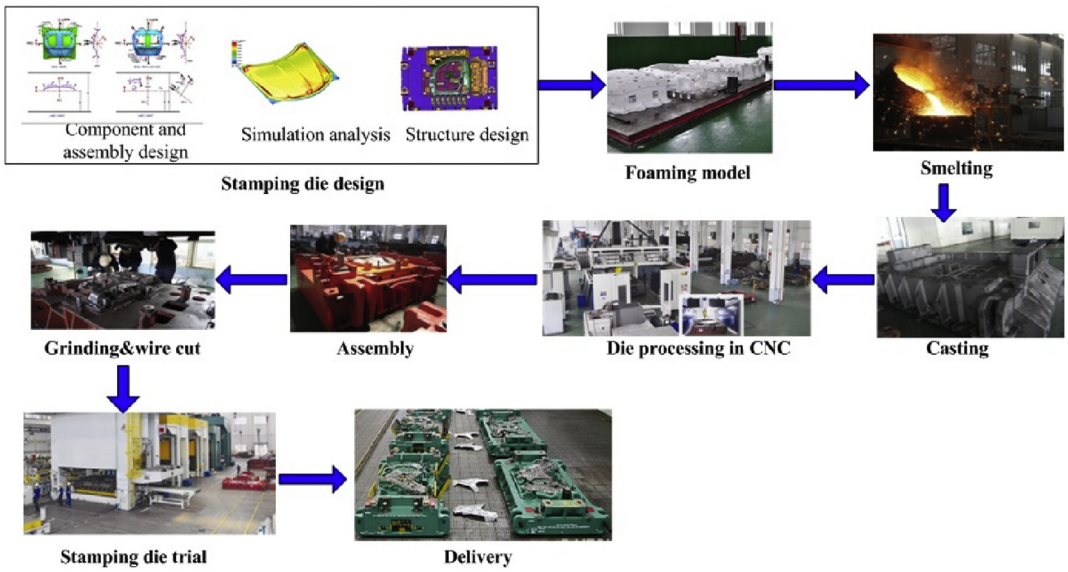

Fig. 1. Schematic illustration of the manufacturing process of automobile stamping dies.

also be included, not only the energy consumption and CO_2 emissions in the production process.

Ahmadi et al. (2016b) investigated the multi-objective eco-design problem, in which several objectives including environmental, economic, and technological are supposed to be optimized simultaneously. They presented a new framework named AMOEA-MAP aiming to solve multi-objective problems efficiently. Two novel operators: a memory-based adaptive partitioning strategy, and a bi-population evolutionary algorithm are proposed. Tests on several tough benchmarks and application on a real-world eco-design problem validated its performance.

In their subsequent work, Ahmadi et al. (2016a) extended the structure of Process Modelling and Life Cycle Assessment by involving the life cycle networks of various energy resources and raw materials. The AMOEA-MAP is adopted to obtain the Pareto-optimal solutions related with cost-environmental impacts. The methodology is demonstrated by the application on a conventional drinking water production plant.

2.3. Summary of literature review

From the review mentioned above, many previous studies have investigated energy consumption reduction in production scheduling. Various mathematical models and algorithms have been developed to solve energy-efficient scheduling problems. However, to the best of our knowledge, in the existing researches on energy-efficient parallel machine scheduling, some researches consider just one objective of minimizing total electricity cost, some researches transform bi-objective or multiple objectives into a weighted sum single objective function, and other researches take either the makespan or the energy (or resource) as side constraints to optimize a single objective problem.

There are rather few researches on bi-objective parallel machine scheduling problems considering makespan and energy consumption simultaneously. There are two researches that are most related to our study, Zeng et al. (2018) and Wu and Che (2018). However, in Zeng et al. (2018), one objective they considered is different from ours. They considered the number of machines used and we considered the makespan. Wu and Che (2018) studied an energy-efficient bi-objective unrelated parallel machine scheduling problem to minimize both makespan and total energy consumption. We focused more on identical parallel machine version based on the practical application in a real-world CNC processing of an automobile stamping die manufacturing shop floor. They proposed a metic differential evolution algorithm combined with problem property based heuristic and local search method. We applied the augmented ε-constraint (an exact method for optimal solutions), proposed a constructive heuristic method and applied the NSGA-II algorithm. The problems with randomly generated data and case data are tested and the results show the promising performance of our methods.

3. Problem description

The deterministic scheduling problem is originated from an automobile stamping die manufacturing. The schematic manufacturing process of automobile stamping die is shown in Fig. 1, which covers the whole process from the design to delivery. In this study, we focus on the identical parallel machine processing for stamping dies in the CNC shop floor. Many different moulds and stamping dies will be casted and transferred to the CNC shop floor with hoists for further processing. Machines in the CNC shop floor have different energy consumption rates since some machines employ green technology while others employ regular technology.

The scheduling problem in the CNC shop floor is described as follows. There are n independent, non-preemptive jobs to be processed on m identical parallel machines in a given $T = |\mathscr{T}|$ time periods, where $\mathscr{T}$ is the set of time periods. At time period 0, all jobs are available for processing. Each machine can only process one job at a time and one job can only be processed on one machine. The processing time p_j of a job j, $j \in \mathscr{N}$ on any alternative machine is same. The energy consumption rate e_i of machines i, $i \in \mathscr{M}$ are different, where $\mathscr{N}$ is the set of jobs, $\mathscr{M}$ is the set of machines, and $n = |\mathscr{N}|$, $m = |\mathscr{M}|$. Also, due to time-of-use (TOU) policy, the given T time periods are divided into K time intervals. One certain time interval may contain one or more time periods with the same energy (especially electricity) price, but the energy price c_k of each time interval k, $k \in \mathscr{K}$ is not necessary same, where $\mathscr{K}$ is the set of time intervals and $K = |\mathscr{K}|$. The duration of time interval k is donated as T_k, $T = \sum_{k \in \mathscr{K}} T_k$.

Algorithm 1
The augmented ε-constraint method.

1: Compute the Ideal point $\mathbf{f}^I = (f_1^I, f_2^I)$ and Nadir point $\mathbf{f}^N = (f_1^N, f_2^N)$.
2: Set $\mathbf{F} = \{(f_1^N, f_2^I)\}$ and $\varepsilon = f_1^N - \triangle$ ($\triangle = 1$ for this problem).
3: **while** $\varepsilon \geq f_1^I$ **do**
4: Solve the ε-constraint problem with $f_1 + s = \varepsilon$ as a constraint and $f_2 - eps \times s$ as the single objective function to optimality.
5: Add the optimal solution value (f_1^*, f_2^*) to $\mathbf{F}$.
6: Set $\varepsilon = f_1^* - \triangle$.
7: **end while**

With the duration time T_k and the energy price c_k of the time interval k, the energy price c'_t of every time period t, $t \in \mathscr{T}$ can be obtained, e.g., if time interval k contains time period t, then the energy price of this period t, c'_t is equal to the energy price of current time interval k, i.e., $c'_t = c_k$.

Energy consumption is occurred on each time period when a machine is processing a job. The energy consumption when machine is idle can be ignored. The scheduling decisions are to assign the jobs to the machines, to sequence the jobs on each machine and to determine the starting time of each job, such that the makespan and the total energy consumptions are minimized simultaneously. Using the three-field notation (Graham et al., 1979), the problem can be denoted by $Pm, TOU||\{C_{max}, TEC\}$, where "$P_m$" represents the identical parallel machine, "*TOU*" represents the time-of-use policy of energy price, "C_{max}" represents the makespan, and "*TEC*" represents the total energy consumption.

On the basis the single-objective model for minimizing *TEC* in Che et al. (2017b), we introduce two variables to characterize the completion time of every job, $C_j, j \in \mathscr{N}$ and the maximum completion time of all jobs, C_{max}. Two other constraints (16) ~ (17) that confine the value of C_j as well as define the relationship between C_j and C_{max} are formulated. The modified bi-objective mathematical model is given in Appendix.

4. Solution approaches

For bi-objective and multi-objective problems, there is no single optimal solution, but rather a series of Pareto optimal solutions which constitute the Pareto front. The following subsections present an augmented ε-constraint method to obtain exact Pareto front, and a constructive heuristic method with a local search strategy and an application of the NSGA-II algorithm to obtain approximate Pareto optimal fronts.

Algorithm 3
The local search strategy.

while $i = 1$ to m **do**
2: Find the jobs on machine i which could be seen as one job or block. Use the set $S_1, S_2, \ldots, S_B$ to record them, where B is the total number of blocks.
3: **for** $b = 1$ to B **do**
4: Search the change of the starting time for S_b which could bring the most reduction of energy consumption on machine i.
5: Update the status of periods for the machine i that are assigned to process.
6: **end for**
7: **end while**

4.1. The augmented ε-constraint method

The ε-constraint method is widely used to solve bi-objective optimization problems. The basic idea of the ε-constraint method is to focus on a single objective and restrict remaining objectives. The augmented ε-constraint method is an improved version (Mavrotas, 2009). Compared to the traditional ε-constraint method, it only generates efficient solutions and it avoids the generation of weakly efficient solutions (Mavrotas, 2009).

According to the idea of the augmented ε-constraint method, we need to obtain the upper bound and lower bound of the objective that are used as constraints. Therefore, the following concepts are needed (Bérubé et al., 2009).

Ideal point: let $\mathbf{f}^I = (f_1^I, f_2^I)$ with $f_1^I = \min\{f_1(\mathbf{X})\}$ and $f_2^I = \min\{f_2(\mathbf{X})\}$, $\mathbf{X} \in \mathbf{Z^V}$, where $\mathbf{Z^V}$ is a V-dimensional decision variable vector;
Nadir point: let $\mathbf{f}^N = (f_1^N, f_2^N)$ with $f_1^N = \min\{f_1(\mathbf{X}) : f_2(\mathbf{X}) = f_2^I\}$ and $f_2^N = \min\{f_2(\mathbf{X}) : f_1(\mathbf{X}) = f_1^I\}$, $\mathbf{X} \in \mathbf{Z^V}$;
Extreme point: $f_1^E = (f_1^I, f_2^N)$ and $f_2^E = (f_1^N, f_2^I)$ are two extreme points on the Pareto front.

For problem $Pm, TOU||\{C_{max}, TEC\}$, $f_1 = min\{C_{max}\}$ can be regarded as a constraint. The detailed method is shown in Algorithm 1. In the method, *eps* is an adequately small number (usually between 10^{-3} and 10^{-6}), s is a positive integer variable. $\triangle$ is a positive constant for reducing ε at each iteration, which is for spanning the whole solution space. In this problem, $\triangle = 1$, since the minimum unit of change value of the makespan is one.

Algorithm 2
The constructive heuristic method.

Step 1. Input the information of machines, jobs, time periods, processing times, and energy prices.
Step 2. Sort the jobs in non-increasing order of processing times, $p_j, j \in \mathscr{N}$, i.e., $p_{[1]} \geq p_{[2]} \geq \ldots \geq p_{[n]}$, where $[j]$ is the jth job in the order.
Step 3. Sort the machines in non-decreasing order of energy consumption rates, $e_i, i \in \mathscr{M}$, i.e., $e_{[1]} \leq e_{[2]} \leq \ldots \leq e_{[m]}$, where $e_{[i]}$ represents the ith energy consumption rate in the order.
Step 4. Define $h_i^t = e_i c'_t$, $i = \{[1], \ldots, [m]\}, t = \{1, \ldots, T\}$.
Step 5. Define $T_{min} = \sum_{j=1}^{n} p_j / m$. Let $NS = \varnothing$ be the set of solutions found by this procedure.
while $T \geq T_{min}$ **do**
for $j = 1 : n$ **do**
Step 6. For job j, search certain time interval with minimum $\sum_{l=t}^{t+p_j-1} h_i^l$ from all idle and continuous time periods on every machine $i = \{[1], \ldots, [m]\}$, with $t = \{1, \ldots, T - (p_j - 1)\}$. This selected time interval on machine i_j^* is the best option for job j since it causes the least energy consumption for processing job j. The earlier starting time is used as the tie-breaking for the same minimum $\sum_{l=t}^{t+p_j-1} h_i^l$.
Step 7. Update the working status of time periods that are assigned for processing job j on machine i_j^* to be busy, so that job $\{[j+1], \ldots, [n]\}$ cannot be assigned to these periods on this machine.
end for
Step 8. Calculate the makespan and total energy consumption of current solution s within current time periods T, and they are denoted as C_{max} and *TEC*, respectively.
Step 9. Add the current solution s to the set *NS*: $NS = NS \cup \{s\}$.
Step 10. Update the time periods: $T = C_{max} - 1$.
end while
Step 11. Obtain the non-dominated solutions in set *NS*.

Time interval	1		2	3		4	5	6	7	8	9		10	11	12
Price per interval	1		3	4		2	3	4	2	1	2		4	1	3
Time period	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Price per period	1	1	3	4	4	2	3	4	2	1	2	2	4	1	3
Schedule1	Job 2				Job 3			Job 4		Job 1					
Schedule2	Job 2				Job 3				Job 4	Job 1					
Schedule3	Job 2					Job 3			Job 4	Job 1					

Fig. 2. Schedules obtained by the constructive heuristic and local search strategy for an instance.

4.2. A constructive heuristic method

For small-scale instances, the augmented ε-constraint method could solve them optimally in acceptable computation time. However, it will be difficult to obtain the optimal Pareto front as the problem scale increases. In this subsection, a constructive heuristic with a local search strategy is proposed to solve large-scale instances. When the makespan is limited by a certain value, the idea for searching an optimal schedule is to avoid the job processing on machines with high energy consumption rates and in the periods with high electricity prices. The proposed constructive heuristic method is shown in Algorithm 2.

Algorithm 4
The initialization procedure.

1: Input the number of jobs n, the number of machines m, the number of time periods T, and population size N. Let $PopSize = 0$.
2: **while** $PopSize < N$ **do**
3: Generate T_{max} randomly from $\left[max\left(\sum_{j\in N_j} p_j/m, max(p_j)\right), T\right]$. Let $ECT_i = 0$ $(i \in \mathcal{M})$ be the current earliest available time for machine i. Let $AM_j = \varnothing$ $(j \in \mathcal{N})$ be the available machine set for job j. Let $AN_i = \varnothing$ $(i \in \mathcal{M})$ be the set of jobs assigned on machine i.
4: **for** every job, j $(j \in \mathcal{N})$, needed to be assigned **do**
5: **for** every machine, i $(i \in \mathcal{M})$ **do**
6: **if** $ECT_i + p_j \le T_{max}$ **then**
7: $AM_j = AM_j \cup \{i\}$.
8: **end if**
9: **end for**
10: Job j is assigned on machine i $(i \in AM_j)$ that is chosen randomly from AM_j. Let $AN_i = AN_i \cup \{j\}$ and $ECT_i = ECT_i + p_j$.
11: **end for**
12: **for** every machine, $i \in \mathcal{M}$ **do**
13: Sort jobs on machine i, i.e., $j \in AN_i$ randomly.
14: $T_{idle} = T_{max} - \sum_{j\in AN_i} p_j$.
15: **for** every job assigned on machine i, $j \in AN_i$ **do**
16: Generate $idlet_j$ randomly from $[0, T_{idle}]$.
17: $T_{idle} = T_{idle} - idlet_j$.
18: **if** $j = [1]$ **then**
19: The starting time of job j: $S_j = idlet_1$.
20: **else**
21: The starting time of job i: $S_j = S_{j-1} + p_{j-1} + idlet_j$.
22: **end if**
23: **end for**
24: **end for**
25: If the obtained solution is feasible, record the assignment and staring time information in the chromosome. Otherwise, regenerate the chromosome from Lines 3–24 until it is feasible.
26: Compute the makespan and TEC of current feasible solution. The generation of one individual is completed.
27: $PopSize = PopSize + 1$
28: **end while**
29: Obtain the initial population.

Step 4 in Algorithm 2 shows that each job is assigned to the time interval with the least energy consumption. It means that the change of the starting time of any job will not lead to the reduction of total energy consumption. However, some jobs are processed on the same machine consecutively, i.e., there are no idle time during these jobs processing. In this case, these jobs could be seen as one job or block so that the change of the starting time may bring the reduction of total energy consumption. With this idea, we present a local search strategy which could improve solutions obtained by Algorithm 2. The detailed local search procedure is shown in Algorithm 3.

In the following, an example is given to show the detailed procedure of the constructive heuristic method and the local search strategy.

Example 1. There is one machine with energy consumption rate $e_1 = 1$ and 4 jobs with the processing times $p_j = \{5, 4, 3, 2\}$ for jobs $j = \{1, \ldots, 4\}$, respectively. There are $K = 12$ time intervals and $T = 15$ time periods. The energy price c_k of every time interval k, the duration time T_k of every time interval k, and the energy price c'_t of every time period t are shown in Fig. 2. For job 1, that demands $p_1 = 5$ time periods, we calculate $\sum_{l=t}^{t+p_1-1} e_1 c'_l = \{13, 14, 16, 17, 15, 12, 12, 11, 11, 10, 12\}$, since $t = \{1, \ldots, T - (p_1 - 1)\} = \{1, \ldots, 11\}$. Hence, job 1 starts processing at $t = 10$ since the total energy consumption is minimum on the machine, with value 10.

Next, for job 2, the continuous and idle time with $p_2 = 4$ periods is only from $t = 1$ to $t = 9 - (p_2 - 1) = 6$ and the corresponding $\sum_{l=t}^{t+p_2-1} e_1 c'_l = \{9, 12, 13, 13, 13, 11\}$. Hence, the starting time of job 2 is $t = 1$.

Next job, job 3, can only be started from $t = \{5, 6, 7\}$. Job 3 is started at $t = 5$, since $\sum_{l=t}^{t+p_3-1} e_1 c'_l = \{9, 9, 9\}$ and the tie-breaking of earlier t is used.

Algorithm 5
The procedure of mutation 1.

1: Input a parent randomly selected.
2: Choose a job j randomly and record its current machine i.
3: Select another different machine k for job j by following Lines 5 ~ 10 of the initialization procedure in Algorithm 4.
4: Re-order the sequence of jobs assigned on machine k randomly.
5: Compute the stating time of every job in machine k by following Lines 13 ~ 23 of the initialization procedure in Algorithm 4.
6: Obtain one offspring.

Algorithm 6
The procedure of Mutation 2.

1: Input a parent randomly selected.
2: Select a machine i randomly on which there are more than one jobs assigned.
3: Choose two jobs with different processing times randomly and swap their processing positions on machine i.
4: Compute the stating time of every job by following the steps in Lines $12 \sim 24$ of the initialization procedure in Algorithm 4.
5: Obtain one offspring.

Finally, job 4 can only be started from $t = 8$. The schedule is shown as Schedule 1 in Fig. 2, with $C_{max} = 14$ and $TEC = 34$.

With Schedule 1, there are some blocks, for instance, $S_1 = \{4,1\}$, $S_2 = \{2,4,1\}$, $S_3 = \{3,2,4\}$. By using the local search strategy, S_1 or S_2 can be shifted to the right one time unit and the obtained schedules are Schedule 2 and Schedule 3, respectively, with $C_{max} = 15$ and $TEC = 33$ for both schedules.

The solution obtained by the constructive heuristic method is a good solution with the smaller TEC. However, when the makespan is limited by a certain value closed to f_1^l, there will be no enough idle time interval for some jobs. For example, in the example mentioned above, $f_1^l = 14$ and the makespan is limited by 15. If the energy price of period 1 is 5, job 3 will be assigned to time interval [2,5] and job 2 will be assigned to time interval [6,8]. There will not be enough time interval for job 4. Therefore, the drawback of the constructive heuristic method is that some solutions (f_1, f_2) could not be obtained when f_1 is closed to f_1^l.

4.3. NSGA-II

Due to the drawbacks of the augmented ε-constraint method and the constructive heuristic method, some other algorithms are needed to tackle the large-scale instances effectively and efficiently.

NSGA-II (Deb et al., 2002) is a well-known evolutionary algorithm and is widely applied for multi-objective optimization problems. Its elitism and diversity preservation mechanism with the non-dominated sorting and crowding distance could help obtain an approximate Pareto front with good optimality and diversity. The following subsections present the detailed implementation procedure for the problem.

4.3.1. Representation of chromosome

The representation of chromosome is a vector with length $2n + 2$, where n is the number of jobs. The first $1 \sim n$ genes are coded with a value from the set $\{1, 2, \ldots, m\}$, representing on which machine job j ($j \in \mathcal{N}$) is processed. Similarly, $n + 1 \sim 2n$ genes are coded with a value from the set $\mathcal{T}$, representing the starting time of job j. The last two genes represent two objective values, C_{max} and TEC, respectively.

Algorithm 7
The procedure of Mutation 3.

1: Input a parent randomly selected.
2: Select a machine i randomly on which there are more than one jobs assigned.
3: Sort all jobs on machine i according to their starting times and obtain the processing sequence.
4: **for** every pair of consecutive jobs **do**
5: Compute the idle time between every pair of consecutive jobs, which is denoted by $idle_{max}$.
6: Generate an integer it from discrete uniform distribution $[0, idle_{max}]$ randomly.
7: Shift the second job in every pair of consecutive jobs it time units to the left.
8: **end for**
9: Obtain one offspring.

4.3.2. Initialization

The initialization procedure aims to generate an initial population consists of n individuals. The constraints stated in the model in Appendix are used to ensure that every chromosome represents a feasible solution. The pseudo code in Algorithm 4 describes the detailed procedure of population initialization and chromosome generation.

4.3.3. Selection

We apply the standard binary tournament selection strategy for choosing parents.

4.3.4. Crossover

Trying to ensure the feasibility of offsprings, the crossover operator is designed with the idea of preserving parent's common job assignment as well as reassigning the rest of jobs randomly. More specifically, if the assignment of machine for the same job in both parent are consistent, then the assignment of this job is kept in offsprings.

4.3.5. Mutation

In order to evolved efficiently in cooperate with crossover, three mutation operators are designed.

(1) **Mutation1:** the main idea is to reassign a job to a new machine. The procedure is described in Algorithm 5.
(2) **Mutation 2:** it aims to improve individuals through exchanging the sequence of two jobs with different processing times on the same machine. The procedure is described in Algorithm 6.
(3) **Mutation 3:** the main idea is to decrease the idle time between consecutive jobs on a machine. The procedure is described in Algorithm 7.

Illustrative examples for mutation operators are given in Figs. 3–5.

4.4. Combination with the constructive heuristic

During the initialization, some individuals are generated by the constructive heuristic method and the local search strategy described in subsection 4.2, while other individuals are generated by following the initialization procedure described in Algorithm 4. Such an adjustment try to combine the advantages of the constructive heuristic method (generating high-quality individuals based on problem structure) and the evolutionary algorithm (searching for better individuals through crossover and mutation procedure).

5. Parameter tuning and computational results

This section presents the computational experiments to show the performance of the methods. The augmented ε-constraint method is coded with Python 3.6 and is run with Gurobi 7.5. The constructive heuristic method and the NSGA-II algorithm are coded in MATLAB R2016(b) on a PC with a 3.3 GHz Intel core i5-4590 processor and 8 GB RAM in Windows 7 system.

5.1. Performance metrics

For a multi-objective optimization, there are two goals: (1) convergence to the Pareto-optimal set and (2) maintenance of

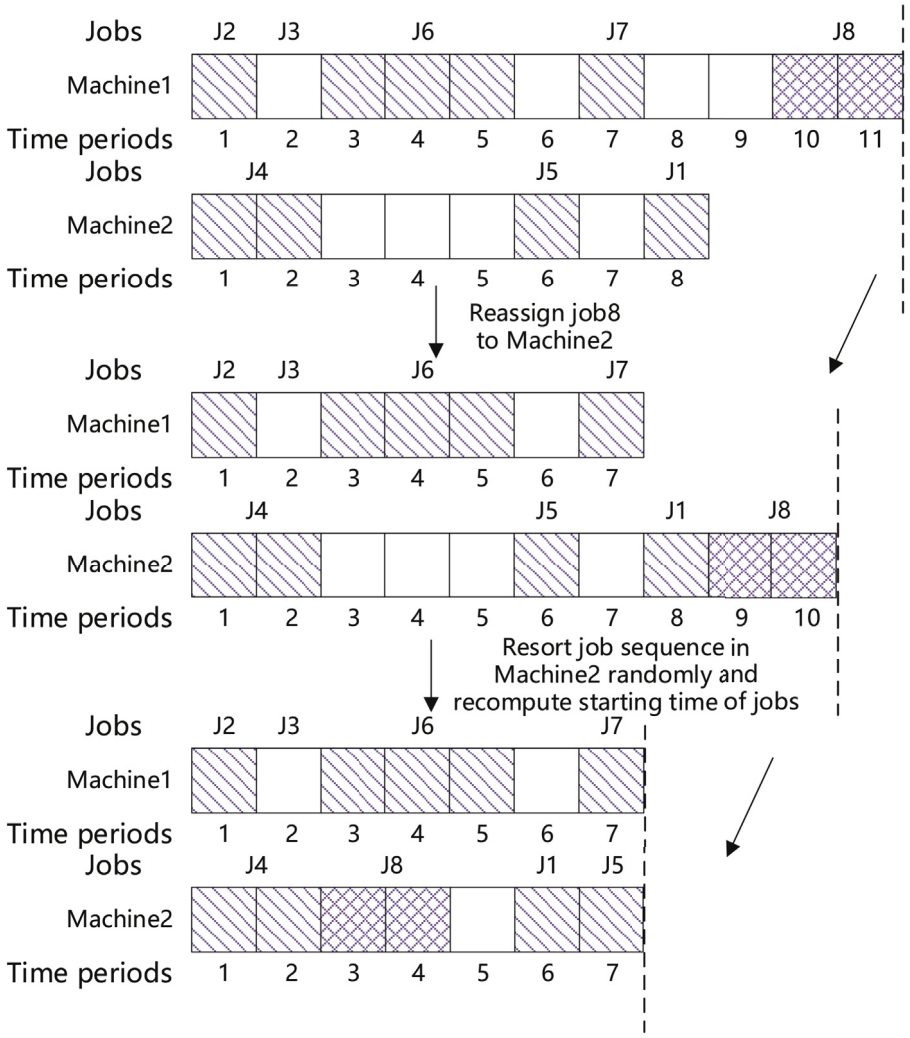

Fig. 3. An illustrative example of Mutation 1.

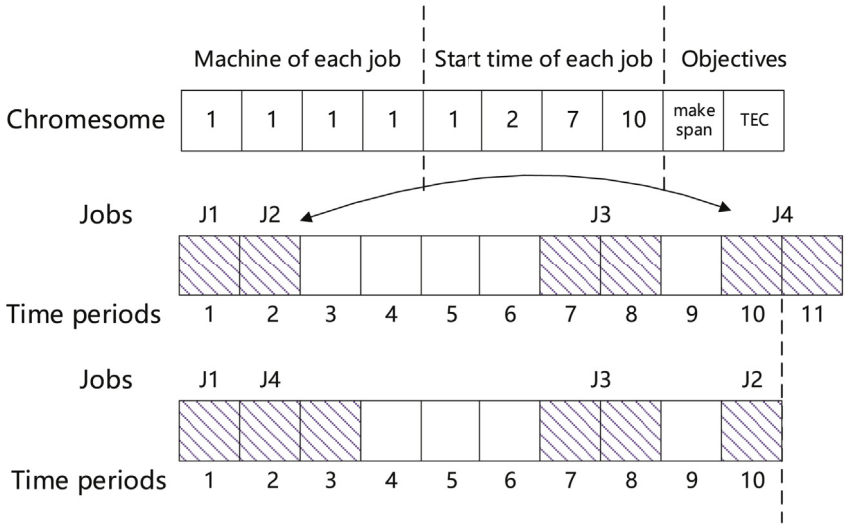

Fig. 4. An illustrative example of Mutation 2.

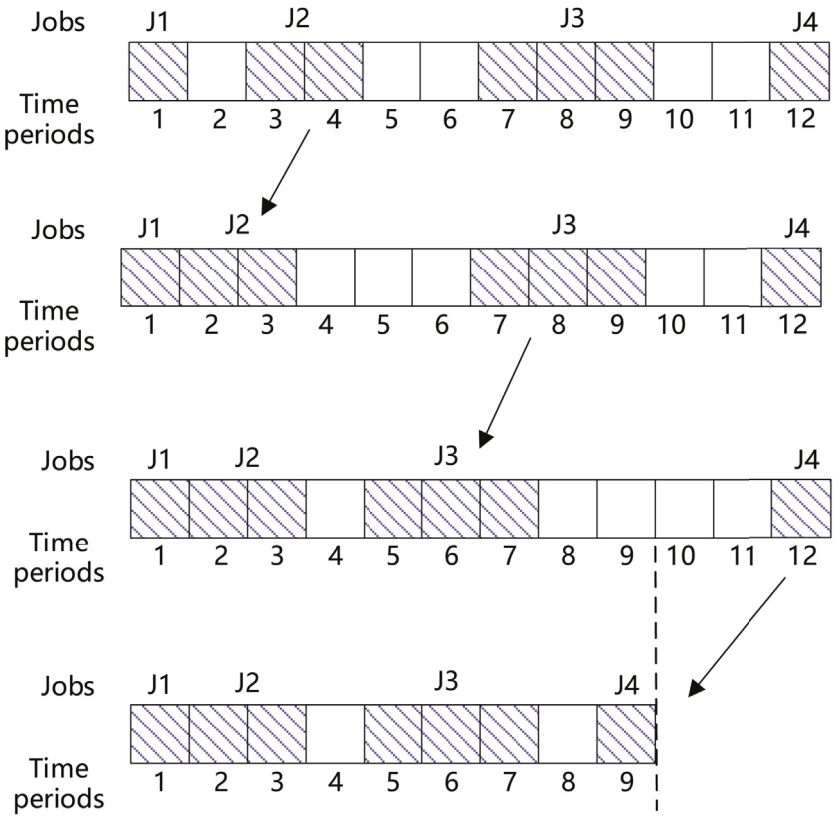

Fig. 5. An illustrative example of Mutation 3.

Table 1
Instance configurations for parameter tuning.

Instance No.	Size	$m \times n$	$p_j \sim U[1, p_{max}]$	T	$e_i \sim U[1, e_{max}]$	$c_k \sim U[1, c_{max}]$
1	Small -scale	2×5	[1, 3]	10	[1, 3]	[1, 3]
2		2×5	[1, 5]	10	[1, 3]	[1, 5]
3		3×5	[1, 3]	10	[1, 3]	[1, 3]
4		3×5	[1, 5]	10	[1, 3]	[1, 5]
5		4×5	[1, 5]	10	[1, 3]	[1, 5]
6		4×5	[1, 5]	10	[1, 5]	[1, 5]
7		4×8	[1, 3]	10	[1, 3]	[1, 3]
8		4×8	[1, 3]	10	[1, 3]	[1, 5]
9		4×10	[1, 3]	20	[1, 5]	[1, 3]
10		4×10	[1, 5]	20	[1, 5]	[1, 5]
11		5×10	[1, 3]	20	[1, 3]	[1, 3]
12		5×10	[1, 5]	20	[1, 5]	[1, 5]
13		5×12	[1, 5]	20	[1, 3]	[1, 5]
14		5×12	[1, 5]	30	[1, 5]	[1, 5]
15		5×15	[1, 5]	30	[1, 7]	[1, 7]
16	Medium- and large -scale	5×30	[1, 3]	30	[1, 5]	[1, 3]
17		5×35	[1, 3]	30	[1, 3]	[1, 5]
18		5×40	[1, 5]	30	[1, 5]	[1, 5]
19		5×45	[1, 5]	50	[1, 5]	[1, 5]
20		8×50	[1, 3]	50	[1, 5]	[1, 5]
21		8×60	[1, 3]	80	[1, 7]	[1, 3]
22		8×70	[1, 5]	80	[1, 3]	[1, 7]
23		8×80	[1, 5]	80	[1, 7]	[1, 7]
24		8×85	[1, 5]	100	[1, 7]	[1, 5]
25		10×100	[1, 5]	300	[1, 7]	[1, 7]
26		10×150	[1, 5]	300	[1, 7]	[1, 7]
27		20×180	[1, 5]	300	[1, 7]	[1, 7]
28		25×200	[1, 3]	300	[1, 7]	[1, 5]
29		30×230	[1, 5]	400	[1, 5]	[1, 7]
30		35×250	[1, 5]	400	[1, 7]	[1, 7]

Table 2
Factors and candidate levels for the NSGA-II algorithm.

Factors	Levels	
	Small-scale instances	Medium- and large-scale instances
PopSize	{30, 50, 80, 100}	{200, 300, 450, 500}
MaxGen	{50, 100, 150, 200}	{100, 150, 200, 300}
P_c	{0.2, 0.3, 0.4, 0.45}	{0.3, 0.35, 0.4, 0.6}
P_{m1}	{0.1, 0.2, 0.25, 0.3}	{0.01, 0.05, 0.15, 0.2}
P_{m2}	{0.05, 0.1, 0.15, 0.2}	{0.01, 0.05, 0.1, 0.15}

diversity in obtained non-dominated solutions (Deb et al., 2002). Appropriate performance metrics are needed to measure the achievements of these two goals. In this paper, the following four performance metrics are used.

The first one is the number of non-dominated solutions N_d.

The second and third ones are diversity introduced in Deb et al. (2000) and Deb et al. (2002). They are defined as follows:

$$\Delta_{D1} = \sum_{i=1}^{N-1} \frac{\left|d_i - \overline{d}\right|}{N-1} \tag{1}$$

where N represents the number of non-dominated solutions, d_i is the Euclidean distance between two consecutive solutions in the obtained non-dominated set of solutions, $\overline{d}$ is the average of all distances d_i, $i \in \{1, \ldots, N-1\}$, since there are $N-1$ consecutive distances for N solutions on the non-dominated front.

$$\Delta_{D2} = \frac{d_f + d_l + \sum_{i=1}^{N-1}\left|d_i - \overline{d}\right|}{d_f + d_l + (N-1)\overline{d}} \tag{2}$$

where d_f and d_l are Euclidean distances between the extreme solutions and the boundary solutions of the obtained non-dominated set.

The smaller values of Δ_{D1} and Δ_{D2} are, the better the diversity of the efficient set is.

The fourth one is the D_R measure (Ishibuchi et al., 2003), which can be used to evaluate the distance of a set of non-dominated solutions to a reference front (i.e., the optimal Pareto front or a near optimal Pareto front). Let S^* be the reference solution set. If the Pareto front is not known, the two (or more if more than two algorithms are used to generate different fronts) fronts are combined and all the non-dominated solutions are selected to form the set S^*. The D_R measure is considered to be the most important one among the four metrics. D_R of a front A is given by

$$D_R(A) = \frac{1}{|S^*|}\sum_{y \in S^*} min\{d_{xy} : x \in A\} \tag{3}$$

where d_{xy} is the Euclidean distance between a solution x in the front A and a solution y in the reference front. For bi-objective optimization, d_{xy} can be calculated as follows:

$$d_{xy} = \sqrt{\left(f_1^*(x) - f_1^*(y)\right)^2 + \left(f_2^*(x) - f_2^*(y)\right)^2} \tag{4}$$

where $f_1^*(\cdot)$ and $f_2^*(\cdot)$ represent the objective values of the makespan and total energy consumption, respectively, which are normalized by using the solutions in the reference front S^*.

$$f_i^*(\cdot) = \frac{f_i(\cdot) - f_i^{min}(S^*)}{f_i^{max}(S^*) - f_i^{min}(S^*)} \tag{5}$$

where $f_i(\cdot)$ represents the ith objective value of the solution $(\cdot)$, $i = \{1, 2\}$. $f_i^{max}(S^*)$ and $f_i^{min}(S^*)$ represent the maximum and minimum values of ith objective in the reference front S^*, respectively.

The smaller the value of D_R is, the better the convergence of the efficient set is.

5.2. Parameter tuning

Parameters affect the performance of the NSGA-II algorithm largely. Hence, it's necessary to tune parameters with appropriate methods.

Taguchi method is one of strong stochastic techniques among design of experiments (DOE) methods. It aims to improve quality and reduce cost in the design of process or production. The main idea of this method is to combine different factors and their degrees by vertical arrays and factorial designs. Then the experiments and analysis are conducted with fewer number of experiments (Shahidi-Zadeh et al., 2017).

5.2.1. Generate sample problems for parameter tuning

The key point for parameters tuning is to test algorithm's performance with the same problems under different parameters. Hence, a set of sample problem instances for experimental testing are firstly generated. For this problem, the following six parameters which affect the problem are considered: (1) number of machines: m; (2) number of jobs: n; (3) the maximum processing time for p_j: p_{max}; (4) time horizon: $T \geq \max(\sum_{j=1}^{n} p_j/m,\ \max(p_j))$; (5) the maximum energy consumption rate for each machine e_i: e_{max}, and (6) the maximum energy price for each time interval c_k: c_{max}. The T_k is taken from $\{2, 3, 5\}$. The values are set as in Table 1. In the table, U represents the discrete uniform distribution.

5.2.2. Determine factors and candidate levels

For the NSGA-II algorithm, the following 5 factors should be considered:

(1) Size of population: *PopSize*;
(2) Maximize generation: *MaxGen*;
(3) Crossover probability: P_c;
(4) Probability of mutation 1: P_{m1};
(5) Probability of mutation 2: P_{m2};

It is noted that the probability of mutation 3, $P_{m3} = 1 - P_c - P_{m1} - P_{m2}$. The candidate levels of above parameters are designed

Table 3
Experimental configuration of Taguchi vertical array $L16(4^5)$.

Experiment No.	Combination of factors' levels				
	PopSize	*MaxGen*	P_c	P_{m1}	P_{m2}
1	1	1	1	1	1
2	1	2	2	2	2
3	1	3	3	3	3
4	1	4	4	4	4
5	2	1	2	3	4
6	2	2	1	4	3
7	2	3	4	1	2
8	2	4	3	1	2
9	3	1	3	4	2
10	3	2	4	3	1
11	3	3	1	2	4
12	3	4	2	1	3
13	4	1	4	2	3
14	4	2	3	1	4
15	4	3	2	4	1
16	4	4	1	3	2

Table 4
The experimental testing results for small-scale instances.

$\overline{D_R}$	Instance No.															$\overline{\overline{D_R}}$
Exp. No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
1	0	0	0	0	0	0	0	0.050	0.040	0.076	0.014	0.023	0.031	0.020	0.038	0.019
2	0	0	0	0	0	0	0	0.016	0.036	0.071	0.014	0.035	0.031	0.020	0.037	0.017
3	0	0	0	0	0	0	0	0.022	0.036	0.071	0.014	0.016	0.030	0.019	0.037	0.016
4	0	0	0	0	0	0	0	0.022	0.030	0.076	0.014	0.033	0.030	0.019	0.038	0.017
5	0	0	0	0	0	0	0	0.080	0.036	0.071	0.014	0.033	0.031	0.019	0.038	0.021
6	0	0	0	0	0	0	0	0.080	0.040	0.071	0.014	0.031	0.031	0.019	0.038	0.022
7	0	0	0	0	0	0	0	0.055	0.032	0.071	0.014	0.021	0.028	0.018	0.037	0.018
8	0	0	0	0	0	0	0	0.080	0.032	0.071	0.014	0.031	0.030	0.020	0.036	0.021
9	0	0	0	0	0	0	0	0.050	0.036	0.076	0.014	0.034	0.030	0.020	0.038	0.020
10	0	0	0	0	0	0	0	0.050	0.040	0.071	0.014	0.028	0.025	0.020	0.038	0.019
11	0	0	0	0	0	0	0	0.080	0.030	0.068	0.014	0.021	0.030	0.019	0.036	0.020
12	0	0	0	0	0	0	0	0.050	0.032	0.068	0.014	0.005	0.030	0.019	0.034	0.017
13	0	0	0	0	0	0	0	0.055	0.028	0.069	0.014	0.038	0.030	0.020	0.038	0.019
14	0	0	0	0	0	0	0	0.050	0.030	0.071	0.014	0.018	0.030	0.019	0.037	0.018
15	0	0	0	0	0	0	0	0.050	0.032	0.060	0.014	0.018	0.030	0.019	0.037	0.017
16	0	0	0	0	0	0	0	0.016	0.025	0.062	0.014	0.025	0.029	0.019	0.037	0.015

Table 5
The experimental results for medium- and large-scale instances.

$\overline{\Delta_{D1}}$	Instance No.															$\overline{\overline{\Delta_{D1}}}$
Exp. No.	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	
1	4.6	9.5	15.0	7.9	5.3	23.7	16.1	16.0	13.4	13.5	23.1	20.6	11.4	27.2	18.4	15.0
2	4.6	9.5	15.0	7.9	5.3	18.2	16.1	16.0	13.8	13.5	22.8	20.6	11.4	27.2	18.4	14.7
3	4.6	9.5	15.0	7.9	5.3	17.0	15.9	16.0	13.5	13.5	22.6	20.9	11.4	27.2	18.4	14.6
4	4.6	9.5	15.0	7.9	5.3	18.2	16.0	16.0	14.1	13.5	22.8	20.9	11.4	27.2	18.4	14.7
5	4.6	9.5	15.0	7.9	5.3	18.2	16.0	16.0	13.6	13.5	22.9	21.0	11.4	27.2	18.4	14.7
6	4.6	10.7	15.0	7.9	5.3	25.8	16.0	16.0	13.8	13.5	22.9	20.3	11.4	27.2	18.4	15.3
7	4.6	9.5	15.0	7.9	5.3	18.2	16.1	16.0	13.6	13.5	23.0	20.6	11.4	27.2	18.4	14.7
8	4.6	12.6	15.0	7.9	5.3	18.2	16.1	16.0	13.9	13.5	22.8	20.6	11.4	27.1	18.4	14.9
9	4.6	9.5	15.0	7.9	5.3	18.2	16.1	16.0	13.5	13.5	22.7	20.6	11.4	27.2	18.4	14.7
10	4.6	9.5	15.0	7.9	5.3	18.2	16.0	16.0	13.4	13.5	22.9	20.6	11.4	27.2	18.6	14.7
11	4.6	9.7	15.0	7.9	5.3	25.8	15.9	16.0	13.6	13.5	23.0	20.7	11.5	27.2	18.6	15.2
12	4.6	9.5	15.0	7.9	5.3	25.8	15.9	16.0	13.6	13.5	23.0	21.1	11.7	27.1	18.6	15.2
13	4.6	9.5	15.0	7.9	5.3	23.1	16.1	16.0	13.4	13.5	23.1	20.6	11.4	27.1	18.6	15.0
14	4.6	9.5	15.0	7.9	5.3	17.0	16.2	16.0	14.5	13.5	23.2	20.8	11.4	27.2	18.6	14.7
15	4.6	9.5	15.0	7.9	5.3	32.0	16.1	16.0	13.8	13.5	23.0	20.7	11.4	27.2	18.6	15.6
16	4.6	9.6	15.0	7.9	5.3	18.2	16.0	16.0	13.9	13.5	22.9	20.6	11.4	28.7	18.6	14.8

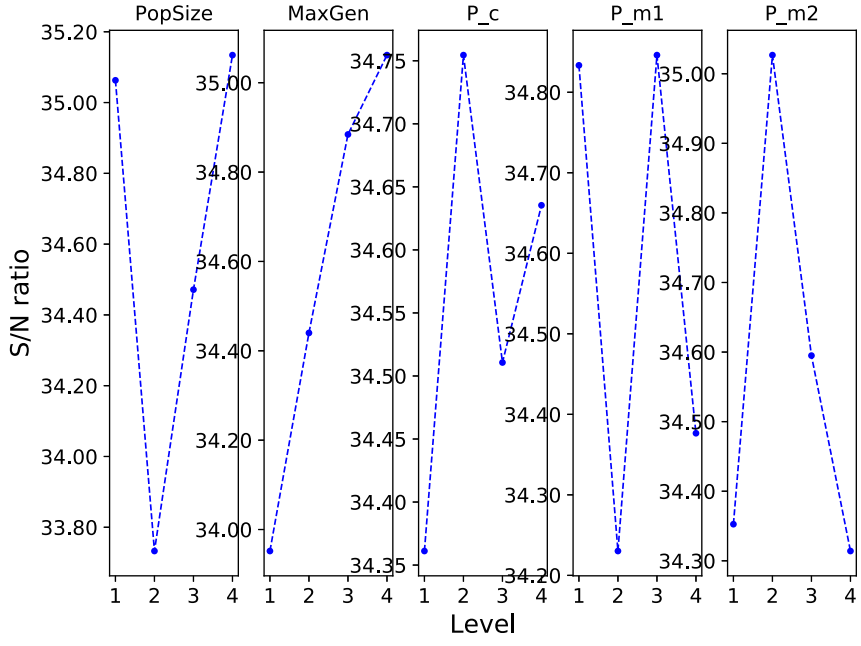

Fig. 6. Main effects plot for S/N ratios for the NSGA-II algorithm: small-scale instances.

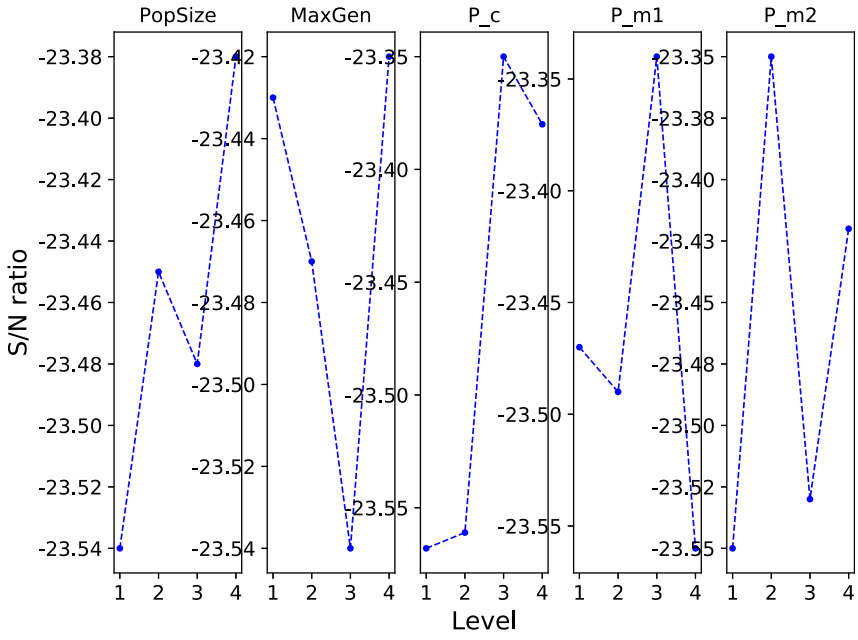

Fig. 7. Main effects plot for S/N ratios for the NSGA-II algorithm: medium- and large-scale instances.

in Table 2.

Table 6
Parameter tuning results for the NSGA-II algorithm.

		Parameters					
		PopSize	*MaxGen*	P_c	P_{m1}	P_{m2}	P_{m3}
Small-scale	Opt. Level	4	4	2	3	2	–
	Opt. Value	100	200	0.3	0.25	0.1	0.35
Medium- and large-scale	Opt. Level	4	4	3	3	2	–
	Opt. Value	500	300	0.4	0.15	0.05	0.4

Table 7
Comparison results for small-scale instances.

No.	AUGMECON				CH					NSGA-II				
	$t(s)$	*Nd*	Δ_{D1}	Δ_{D2}	$t(s)$	*Nd*	Δ_{D1}	Δ_{D2}	D_R	$t(s)$	*Nd*	Δ_{D1}	Δ_{D2}	D_R
1	2.24	11	3.32	0.55	0.10	13	3.52	0.64	0.05	16.50	13	4.83	0.73	0.04
2	0.68	6	27.09	0.71	0.08	13	13.99	0.87	0.06	15.84	14	14.44	0.87	0.06
3	1.91	9	12.80	0.84	0.12	8	6.13	0.67	0.07	17.08	9	9.61	0.63	0.03
4	2.34	11	22.87	0.83	0.13	11	16.16	0.88	0.04	18.50	12	21.17	0.84	0.01
5	1.11	6	5.73	0.45	0.10	5	1.50	0.34	0.12	1.11	6	5.73	0.45	0
6	2.11	8	27.86	0.59	0.13	8	24.04	0.75	0.05	2.11	8	27.86	0.59	0
7	3.86	16	7.36	0.65	0.19	29	3.18	0.55	0.04	20.07	32	4.92	0.77	0.04
8	8.84	22	8.45	0.68	0.26	39	3.72	0.74	0.05	19.94	39	4.94	0.73	0.02
9	6.04	12	3.81	0.36	0.14	15	9.92	0.75	0.07	19.26	13	12.24	0.79	0.05
10	3.66	11	16.43	0.80	0.19	16	8.76	0.82	0.05	19.15	17	13.69	0.95	0.05
11	3.54	10	24.20	1.03	0.21	13	8.03	0.81	0.08	20.95	15	14.14	0.94	0.04
12	7.80	13	26.32	0.83	0.29	16	16.57	0.87	0.04	21.80	17	23.03	0.93	0.03
13	20.76	14	10.91	0.63	0.18	26	4.79	0.67	0.05	19.44	27	6.55	0.74	0.05
14	28.57	23	16.01	0.92	0.27	33	9.32	0.78	0.02	21.80	34	9.98	0.81	0.02
15	49.26	15	29.03	0.78	0.19	23	21.42	0.90	0.04	20.66	24	19.29	0.82	0.03
16	19.53	20	11.65	0.73	0.38	26	6.36	0.62	0.04	22.03	27	9.84	0.76	0.03
17	14.57	16	25.08	0.97	0.31	25	12.28	0.94	0.04	22.22	26	19.47	1.08	0.04
18	476.26	26	10.90	0.82	0.50	29	9.85	0.80	0.03	23.68	31	16.36	1.01	0.02
19	155.71	19	23.25	1.07	0.22	29	12.61	0.92	0.04	21.08	30	12.18	0.89	0.04
20	386.80	26	15.31	0.91	0.35	34	7.75	0.61	0.02	22.52	36	7.90	0.65	0.02
21	9051.64	20	24.30	0.89	0.35	34	15.53	0.93	0.04	23.71	35	16.23	0.93	0.03
22	77.78	23	15.93	0.94	0.60	41	7.14	0.77	0.03	25.97	44	7.44	0.81	0.03
23	30.78	15	11.99	0.64	0.32	20	3.71	0.51	0.05	24.01	21	6.86	0.53	0.04
24	11,899.71	33	16.05	0.91	0.79	42	11.93	0.84	0.02	27.71	43	12.10	0.84	0.02
25	551.14	13	40.58	0.88	0.19	18	12.81	0.41	0.07	22.57	22	15.14	0.57	0.06
26	18,357.87	23	13.16	0.77	0.47	34	7.54	0.66	0.05	24.70	38	7.68	0.70	0.03
27	26.69	16	31.06	1.08	0.27	21	15.83	0.71	0.04	23.97	23	16.34	0.75	0.04
28	7359.89	31	22.01	0.96	0.78	46	12.66	0.90	0.02	27.59	48	15.21	0.96	0.02
29	4247.00	24	31.71	1.04	0.54	33	23.39	1.01	0.03	27.39	34	24.38	1.00	0.03
30	165.14	24	28.47	1.02	0.76	31	18.36	0.92	0.02	28.18	32	18.85	0.91	0.02

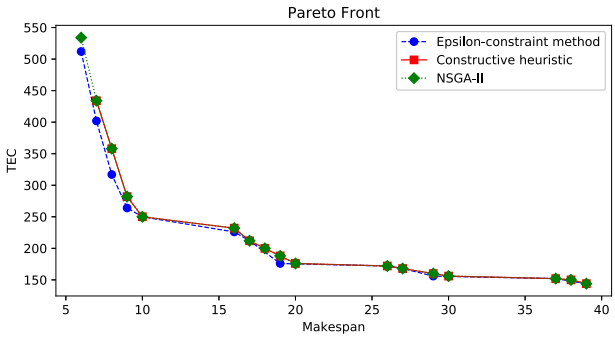

Fig. 8. Illustration of Pareto fronts obtained by three different algorithms for small-scale problem instance #12.

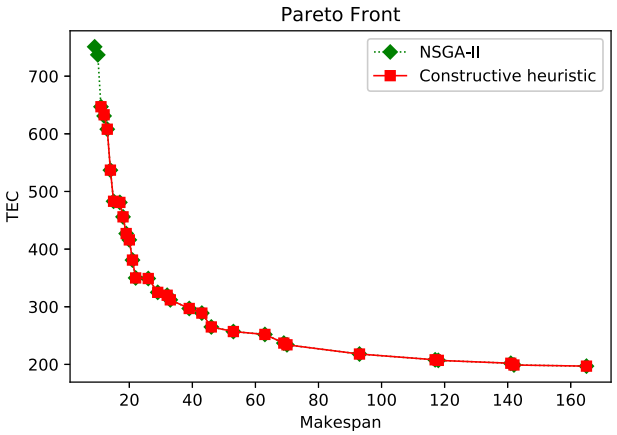

Fig. 9. Illustration of Pareto fronts obtained by different algorithms for problem instance #57.

5.2.3. Design experiments according to Taguchi method

According to Taguchi method, the number of needed degree of freedom, $Degree_f$, should be calculated as $Degree_f = n_f*(n_l - 1) + 1$, where n_f is the number of factors, n_l is the number of levels. For the NSGA-II algorithm, there are $n_f = 5$ factors with $n_l = 4$ levels to be considered. Hence $Degree_f = 16$, which means that Taguchi vertical array $L16(4^5)$ will be selected. The detailed experimental configuration of $L16(4^5)$ is shown in Table 3, in which columns 2∼6 are the levels selected for parameters.

5.2.4. The experimental testing

We perform these 16 experiments for small-scale instances, and medium- and large-scale instances, separately. The testing results for small-scale instances are shown in Table 4. In the table, "Exp. No." is the experiment number. For each instance, every experiment is performed 10 times independently. The ε-constraint method is applied to obtain exact Pareto front, so D_R can be calculated with the exact Pareto front as the reference front. In the table, $\overline{D_R}$ is the average D_R of 10 independent experiments for each instance. $\overline{\overline{D_R}}$ listed in the last column are averages value of $\overline{D_R}$ for fifteen small-scale problem instances.

The results for medium- and large-scale instances are listed in Table 5. Since it is difficult to solve medium- and large-scale instances to optimality by the ε-constraint method in a reasonable computation time, only the NSGA-II algorithm is applied to obtain approximate Pareto front. Hence, only Δ_{D1} is used.

Table 8
Comparison results for medium- and large-scale instances.

No.	*CPU*(*s*)		*Nd*		Δ_{D1}		Δ_{D2}		$D_R \times 10^{-3}$	
	CH	NSGA-II	CH	NSGA-II	CH	NSGA-II	CH	NSGA-II	CH	NSGA-II
31	7.41	85.96	47	49.1	5.27	7.65	0.87	0.95	7.74	0.84
32	11.73	120.75	52	51.7	5.51	6.84	0.87	0.86	6.13	0.69
33	30.66	247.90	55	57.0	11.58	14.63	0.82	0.85	4.40	0
34	27.05	235.14	59	61.0	20.66	22.35	0.79	0.78	2.21	0
35	25.80	265.69	46	48.0	16.47	20.47	0.72	0.72	4.82	0
36	6.35	88.44	21	23.1	3.46	7.64	0.62	0.67	27.50	0
37	15.26	144.23	31	33.9	14.87	19.39	0.82	0.87	12.21	0.20
38	37.09	229.96	61	61.2	16.54	24.11	0.99	1.08	8.05	0.89
39	51.32	293.35	55	57.0	15.65	20.65	0.84	0.87	5.57	0
40	60.27	336.60	59	61.0	30.74	33.03	0.93	0.93	2.51	0
41	6.97	92.06	13	15.3	8.20	19.94	0.77	0.77	51.21	0
42	19.19	149.00	26	28.0	6.87	18.78	0.79	0.96	20.91	0
43	35.61	224.09	48	50.0	11.82	23.61	0.88	1.00	12.58	0
44	62.18	317.30	61	63.0	22.08	28.61	1.00	1.07	5.05	0
45	86.87	392.74	62	65.0	20.61	22.81	0.93	0.93	3.26	0
46	8.38	85.07	26	27.5	4.31	6.20	0.67	0.64	17.63	0.89
47	32.06	164.64	66	68.8	5.95	7.43	0.72	0.75	4.06	0.01
48	80.12	275.19	81	84.0	7.11	8.31	0.73	0.76	2.83	0
49	220.75	587.32	159	156.9	6.96	7.41	0.83	0.83	0.66	0.17
50	202.55	516.45	139	142.0	10.13	10.64	0.80	0.82	0.53	0.03
51	11.14	99.46	12	14.0	8.07	11.36	0.87	0.82	42.35	0
52	39.19	195.41	19	21.0	5.68	13.16	0.68	0.68	39.10	0
53	125.31	367.04	54	56.0	10.51	15.57	0.74	0.79	6.31	0
54	232.41	565.33	50	52.0	13.00	15.04	0.68	0.67	3.69	0
55	346.76	740.10	86	87.0	10.78	15.11	0.76	0.83	3.41	0.13
56	13.56	114.47	12	14.0	4.04	9.56	0.63	0.68	40.76	0
57	43.72	206.24	28	30.0	11.93	13.76	0.66	0.62	11.73	0
58	153.29	420.64	48	50.0	9.68	13.11	0.75	0.77	5.76	0
59	185.23	513.56	33	37.0	13.17	17.12	0.72	0.74	11.47	0
60	426.57	871.75	75	80.2	15.37	17.09	0.75	0.78	3.41	0

Table 9
p_values of the hypothesis tests ($\alpha = 0.05$).

Null hypothesis (H_0)	Instance No.	Performance metrics				
		CPU(*s*)	*Nd*	Δ_{D1}	Δ_{D2}	D_R
CH performs better than NSGA-II	31	1.0	0.0	1.0	1.0	0.0
	32	1.0	0.9	1.0	0.0	0.0
	33	1.0	0.0	1.0	1.0	0.0
	34	1.0	0.0	1.0	0.0	0.0
	35	1.0	0.0	1.0	1.0	0.0
	36	1.0	0.0	1.0	1.0	0.0
	37	1.0	0.0	1.0	1.0	0.0
	38	1.0	0.8	1.0	1.0	0.0
	39	1.0	0.0	1.0	1.0	0.0
	40	1.0	0.0	1.0	0.0	0.0
	41	1.0	0.0	1.0	0.2	0.0
	42	1.0	0.0	1.0	1.0	0.0
	43	1.0	0.0	1.0	1.0	0.0
	44	1.0	0.0	1.0	1.0	0.0
	45	1.0	0.0	1.0	1.0	0.0
	46	1.0	0.0	1.0	0.0	0.0
	47	1.0	0.0	1.0	1.0	0.0
	48	1.0	0.0	1.0	1.0	0.0
	49	1.0	1.0	1.0	0.7	0.0
	50	1.0	0.0	1.0	1.0	0.0
	51	1.0	0.0	1.0	0.0	0.0
	52	1.0	0.0	1.0	1.0	0.0
	53	1.0	0.0	1.0	1.0	0.0
	54	1.0	0.0	1.0	0.0	0.0
	55	1.0	0.0	1.0	1.0	0.0
	56	1.0	0.0	1.0	1.0	0.0
	57	1.0	0.0	1.0	0.0	0.0
	58	1.0	0.0	1.0	1.0	0.0
	59	1.0	0.0	1.0	1.0	0.0
	60	1.0	0.0	1.0	1.0	0.0
Percentage of rejected H_0		0.00%	90.00%	0.00%	23.33%	100.00%

5.2.5. Analysis of the experimental results

Then, the S/N ratio shown in Eq. (6) is to be maximized. In the equation, n is the number of levels, y_i is the performance metric for the ith level of certain factor.

$$S/N = -10log\left(1/n\sum_{i=1}^{n} y_i^2\right) \tag{6}$$

With the results in Tables 4 and 5, the main effects plot for S/N ratio values are shown in Figs. 6 and 7, respectively.

According to the Taguchi method, for every factor, the level with the maximum S/N ratio is the best one. So we can find the optimal level for each factor from Figs. 6 and 7, and the most appropriate parameters for the NSGA-II algorithm are suggested in Table 6, in which "Opt. Level" and "Opt. Value" are the chosen level and the corresponding value of each parameter.

Table 10
The power of 17 machines (unit: kW).

Machine No.	Power	Machine No.	Power
1	48	10	52
2	48	11	40
3	48	12	68
4	60	13	68
5	48	14	50
6	48	15	40
7	64	16	40
8	52	17	48
9	80	/	/

5.3. Computational results

5.3.1. Data generation

The parameters related to the problem include the number of jobs n, the number of machines m, the time horizon T, energy consumption rate of machines e_i, energy price of time intervals c_k, processing times of jobs p_j and time intervals T_k. The size of problem is denoted as $[m, n, T]$.

We consider the following combinations of above parameters, for small-scale instances: $[m, n, T] : m \in \{3, 5, 7\}, n \in \{6, 10, 15, 20, 25\}, T \in \{50, 80\}$ and for medium- and large-scale instances: $[m, n, T] : m \in \{8, 16, 25\}$, $n \in \{30, 60, 100, 150, 200\}$, $T \in \{100, 300\}$, which results in $3 \times 5 \times 2 = 30$ small-scale instances and 30 medium- and large-scale instances. p_j is randomly generated in $U[1, 4]$. c_k and e_i are randomly generated in $\{1, 2, 3, 4\}$ and $\{1, 2, 3\}$, respectively. T_k is taken from $\{2, 3, 5\}$. These sixty problems are numbered from 1 to 60. All the data of instances and related code can be downloaded via the following link: pan.baidu.com/s/1u4Q4PDzJmqa8m_5dh8uocw.

5.3.2. Comparison results for small-scale instances

The performance of the augmented ε-constraint method (AUGMECON), the constructive heuristic method (CH) and the NSGA-II algorithm (NSGA-II) are compared. The results are shown in Table 7. In the table, there are five criteria including the four performance metrics introduced in subsection 5.1. "t(s)" is the computation time in CPU seconds.

The results show that the NSGA-II algorithm performs relatively better than the CH in terms of D_R. Their performance in terms of *Nd*, Δ_{D1} and Δ_{D2} are close for most instances. The ε-constraint method can obtain the optimal Pareto front for all these thirty instances.

Table 11
The processing time of 170 moulds (unit: h).

Job No.	p_j	Job No.	p_j	Job No.	p_j	Job No.	p_j	Job No.	p_j	Job No.	p_j
1	45	31	12	61	18	91	17	121	4	151	4
2	9	32	12	62	33	92	7	122	6	152	16
3	31	33	25	63	29	93	3	123	11	153	18
4	13	34	42	64	5	94	28	124	12	154	1
5	12	35	86	65	8	95	10	125	10	155	39
6	35	36	17	66	48	96	12	126	3	156	4
7	36	37	17	67	8	97	13	127	6	157	9
8	14	38	19	68	19	98	12	128	5	158	2
9	18	39	19	69	26	99	5	129	1	159	29
10	37	40	20	70	47	100	4	130	4	160	1
11	43	41	18	71	94	101	23	131	4	161	1
12	18	42	70	72	18	102	11	132	8	162	2
13	59	43	19	73	15	103	4	133	2	163	2
14	32	44	28	74	39	104	15	134	2	164	20
15	18	45	37	75	5	105	41	135	6	165	44
16	74	46	18	76	13	106	2	136	5	166	4
17	54	47	93	77	15	107	5	137	3	167	3
18	47	48	47	78	43	108	10	138	3	168	9
19	79	49	43	79	32	109	12	139	6	169	7
20	33	50	91	80	26	110	2	140	18	170	27
21	16	51	45	81	21	111	12	141	2	/	/
22	6	52	15	82	47	112	13	142	10	/	/
23	5	53	19	83	86	113	9	143	2	/	/
24	13	54	51	84	38	114	20	144	3	/	/
25	32	55	13	85	47	115	11	145	3	/	/
26	15	56	13	86	5	116	6	146	12	/	/
27	6	57	34	87	6	117	6	147	3	/	/
28	4	58	15	88	19	118	13	148	7	/	/
29	15	59	3	89	20	119	10	149	5	/	/
30	45	60	11	90	14	120	2	150	10	/	/

However, as expected, the computation time increases as the problem scale increases. For instance, for problem #26 with size $[5, 25, 50]$, it takes the ε-constraint method more than 18,000 s to obtain the optimal Pareto front. Therefore, for medium- and large-scale instances, only the constructive heuristic method and the NSGA-II algorithm are applied.

In addition, as an example, Fig. 8 illustrates the Pareto fronts achieved by these three different methods for a sample small-scale problem instance #12.

5.3.3. Comparison results for medium- and large-scale instances

The constructive heuristic method and the NSGA-II algorithm are used to solve the medium- and large-scale instances. For each instance, the NSGA-II algorithm run 10 times by considering the randomness, and the average values of performance metrics are recorded. The fronts obtained by two algorithms mentioned above are combined and all the non-dominated solutions are used to form the set S^* to compute D_R. The results are shown in Table 8. Fig. 9 illustrates two sample Pareto fronts achieved by these two methods for problem instance #57.

Statistical hypothesis tests are conducted at the level of significance $\alpha = 0.05$. The null hypothesis H_0 is that the constructive heuristic method performs better than the NSGA-II algorithm. The results are shown in Table 9. The results show that the NSGA-II algorithm performs better than the CH in terms of Nd and D_R, which means that the NSGA-II algorithm can obtain more non-dominated solutions with better convergency. On the contrary, the CH method is much faster and can generate approximate Pareto fronts with better diversity, since it performs better than the NSGA-II algorithm in terms of $CPU(s)$, Δ_{D1} and Δ_{D2}.

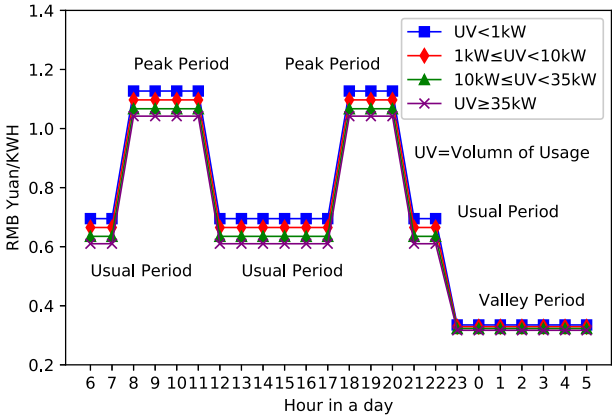

Fig. 10. Electricity market price for industry during the non-summer time in Shanghai, China. (Source: Shanghai Electricity Price, 2017; www.sh.sgcc.com.cn/. Accessed on 5 May 2018.)

5.4. A case study

The following case is originated from a real-world scheduling problem in an automobile stamping die manufacturing shop floor. Its data demonstrates some common characteristics of the automobile stamping die manufacturing. However, we have pre-processed the data for some commercial confidentiality reasons.

5.4.1. Data of the case

In this shop floor, there are 17 identical parallel machines with

Table 12
The performance metrics of proposed algorithms for solving the case.

	Nd	Δ_{D1}	Δ_{D2}	$D_R \times 10^{-3}$
CH	83	171.7787	0.6911	3.90
NSGA-II	85	185.5820	0.6714	0.119

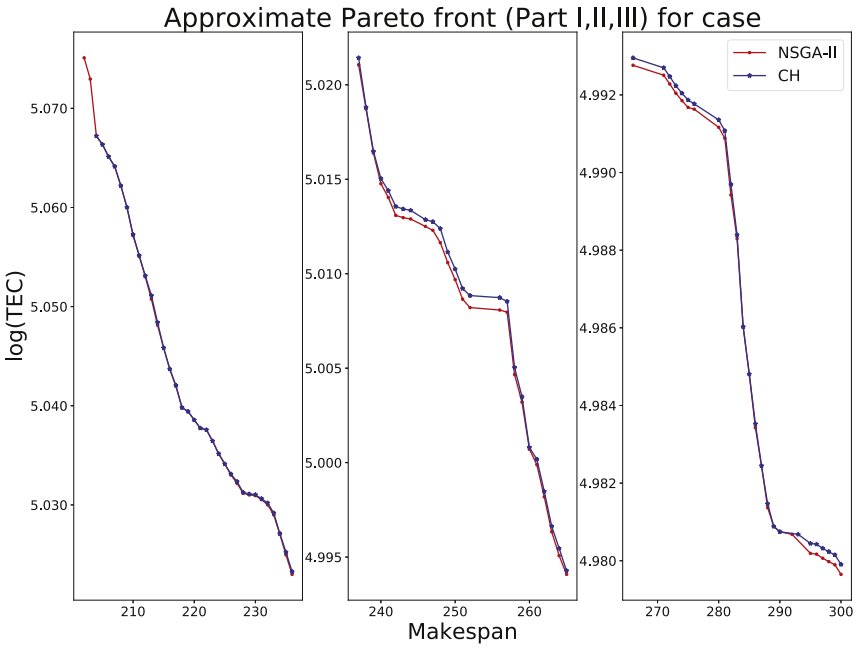

Fig. 11. Illustration of Pareto fronts obtained by two algorithms for the case.

same functions but with different energy consumption rates. Since some machines are purchased during the later stage of capacity expansion so they are employed green technology, while others purchased earlier are employed regular technology. All of these 17 machines are fully flexible and can process each job. The powers of machines are listed in Table 10. In the table, "/" represents that there are no more machines.

The automobile stamping die processing usually takes quite a long time, since the stamping dies are very large and heavy. Some are even more than 10 t. In this case, there are 170 moulds (i.e., jobs) need to be processed in the shop floor, and their processing times are listed in Table 11. In the table, "/" represents that there are no more jobs.

Electricity is the main energy and the electricity price for industry is dynamic. In this case, we use the non-summer energy price for industry in Shanghai, which is shown in Fig. 10. The electricity price varies hourly and presents the price-wise linear shape. The more the customers use during the peak period, the higher the electricity cost will be charged. Note that the average processing time p_j is as long as 20 h, and the changes of electricity prices will influence the total energy consumption largely.

5.4.2. Computational results of the case

For the case, the NSGA-II algorithm and the constructive heuristic method are applied. The computational results are shown in Table 12. The Pareto fronts are shown in Fig. 11.

An approximate Pareto front with 85 non-dominated solutions is obtained by the NSGA-II algorithm, while 83 solutions are obtained by the constructive heuristic method. Two solutions on the upper left side, (202, 118,638.5) and (203, 118,448.884) cannot be found by the constructive heuristic method. Considered that the completion time of these two solutions are near to the minimal makespan ($max(\sum_{j=1}^{n} p_j/m, max(p_j)) = 201$), such a phenomenon echoes the main drawback of the constructive heuristic method mentioned in Section 4.2.

In addition, with the same makespan, most solutions obtained by NSGA-II has a lower TEC value than that of the constructive heuristic method, especially those on the lower right side. The D_R measure of the NSGA-II algorithm and the constructive heuristic method are 1.19×10^{-4} and 3.90×10^{-3} respectively, which indicates that the NSGA-II algorithm has a better convergence compared to the constructive heuristic method.

As for computation time, it takes about 10 min to evolve a 500-individual population to 300 generations in the NSGA-II procedure. In contrast, the constructive heuristic method can obtain the approximate Pareto front in around 5 min.

6. Conclusions and future work

Originated from a real-world CNC processing of an automobile stamping die manufacturing shop floor, we studied a bi-objective identical parallel machine scheduling problem with different energy consumption rates of machines and time-of-use electricity prices, to minimize the total energy consumption and makespan simultaneously. An augmented ε-constraint method is applied for obtaining the exact Pareto fronts for small-scale instances, based on the adaption of the mathematical model in Che et al. (2017b). To tackle medium- and large-scale instances, a constructive heuristic method with a local search strategy is constructed according to the characteristics of the treated problem. Moreover, the detailed design of initialization, cross-over and three types of mutation procedures are accomplished so that the NSGA-II algorithm is adapted to the problem in a thorough manner. After the parameter tuning of NSGA-II with the Design of Experiments and Taguchi method, the computational experiments are conducted. The results on randomly generated data as well as the real-world case study show the promising performance of the NSGA-II and the constructive heuristic method.

For the future research, some other meta-heuristic algorithms could be developed, and the properties of the problem could be further explored to speed up the search for optimal Pareto front. Furthermore, the unrelated or unique parallel machine problems with sequence-dependent setup times could also be investigated as an extension.

Acknowledgement

The authors are grateful to the Editor and referees for their helpful comments, which improves the quality of the paper largely. This work is the overall extension of the primary idea of the conference paper in Wang and Ma (2016). This work was supported by the National Natural Science Foundation of China under Grants 71701144, 71571134 and 71571135. The work was also supported by the Fundamental Research Funds for the Central Universities.

Appendix

The model of Che et al. (2017b) is applied to our stamping die manufacturing problem to minimize the makespan and the total energy consumption (TEC) simultaneously. The detailed formulation is shown as below.

Indices:

j: index of job, $j\in\{1,\ldots,n\}$.
i: index of machine, $i\in\{1,\ldots,m\}$.
k: index of time interval, $k\in\{1,\ldots,K\}$.

Parameters:

n: the number of jobs.
m: the number of machines.
K: the number of time intervals.
p_j: the processing time of job j.
e_i: the energy consumption rate of machine i.
c_k: the energy price of time interval k.
T_k: the duration time of time interval k.
$\mathscr{N}$: the set of jobs, i.e., $\mathscr{N}=\{1,\ldots,n\}$.
$\mathscr{M}$: the set of machines, i.e., $\mathscr{M}=\{1,\ldots,m\}$.
$\mathscr{K}$: the set of time intervals, i.e., $\mathscr{K}=\{1,\ldots,K\}$.
τ_j: the possible maximum number of periods that job j can span, computed as $\tau_j=p_j/\min_{k\in\mathscr{K}}\{T_k\}$.
L: a large enough positive integer, it is safely to set $L=\sum_{k\in\mathscr{K}}T_k+\sum_{j\in\mathscr{N}}p_j$.

Variables:

C_{max}: makespan.
C_j: the completion time of job j.
x_{ijk}: the actual processing time of job j in time interval k of machine i.
y_{ijk}: equal to 1 if job j is processed in time interval k on machine i, 0 otherwise.
v_{ij}: equal to 1 if job j is processed on machine i, 0 otherwise.

MILP:

$$\min \quad f_1=C_{max}; \tag{7}$$

$$\min \quad f_2=\sum_{j=1}^{n}\sum_{i=1}^{m}\sum_{k=1}^{K}e_ic_kx_{ijk}; \tag{8}$$

subject to:

$$\sum_{k=1}^{K}\sum_{i=1}^{m}\left(x_{ijk}\Big/p_j\right)=1;\quad \forall j\in\mathscr{N}; \tag{9}$$

$$x_{ijk}\le p_jy_{ijk};\quad \forall j\in\mathscr{N},i\in\mathscr{M},k\in\mathscr{K}; \tag{10}$$

$$\sum_{j=1}^{n}x_{ijk}\le T_k;\quad \forall i\in\mathscr{M},k\in\mathscr{K}; \tag{11}$$

$$\sum_{l=k+2}^{K}y_{ijl}\le K\Big(1-y_{ijk}+y_{ij(k+1)}\Big);\quad \forall j\in\mathscr{N},i\in\mathscr{M},k\in\mathscr{K}\backslash\{K-1,K\}; \tag{12}$$

$$x_{ijk}\ge T_k\Big(y_{ij(k-1)}+y_{ij(k+1)}-1\Big);\quad \forall j\in\mathscr{N},i\in\mathscr{M},k\in\mathscr{K}\backslash\{1,K\}; \tag{13}$$

$$v_{ij}\ge\sum_{k=1}^{K}y_{ijk}\Big/\tau_j;\quad \forall j\in\mathscr{N},i\in\mathscr{M}; \tag{14}$$

$$\sum_{i=1}^{m}v_{ij}=1;\quad \forall j\in\mathscr{N}; \tag{15}$$

$$\left(1-\sum_{i=1}^{m}y_{ijk}\right)L+C_j\ge\sum_{l=1}^{k-1}T_l+\sum_{i=1}^{m}x_{ijk};\quad \forall j\in\mathscr{N},k\in\mathscr{K}\backslash\{1\}; \tag{16}$$

$$C_{max}\ge C_j;\quad \forall j\in\mathscr{N}; \tag{17}$$

$$x_{ijk},C_j,C_{max}\ge 0;\quad \forall j\in\mathscr{N},i\in\mathscr{M},k\in\mathscr{K}; \tag{18}$$

$$y_{ijk},v_{ij}\in\{0,1\};\quad \forall j\in\mathscr{N},i\in\mathscr{M},k\in\mathscr{K}; \tag{19}$$

The objective functions (7) and (8) stand for the makespan and the *TEC* respectively. Constraints (9)–(11) are related to the processing time constraints, among which constraints (9) ensure that the total processing time of a job assigned in all periods on all machines is equal to its corresponding processing time. Constraints (10) ensure that if a job j is not processed in period k on machine i (i.e., $y_{ijk}=0$), then the actual processing time of job j in period k on machine i should take the value of 0. Constraints (11) guarantee that the total processing time of all jobs assigned in a period on any machine cannot exceed its duration. Constraints (12) ensure that if job j is processed on some machine across more than one period, then these periods must be continuous. Constraints (13) guarantee that if a job is processed in periods $k-1$ and $k+1$ on some machine, then the middle period k should be fully occupied by the same job due to the non-preemption assumption. Constraints (14) ensures the consistency in the definitions of binary variables v_{ij} and y_{ijk}. Constraints (15) guarantee that each job can only be processed on one machine. Constraints (16) define the completion time of job j. Constraints (17) define C_{max}. Constraints (18) and (19) give the range of decision variables.

It can be found that the problem $Pm,TOU||\{C_{max},TEC\}$ can be reduced to $Pm||C_{max}$ if we set $c_k=c$ and $p_j=p$. Hence the problem $Pm,TOU||\{C_{max},TEC\}$ is NP-hard, since $Pm||C_{max}$ is NP-hard (Garey and Johnson, 1978).

References

Ahmadi, A., Tiruta-Barna, L., Benetto, E., Capitanescu, F., Marvuglia, A., 2016a. On the importance of integrating alternative renewable energy resources and their life cycle networks in the eco-design of conventional drinking water plants. J. Clean. Prod. 135, 872–883.

Ahmadi, A., Tiruta-Barna, L., Capitanescu, F., Benetto, E., Marvuglia, A., 2016b. An archive-based multi-objective evolutionary algorithm with adaptive search space partitioning to deal with expensive optimization problems: application to process eco-design. Comput. Chem. Eng. 87, 95–110.

Bé rubé, J., Gendreau, M., Potvin, J., 2009. An exact εh-constraint method for bi-objective combinatorial optimization problems: application to the traveling salesman problem with profits. Eur. J. Oper. Res. 194 (1), 39–50.
Bruzzone, A., Anghinolfi, D., Paolucci, M., Tonelli, F., 2012. Energy-aware scheduling for improving manufacturing process sustainability: a mathematical model for flexible flow shops. CIRP Ann. - Manuf. Technol. 61, 459–462.
Che, A., Wu, X., Peng, J., Yan, P., 2017a. Energy-efficient bi-objective single-machine scheduling with power-down mechanism. Comput. Oper. Res. 85, 172–183.
Che, A., Zhang, S., Wu, X., 2017b. Energy-conscious unrelated parallel machine scheduling under time-of-use electricity tariffs. J. Clean. Prod. 156, 688–697.
Che, A., Zeng, Y., Lyu, K., 2016. An efficient greedy insertion heuristic for energy conscious single machine scheduling problem under time-of-use electricity tariffs. J. Clean. Prod. 129, 565–577.
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.
CSY, 2016. China Statistical Yearbook 2015 (Online). www.stats.gov.cn/tjsj/ndsj/2016/indexeh.htm. Accessed on 5 May 2018.
Dai, M., Tang, D., Giret, A., Salido, M., Li, W., 2013. Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm. Robot. Comput. Integrated Manuf. 29 (5), 418–429.
Deb, K., Agarwal, S., Pratap, A., Meyarivan, T., 2000. Parallel Problem Solving from Nature. A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II, vol. 6, pp. 849–858.
Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. 6 (2), 182–197.
Ding, J., Song, S., Zhang, R., Chiong, R., Wu, C., 2016a. Parallel machine scheduling under time-of-use electricity prices: new models and optimization approaches. IEEE Trans. Autom. Sci. Eng. 13 (2), 1138–1154.
Ding, J., Song, S., Wu, C., 2016b. Carbon-efficient scheduling of flow shops by multi-objective optimization. Eur. J. Oper. Res. 48 (3), 758–771.
EIA, 2013. International Energy Outlook 2013 (Online). www.eia.gov/forecasts/ieo/pdf/0484(2013).pdf. Accessed on 5 May 2018.
Fang, K., Lin, B., 2013. Parallel-machine scheduling to minimize tardiness penalty and power cost. Comput. Ind. Eng. 64 (1), 224–234.
Fang, K., Uhan, N., Zhao, F., Sutherland, J., 2016. Scheduling on a single machine under time-of-use electricity tariffs. Ann. Oper. Res. 238, 199–227.
Gahm, C., Denz, F., Dirr, M., Tuma, A., 2016. Energy-efficient scheduling in manufacturing companies: a review and research framework. Eur. J. Oper. Res. 248 (3), 744–757.
Garey, M.R., Johnson, D.S., 1978. “Strong” NP-completeness results: motivation, examples, and implications. J. ACM 25 (3), 499–508.
Giret, A., Trentesaux, D., Prabhu, V., 2015. Sustainability in manufacturing operations scheduling: a state of the art review. J. Manuf. Syst. 37 (1), 126–140.
Graham, R., Lawler, E., Lenstra, J., Rinnooykan, A., 1979. Optimization and approximation in deterministic sequencing and scheduling: a survey. Ann. Discrete Math. 5, 287–326.
He, Y., Li, Y., Wu, T., Sutherland, J., 2015. An energy-responsive optimization method for machine tool selection and operation sequence in flexible machining job shops. J. Clean. Prod. 87, 245–254.
Ishibuchi, H., Yoshida, T., Murata, T., 2003. Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. IEEE Trans. Evol. Comput. 7 (2), 204–223.
Ji, M., Wang, J., Lee, W., 2013. Minimizing resource consumption on uniform parallel machines with abound on makespan. Comput. Oper. Res. 40, 2970–2974.
Jia, Z., Zhang, Y., Leung, J., 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.
Li, K., Shi, Y., Yang, S., Cheng, B., 2011. Parallel machine scheduling problem to minimize the makespan with resource dependent processing times. Appl. Soft Comput. 11 (8), 5551–5557.
Li, Z., Yang, H., Zhang, S., Liu, G., 2016a. Unrelated parallel machine scheduling problem with energy and tardiness cost. Int. J. Adv. Manuf. Technol. 84 (1), 213–226.
Li, K., Zhang, X., Leung, J., Yang, S., 2016b. Parallel machine scheduling problems in green manufacturing industry. J. Manuf. Syst. 38, 98–106.
Lin, W., Yu, D., Zhang, C., Liu, X., Zhang, S., Tian, Y., Liu, S., Xie, Z., 2015. A multi-objective teaching-learning-based optimization algorithm to scheduling in turning processes for minimizing makespan and carbon footprint. J. Clean. Prod. 101, 337–347.
Liu, C., Yang, J., Lian, J., Li, W., Evans, S., Yin, Y., 2014b. Sustainable performance oriented operational decision-making of single machine systems with deterministic product arrival time. J. Clean. Prod. 85, 318–330.
Liu, Y., Dong, H., Lohse, N., Petrovic, S., Gindy, N., 2014a. An investigation into minimising total energy consumption and total weighted tardiness in job shops. J. Clean. Prod. 65, 87–96.
Luo, H., Du, B., Huang, G., Chen, H., Li, X., 2013. Hybrid flow shop scheduling considering machine electricity consumption cost. Int. J. Prod. Econ. 146, 423–439.
Mansouri, S., Aktas, E., Besikci, U., 2016. Green scheduling of a two-machine flowshop: trade-off between makespan and energy consumption. Eur. J. Oper. Res. 248 (3), 772–788.
Merkert, L., Harjunkoski, I., Isaksson, A., Saynevirta, S., Saarela, A., Sand, G., 2015. Scheduling and energy-industrial challenges and opportunities. Comput. Chem. Eng. 72, 183–198.
Mavrotas, G., 2009. Effective implementation of the e-constraint method in multi-objective mathematical programming problems. Appl. Math. Comput. 213, 455–465.
Mouzon, G., 2008. Operational Methods and Models for Minimisation of Energy Consumption in a Manufacturing Environment. Ph.D. thesis. Wichita State University, Wichita, the United States of America.
Mouzon, G., Yildirim, M., Twomey, J., 2007. Operational methods for minimization of energy consumption of manufacturing equipment. Int. J. Prod. Res. 45 (18–19), 4247–4271.
Shahidi-Zadeh, B., Tavakkoli-Moghaddam, R., Taheri-Moghadam, A., Rastgar, I., 2017. Solving a bi-objective unrelated parallel batch processing machines scheduling problem: a comparison study. Comput. Oper. Res. 88, 71–90.
Shrouf, F., Ordieres-Meré, J., García-Sánchez, A., Ortega-Mier, M., 2014. Optimizing the production scheduling of a single machine to minimize total energy consumption costs. J. Clean. Prod. 67, 197–207.
Tang, D., Dai, M., Salido, M., Giret, A., 2016. Energy-efficient dynamic scheduling for a flexible flow shop using an improved particle swarm optimization. Comput. Ind. 81, 82–95.
Wang, S., Ma, S., 2016. An energy-efficient scheduling method for a parallel-machine scheduling problem in automobile mould machining. In: CIE46 Proceedings: 1613-1621. 46th International Conferences on Computers and Industrial Engineering, 29-31 October 2016, Tianjin, China.
Wang, Y., Wang, M., Lin, S., 2017. Selection of cutting conditions for power constrained parallel machine scheduling. Robot. Comput. Integrated Manuf. 43, 105–110.
Wu, X., Che, A., 2018. A memetic differential evolution algorithm for energy-efficient parallel machine scheduling (in press). Omega. https://doi.org/10.1016/j.omega.2018.01.001. Accessed on 12 May 2018.
Yildirim, M., Mouzon, G., 2012. Single-machine sustainable production planning to minimizing total energy consumption and total completion time using a multiple objective genetic algorithm. IEEE Trans. Eng. Manag. 59 (4), 585–597.
Zeng, Y., Che, A., Wu, X., 2018. Bi-objective scheduling on uniform parallel machines considering electricity cost. Eng. Optim. 50 (1), 19–36.
Zhang, R., Chiong, R., 2016. Solving the energy-efficient job shop scheduling problem: a multi-objective genetic algorithm with enhanced local search for minimizing the total weighted tardiness and total energy consumption. J. Clean. Prod. 112, 3361–3375.
Zhang, H., Zhao, F., Fang, K., Sutherland, J., 2014. Energy-conscious flow shop scheduling under time-of-use electricity tariffs. CIRP Ann. - Manuf. Technol. 63 (1), 37–40.