Journal of Cleaner Production 172 (2018) 3249–3264

Contents lists available at ScienceDirect

Journal of Cleaner Production

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

A green scheduling algorithm for flexible job shop with energy-saving measures

Xiuli Wu*, Yangjun Sun

Department of Logistics Engineering, School of Mechanical Engineering, University of Science and Technology Beijing, Beijing, China

ARTICLE INFO

Article history:
Received 21 February 2017
Received in revised form
21 August 2017
Accepted 31 October 2017
Available online 2 November 2017

Keywords:
Flexible job shop scheduling problem
Energy-saving measure
Turn-on/off machines
Multi-speed machine
A green scheduling heuristic

ABSTRACT

We study how to save energy from the viewpoint of operation management. When to turn-on/off machines and which speed level to choose are two measures we employ to save energy. We focus on the flexible job shop scheduling problem. To begin with, a model is formulated for the flexible job shop scheduling problem when the two energy-saving measures are under consideration. An energy consumption model is proposed to compute the energy consumption for a machine in different states. Then, a non-dominated sorted genetic algorithm is developed to solve the problem. In the non-dominated sorted genetic algorithm, a green scheduling heuristic is presented to optimize the makespan, the energy consumption and the numbers of turning-on/off machines simultaneously. Finally, the comprehensive experiment results prove that the proposed model and the algorithm can solve the problem effectively and efficiently.

1. Introduction

Reducing greenhouse gas emissions is one of the most important challenges facing by the manufacturing industry today. According to US Energy Information Administration (EIA, 2010), the industrial sector currently contributes about one-half of the world's total energy consumption, which has almost doubled over the last 60 years. In China, for instance, the average proportion of industrial GDP, 40.1%, is obtained by consuming 67.9% of national energy and emitting 83.1% of national carbon dioxide since 1978 (Chen, 2009). Energy-saving and emission-reduction has attracted more and more concerns from governments and researchers recently. For example, China "13th Five-Year Plan" (2016) requires the energy consumption in 2020 to be reduced by 15% as compared with 2015. The existed study on reducing manufacturing energy consumption has largely centered on developing more energy efficient machines or processes, such as the work of Haapala et al. (2009), Diarra et al. (2010), and Nava et al. (2010). Gutowski et al. (2005) observed that the total energy requirement for the active removal of material can be quite small compared to the supportive process needed for operating a machine, and that more than 85% of the energy is consumed by non-machining operations that are not directly related to the actual production of parts in the Toyota Motor Corporation (a mass production environment). This implies that the attention should be paid to developing new operational methods at the system level and may potentially realize significant energy reduction. It is well known that the production scheduling plays an important role in operation management. If we could make an energy-saving scheduling solution, it would definitely be an effective approach to realize energy-saving and emission-reduction.

Following this way, researchers have carried out a few studies and have proposed some approaches. Generally, these approaches can be classified into three groups.

(1) Turn off the idle machines. Turning off the machines which will be kept idle for a certain time can save energy effectively. One needs to make the decision when to turn-on/off machines besides the traditional scheduling task. One of the earliest studies can be found in Mouzon et al. (2007). They built a scheduling rule and a scheduling model for a single machine scheduling problem. A neural network was employed to predict the arrival of the next job in order to decide whether to turn off the machine during the next idle time slot. This work has further been extended in 2008, in which they proposed a break-even duration concept (Yildirim and Mouzon, 2008). If the idle time slot on the machine is less than the break-even duration, keep the

* Corresponding author.
E-mail addresses: wuxiuli@ustb.edu.cn (X. Wu), ustb_sunyj@163.com (Y. Sun).

https://doi.org/10.1016/j.jclepro.2017.10.342
0959-6526/

machine idle; otherwise, it is wise to turn off the machine in order to save energy. Che et al. (2017) studied the single machine scheduling problem with energy consumption as the optimization objective. The machine could be turned off to save energy. Dai et al. (2013) solved the flexible flow shop scheduling problem with a genetic simulated annealing algorithm. They proposed an energy-saving difference coefficient to control when to turn off a machine.

(2) Slow down the speed level without postponing the makespan. Different speed level consumes different amounts of energy. It is wise to slow down the speed level of the non-bottleneck machines. Hence, the multi-speed machine scheduling problem appears. E.g. Che et al. (2015) studied the multi-speed single machine scheduling problem with the due date as one of constraints. They employed the CPLEX software to solve this problem. Mansouri et al. (2016) studied the 2-machine flow shop scheduling problem with multi-level speed. They built a multi-objective mixed integer programming model and designed a heuristic to solve the model. Fang et al. (2011) also studied the 2-machine flow shop scheduling problem. Later, they discussed in detail the flow shop scheduling problem with both the discrete speed level and the continuous speed level (Fang et al., 2013). Besides the turning-on/off measure, Dai (2015) subdivided the energy consumption into the machine turning-on energy consumption, the idle energy consumption and the processing energy consumption. A particle swarm optimization algorithm was proposed to solve the dynamic flow shop scheduling problem when the new job arrival and machine breakdown were under consideration. Zhang and Chiong (2016) studied the job shop scheduling problem. They assumed that the processing time decreased with an increasing speed. A hybrid genetic algorithm integrating the local search was proposed to minimize the total tardiness and the energy consumption. Zhang et al. (2015) also studied whether to turn off an idle numeric control machine. They discussed the energy consumption in 7 states and proposed a genetic algorithm. Zhang et al. (2012) solved the static and dynamic scheduling problems in flexible manufacture system. They used a programming software to optimize the makespan and the energy consumption.

(3) Save energy by off-peak production. In peak, the electricity cost is much higher and the pollution is intensive. If the production is scheduled at off-peak, a substantial quantity electricity cost can be reduced and the pollution can also be relieved. Scheduling problems in this group don't make too much challenge except to shift the production horizon to the off-peak. Pach et al. (2014) made an energy-saving scheduling in flexible manufacturing system by turning off machines in time and processing in off-peak. Shrouf et al. (2014) proposed a genetic algorithm to solve a single machine scheduling problem with the energy-saving objective, in which the energy cost was different at different time. Cheng et al. (2017) studied the single machine batch scheduling problem. They considered the different electricity price and built a bi-objective mixed integer programming model.

In summary, the study on scheduling problems with energy consumption and the traditional objective as optimization objectives has just started and mainly focused on single machine scheduling problem and flow shop scheduling problem. There is, however, little work on the flexible job shop scheduling problem (FJSP). Therefore, this paper will contribute in the following: (1) an optimization model is formulated for the flexible job shop scheduling problem with multi-speed machine. In the model, turning off machines if there is enough idle time and selecting a suitable speed level are two measures to employed to save energy; (2) an energy consumption model for a machine is built which is composed of the turning-on/off energy consumption, the idle energy consumption, the processing energy consumption and the standby energy consumption; (3) a green scheduling heuristic is proposed to save energy without postponing the makespan; and (4) the non-dominated sorted genetic algorithm (NSGA-II) integrating a green scheduling heuristic is employed to solve the model.

The rest of this paper is organized as follows. The flexible job shop scheduling problem with energy-saving measure is introduced in Section 2. An optimization model is formulated in Section 3. The NSGA-II integrating a green scheduling heuristic is proposed in Section 4. The case study is in Section 5. Conclusions and the future work are presented in Section 6.

2. Problem description

Flexible job shop scheduling problem with energy-saving measures (FJSP-ESM) can be described as follows: n jobs will be processed by m machines. Each machine has diverse kinds of processing speed, which will consume correspondingly different amounts of energy. The $i-$ th job is composed of n_i operations and each operation can be processed by one or more machines. Each operation consumes different amounts of energy at different speed level. The energy consumption is proportional to the processing speed. One can choose to keep the machine idle or to turn off the machine between two adjacent processing tasks. Turning-on/off one machine will consume an extra amount of energy. A small amount of energy will be consumed when machines are kept idle. Frequent turning-on/off one machine will do harm to its age, so it would be wise to keep a machine being idle for a certain time. The task of FJSP-ESM is to assign machines to operations, to sequence the operations on each machine, to decide the processing speed for each operation, and to decide when to turn-on/off machines. The optimization objectives are the makespan, the energy consumption and the total numbers of turning-on/off machines.

To make it easier to understand, an instance of FJSP-ESM is shown in Table 1. There are 5 jobs and each comprises 3 operations. 5 machines are available and each has 3 speed levels. The data in Table 1 is the processing time on each available machine with different speed level. Take the first operation of job1 as an example. It can be processed on two machines, M1 and M2. The processing time with the three speed levels on M1 are 14, 11, and 9 time units respectively. Similarly, the processing time with the three speed levels on M2 are 12, 10, and 8 time units respectively. The power parameters for each machine are listed in Table 2, including the processing power for different speed levels, the idle power, the standby power, the turning-on/off energy consumption and the

Table 1
An instance of FJSP-ESM.

	Operation 1		Operation 2		Operation 3	
Job1	M1	14,11,9	2	11,9,7	1	14,11,9
	M2	12,10,8	4	14,11,9	4	6,5,4
Job2	M1	5,4,3	2	9,8,6	1	3,2,2
	M3	9,8,6	3	6,5,4	2	5,4,3
	M4	11,9,7	–	–	3	2,1,1
Job3	M3	5,4,3	1	14,11,9	3	9,8,6
	M4	8,6,5	5	12,10,8	5	8,6,5
Job4	M2	9,8,6	4	5,4,3	1	2,1,1
	M5	5,4,3	5	11,9,7	5	5,4,3
Job5	M2	8,6,5	1	9,8,6	2	6,5,4
	M5	11,9,7	2	6,5,4	3	3,2,2
	–	–	4	8,6,5	–	–

Table 2
Machine parameters.

	speed 1		speed 2		speed 3		The standby	Turning on/off	The threshold
	Processing	Idle	Processing	Idle	Processing	Idle			
Machine 1	1230	230	1510	320	2270	370	20	2600	7
Machine 2	1160	180	1500	280	1820	350	22	2530	8
Machine 3	1150	190	1390	300	1880	350	25	2560	6
Machine 4	1380	230	1920	330	2340	390	30	2740	7
Machine 5	1040	220	1500	310	2220	380	25	2640	6

threshold to determine if machine can be turned off. From Table 2, we can see that the processing power at the three speed levels on machine 1 are 1230 W, 1510 W, and 2270 W, respectively. The idle power is 230 W, 320 W and 370 W, respectively. The standby power is 20 W. The turning-on/off energy consumption is 2600 W•min. The threshold to turn off machine M1 is 7 time units.

3. The formulation of FJSP-ESM

3.1. Notations

Notations are listed in Table 3.

Table 3
Notations.

Symbol	Descriptions
m	The number of machines
n	The number of jobs
n_i	The number of the operations for job i
m_{ij}	The available machines for operation O_{ij}
a, i, h	The index for jobs, $a, i, h = 1, 2, \ldots, n$
b, j, l	The index for operations, $b, j, l = 1, 2, \ldots, \max\{n_a, n_i, n_h\}$
k	The index for machines, $k = 1, 2, \ldots, m$
q	The index for the processing speeds
P_{kq}	The processing power of machine k under speed q
Z_{kq}	The idle power of machine k with speed q
$P_k(t)$	The input power of machine k at time t
O_{ij}	The $j-$th operation of job i
X_{ijkq}	$X_{ijkq} = 1$ if the operation O_{ij} is processed on machine k with speed ,q otherwise, $X_{ijkq} = 0$
Y_{ijhl}	$Y_{ijhl} = -1$ if operation O_{ij} is the precedence of operation O_{hl}; $Y_{ijhl} = 1$if operation O_{hl} is the precedence of operation O_{ij}; otherwise, $Y_{ijhl} = 0$
C_{ijkq}	The ending time of operation O_{ij} on machine k with speed q
T_{ijkq}	The processing time of operation O_{ij} on machine k with speed q
$C_{\max}$	The makespan
Q	The total energy consumption
Q_{ij}	The energy consumption for operations O_{ij}
Q_{ck}	The total processing energy consumption
Q_{dk}	The energy consumption of machine k when in the idle state
Q_{gk}	The energy consumption for turning-on/off machine k
T_{k1}	The duration to turn on machine k
T_{k2}	The duration to turn off machine k
E_k	The time to turn on machine k at the beginning of a schedule
F_k	The time to turn off machine k at the ending of a schedule
Q_{fk}	The energy consumption of machine k when in the standby state
G	The total numbers of turning-on/off machines
g_k	The total numbers of turning-on/off machine k
E_{k1}	The latest time to turn on machine k
GT_{kq}	The breakeven duration to determine if machine k in the idle state with speed q can be turned off
U_{ijhl}	$U_{ijhl} = 1$ if the machine is turned off between operation O_{ij} and operation O_{hl}; otherwise, $U_{ijhl} = 0$
Q_{dkijhl}	The energy consumption between operation O_{ij} and operation O_{hl} when machine k is kept idle
H_k	A threshold for machine k to be turned off
Z_{fk}	The standby power for machine k

3.2. Assumptions

1) All machines are ready at $t = 0$;
2) All raw material is ready at $t = 0$;
3) The precedence relation between operations won't change;
4) All operations can be processed by the available machines with any speed level;
5) Each machine can only process one job at a certain moment;
6) A job can only be processed by one of the available machines with a chosen speed level at a certain moment;
7) Once start, the process cannot be interrupted.

3.3. The energy consumption model

In FJSP-ESM, machines have diverse kinds of speed and consume correspondingly different amounts of energy. In the idle time slot between two processing tasks, one can choose to turn off the machine. Hence, for one machine, the input power changes with its state. There are four states in total: the turning-on/off state, the processing state, the idle state and the standby state (Fig. 1).

1) the turning-on/off state

Frequent turning-on/off one machine will do harm to its performance and age. As a result, there is a restriction on turning-on/off machines. When a machine is turned on or turned off, the energy is consumed to active or to stop the parts of the machine (Dai, 2015). When a machine is scheduled to process jobs ($\max_{i,j,q}(X_{ijkq}) = 1$), it needs to be turned on and consumes energy. After it finishes all the tasks, it needs to be turned off and consumes energy too. In a scheduling, the energy consumption of turning-on/off machine k (i.e.Q_{gk}) is related with the total numbers of turning-on/off machine k (i.e.g_k), which can be computed with Eq. (1). The time to turn on machine k(i.e. E_k) and the time to turn off machine k(i.e.F_k) can be computed with Eqs. (2) and (3).

$$Q_{gk} = g_k \left[\int_{E_k}^{E_k+T_{k1}} P_k(t)dt + \int_{F_k-T_{k2}}^{F_k} P_k(t)dt \right] \max_{i,j,q} \left(X_{ijkq}\right) \tag{1}$$

$$E_k = \min_{i,j,q}\left(X_{ijkq}\left(C_{ijkq} - T_{ijkq}\right)\right) \tag{2}$$

$$F_k = \max_{i,j,q}\left(X_{ijkq}C_{ijkq}\right) \tag{3}$$

2) the idle state

When a machine is in the idle state, the energy consumption mainly results from the main drive system (Dai, 2015).

Theoretically, the energy consumption in the idle state is the product of the idle power and the idle time duration. The idle time duration is the duration between two adjacent tasks. The idle power changes with the preceding processing speed q. At the same processing speed, the idle power is nearly constant (He, 2007). Hence the idle energy consumption (i.e.Q_{dk}) can be computed with Eq. (4), in which the energy consumption between operation O_{ij} and operation O_{hl} when machine k is kept idle (i.e.Q_{dkijhl}) is computed with Eq. (5).

The detail for computing Q_{dkijhl} is as following. If $U_{ijhl}=0$, the machine will be kept idle all the time. If $U_{ijhl}=1$, it means that there is a scheduled turning-on/off. Here, there are two cases for the machine. The first is the machine is turned off immediately after the preceding operation is finished and it won't be turned on until the succeeding operation begins. No energy is consumed for this case. The latter is the machine is first kept idle for a while and then is turned off when the time duration is more than the threshold for the machine k to be turned off (H_k). Energy will be consumed for this case. The duration for keeping idle is the difference between the ending time and the starting time of an idle state. The starting time is the ending time of the preceding operation. The ending time is the larger of $(E_{k1}+H_k)$ and the ending time of the preceding operation. The coefficients D, D_1, D_2 are common factors. The latest time to turn on machine k(i.e.E_{k1}) is computed with Eq. (6). There are also two cases for computing E_{k1}. The first is to search if there exists an operation O_{ab} which satisfies the following four conditions simultaneously: a) it is scheduled on machine k, b) it is before the two adjacent operations O_{ij} and O_{hl}, c) there is a turning-on/off between the O_{ij} and O_{hl}, and d) there is only one turning-on/off between the O_{ab} and O_{ij}or O_{hl}. If there exists an operation O_{ab}, compute E_{k1} with Eq. (7). $E_{k1}$is the earliest starting time for all the operations O_{ab}. A means the earliest starting time between two adjacent operations O_{ij} and O_{hl}, which is computed with Eq. (8). Otherwise, if there doesn't exist such an operation O_{ab}, $E_{k1}=E_k$.

$$Q_{dk}=\sum_{i,h}^{n}\sum_{j,l}^{\max\{n_i,n_h\}}Q_{dkijhl} \tag{4}$$

3) the processing state

When a machine is in the processing state, the energy is consumed for processing. Stute and Limde (1955) stated that there is an approximate linear relationship between the total input and the output power after analyzing a lot of experiments results. That is, the energy consumption of scheduling is not merely the sum of processing energy consumption, the idle energy consumption and the turning-on/off energy consumption. From the viewpoint of energy consumption, there is a linear relationship between the total energy consumption and the processing energy consumption. Hence the processing energy consumption on machine k(i.e.Q_{ck}) can be computed with Eq. (9), in which the coefficient β usually between 1.15 and 1.25 (Liu et al., 1995). The processing power and the processing duration are determined by speed q.

$$Q_{ck}=\beta\sum_{q}P_{kq}\left(\sum_{i}^{n}\sum_{j}^{n_i}X_{ijkq}T_{ijkq}\right) \tag{9}$$

4) the standby state

When a machine is in the standby state, the standby power is usually consumed to keep a ready condition and to provide the convenience for the normal use. (China Certification Center for Energy Conservation Product (CECP), 2002). Generally, the standby power is constant when a machine is in the standby state. The standby energy consumption Q_{fk} is computed with Eq. (10). Z_{fk} is the standby power for machine k.

$$Q_{fk}=C_{\max}Z_{fk} \tag{10}$$

5) The total energy consumption

The total energy consumption is composed of four parts: the turning-on/off energy consumption, the idle energy consumption,

$$Q_{dkijhl}=\begin{cases} D\left(D_1\left(\max\left(E_{k1}+H_k, C_{ijkq_{ij}}\right)-C_{ijkq_{ij}}\right)+D_2\left(\max\left(E_{k1}+H_k, C_{hlkq_{hl}}\right)-C_{hlkq_{hl}}\right)\right) & ,U_{ijhl}=1 \\ D\left(D_1\left(C_{hlkq_{hl}}-T_{hlkq_{hl}}-C_{ijkq_{ij}}\right)+D_2\left(C_{ijkq_{ij}}-T_{ijkq_{ij}}-C_{hlkq_{hl}}\right)\right) & ,U_{ijhl}=0 \end{cases} \tag{5}$$

$$\text{where}, D=X_{ijkq_{ij}}X_{hlkq_{hl}}\left(Y_{ijhlk}/2\right), D_1=Z_{kq_{ij}}\left(Y_{ijhlk}-1\right), D_2=Z_{kq_{hl}}\left(Y_{ijhlk}+1\right)$$

$$E_{k1}=\begin{cases} \min\limits_{a,b}\left(C_{abkq_{ab}}-T_{abkq_{ab}}\right)X_{abkq_{ab}}X_{ijkq_{ij}}X_{hlkq_{hl}} & ,U_{abhl}+U_{abij}=1\cap U_{ijhl}=1\cap 0<C_{abkq_{ab}}<A \quad (6) \\ E_k & ,\text{other} \quad (7) \end{cases}$$

$$A=\left(Y_{ijhlk}/2\right)\left(\left(Y_{ijhlk}+1\right)\left(C_{hlkq_{hl}}-T_{hlkq_{hl}}\right)+\left(Y_{ijhlk}-1\right)\left(C_{ijkq_{ij}}-T_{ijkq_{ij}}\right)\right) \tag{8}$$

the processing energy consumption and the standby energy consumption. Hence the total energy consumption for one single machine is the sum of them (Eq. (11))

$$Q_k=Q_{gk}+Q_{dk}+Q_{ck}+Q_{fk} \tag{11}$$

3.4. The formulation of the FJSP-ESM optimization model

The makespan is always the most important production goal. Therefore, we will optimize three objectives simultaneously for the FJSP-ESM as shown in Eq. (12), in which $C_{\max}$ is the makespan, Q is the energy consumption; and G is the total numbers of turning-on/off machines.

$$f = \min(C_{\max}, Q, G) \tag{12}$$

$$C_{\max} = \max_{i,j} X_{ijkq} C_{ijkq} \tag{13}$$

$$E_k = \min_{i,j,q}\left(X_{ijkq}\left(C_{ijkq} - T_{ijkq}\right)\right) \tag{20}$$

$$F_k = \max_{i,j,q}\left(X_{ijkq} C_{ijkq}\right) \tag{21}$$

$$Q_{dk} = \sum_{i,h}^{n} \sum_{j,l}^{\max\{n_i,n_h\}} Q_{dkijhl} \tag{22}$$

$$Q_{dkijhl} = \begin{cases} D\left(D_1\left(\max\left(E_{k1}+H_k, C_{ijkq_{ij}}\right) - C_{ijkq_{ij}}\right) + D_2\left(\max\left(E_{k1}+H_k, C_{hlkq_{hl}}\right) - C_{hlkq_{hl}}\right)\right) & , U_{ijhl} = 1 \\ D\left(D_1\left(C_{hlkq_{hl}} - T_{hlkq_{hl}} - C_{ijkq_{ij}}\right) + D_2\left(C_{ijkq_{ij}} - T_{ijkq_{ij}} - C_{hlkq_{hl}}\right)\right) & , U_{ijhl} = 0 \end{cases} \tag{23}$$

where, $D = X_{ijkq_{ij}} X_{hlkq_{hl}}\left(Y_{ijhlk}/2\right), D_1 = Z_{kq_{ij}}\left(Y_{ijhlk} - 1\right), D_2 = Z_{kq_{hl}}\left(Y_{ijhlk} + 1\right)$

$$E_{k1} = \begin{cases} \min_{a,b}\left(C_{abkq_{ab}} - T_{abkq_{ab}}\right) X_{abkq_{ab}} X_{ijkq_{ij}} X_{hlkq_{hl}} & , U_{abhl} + U_{abij} = 1 \cap U_{ijhl} = 1 \cap 0 < C_{abkq_{ab}} < A \quad (24) \\ E_k & , \text{other} \quad (25) \end{cases}$$

$$Q = \sum_{k=1}^{m}\left(Q_{gk} + Q_{dk} + Q_{ck} + Q_{fk}\right) \tag{14}$$

$$G = \sum_{k=1}^{m} g_k \tag{15}$$

s.t.

$$C_{ij} - T_{ij} \geq C_{i(j-1)} \tag{16}$$

$$\begin{aligned} &\left(Y_{ijhlk}/2\right)\left(Y_{ijhlk} - 1\right)\left(C_{hl} - C_{ij} - T_{hlkq}\right) X_{ijkq_{ij}} X_{hlkq_{hl}} \\ &+ \left(Y_{ijhlk}/2\right)\left(Y_{ijhlk} + 1\right)\left(C_{ij} - C_{hl} - T_{ijkq}\right) X_{ijkq_{ij}} X_{hlkq_{hl}} \geq 0 \end{aligned} \tag{17}$$

$$\sum_{k=1}^{m} X_{ijkq} = 1 \tag{18}$$

$$Q_{gk} = g_k\left[\int_{E_k}^{E_k+T_{k1}} P_k(t)dt + \int_{F_k-T_{k2}}^{F_k} P_k(t)dt\right] \max_{i,j,q}\left(X_{ijkq}\right) \tag{19}$$

$$\begin{aligned} A = {} & \left(Y_{ijhlk}/2\right)\left(\left(Y_{ijhlk}+1\right)\left(C_{hlkq_{hl}} - T_{hlkq_{hl}}\right) \right. \\ & \left. + \left(Y_{ijhlk}-1\right)\left(C_{ijkq_{ij}} - T_{ijkq_{ij}}\right)\right) \end{aligned} \tag{26}$$

$$Q_{ck} = \beta \sum_{q} P_{kq}\left(\sum_{i}^{n} \sum_{j}^{n_i} X_{ijkq} T_{ijkq}\right) \tag{27}$$

$$Q_{fk} = C_{\max} Z_{fk} \tag{28}$$

$$\begin{aligned} & U_{ijhl} X_{ijkq} X_{hlkq} |Y_{ijhlk}|\left(\max\left(C_{ijkq_{ij}} - T_{ijkq_{ij}}, C_{hlkq_{hl}} - T_{hlkq_{hl}}\right) \right. \\ & \left. - \max\left(\left(H_k + E_{k1}\right), \min\left(C_{ijkq_{ij}}, C_{hlkq_{hl}}\right)\right)\right) \geq G T_{kq} \end{aligned} \tag{29}$$

$$g_k = \sum_{i,h}^{n} \sum_{j,l}^{\max\{n_i,n_h\}} U_{ijhl} X_{ijkq} X_{hlkq}\left|Y_{ijhlk}\right|, \forall q \tag{30}$$

$$X_{ijkq} = \begin{cases} 1 & \text{operation } O_{ij} \text{ is processed with speed } q \text{ on machine } k \\ 0 & \text{otherwise} \end{cases} \tag{31}$$

$$Y_{ijhlk} = \begin{cases} 1 & \text{operation } O_{ij} \text{ is the sucessor of operation } O_{hl} \text{ on machine } k \\ 0 & \text{operation } O_{ij} \text{ and operation } O_{hl} \text{ are not adjacent} \\ -1 & \text{operation } O_{ij} \text{ is the preceding of operation } O_{hl} \text{ on machine } k \end{cases} \tag{32}$$

$$U_{ijkq} = \begin{cases} 1 & \text{The machine will be turned off between the operation } O_{ij} \text{ and operation } O_{hl} \\ 0 & \text{The machine won't be turned off between the operation } O_{ij} \text{ and operation } O_{hl} \end{cases} \tag{33}$$

$$C_{ijkq} \geq 0 \tag{34}$$

$$T_{ijkq} \geq 0 \tag{35}$$

Eq. (13) is the makespan objective. Eq. (14) is the total energy consumption. Eq. (15) is the total numbers of turning-on/off mahcines. Eq. (16) shows the precedence relation. Eq. (17) represents that one machine can't process more than one jobs at the same time. Eq. (18) is the decision variable to constrain an operation can only be processed by one of the available machines. Eqs. (19)–(28) is to compute the energy consumption which has been discussed in section 3.3. Eq. (29) defines the duration between the latest turning-off time and the next turning-on time must exceed the breakeven duration to maintain the performance of one machine. Eq. (30) is to compute the total numbers of turning-on/off machines. Eqs. (31)–(35) are the decision variable constraints.

4. The non-dominated sorted genetic algorithm

4.1. The framework

The flexible job shop scheduling problem with more than 2 jobs is proved to be a NP-hard problem (Ho and Tay, 2004). Hence, heuristics and meta-heuristics are usually employed to achieve an approximately optimal solution (Wu and Wu, 2017). The genetic algorithm (GA) is based on the "survival of the fittest". It is a highly parallel, random and adaptive optimization algorithm. The solution of a problem is represented as a chromosome. Through the population evolution (selection, crossover and mutation), GA searches the optimal solution. For multi-objective optimization problems, the NSGA-II has been proved to be an effective algorithm (Deb et al., 2002). The basic idea of the NSGA-II is as following: first, initialize a population and evaluate the population with a green scheduling heuristic. Then, cross and mutate the population. Next, evaluate the individuals, combine the parent and the offspring populations, sort the non-dominated solutions and compute the crowding distance. Finally, select the individuals into the next generation based on the non-dominated sorting and the crowding distance. Repeat until the termination condition is satisfied.

The pseudo-code is as follows.

```
Initialize a population R with N individuals; record the current iteration
number as s , the total iterations S
    Evaluate R
    while s < S do
        if p_c > rand
            crossover, R' ← R
        end if
        if p_m > rand
            mutation, R" ← R'
        end if
        evaluate R"
        combine R and R"
        execute the non-dominated sorting
        calculate the crowding distance
        select the new population
    end while
    output the Pareto solutions
```

p_c : the probability of crossover; p_m : the probability of mutation.

4.2. The details of NSGA-II

4.2.1. Encoding

Encoding represents the problem as a chromosome. The operation based encoding is employed to encode a FJSP-ESM to a row vector (Wu and Wu, 2017). Each chromosome consists of $n \times \max\{n_i\}$ numbers and each number corresponding a job occurs $\max\{n_i\}$ times. A surplus part represents those jobs whose operation number are less than $\max\{n_i\}$. The surplus part is referred to as virtual operations that don't take any machine time. The $j-$th occurrence of the number i represents operation O_{ij}. Take the instance mentioned before (see Table 1), the FJSP-ESM can be represented as follows.

1	2	5	4	3	4	2	1	5	3	3	2	1	5	4
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
O_{11}	O_{21}	O_{51}	O_{41}	O_{31}	O_{42}	O_{22}	O_{12}	O_{52}	O_{32}	O_{33}	O_{23}	O_{13}	O_{53}	O_{43}

4.2.2. A green scheduling heuristic

In FJSP-ESM, each operation can be processed by the available machines with all levels of speed. Hence, how to assign a suitable machine with an optimal speed level is the main concern when to decode a chromosome to a scheduling solution. In our encoding method, the speed level has not been taken into account. So the decoding procedure needs to determine the speed level besides the machine assignment. Here, a green scheduling heuristic is proposed to deal with this problem. Before the scheduling heuristic is proposed, the idle time first rule (ITPR) and the turning-on/off schedule heuristic are presented.

1) Idle time first rule

The idle time first rule means that if an operation can be assigned into one of the idle time slot while the precedence relation is satisfied, the idle time is first preferred (Wu and Wu, 2017).

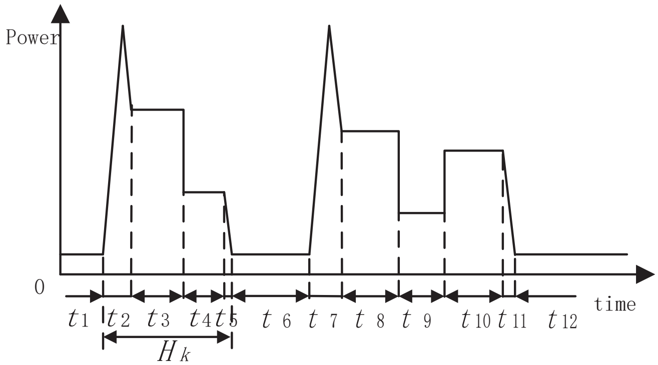

Fig. 1. The power distribution for different states.

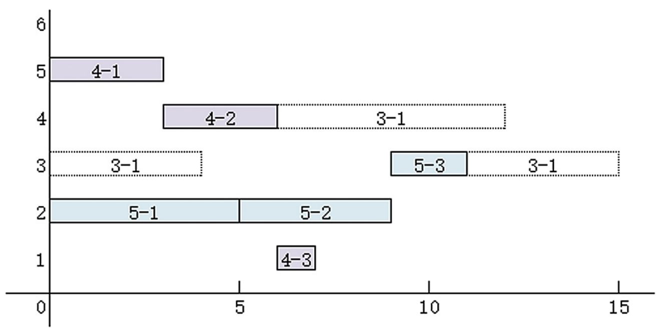

Fig. 2. An example of ITFR.

Otherwise, append the operation to the end of the current sequence. To explain it clearer, an example is shown in Fig. 2. Operations O_{51}, O_{52}, O_{41}, O_{42}, O_{43} and O_{53} have been sequenced and now it is the operation O_{31} turn to be sequenced. As can be seen from Fig. 2, if we don't consider the existed idle time slots, machine 4 has the highest priority. However, if the idle time slot on machine 3 is considered, the processing time for operation O_{31} will be greatly reduced. There is only operation O_{53} on machine 3 leading to an idle time slot from time 0 to time 9 units. The longest processing time for operation O_{31} is 5 units. Operation O_{31} is the first operation of job J_3, hence it can start from the very beginning. The machine's speed level and the processing time will be determined by the following two scheduling heuristics.

2) Machine turning-on/off scheduling heuristic

If one machine is kept being idle for a period, it is wise to turn off it to save energy and to reduce carbon emission. On the other hand, frequent turning-off machines will definitely harm the machine performance and age. Hence, a heuristic is proposed to schedule when to turn on/off machines.

Step 1: judge whether the length of an idle time slot is more than the breakeven duration GT_{kq}. If yes, it is possible to turn off the machine, go to step 2; otherwise, ends.
Step 2: compare the difference between the starting time of the idle time slot and the latest turning-on time of the machine. If the difference exceeds the predefined threshold H_k, turn off the machine at the starting time of the idle time slot directly. Otherwise, go to step 3.
Step 3: let the latest turning-on time of the machine be $t1$ and the starting time of the successor operation $t3$. compute $t2 = t1 + H_k$. If $t3 - t2 > GT_{kq}$, turn off the machine at the time $t1 + H_k$. Otherwise, keep the machine idle.

The pseudo-code is as follows.

for $c' = 1{:}num2 - 1$
  if $TI > GT_{kq}$
    if $C_{c'kq_{c'}} - T_{c'kq_{c'}} > t1$
      $U_{c'(c'+1)} = 1$
    else
      $t2 = t1 + H_k$
      if $t3 - t2 > GT_{kq}$
        $U_{c'(c'+1)} = 1$
      else
        $U_{c'(c'+1)} = 0$
      end if
    end if
  else
    $U_{c'(c'+1)} = 0$
  end if
end for

TI :the idle time slot between operation c' and its successor operation on machine k ; $num2$: the scheduled operation number after operation O_{ij} is scheduled.

To make the turning-on/off scheduling heuristic easier to understand, an example is shown in Fig. 3. After operations O_{11}, O_{21}, O_{51}, O_{52}, O_{41}, O_{31}, O_{12}, O_{53} and O_{13} are sequenced, there exists an idle time slot from 5 to 16 units time on machine 4. This idle time slot is more than the breakeven duration (assume $GT_{kq} = 7.02$) of machine 4. So, it is possible to turn off machine 4. Furthermore, the latest turning-on time on machine 4 is at the beginning, i.e. $t1 = 0$. The threshold H_k is 7. $t1 + H_k = 7$. The rest idle time duration is 9 units (from 7 to 16), which is more than the breakeven duration. Hence it is wise to turn off machine 4 from time 9 and turn it on at time 16.

3) The green scheduling heuristic

A green scheduling heuristic is proposed combining ITFR and the machine turning-on/off schedule heuristic. The main idea is to slow down the machines on the non-critical path with the makespan unaffected in order to save energy. To achieve this aim, two rules are developed. First check if there is an idle time slot into which the current operation can be inserted. If yes, select rule 1; otherwise, select rule 2.

Rule 1: keep the current turning-on/off arrangement unchanged as far as possible and select those machines and the respective speed level whose energy consumption is the lowest;

Rule 2: select a machine which can process the current operation as early as possible and then append the current operation to the scheduled sequence on the machine.

The procedure of the green scheduling heuristic is as follows:

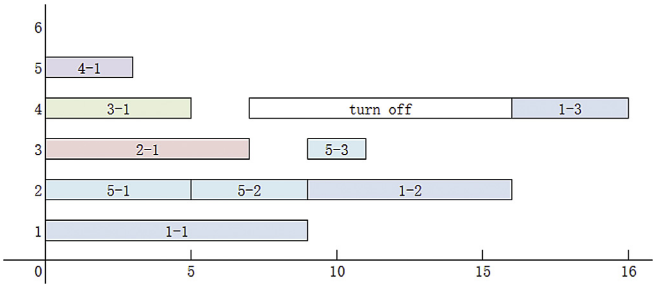

Fig. 3. An example of the turning-on/off scheduling heuristic.

Input: the individual r whose length is $len(r)$.

Output: An optimal scheduling solution in which the starting time, the ending time, the selected machine and the speed level for each operation are determined.

Step 1: Obtain the individual r and set $z = 1$;

Step 2: Repeat when z is less than $len(r)$;

Step 2.1. Obtain the z-th gene $r(z)$;

Step 2.2. Determine the operation index $j(=count(r(z)))$ according to the appearing times of $r(z)$. Search the available machines for the current operation $O_{r(z)j}$, and compute the completion time, the processing energy consumption, the current makespan, the energy consumption change and the total load of the machine with ITFR. For those machines on which the current operation can be inserted into the idle time slot, according to Rule 1, go to step 2.3; for those machines that the current operation can only be appended at the rear of the sequence, according to rule 2, go to step 2.4.

Step 2.3. On these available machines, before the current operation is inserted into an idle time slot, we should check if the existed machine turning-on/off schedule is interrupted or not. There are three scenarios: totally interrupted, non-interrupted and partially interrupted. For the first two, go to step 2.3.1 directly. For the third, first select those available machines on which the turning-on/off schedule is not interrupted and then go to step 2.3.1.

Step 2.3.1: Select the machine whose total energy consumption is the smallest. If there are more than one alternative, go to step 2.3.2. Otherwise, go to step 2.3.4.

Step 2.3.2: Select the machine in which the energy consumption for processing the current operation is the smallest. If there are more than one alternative, go to step 2.3.3. Otherwise go to step 2.3.4.

Step 2.3.3: Select the machine whose load up to now is the smallest. If the machine and speed are not unique, select randomly one machine and go to step 2.3.4; otherwise, go directly to step 2.3.4.

Step 2.3.4: Insert the operation $O_{r(z)j}$ and determine its beginning time $st(r(z),j)$ and the ending time $en(r(z),j)$. Update the available time and the idle time of machine k. Go to step 3.

Step 2.4. Select a machine to append the current operation according to the following rules.

Step 2.4.1: Select the machine with a certain speed level on which the current operation can be completed at the earliest time. If there are more than one alternative, go to Step 2.4.2. Otherwise, go to step 2.4.5.

Step 2.4.2: Select the machine on which the processing time of the current operation is the shortest. If there are more than one alternative, go to step 2.4.3. Otherwise, go to step 2.4.5.

Step 2.4.3: Select the machine whose ending time is the earliest. If there are more than one alternative, go to step 2.4.4. Otherwise, go to step 2.4.5.

Step 2.4.4: Select the machine whose load up to now is the smallest. If there are more than one alternative, select randomly a machine and go to step 2.4.5.

Step 2.4.5: Append the operation $O_{r(z)j}$ at the end of machine k and set its beginning time $st(r(z),j)$ to be the ending time of the machine $(mach(k))$. Update its ending time $en(r(z),j)$ and the ending time $mach(k)$ of machine k. Go to step 3.

Step 3: Obtain the next gene, set $z = z + 1$, and return to step 2.1.

The pseudo code is shown as follows.

$TP = 0; TM = 0;$
for $e = 1 : n * n_i$
  $TP = \max_j (C_{ijk_{ij}q_{ij}})$
  $TM = \max_{h,l} (C_{hlk_{hl}q_{hl}})$
  Determine O_{ij}
  for $f = 1 : m_{ij}$
    //*calculate k, q, C_{ijkq}, Q_{ijkq}, Q_k with ITFR
    $C_{ijkq} - T_{ijkq} = \max(TP, TM)$
    if $TP < TM$
      for $c = 1 : num1 - 1$
        if $T_{ijkq} \geq TI$
          $C_{ijkq} = C_{ckq_c} + T_{ijkq}$
          $Q_{ijkq} = T_{ijkq} * P_{ijkq}$
          Break
        end if
      end for
    end if
    record the turning-on/off on machine k and compute the energy consumption before and after the operation O_{ij} is scheduled
  end for
  if (there exists an idle in the machine that can be inserted into the operation)
    decode according to step 2.3
  else
    decode according to step 2.4
  end if
  save k, q, C_{ijkq}, Q_{ijkq}
end for

TP: the completion time of the preceding operation of the same job;

TM: the current makespan on the same machine ;

TI : the idle time between two adjacent operations ;

$num1$: the number of operations already scheduled on machine k between TP and TM ;

$num2$: the number of operations already scheduled on machine k after operation O_{ij} is scheduled.

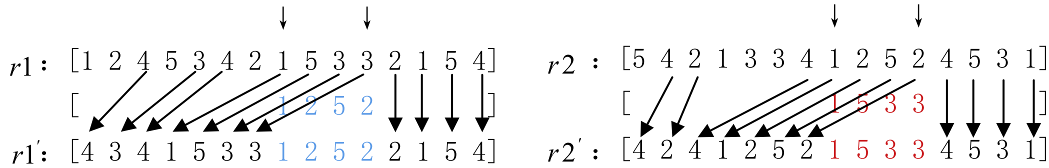

Fig. 4. Linear order crossover.

4.2.3. Crossover

Two chromosomes are crossed to generate offspring in order to explore some new solution space. We employ linear order crossover (LOX) (Akay and Yao, 2013). Select two positions randomly. Exchange the substring between the positions. Delete the genes beyond the positions which are inherited from the other parent. Insert the rest into an offspring from left to right skipping the inherited part.

For example, in Fig. 4, select position 8 and 11. Exchange the substring between position 8 and 11. For the first parent *r1*, the rest substring is [4 3 4 1 5 3 3 2 1 5 4]. Insert it into offspring 1 from left to right skipping the inherited part ‘1 2 5 2’. Similarly, the rest substring for the second parent is [4 2 4 1 2 5 2 4 5 3 1]. Insert it into offspring 2 from left to right skipping the inherited part ‘1 5 3 3’.

4.2.4. Mutation

Mutation is to exploit the solution space. Different mutations generate a neighbor in different directions. Four methods are designed to fully exploit the solution space. The swap and insertion mutation methods generate close neighbors while the inversion and displacement mutation methods generate distant neighbors. These four methods are randomly selected in each iteration.

(1) Swap method

In the traditional swap method, only two randomly selected genes are swapped. This might generate a very close neighbor especially for the large-scale problems. Hence, we perform the swap method several times, that is, $2*d$ genes are swapped where d is a coefficient. For example, e.g. in Fig. 5, when $d = 2$, random select four locations. The genes with the same color are swapped in turn. i.e. “1” and “4” are swapped, “5” and “3” are swapped.

(2) Insertion

The insertion method is to select randomly d genes and d positions. Insert the genes into the positions. When d is small, this method can generate a close neighbor with a small change. With d increases, a distant neighbor with a bigger change is generated. An example is shown in Fig. 6, select randomly two genes 1 and 4 and two positions 6 and 11. First insert the gene 1 into the position 6 and then insert the gene 4 into the position 11, thus a new neighbor is generated.

(3) Inversion

The inversion method is to inverse the genes between two randomly selected positions. As shown in Fig. 7, position 4 and 8 are randomly selected. The genes between the two positions ‘5 3 4 2 1’ is inversed to be ‘1 2 4 3 5’.

(4) Displacement

The displacement method is to displace two randomly selected sub-sequences. As shown in Fig. 8 select three positions 4, 8 and 11 and then displace the sub-sequence “5 3 4 2 1” and “5 3 3” to get a new neighbor.

4.2.5. Determining the non-dominated level and the crowding distance

In NSGA-II, the non-dominated level represents the dominance among individuals in the population. The crowding distance represents the sum of the relative distances for all the objectives between two individuals in the same dominance level (Deb et al., 2002). The non-dominance level and the crowding distance determine whether an individual can be selected into the next generation. The parent population R and the offspring population R'' are combined as a new population R'''. First compute the non-dominated level and the crowding distance of each individual,

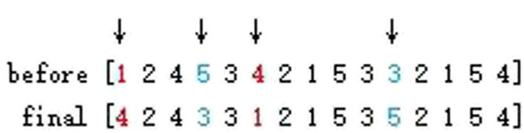

Fig. 5. Swap.

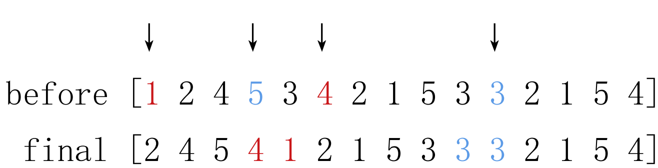

Fig. 6. Insert.

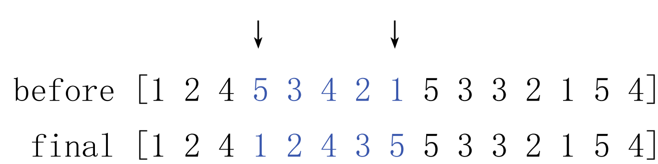

Fig. 7. Inversion.

Fig. 8. Displacement.

and then determine whether the individual is selected into the next generation.

The algorithm to compute the non-dominated level is as follows.

Step 1. Initialize a dominance table zp with $2N$ rows and $2N$ columns. $zp(x,y)$ represents the dominance between the individual x and the individual y. If the individual x dominates the individual y, $zp(x,y) = -1$; If the individual y dominates the individual x,$zp(x,y) = 1$; otherwise, $zp(x,y) = 0$.
Step 2. After the dominance table is inserted, search each row in turn. Initialize $w = 1$. If the entries in row x are all less or equal to 0, it means that there is no individual which dominates the individual x, the non-dominated level for row x is w.After the first run search, delete the rows and columns temporarily whose levels has been determined. Update $w = w + 1$. Repeat the procedure until the non-dominated level for all the individuals are determined.

The pseudo code is shown as follows.

```
for  x = 1 : 2N
        for  y = (x + 1) : 2N
        if  x ≺ y , zp(x, y) = -1 ; If  y ≺ x , zp(x, y) = 1 ; else, zp(x, y) = 0
        end for
end for
w = 0
while (control table is not empty)
        w = w + 1
        N' = size(zp,1)
        for  x = 1: N'
        if (there is no number greater than 0 in line  x  )
              w_x = w
        end if
        end for
    delete line  x  in  zp
end while
```

The crowding distance determines the quality of the individuals in the same dominance level. The individual crowding distance in one dominance level is computed as follows.

Step 1: For all the individuals in the same non-dominated level, execute the step 1.1, 1.2 and 1.3 and then go to step 2;

Step 1.1: Sort the individuals according to the objective value;
Step 1.2: For the non-edge individual x, compute the crowding distance with Eq. (36), where $K_B(x)$ represents the *B-th* objective value for individual x, u_{Bx} is the crowding distance for the *B-th* objective of individual x and $K_{B\max}$ is the maximum value of the *B-th* objective.

$$u_{Bx} = (K_B(x+1) - K_B(x-1))/K_{B\max} \tag{36}$$

Step 1.3:For the edge individuals, the crowding distance is defined as the maximum crowding distance of the non-edge individuals plus a constant to ensure that the crowding distance is larger than that of the non-edge individuals,

Step 2: The crowding distance of an individual x is the sum of the crowding distance of the different objectives for one individual, i.e.$v_x = \sum_{B=1}^{\varpi} u_{Bx}$;
Step 3: After the individuals in the same non-dominated level are computed, repeat the above procedure until all of the individual are dealt with.

The pseudo code is shown as follows.

```
for  ii = 1:  max(w)
        for  B = 1:  ϖ
        sort(K_B)
        if x is non-edge individuals
              u_Bx = (K_B(x + 1) − K_B(x − 1))/K_Bmax
        else
              u_Bx = max(u_B) + φ
        end if
        end for
        for  x = 1:  size(K_B,1)
        v_x = Σ_{B=1}^{ϖ} u_Bx
        end for
end for
```

ϖ : the number of objective value; ϕ : a constant

4.2.6. Selection

Selection ensures the survival of the fittest. After the non-dominated level and the crowding distance of the individuals in the combined population R''' are determined, some individuals are selected into the next generation with the tournament selection. The procedure is as follows.

Step 1: If the number of the individuals with the minimal non-dominated level ($num3$) exceeds the population size N, select N individuals with the tournament selection method; otherwise, the individuals with the minimal non-dominated level are selected directly into the next generation. The rest are selected according to step 2;
Step 2: Select $num4$ individuals with the 2-size tournament selection. If the non-dominated levels for two individuals are different, select the one with a smaller non-dominated level; otherwise, select the one with a bigger crowding distance. The selected individuals will be reselected in step 3;
Step 3: Delete those individuals with the higher non-dominated level. If the non-dominated levels for two individuals are the same, delete the one with the less crowding distance. The number of the kept individuals is the difference between the population size and the number of the Pareto individuals.
Step 4: Combine the Pareto individuals and the individuals selected in step 3 to form the offspring population.

The pseudo code is as follows.

```
If num3 > N
    Select N individuals with the minimal non-dominated level with
     tournament method
else
    the num3 individuals with the minimal non-dominated level are
     selected directly into the next generation
    for x = 1 : 2 : (size(R''',1)-num3)
        select the individual according to the non-dominated level
    end for
    if num4 > N - num3
        delete the individuals according to the non-dominated level and
         crowding distance
    end if
end if
```

num3 : the number of Pareto solutions;

num4 : the number of select individual in step 2;

4.3. The time computational complexity

The time requirement of the algorithm is called the time complexity of the algorithm. It is an important index to evaluate the feasibility of the algorithm. Here, we analyze the computational complexity of each block first and then evaluate the computational complexity of the NSGA-II.

According to the green scheduling heuristic, the computational complexity of the green scheduling heuristic in the worst case is:$f_1 = o(2n^2\max\{\mathrm{n_i}\}^2 m + 9mn\max\{\mathrm{n_i}\} + 10n\max\{\mathrm{n_i}\} + 1)$.

The computational complexity of the crossover is:$f_2 = o(Nn\max\{\mathrm{n_i}\}//2 + 3\mathrm{N}/2)$.

The computational complexity of the mutation is:$f_3 = o(3Nn\max\{\mathrm{n_i}\} + N)$.

The computational complexity to determine the non-dominant level is:$f_4 = o(9N^2 + 3N + 1)$.

The computational complexity to compute the crowding distance is:$f_5 = o(\varpi N^2 + 2\varpi N + N^2 + N)$.

The computational complexity of the selection is:$f_6 = o(3N + 4)$.

Hence the computational complexity of the NSGA-II is:

$$o(1 + Nf_1 + S(f_2 + f_3 + Nf_1 + 1 + f_4 + f_5 + f_6) + 1) = o\left(2SNmn^2\max\{\mathrm{n_i}\}^2\right)$$

Hence the computational complexity of NSGA-II is $o(2SNmn^2\max\{\mathrm{n_i}\}^2)$ which means that the following factors influence the complexity of the NSGA-II. They are: the number of generation, the number of population, the number of machines, the number of jobs and the maximal operation number.

5. Case study

5.1. Design of experiments

5.1.1. Experiment setting

All the experiments were conducted in a desktop computer with an Intel Core i3-3240, 3.40 GHz CPU,4.00G RAM, Win7 64 OS, and Matlab©.

5.1.2. Data source

The benchmark instances from Brandimarte (1993) are employed as the test data. But to test the performance of the proposed algorithm, it is necessary to modify the instances. Since there is only a single speed level for each machine, a coefficient vector [1.5 1.2 1] is multiplied to extend the single speed level to three levels. The processing power and the idle power for each level are given in column 2–7 respectively in Table 4. Besides, the turning-on/off energy consumption, the standby power and the turning off threshold are listed in the last three columns.

5.1.3. Parameters setting

The Parameters for the NSGA-II are set as follows. The population R is initialized at random. The probability of crossover P_c and the probability of mutation P_m are the same as those in Zhang and Chiong (2016); In order to explore the sufficient solution space and meanwhile to avoid too long running time, the total iterations S is 5000 and the population size N is 100. β is the coefficient which is same as that in Zhang et al. (2012).

5.1.4. The aim

The aim of the case study is to test the performance of NSGA-II, to analyze the performance of the proposed green scheduling heuristic and to discuss the role of the energy-saving measures. The indicator is whether the NSGA-II can obtain a group of high quality Pareto solutions within a limited time.

Table 4
The power distribution for each machine.

	Level 1		Level 2		Level 3		Turning-On/off	Standby	H_k
	processing	idle	processing	idle	processing	idle			
Machine 1	1230	230	1510	320	2270	370	2600	20	7
Machine 2	1160	180	1500	280	1820	350	2530	22	8
Machine 3	1150	190	1390	300	1880	350	2560	25	6
Machine 4	1380	230	1920	330	2340	390	2740	30	7
Machine 5	1040	220	1500	310	2220	380	2640	25	6
Machine 6	1270	230	1560	270	2260	370	2600	27	8
Machine 7	1170	220	1510	300	2160	400	2570	22	7
Machine 8	1000	170	1210	290	1690	350	2550	20	8
Machine 9	1300	250	1770	320	2510	400	2860	30	7
Machine 10	1360	250	1960	310	2510	380	2840	28	8
Machine 11	1350	240	1850	340	2440	390	2800	22	7
Machine 12	1030	190	1480	280	1920	320	2630	21	6
Machine 13	1310	230	1860	310	2290	390	2860	28	8
Machine 14	1060	200	1450	300	1960	400	2760	29	8
Machine 15	1450	300	2090	350	2970	400	3050	30	8

5.2. The experiment results

5.2.1. Testing the benchmark results

The experiment results for the 10 benchmark instances (Brandimarte, 1993) are reported in Table 5. The second and third columns report the Pareto solution numbers and each Pareto solutions in term of $(C_{\max}, Q, G)$, respectively.

To compare the performance of the NSGA-II, we report the Pareto front for MK01 in Fig. 9. The makespan objective is on the horizontal x-axis, the energy consumption objective is on the y-axis and the total numbers of turning-on/off machines is on the z-axis. The green star represents the non-Pareto solution and the red circle represents the Pareto solutions. As can be seen from Fig. 9, the solution space is quite large, in which the makespan is between 40 and 70, the energy consumption is between 6 and 9, and the total numbers of turning-on/off mahcines is between 5 and 9. Since the total numbers of turning-on/off mahcines is an integer, all the solutions are distributed into 5 layers along the z-axis. The Pareto solutions are distributed into 3 out of 5 layers. There is no Pareto solutions whose total numbers of turning-on/off mahcines equals 8 or 9. This result indicates that the total numbers of turning-on/off mahcines is not the more the better and too many total numbers of turning-on/off mahcines cannot balance the makespan the energy consumption very well. A suitable total numbers of turning-on/off mahcines, e.g. 7, can minimize the makespan and the energy consumption simultaneously. On the other hand, the number of Pareto solutions is enough and the Pareto front is well-distributed. In all, this group experiment proves that the green scheduling heuristic can balance the makespan, the energy

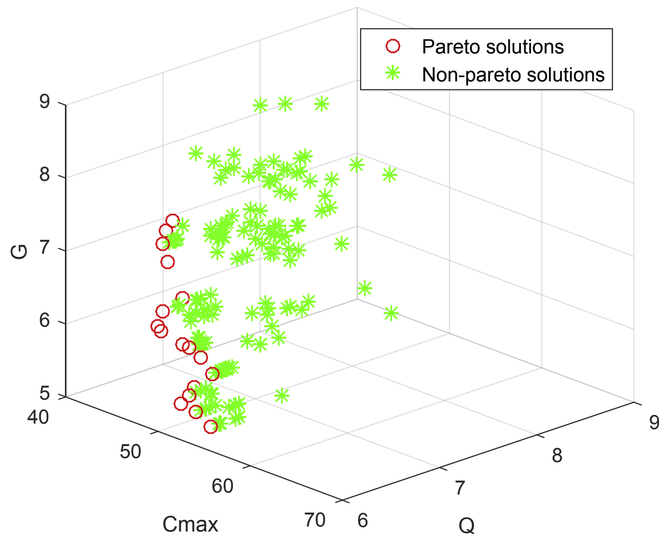

Fig. 9. Pareto solutions.

Table 5
The results for the benchmark instances.

	Pareto numbers	Pareto solutions
MK01	17	(44,6.62,7), (47,6.39,7), (52,6.24,6), (42,7.00,6), (46,6.42,6), (49,6.34,6), (49,6.48,5), (47,6.52,5), (45,6.85,5), (44,7.13,5), (46,6.70,5), (52,6.35,5), (50,6.32,6), (45,6.48,6), (42,6.85,7), (41,7.01,7), (43,6.72,6)
MK02	8	(29,5.87,6), (33,5.74,6), (30,5.82,6), (31,5.81,6), (33,5.68,7), (28,6.03,6), (31,5.74,7), (32,5.79,6)
MK03	12	(213,39.35,11), (213,38.83,12), (204,38.82,13), (204,37.95,14), (213,38.34,13), (204,37.54,16), (204,42.33,9), (204,37.59,15), (204,41.93,10), (204,39.37,12), (204,40.86,11), (213,41.44,10)
MK04	26	(67,14.57,10), (71,14.36,8), (70,14.38,9), (72,14.94,7), (81,13.80,7), (70,15.41,7), (67,14.63,9), (74,13.92,8), (69,14.48,8), (67,14.38,11), (71,14.25,9), (80,13.82,7), (67,14.71,8), (71,15.12,7), (73,14.14,7), (81,13.68,8), (68,14.48,10), (76,13.86,9), (79,13.77,9), (68,14.56,9), (73,13.95,9), (74,14.01,7), (73,13.97,8), (69,14.32,10), (80,13.71,8), (75,13.89,7)
MK05	11	(181,28.08,4), (186,27.84,4), (208,27.50,4), (191,27.65,4), (184,27.86,4), (189,27.78,4), (190,27.75,4), (205,27.54,4), (178,28.72,4), (196,27.64,4), (197,27.63,4)
MK06	50	(98,14.61,15), (73,16.84,10), (77,16.02,10), (69,17.34,11), (85,14.78,13), (67,16.88,13), (70,16.33,13), (71,16.52,11), (73,16.11,12), (104,14.40,13), (78,15.69,10), (78,15.68,11), (81,15.58,8), (73,15.90,13), (87,14.65,10), (69,16.99,12), (103,14.48,14), (77,16.10,9), (85,14.82,11), (87,14.69,9), (103,14.54,13), (74,16.01,12), (75,16.16,10), (85,14.86,10), (98,14.59,16), (79,15.55,9), (104,14.38,14), (84,14.82,13), (86,14.83,9), (69,16.56,13), (71,16.03,13), (82,15.30,9), (83,15.19,9), (72,17.12,10), (84,14.96,12), (77,15.68,12), (83,15.01,13), (80,15.70,8), (73,16.30,11), (81,15.38,10), (70,16.40,12), (87,14.65,11), (74,16.45,10), (67,16.40,14), (86,14.74,10), (70,17.08,11), (78,15.76,9), (83,15.05,12), (69,16.09,15), (78,15.77,8)
MK07	8	(153,27.87,5), (145,28.51,5), (156,27.64,5), (148,28.33,5), (161,26.84,5), (157,27.13,5), (165,26.73,5), (158,26.95,5)
MK08	68	(570,106.40,26), (593,104.83,29), (546,107.90,27), (533,108.22,26), (560,107.09,27), (526,108.80,35), (561,106.76,26), (530,109.09,29), (523,109.45,28), (587,105.53,28), (570,106.73,24), (526,108.91,34), (561,106.71,27), (550,109.94,23), (530,109.25,28), (579,106.11,28), (585,104.96,30), (526,109.31,33), (533,108.43,25), (584,105.57,28), (564,107.22,24), (557,107.14,26), (552,107.81,25), (568,106.63,26), (570,106.56,25), (562,107.24,24), (582,105.66,28), (557,107.42,24), (539,108.00,26), (569,106.58,25), (537,108.00,27), (530,110.68,25), (584,105.05,31), (531,109.13,28), (539,108.22,25), (555,107.72,28), (561,106.92,25), (587,104.89,29), (536,108.41,25), (532,110.62,25), (593,105.47,28), (532,109.18,26), (569,106.42,27), (532,108.75,36), (571,106.15,30), (557,107.13,27), (523,110.04,26), (561,106.71,28), (567,106.61,27), (574,105.64,29), (579,105.47,29), (531,109.03,29), (565,107.22,24), (569,106.42,26), (571,106.26,29), (576,106.66,24), (578,105.09,30), (575,105.52,32), (528,108.98,33), (570,106.38,30), (576,106.50,25), (574,105.53,30), (561,106.70,29), (536,108.57,24), (531,109.31,26), (568,106.79,25), (531,109.15,27), (557,107.25,25)
MK09	71	(323,100.97,18), (368,95.25,24), (332,98.71,20), (404,93.01,26), (416,93.93,19), (328,99.85,24), (416,92.62,21), (398,93.93,23), (336,98.08,15), (413,92.38,25), (364,95.53,22), (353,95.88,22), (416,91.46,22), (396,93.42,26), (429,91.44,23), (398,93.76,24), (332,99.58,19), (330,99.93,21), (413,92.09,26), (422,91.41,26), (328,100.23,22), (344,95.99,21), (370,94.84,22), (365,94.84,25), (363,95.17,26), (424,96.27,18), (330,99.49,22), (329,99.81,23), (338,97.29,17), (336,96.37,18), (413,93.07,23), (329,100.27,20), (408,93.24,23), (331,99.65,19), (415,92.18,24), (405,93.18,24), (422,91.17,28), (353,96.38,17), (327,99.69,26), (334,105.97,14), (332,105.59,16), (339,97.05,16), (429,95.93,18), (320,102.60,21), (334,105.95,15), (336,97.53,16), (324,100.14,24), (354,95.87,25), (401,93.74,24), (341,96.67,17), (405,93.28,23), (404,93.17,25), (329,100.57,19), (437,95.85,18), (334,102.73,16), (324,100.59,23), (415,92.33,22), (355,95.67,19), (327,100.70,22), (403,93.55,25), (377,94.37,25), (369,95.26,22), (415,92.68,21), (398,94.10,22), (398,93.60,25), (404,93.54,23), (324,100.78,17), (336,98.30,14), (413,92.59,24), (322,102.12,21), (333,97.45,19)
MK10	33	(267,83.70,20), (280,84.08,15), (272,83.60,22), (262,84.77,14), (264,84.32,22), (265,84.36,20), (267,84.15,19), (268,84.35,17), (282,83.35,21), (258,84.53,18), (286,83.03,21), (281,83.38,19), (280,83.72,18), (280,83.92,17), (243,85.90,15), (264,84.37,21), (246,84.96,12), (245,85.18,14), (242,87.42,15), (282,83.34,22), (252,84.84,13), (279,84.30,17), (279,84.41,16), (275,83.49,20), (261,84.49,18), (291,83.29,19), (249,84.89,15), (279,83.45,19), (267,84.39,18), (268,83.81,18), (252,84.55,15), (244,85.38,14), (281,83.63,18)

consumption and the total numbers of turning-on/off mahcines very well and can provide a set of Pareto solutions for decision makers. The decision makers can employ a multi-criteria decision method, such as analytic hierarchy process to choose an optimal compromise solution.

Fig. 10 is the Gantt chart of the first Pareto solution for instance MK01. Among them, the horizontal coordinate represents time, the vertical coordinate represents the machine index, and each block represents an operation. The number in the first row in a block is the job index, the number in the second row is an operation index and the color represents the speed level. For example, the box in upper left corner represent the first operation for job 10 is scheduled to be processed on machine 6 with speed level 2. The yellow block indicates the turning-off interval. E.g. the yellow block on machine 5 indicates that machine 5 is scheduled to be turned off from time 6–21 units. The reason why machine 5 isn't turned off at once when operation O_{51} finishes is because of the threshold H_k. According to the green scheduling heuristic, the operations on the bottleneck machine (machine 2 and 4, e.g.) are scheduled to be processed with the highest speed level in order to minimize the makespan objective; on the other hand, the operations on the non-bottleneck machines (machine 1 and 6, e.g.) are scheduled to be processed with the lower speed levels in order to save energy. From the scheduling result, we can see how the green scheduling heuristic balance the contradiction between the makespan and the energy consumption. The operations on the bottleneck machine are scheduled to be processed with the highest speed level to minimize the makespan. Hence, in order to save energy, the green scheduling heuristic selects a lower speed level for operations on the non-bottleneck machines and turn off the idle machines when necessary.

5.2.2. Evaluating the green scheduling heuristic

In order to study whether the green scheduling heuristic is reasonable, we replace the green scheduling heuristic with a single objective scheduling heuristic in the NSGA-II. The single objective scheduling heuristic is to decode the individual with the time or the energy objective in step 2.3 and 2.4 of the green scheduling heuristics, respectively. The results are shown in Table 6, and the Pareto solutions from the two single objective scheduling heuristics are shown in Fig. 11 and Fig. 12, respectively. The comparison of the three decoding methods is shown in Fig. 13. It can be seen that the Pareto solution with the two decoding methods is not better than that with the green scheduling heuristic. Quantitatively, the Pareto

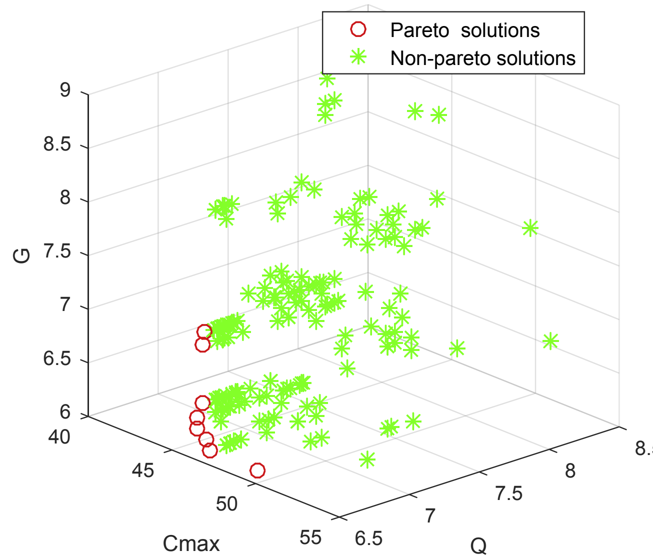

Fig. 11. Decode with time.

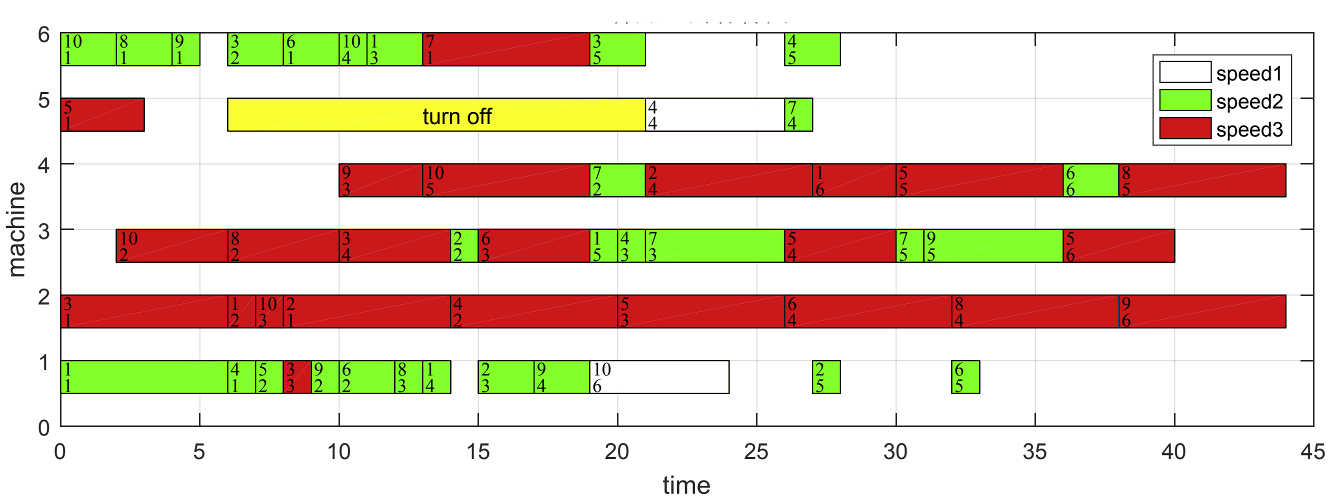

Fig. 10. Gantt chart for MK01.

Table 6
Results with the three decoding methods for MK01instance.

Decoding methods	Pareto number	Pareto solutions
Time-based	8	(42,7.07,6), (43,6.92,6), (49,6.63,6), (46,6.59,7), (44,6.81,6), (46,6.65,6), (45,6.73,7), (45,6.74,6)
Energy-based	14	(92,5.90,9), (86,5.92,6), (74,6.02,7), (80,5.93,7), (80,5.93,8), (81,5.96,6), (77,5.96,7), (87,5.92,6), (78,5.96,6), (78,5.94,7), (77,5.98,6), (89,5.91,8), (89,5.90,9), (81,5.91,7)
The green scheduling heuristic	17	(44,6.62,7), (47,6.39,7), (52,6.24,6), (42,7.00,6), (46,6.42,6), (49,6.34,6), (49,6.48,5), (47,6.52,5), (45,6.85,5), (44,7.13,5), (46,6.70,5), (52,6.35,5), (50,6.32,6), (45,6.48,6), (42,6.85,7), (41,7.01,7), (43,6.72,6)

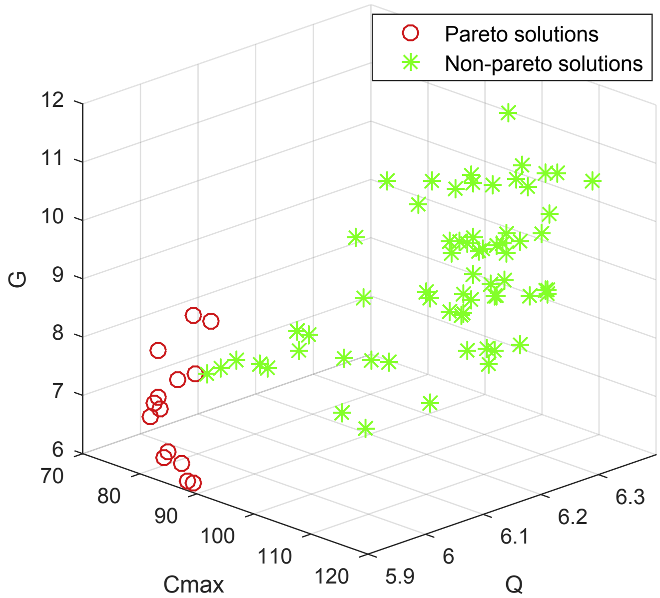

Fig. 12. Decode with energy.

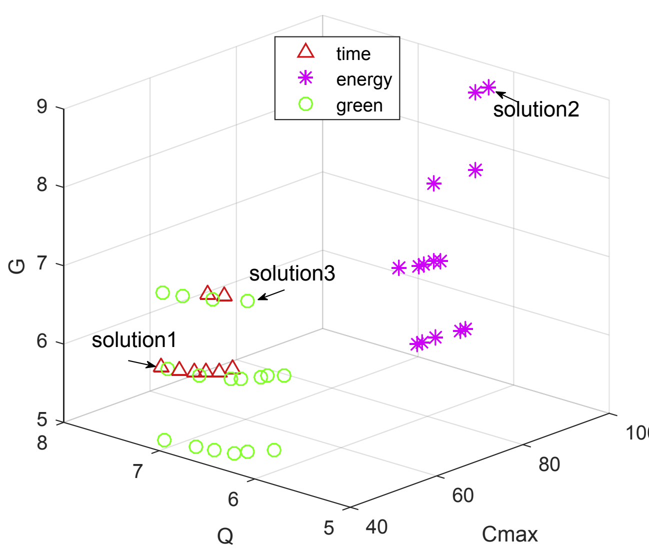

Fig. 13. Comparison among the three decoding methods.

solution numbers resulting from the green scheduling heuristic is more than that resulting from the other two scheduling heuristics. Qualitatively, the Pareto solutions resulting from the green scheduling heuristic dominate most of the Pareto solutions resulting from the other two scheduling heuristics. Although some Pareto solutions resulting from the green scheduling heuristic and the energy-based scheduling heuristic are dominated by each other, the Pareto solutions resulting from the green scheduling heuristic can save more energy. The makespan objectives for the Pareto solutions resulting from the energy-based scheduling heuristic are distributed between 70 and 100, while those from the green scheduling heuristic are distributed between 40 and 70. It does save energy, but too long makespan is not what most of the operation managers expect.

Further, we select three typical solutions to study the difference of the three decoding methods. Solution 1 comes from the time-based scheduling heuristic and its three objectives are (42, 7.07, 6). Solution 2 comes from the energy-based decoding method and its three objectives are (92,5.90,9). Solution 3 comes from the green scheduling heuristic and its three objective are (47,6.39,7). Fig. 14 is the histograms for comparing the total energy consumption (Q), the total processing energy consumption (Q_{ck}), the total energy consumption for keeping machines idle (Q_{dk}) and the total energy consumption for turning-on/off machines (Q_{gk}) respectively. Since the total energy consumption for keeping machines standby (Q_{fk}) is relatively too small, it isn't included into the comparison histogram. The orders for the total energy consumption and the processing energy consumption for the three solutions are the same, but the orders for the other two energy consumption (Q_{dk} and Q_{gk}) are different. Solution 2 has the lowest Q_{dk} but has the highest Q_{gk}. Solution 1 has the highest Q_{dk} but has the lowest Q_{gk}. This indicates that the Q_{dk} can be decreased by increasing the Q_{gk}. The Q_{dk} and the Q_{gk} are much less than the processing energy, so they has less impact on the total energy consumption. The processing energy is crucial to the total energy consumption. Different speed level consumes different amounts of energy, so one can see that the green scheduling heuristic (1) reduces the processing energy consumption by selecting a suitable speed level and (2) reduces the total energy consumption for keeping machines idle by selecting suitable machine. Therefore, the logic of the green scheduling heuristic is reasonable.

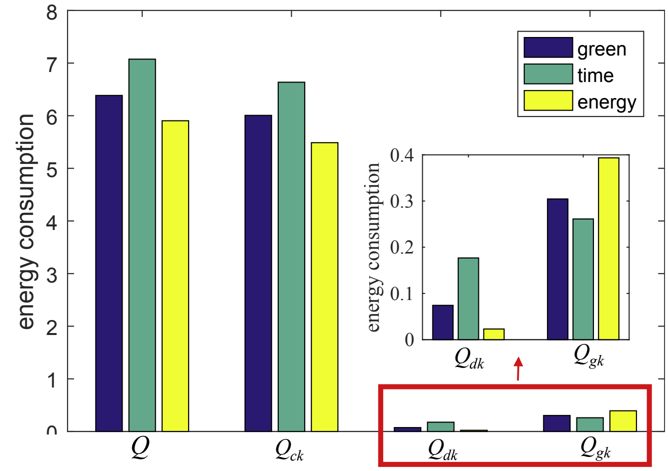

Fig. 14. The histograms for comparing the energy consumption.

5.2.3. Evaluating the role of the energy-saving measures

To evaluate the role of the energy-saving measure, we also compare the energy consumption difference before and after the energy-saving measure are employed. Take the first Pareto solution of MK08 as an example, the non-processing energy consumption before and after the turning-on/off scheduling is made is compared in Fig. 15. Similarly, the standby energy consumption Q_{fk} isn't also included into the comparison histogram. Without the energy-saving measure, the total energy consumption for keeping machines idle is 7.23 units. When the energy-saving measure is employed, it reduces to 1.76 units. The decreasing rate is more than 75%. The total energy consumption for turning-on/off machines changes from 0.04 to 1.15 and the total energy consumption for keep machines idle changes from 6.83 to 0.61 units. The energy-saving measure reduces the idle energy consumption greatly, although it increases the turning-on/off energy a little. Hence, it can be easily concluded that the energy-saving measures are very effective to save energy.

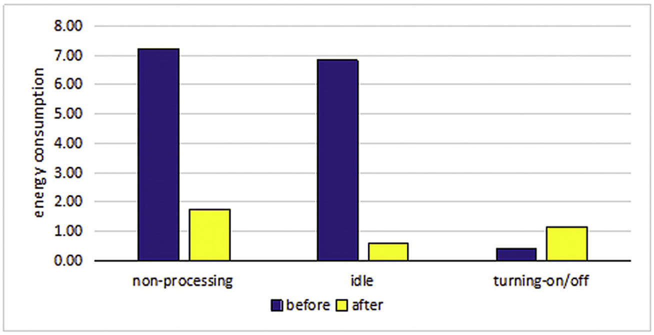

Fig. 15. The change of the non-processing energy consumption with or without energy-saving measures.

Table 7
Comparison of the results with NSGA-II and GSAA.

	Pareto numbers		$\min(C_{\max})$		$\min(Q)$		$\min(G)$	
	NSGA-II	GSAA	NSGA-II	GSAA	NSGA-II	GSAA	NSGA-II	GSAA
MK01	17	3	41	51	6.24	6.36	5	7
MK02	8	3	28	46	5.68	7.24	6	6
MK03	12	5	204	243	37.54	46.28	9	18
MK04	26	2	67	81	13.68	13.95	7	11
MK05	11	4	178	198	27.50	23.82	4	4
MK06	50	2	67	133	14.38	24.40	8	24
MK07	8	4	145	207	26.73	30.85	5	5
MK08	68	4	523	580	104.83	94.45	23	38
MK09	71	3	320	415	91.17	89.53	14	32
MK10	33	4	242	355	83.03	84.03	12	29

5.2.4. Comparing the proposed NSGA-II with the state of the art

In this section, the proposed NSGA-II is compared with the state of the art. Since the FJSP-ESM is a very novel problem, the related study is very few. To our best knowledge, we found that Dai et al. (2013) developed a Genetic-simulated annealing algorithm (GSAA) to solve the FJSP-ESMS. We run the GSAA to solve the benchmark instances from Brandimarte (1993). Table 7 reports the results, including the Pareto numbers, $\min(C_{\max})$, $\min(Q)$ and $\min(G)$. It is clear that the proposed NSGA-II outperforms the GSAA in most cases. More solution space is explored with the NSGA-II. The makespan and the total numbers of turning-on/off mahcines of the solutions are minimized fully. For most of instances except the instance MK05 and MK08, the NSGA-II minimized the energy consumption objective. Fig. 16 shows the comparison for the makespan and the energy consumption objectives. One can see that the results with the NSGA-II are better than those with the GSAA except the MK05 instance in term of the Pareto solution number and quality. The perfect performance of the NSGA-II can be concluded. Furthermore, Fig. 17 compares the Pareto solution numbers. NSGA-II generates more Pareto solutions than the GSAA does, especially for instance MK08 and MK09, which means that a decision maker has more candidates to select. It is always popular for those in charge of the operation management. To sum up, we can confident to conclude that the NSGA-II is better than the GSAA.

6. Conclusions

It is difficult to find the optimal or near-optimal solution for the FJSP-ESM. A multi-objective optimization model is built to optimize

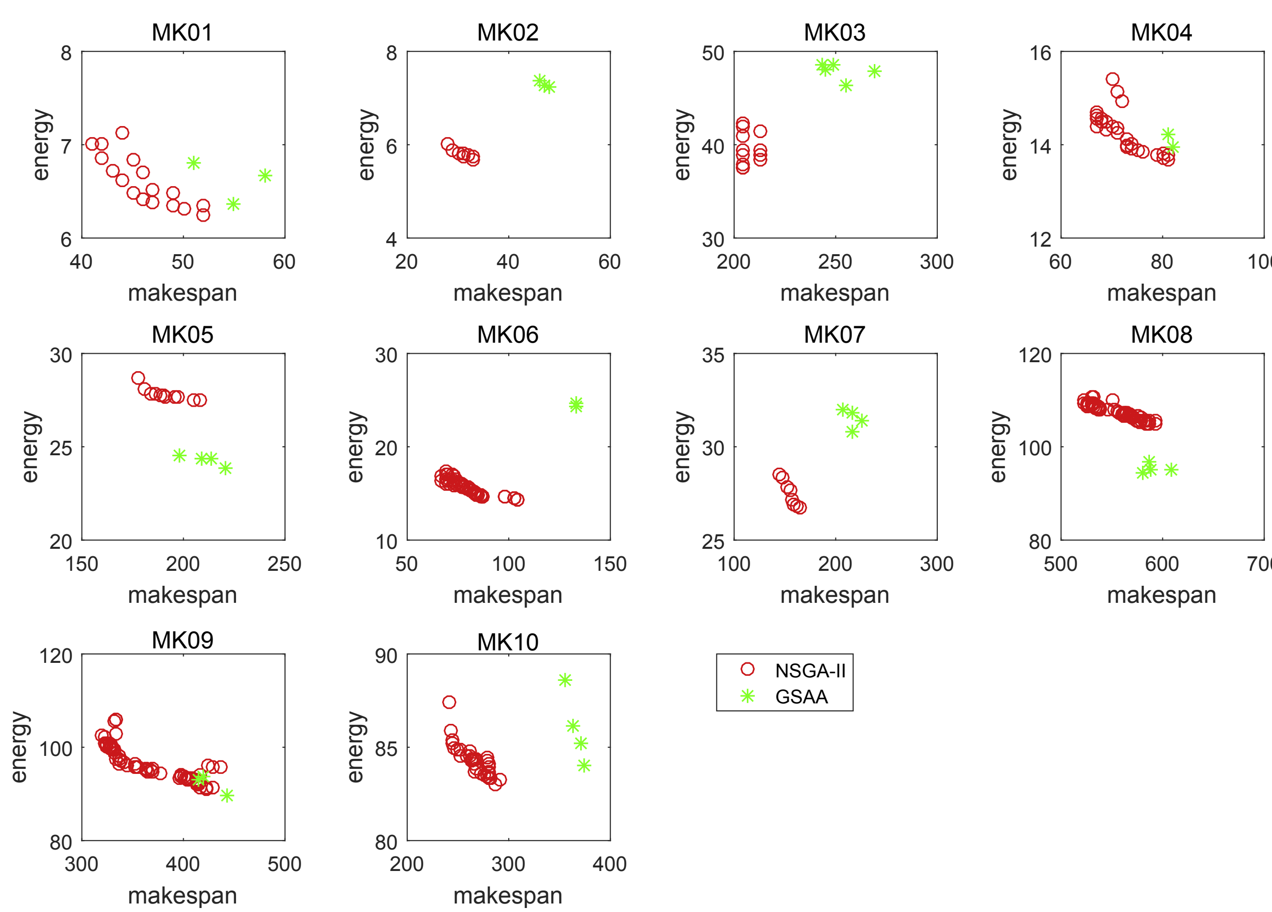

Fig. 16. The comparison in terms of the makespan and the energy consumption with NSGA-II and GSAA.

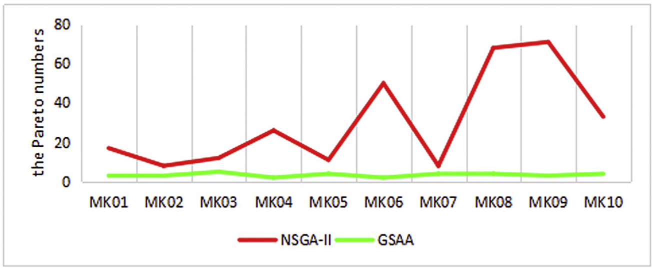

Fig. 17. The comparison in terms of the Pareto numbers with NSGA-II and GSAA.

the total energy consumption and the total numbers of turning-on/off mahcines besides the traditional makespan objectives. A NSGA-II algorithm is employed as the multi-objective optimization algorithm, in which a green scheduling heuristic is proposed to optimize the three objectives simultaneously. The results of the case study verify the performance of the green scheduling heuristic. By comparing with the state of the art, the NSGA-II algorithm is proved to outperform the GSAA.

Production scheduling considering energy-saving measures is important to realize energy-saving and emission-reduction, but the study is not enough especially on FJSP. This study focuses only on the static FJSP, but more dynamic factors will appear stochastically which will disturb the production. Besides, we only considered two energy-saving measures. In the future, it will be interesting to investigate on the following issues:

1) To integrate more practical constraints such as the random breakdown or rush orders into the optimization model,
2) To explore some new energy-saving measures, and
3) To improve the NSGA-II and the green scheduling heuristic.

Acknowledgements

We are very grateful to the editor and the three anonymous reviewers for their constructive comments which have helped improved this manuscript greatly. This work was supported by the National Natural Science Foundation of China under Grant [Grant No.51305024].

References

Akay, B., Yao, X., 2013. Recent advances in evolutionary algorithms for job shop scheduling. Autom. Sched. Plan. 191–224.
Brandimarte, P., 1993. Routing and scheduling in a flexible job shop by Tabu search. Ann. Oper. Res. 41, 157–183.
China Certification Center for Energy Conservation Product (CECP), 2002. Review standby power consumption of products at home and abroad. Energy Conserv. Environ. Prot. 38–40.
Che, A., Lv, K., Levner, E., Kats, V., 2015. Energy consumption minimization for single machine scheduling with bounded maximum tardiness. In: ICNSC 2015-2015 IEEE 12th Int. Conf. Networking, Sens. Control, pp. 146–150.
Che, A., Wu, X., Peng, J., et al., 2017. Energy-efficient bi-objective single-machine scheduling with power-down mechanism. Comput. Oper. Res. 172–183.
Chen, S., 2009. Engine or drag: can high energy consumption and CO2 emission drive the sustainable development of Chinese industry? Front. Econ. China 4, 548–571.
Cheng, J., Chu, F., Liu, M., et al., 2017. Bi-criteria single-machine batch scheduling with machine on/off switching under time-of-use tariffs. Comput. Ind. Eng. https://doi.org/10.1016/j.cie.2017.04.026.
Dai, M., Tang, D., Giret, A., Salido, M.A., Li, W.D., 2013. Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm. Robot. Comput. Integr. Manuf. 29, 418–429.
Dai, M., 2015. Research on Energy-efficient Process Planning and Scheduling. Doctor of Philosophy. Nanjing University of Aeronautics and Astronautics.
Deb, K., Member, A., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A Fast Elitist Multiobjective Genet. Algorithm 6, 182–197.
Diarra, D.C., Jeswiet, J., Astle, B., Gawel, D., 2010. Energy consumption and Co2 emissions for manufacturing compressed air systems. Trans. NAMRI/SME 38, 767–773.
Fang, K., Uhan, N., Zhao, F., Sutherland, J.W., 2011. A new approach to scheduling in manufacturing for power consumption and carbon footprint reduction. J. Manuf. Syst. 30, 234–240.
Fang, K., Uhan, N.A., Zhao, F., Sutherland, J.W., 2013. Flow shop scheduling with peak power consumption constraints. Ann. Oper. Res. 206, 115–145.
Gutowski, T., Murphy, C., Allen, D., Bauer, D., Bras, B., Piwonka, T., Sheng, P., Sutherland, J., Thurston, D., Wolff, E., 2005. Environmentally benign manufacturing: observations from Japan, Europe and the United States. J. Clean. Prod. 13, 1–17.
Haapala, K.R., Rivera, J.L., Sutherland, J.W., 2009. Reducing environmental impacts of steel product manufacturing. Trans. NAMRI/SME 37, 419–426.
He, Y., 2007. Study on the Optimal Scheduling Method of Machining Tasks for Green Manufacturing. Doctor of philosophy. Chongqing University.
Ho, N.B., Tay, J.C., 2004. GENACE : an efficient cultural algorithm solving the flexible job-shop problem. Congr. Evol. Comput. 2004. CEC2004 1759–1766.
Liu, F., Xu, Z.J., Dan, B., 1995. Energy Characteristics of Mechanical Manufacturing System and its Application. China Machine Press, Beijing.
Mansouri, S.A., Aktas, E., Besikci, U., 2016. Green scheduling of a two-machine flowshop: trade-off between makespan and energy consumption. Eur. J. Oper. Res. 248, 772–788.
Mouzon, G., Yildirim, M.B., Twomey, J., 2007. Operational methods for minimization of energy consumption of manufacturing equipment. Int. J. Prod. Res. 45, 4247–4271.
Nava, P., Jeswiet, J., Kim, I.Y., 2010. Calculation of carbon emissions in metal forming manufacturing processes with eco-benign lubrication. Trans. NAMRI/SME 38, 751–758.
Pach, C., Berger, T., Sallez, Y., Bonte, T., Adam, E., Trentesaux, D., 2014. Reactive and energy-aware scheduling of flexible manufacturing systems using potential fields. Comput. Ind. 65, 434–448.
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.
Stute, H., Limde, E., 1955. Uber Antriebe und Wirkungsgrade bei Werkzeugmaschinen. Ind. Anz. 5–8.
The 13th Five-Year Plan for Economic and Social Development of the People's Republic of China, 2016. People's Daily, pp. 1–78.
U.S. Energy Information Administration (EIA), 2010. Annual energy review 2009. Energy. https://doi.org/10.1021/es103270a.
Wu, X., Wu, S., 2017. An elitist quantum-inspired evolutionary algorithm for the flexible job-shop scheduling problem. J. Intell. Manuf. 28 (6), 1441–1457.
Yildirim, M.B., Mouzon, G., 2008. A framework to minimize total energy consumption and total tardiness on a single machine. Int. J. Sustain Eng. 2, 105–116.
Zhang, L., Li, X., Gao, L., Zhang, G., Wen, X., 2012. Dynamic scheduling model in FMS by considering energy consumption and schedule efficiency. In: Proc. 2012 IEEE 16th Int. Conf. Comput. Support. Coop. Work Des. CSCWD 2012, pp. 719–724.
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, Z., Tang, R., Peng, T., Tao, L., Jia, S., 2015. A method for minimizing the energy consumption of machining system: integration of process planning and scheduling. J. Clean. Prod. 137, 1647–1662.