Energy- and labor-aware flexible job shop scheduling under dynamic electricity pricing: A many-objective optimization investigation

Xu Gong[1],*, Toon De Pessemier[1], Luc Martens[1], Wout Joseph[1]

[1]*Department of Information Technology, Ghent University/imec, Technologiepark 15, 9052 Ghent, Belgium*
* E-mail*:* xu.gong@outlook.com. Tel.: +86 183 2341 0408. Fax.: +32 9 33 14899.

Abstract

Energy-aware production scheduling is a promising way to adapt the factories' energy consumption behavior to the volatile electricity prices in the demand response initiative of smart grids. However, it may not be economical by simply scheduling production loads to the periods with lower electricity prices, as these periods often have higher labor wage, e.g., nights and weekends. Based on this gap, this paper proposes a many-objective integrated energy- and labor-aware flexible job shop scheduling model. Many objectives refer to the number of optimization objectives surpasses three (i.e., five objectives: makespan, total energy cost, total labor cost, maximal workload, and total workload), whereas the existing energy-aware production scheduling research is limited within three objectives. To enable energy awareness in the conventional production scheduling algorithms, a state-based shop floor wide energy model is proposed. To enable labor awareness, the number and type of human workers are matched to the scheduled production loads, with varying labor wage over shifts. As one of the most complex shop floor configurations, the partial flexible job shop further considers job recirculation and operation sequence-dependent machine setup times. The recently-proposed nondominated sorting genetic algorithm-III (NSGA-III) is tailored for this many-objective optimization problem (MaOP), including scheduling solution encoding and decoding, crossover, mutation, and solution evaluation using the energy- and labor-aware discrete-event simulation framework. Through numerical experiments under real-time pricing (RTP) and time-of-use pricing (ToUP), insights are statistically obtained on the relation among these five production objectives; the effectiveness and efficiency of NSGA-III in solving a MaOP are also demonstrated. This proposed scheduling method can be used to automated and enhance the decision making of factory managers in jointly allocating machine, human worker, and energy resources on the shop floor, such that the production cost is minimized even under time-varying electricity and labor prices.

Keywords: Sustainable production scheduling; labor scheduling; demand response; many-objective optimization; evolutionary computation

1. Introduction

The adaptation of end users' consumption behaviors to volatile electricity prices is part of the initiative in smart grids, called price-based demand response (DR) [1]. In price-based DR, electricity users are encouraged to perform such adaption for energy cost reduction and grid stabilization. A common approach to perform DR on the shop floor is to integrate energy awareness to conventional production scheduling algorithms. The energy awareness is enabled by empirical energy modeling based on the measured machine power consumption data [2]. With energy-aware production scheduling, the cost for executing production tasks on the shop floor can be minimized under dynamic electricity prices, while the conventional constraints are satisfied [2-4].

However, compared to residential DR, industrial DR is intrinsically more complex for several reasons. Firstly, there are more constraints in modeling the production activities, due to the interdependency of machines, a common lack of detailed energy data of each machine, the due time which is often a hard constraint, etc. Secondly, factory managers are less open to production load shifting or reorganization, as this affects the production and the economic benefits of factories. Thirdly, besides machines, human workers are usually involved in DR, increasing the modeling complexity and the economic impact. Consequently, current DR research focuses on household applications, while fewer studies investigate industrial DR [5].

Kim, et al. [6] mentioned that the implementation of industrial DR by production scheduling should consider the potentially increasing labor cost. The intrinsic trade-off relation between the energy and labor costs was pointed out in [7]: the lower-priced periods for the electricity consumption, e.g., nights and weekends, are usually higher-priced periods for the labor. This trade-off was further statistically quantified in an extensive sensitivity analysis with empirical data from a Belgium plastic bottle manufacturer [8]. The energy cost and labor cost were recommended to be jointly considered in a production scheduling algorithm. Despite this highlight, there is still little investigation on integrated energy- and labor-production scheduling, compared to the extensive recent studies on energy-aware production scheduling.

A flexible job-shop is one of the most complex shop floor configurations for production [9]. A flexible job shop scheduling problem (FJSSP) can be further classified into two categories: (1) full FJSSP, and (2) partial FJSSP. In a full FJSSP, each operation can be performed by all machines; while in a partial FJSSP, each operation has its own set of capable machines, not necessarily all machines. Although both type of problems are NP-complete, a partial FJSSP is more complex than a full FJSSP [10]. The complexity of a partial FJSSP will even increase by considering other practical factors. For instance, if job recirculation is enabled, a job may return to the machine that has processed it in an early stage. This is usually the case for semiconductor manufacturing, where the wafer is fabricated one layer after another [9].

A FJSSP should rapidly be solved by an optimization algorithm, as production scheduling is time-sensitive compared to the long-term production planning. An evolutionary algorithm (EA) intrinsically complies with this requirement, since it aims to fast obtain a high-quality optimization solution or solution set, without guaranteeing the exact optimum. On the one hand, an integrated energy- and labor-aware FJSSP calls for multiple optimization objectives. This is not only because the integration of energy and labor awareness triggers more production objectives, but also due to the fact that production objectives often vary on the shop floor depending on the preference of a specific decision marker or the compromised preference of multiple decision makers [11]. Therefore, the number of optimization objectives easily exceeds three, the existing multi-objective FJSSP research is limited within three objectives.

On the other hand, many-objective optimization has become an active research topic in multiobjective evolutionary algorithms (MOEAs). This is due to the new challenges faced by EAs [12] in (1) the search for Pareto optimal solutions, (2) the approximation of the entire Pareto front, (3) the presentation of obtained solutions, (4) the choice of a single final solution, and (5) the evaluation of search algorithms. Therefore, many-objective optimization problems (MaOPs) have been defined to denote the multiobjective optimization problems (MOPs) with over three objectives [13, 14]. In a many-objective retrofit planning problem [15], the widely-used NSGA-II was proven to have poor convergence and diversity performance. While the many-objective optimization research is focused on designing a generic EA that can be widely used without any deep domain knowledge, there is little research on tailoring an EA for a real MaOP, e.g., integrated energy- and labor-aware FJSSP.

This paper fills these gaps in the following aspects. (1) Compared to existing FJSSPs, energy and labor awareness are additionally modeled and integrated; job recirculation, operation sequence-dependent machine setup times, and partial flexible job shop are considered in the proposed model, named EL-FJSSP. Though the EL-FJSSP becomes more complex, it is more practical and flexible for industrial applications, aiming to help factories to additionally reduce the energy and labor costs under dynamic electricity pricing. (2) Discrete-event simulation (DES) is used to build a digital twin of the energy- and labor-aware flexible job shop. This not only enables flexible modeling of a complex production problem on the shop floor, but also facilitates fast yet accurate calculation of all the optimization objectives. (3) For the first time, the nondominated sorting genetic algorithm-III (NSGA-III) is tailored for many-objective production scheduling. (4) Through numerical experiments under different dynamic electricity pricing schemes, the underlying relations among the important production objectives are quantitatively revealed, i.e., makespan, total energy cost, total labor cost, maximal workload, and total workload. The effectiveness and efficiency of tailoring a NSGA-III in solving a many-objective EL-FJSSP are demonstrated.

The remainder of this paper is organized as follows. Sect. 2 gives the literature review on energy-aware (flexible) job shop scheduling. Sect. 3 describes the proposed EL-FJSSP, with a focus on how to integrate energy and labor awareness to a conventional FJSSP. Sect. 4 introduces the tailored NSGA-III. Sect. 5 presents the numerical experiments. Sect. 6 draws the conclusion.

2. Literature Review on Energy-Aware (Flexible) Job Shop Scheduling

Table 1 analyzes the recent research on energy-aware (flexible) job shop scheduling in four aspects: energy model, labor model, number of objectives as well as the specific objectives, and optimization method. Generally, the gaps lie in little investigation of dynamic energy prices and labor in the scheduling model, and a small number of objectives.

Regarding the energy model, only a few studies use complete power states (including startup and shutdown) to mimic the dynamic energy consumption behavior of machines. May, et al. [16] propose a set of power states, which are mapped to machine operation states. These power states include off, idle, standby, setup, and working, while ramp-up and ramp-down are modeled as the transitions

Table 1
Feature analysis of recent literature on energy-aware job shop scheduling methods

Literature	Energy model		Labor model		Number of objectives	Objectives	Optimization method
	Power state	Energy price	Shift	Personnel			
Dai, et al. [19]	No	No	No	No	Two	Energy consumption, makespan	Metaheuristic
Moon and Park [20]	No	Yes	No	No	One	Production cost	MIP + CP
Firas and Denis [21]	Partial	No	No	No	Three	Makespan, maximal workload, total workload	Metaheuristic
Garcia-Santiago, et al. [22]	Partial	No	No	No	One	Energy consumption	Metaheuristic
He, et al. [23]	No	No	No	No	Two	Makespan, energy consumption	Metaheuristic
Kemmoé, et al. [24]	Partial	No	No	No	One	Makespan	LP
Liu, et al. [25]	Partial	Yes	No	No	Three	Total weighted tardiness, total non-processing electricity consumption, electricity cost	Metaheuristic
May, et al. [16]	Yes	No	No	No	Two	Makespan, total energy consumption/worthless energy consumption	Metaheuristic
Stock and Seliger [17]	Yes	No	No	No	Two	Energy consumption, peak power loads	Metaheuristic
Tang and Dai [26]	Partial	No	No	No	One	Makespan	Metaheuristic
Liu, et al. [18]	Yes	No	No	No	Two	Total weighted tardiness, total non-processing electricity consumption	Metaheuristic
Salido, et al. [27]	No	No	No	No	Two	Makespan, energy consumption	Metaheuristic
Xu, et al. [28]	Partial	No	No	No	Five	Total time, total cost, material consumption, energy consumption, product quality	Metaheuristic
Zhang and Chiong [3]	Partial	No	No	No	Two	Total weighted tardiness, energy consumption	Metaheuristic
Giglio, et al. [29]	No	No	No	No	One	Total cost	Heuristic
Kim, et al. [6]	Partial	Yes	Yes	Yes	One	Total energy cost	Metaheuristic
Mokhtari and Hasani [30]	No	No	No	No	Three	Total completion time, total availability of the system, energy consumption	Metaheuristic
Salido, et al. [31]	No	No	No	No	One	Weighted sum of makespan and energy consumption	Metaheuristic
Yin, et al. [32]	No	No	No	No	Three	Makespan, total energy consumption, noise emission	Metaheuristic
Zhang, et al. [33]	Yes	No	No	No	One	Total energy consumption	Metaheuristic + AI
Zhang, et al. [4]	No	No	No	No	Three	Makespan, total workload, total energy consumption	Game theory

between off and idle. In [17], four states are considered: process, idle, standby, and switch on/off. Compared to the idle state, the standby state is not ready for operations, since required components stay powered off, e.g., auxiliary systems. Liu, et al. [18] consider a power input model for a machine and turn off/on states. This power input model encompasses idle, switch, and cutting states, where the switch state is a transition to runtime mode. Besides processing and standby states, startup and shutdown states are considered in [33]. The duration of the latter two states enables the switching off-on decision: a machine is switched off if the duration between two adjacent operations is longer than the breakeven duration (i.e., energy consumption of shutdown and startup divided by standby power).

Although studies with partial power states generally consider idle and processing states, they ignore some basic power states, such as shutdown [21, 22] as well as startup and shutdown states [26]. The investigations that neglect power states often classify the total machine energy consumption into unload and cutting energy [19, 23]. The unload power corresponds to activities prepared for processing, such as loading, unloading, positioning, and changing cutting tools. The cutting power corresponds to actual cutting operations. However, as highlighted in [23], this energy modeling method depends on specific machining processes, hindering its general application.

As exhibited in Table 1, even fewer studies consider dynamic energy prices, despite the economic impact of converting the energy consumption to the energy cost [2]. For production during a Rolling Blackout policy [25], the electricity price is more expensive, due to private power generation which fills the shortage of public power supply. In time-of-use pricing (ToUP) and critical-peaking pricing (CPP), the electricity price in a day may have several levels. The difference between levels is much larger in CPP than in ToUP [20]. Besides ToUP, real-time pricing (RTP) is considered in [6], where the electricity price varies more often than ToUP, e.g., every hour in the next day [2]. A common attempt of these studies is to schedule production loads such that the incurred energy cost is minimized under volatile electricity prices, while satisfying the defined production constraints.

Table 1 also reveals a lack of relevant studies on the labor aspect, although machines commonly interact with human workers in practice and an optimal match between these two types of resources is required for economical production. Consideration of human workers requires the fragmentation of a continuous scheduling horizon into discrete labor shifts, which may be further separated by periods forbidding production, e.g., weekends and holidays. Therefore, this needs novel modeling and effective optimization with more constraints. Garcia-Santiago, et al. [22] mentioned that a human operator is required to change the ancillary part of a machine upon a change of product type, but they did not investigate how to economically match machine and human resources. The only relevant literature that considers the labor aspect is [6]. Nevertheless, simple scenario comparisons were performed on whether increasing production and operators in the day or in the night is economical. No hint was given on how to integrate labor in an energy-aware (flexible) job shop scheduling model [6].

In addition, Table 1 demonstrates that all the literature except [28] is limited within three optimization objectives. Despite up to five objectives in Xu, et al. [28], many-objective optimization was not explicitly investigated. A common objective in these investigations is makespan, as it not only directly measures the time to complete a set of jobs, but also is linked to other important production metrics, e.g., earliness/tardiness of finished jobs and workload. The energy-related objectives include minimization of total energy consumption to process a set of jobs [3, 16, 19, 22, 28, 33], non-processing energy [16, 25], peak power consumption [17], and energy cost [20, 25].

Concerning the optimization method, most studies used metaheuristics (Table 1). A genetic algorithm (GA) [19, 21, 27, 32] as well as its variants NSGA-II [18, 25] and memetic algorithm (MA) [3, 31] are the most prevalent among these metaheuristics. Besides, Xu, et al. [28] used a bee algorithm, Garcia-Santiago, et al. [22] utilized a harmony search algorithm, and [23] employed a nested partitions algorithm. Furthermore, two metaheuristics are often hybridized for enhanced search capability in the solution space. For instance, a GA was hybridized with a simulated annealing algorithm in [26, 30], an artificial immune algorithm was hybridized with a simulated annealing in [17], and NSGA-II was hybridized with SPEA-II in [16]. One important reason for this prevalent usage of metaheuristics is that production scheduling on the shop floor intrinsically requires fast yet high-quality decision making [22]. This also explains why optimization by a standard solver [20, 24] or a heuristic [29] is not extensively used, of which the computation time and tractability are very sensitive to additional constraints and an increasing problem size.

3. Problem modeling

The proposed problem is to perform energy- and labor-aware flexible job shop scheduling under dynamic electricity prices (EL-FJSSP). All jobs have a common release time and must be finished before a common due time. Labor shifts are considered while job and changeover split are allowed, such that a job/changeover may be split into multiple subparts by labor-forbidden periods, e.g., weekends and holidays. Job recirculation is allowed, i.e., a job may revisit the same machine even if it has been processed once by this machine. Five objectives need to be simultaneously optimized: makespan (C_{max}), total energy cost (TEC), total labor cost (TLC), maximal machine workload (MWL), and total machine workload (TWL), as defined in Eq. (1). These five objectives are chosen because TEC and TLC are two necessary metrics to quantify the impact of integrating energy and labor awareness to a production scheduling problem, while C_{max}, MWL, and TWL are three conventional performance metrics [34] indicating the overall production time, the maximal workload of a machine, and the overall workload of all machines, respectively.

$$\min(C_{\max}, TEC, TLC, MWL, TWL) \tag{1}$$

3.1 Objectives

C_{max} is defined as the time when the last job is completed:

$$C_{\max} = \max(C_1,\ C_2,\ ...,\ C_{\mathrm{n}}) \tag{2}$$

where C_j is the completion time of job *j*. C_{max} is closely related to the throughput of a production system, while throughput maximization is of the utmost importance and managers are often measured how well they do so.

TEC comprises the energy cost of all machines for processing operations of all jobs $(\sum_{i=1}^{m} ECO_i)$, machine changeover/setup $(\sum_{i=1}^{m} ECC_i)$ and idling between adjacent operations $(\sum_{i=1}^{m} ECI_i)$.

$$TEC = \sum_{i=1}^{m} (ECO_i + ECC_i + ECI_i) \tag{3}$$

TLC depends on the assigned labor shifts for a schedule and the type of human workers in each shift. It comprises the labor cost of all human workers who are involved in these shifts. It is calculated by Eq. (4). The binary variable θ_{sh}^{pt} indicates whether a personnel type (*pt*) will be included in a shift (*sh*) according to the power states assigned to the duration of this shift. A *pt* is identified by the skill that is linked to the worker. The number of workers in type *pt* is indicated by N_{sh}^{pt}.

$$TLC = \sum_{i=1}^{m} \sum_{sh=ST}^{DT} \sum_{pt \in PT} \theta_{sh}^{pt} \cdot W_{sh}^{pt} \cdot N_{sh}^{pt} \tag{4}$$

MWL and *TWL* are two workload-related objectives for a (flexible) job shop [21, 35]. *MWL* indicates the maximum working time spent on any machine, as formulated in Eq. (5). *TWL* measures the total workload of machines, which represents the total working time of all machines, as described in Eq. (6).

$$MWL = \max_{1 \le i \le m}(W_i) \tag{5}$$

$$TWL = \sum_{i=1}^{m} W_i \tag{6}$$

3.2 Total energy cost

The energy cost for all operations on machine *i* (ECO_i) is accumulated according to the operation sequence of machine *i* (π_i), where π_i contains a sequence of indexes (*i, j, k, l*) of these operations. As defined in Eq. (7), the energy consumption for processing operation (*i, j, k, l*) is mapped to the corresponding electricity pricing slots via the time slot indicator β_{ts}, such that the energy cost for processing operation (*i, j, k, l*) is accumulated over these slots. The production-prohibited period (e.g., weekend and holiday) is integrated due to the consideration of labor. If it exists ($\lambda_{\mathrm{wp}} = 1$), an operation may be split by this special period into multiple subparts with machine power-off&on in-between adjacent subparts. A set of sequential power states ($S_{off\&on}$) is involved in powering off&on a machine. Each power state (*s*) is characterized by an averaged power level (P_s) and an averaged time duration (T_s), i.e., power profile. In Eqs. (8-9), the current time is mapped to the corresponding electricity pricing slot.

$$ECO_i = \sum_{(i,j,k,l)\in\pi_i} \sum_{ts=STSO_{ijkl}^{n}}^{ETSO_{ijkl}^{n}} EP_{ts} \cdot \left(\sum_{n=1}^{NSO_{ijkl}} \sum_{t=STO_{ijkl}^{n}}^{ETO_{ijkl}^{n}} \beta_{ts} \cdot P_p \cdot t + \lambda_{wp} \cdot \sum_{n=2}^{NSO_{ijkl}} \sum_{t=STO_{ijkl}^{n}}^{ETO_{ijkl}^{n}} \sum_{s\in S_{off\&on}} \beta_{ts} \cdot P_s^t \cdot t \right) \tag{7}$$

$$\beta_{ts} = \begin{cases} 1, \text{ if } t \in [ts \cdot D, (ts+1) \cdot D) \\ 0, \text{ otherwise} \end{cases} \tag{8}$$

$$ts = \lfloor (t - ST)/D \rfloor,\ t \in [ST,\ ST + \delta t,\ ...,\ DT - \delta t,\ DT] \tag{9}$$

The energy cost for all changeovers on machine *i* (ECC_i) is described in Eq. (10). Analogous to Eq. (7), it is accumulated over all changeovers on machine *i* by mapping the energy consumption of each changeover to the energy cost under dynamic electricity prices. The spit of one changeover into multiple subparts by a special period is also considered, such that the additional energy cost for powering off&on machines before and after this special period is also calculated. A set of sequential power states (S_c) may be involved in a changeover. Note that the machine does not have to perform a changeover if the changeover duration is zero, i.e., $STC_{ijkl} = ETC_{ijkl}$; otherwise, the changeover is performed right before the start of operation (*i, j, k, l*).

$$ECC_i = \sum\nolimits_{(i,j,k,l)\in\pi_i} \sum\nolimits_{ts=STSC_{ijkl}}^{ETSC_{ijkl}} EP_{ts} \cdot \left(\sum\nolimits_{n=1}^{NSC_{ijkl}} \sum\nolimits_{t=STC_{ijkl}}^{ETC_{ijkl}} \sum\nolimits_{s\in S_c} \beta_{ts} \cdot P_s^t \cdot t + \lambda_{wp} \cdot \sum\nolimits_{n=2}^{NSC_{ijkl}} \sum\nolimits_{t=STC_{ijkl}}^{ETC_{ijkl}} \sum\nolimits_{s\in S_{off\,\&on}} \beta_{ts} \cdot P_s^t \cdot t \right) \tag{10}$$

The energy cost for the idling of machine *i* (*ECI$_i$*) is accumulated in Eq. (11), similar to Eqs. (7, 10). The machine idling encompasses three cases, as described in Eqs. (12-14). (1) α_{ijkl} = 0: no machine idling follows operation (*i*, *j*, *k*, *l*), i.e., the set of idling power state is empty ($S_I = \varnothing$). Consequently, a potential changeover immediately starts upon the end of operation (*i*, *j*, *k*, *l*). (2) $\alpha_{ijkl} = 1$: an idle mode is set between operations (*i*, *j*, *k*, *l*) and (*i*, *j'*, *k'*, *l+1*), i.e., $S_I = S_{idle}$. (3) α_{ijkl} = 2: the off mode with an off duration (d_{off}) is set between operations (*i*, *j*, *k*, *l*) and (*i*, *j'*, *k'*, *l+1*), i.e., $S_I = S_{off}$.

$$ECI_i = \sum\nolimits_{(i,j,k,l)\in\pi_i} \sum\nolimits_{ts=STSI_{ijkl}}^{ETSI_{ijkl}} EP_{ts} \cdot \left(\beta_{ts} \cdot \sum\nolimits_{t=STS_{ijkl}}^{ETS_{ijkl}} \sum\nolimits_{s\in S_I} P_s^t \cdot t \right) \tag{11}$$

$$S_I = \begin{cases} \varnothing, if\ \ \alpha_{ijkl} = 0 \\ S_{idle}, if\ \ \alpha_{ijkl} = 1 \\ S_{off}, if\ \ \alpha_{ijkl} = 2 \end{cases} \tag{12}$$

$$STI_{ijkl} = ETO_{ijkl} \begin{cases} = ETI_{ijkl}, \text{if } \alpha_{ijkl} = 0 \\ \neq ETI_{ijkl}, \text{otherwise} \end{cases} \tag{13}$$

$$ETI_{ijkl} = \begin{cases} ETO_{ijkl} = STI_{ijkl}, if\ \ \alpha_{ijkl} = 0 \\ STI_{ijkl} + \sum\nolimits_{s\in S_{idle}} T_s + d_{idle}, if\ \ \alpha_{ijkl} = 1 \\ STI_{ijkl} + \sum\nolimits_{s\in S_{off}} T_s + d_{off}, if\ \ \alpha_{ijkl} = 2 \end{cases} \tag{14}$$

3.3 Total labor cost

The duration of a shift is defined by Eq. (16), i.e., 24 hours divided by the number of shifts in a day $(|\mathrm{SH}|)$. SH is the set of shifts in a day which may be a weekday and a weekend day. If weekend production is allowed, the labor wage in each shift is higher on a weekend day than on a weekday. The labor wage in a day also varies, which increases in the night shift than in the other shifts on the same day.

$$\delta sh = 24 / |\mathrm{SH}| \tag{16}$$

Within one shift (*sh*), once a personnel type (*pt*) is required by an involved power state (*s*), this pt will be included in this sh $(\theta_{sh}^{pt} = 1)$. Otherwise, the binary personnel occupation indicator θ_{sh}^{pt} is zero.

Workers, machine production, and machine power consumption are coupled by the link between the skills required by different machine operations. In this manner, these three decision variables are fully integrated, compared to the existing literature which performs independent decision making on these decision variables. Table 3 presents the mapping between generic power states and

Table 3
Mapping between generic machine power states and the required type of personnel

Current power state	Trigger event	Destination power state	Next power state	Type of personnel
Off	Power-on	Ready	Startup	None
Startup	Automatic	Ready	Ready	Operator
Ready	Idling Produce	Ready Production	Ready Production	Operator
Production	Automatic (when current job is completed)	Ready	Ready	Operator, quality checker (for jobs whose last operation is performed on this machine)
Shutdown	Automatic	Off	Off	Operator

the type of personnel, while Fig. 5 presents the generic power state-based energy model that accommodates these three decision variables. Both the energy model and the personnel type are generic in Table 3 and Fig. 5, and can be further extended to specific cases.

3.4 Operation, job, and changeover

A job j has a fixed sequence of operations G_j. To complete a job, each of these operations must be performed by one qualified machine. Operations of distinct jobs may follow an arbitrary sequence on a machine but with a common release time (*RT*) and due time (*DT*). Recirculation is allowed. From a job perspective, this means that a job may contain multiple identical operations and return to a previous machine it passed through. Due to the weekend production constraint introduced by the labor model, an operation (i, j, k, l) contains one or more sub-durations, as indicated in Eq. (17).

$$DO_{ijkl} = \sum\nolimits_{n=1}^{NSO_{ijkl}} \left(ETO_{ijkl}^{n} - STO_{ijkl}^{n} \right),\ i \in M,\ j \in J, k \in K, l \in I_i \tag{17}$$

The last operation on a machine must be completed before *DT*, considering the duration to power off this machine (Eq. 18).

$$ETO_{ijkl} + D_{poff}^{i} \le DT, i \in M,\ j \in J, k \in K, l = \left| I_i \right| \tag{18}$$

As defined by Eq. (19), a machine changeover is required between adjacent operations of different types. It starts right before the upcoming operation on a machine. $ETC_{ijkl}^{NSC_{ijkl}}$ is the end time of last subpart of the changeover before operation (i, j, k, l).

$$ETC_{ijkl}^{NSC_{ijkl}} = STJ_{ijk'l}^{1}, i \in M,\ j \in J, k, k' \in K, l \in I_i \setminus \{1\} \tag{19}$$

Analogously, the duration of a changeover is the sum of one or multiple sub-durations, as described in Eq. (20).

$$DC_{ijkl} = \sum\nolimits_{n=1}^{NSC_{ijkl}} \left(ETC_{ijkl}^{n} - STC_{ijkl}^{n} \right), i \in M,\ j \in J, k \in K, l \in I_i \setminus \{1\} \tag{20}$$

3.5 Machine

A machine is assumed to have sufficient material supply and have no breakdown. It cannot simultaneously perform multiple operations and does not allow preemption (Eq. 21). An idle mode is only applicable to a period that can accommodate it (Eq. 22). If weekend production is prohibited, the machine must be powered off, as indicated in Eq. (23). The changeover/setup time depends on the sequence of adjacent operations on a machine and is randomly generated from a set of setup times for experiments in this paper, as defined in Eq. (24).

$$ETO_{ijkl}^{NSO_{ijkl}} < STO_{ij'k'(l+1)}^{1}, i \in M,\ j, j' \in J, k, k' \in K, l \in I_i \tag{21}$$

$$\sum\nolimits_{s \in SI_{ij}} D_s \le STC_{ijk(l+1)}^{1} - ETJ_{ijkl}^{NSJ_{ijkl}}, i \in M,\ j \in J, k \in K, l \in I_i \tag{22}$$

$$P_s^t = 0, if\ (\lambda = 1)\ \&\ (\forall t \in \text{weekends}) \tag{23}$$

$$ST_{(i,j,k,l) \to (i,j',k',l+1)} = rand(T_{setup}), i \in M,\ j, j' \in J, k, k' \in K, l \in I_i \tag{24}$$

4. Solution algorithm: tailored NSGA-III

This section is the first attempt to tailor a NSGA-III [36] in solving a many-objective FJSSP. Fourfold issues are specifically tackled in this attempt. (1) How to represent an EL-FJSSP scheduling solution in an evolutionary search? This solution representation not only aims for a conventional flexible job shop, but also should allow for job recirculation on the same machine in the problem domain and promote diversity in a population during an evolutionary search. (2) How to effectively decode a potentially complex chromosome to a corresponding scheduling solution? (3) How to define crossover and mutation operators? (4) How to efficiently evaluate a scheduling solution given the many-objective and highly-constrained FJSSP?

4.1 Scheduling solution encoding

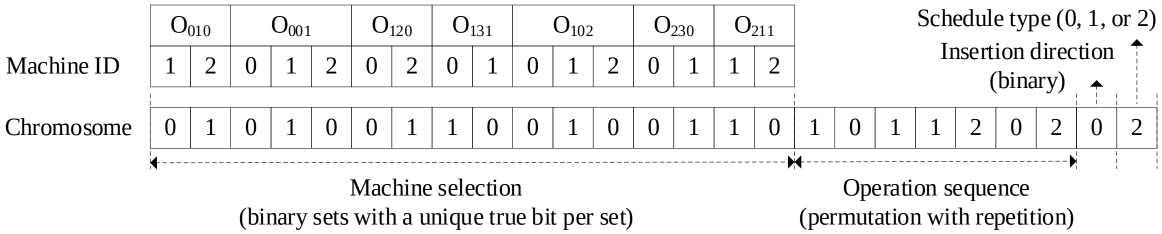

O_{010}
O_{001}
O_{120}
O_{131}
O_{102}
O_{230}
O_{211}
Machine ID
1 2 0 1 2 0 2 0 1 0 1 2 0 1 1 2
Chromosome
0 1 0 1 0 0 1 1 0 0 1 0 0 1 1 0 1 0 1 1 2 0 2 0 2
Schedule type (0, 1, or 2)
Insertion direction (binary)
Machine selection (binary sets with a unique true bit per set)
Operation sequence (permutation with repetition)

Figure 1 Heterogeneous chromosome representation. For an operation O_{jkl}, *j* represents a job, *k* represents a type of operation, and *l* represents the sequence of operation *k* in job *j*.

A canonical evolutionary algorithm usually produces some infeasible solutions after recombination by crossover and mutation. Consequently, a repair mechanism is often required for remedy, although this slows down the evolutionary search. In this paper, a solution is encoded by four heterogeneous chromosome segments, such that these repair efforts are removed, while guaranteeing the feasibility of all solutions.

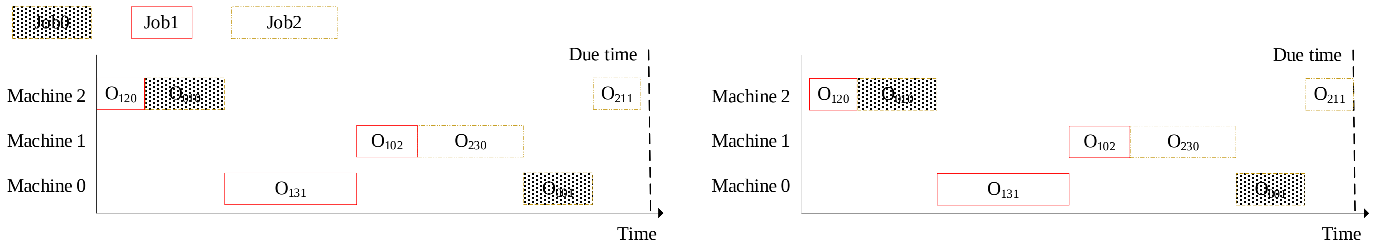
In the example of Fig. 1, job0 has two operations (O_{010}, O_{001}), job1 has three operations (O_{120}, O_{131}, O_{102}), and job2 has two operations (O_{230}, O_{211}). These operations are aligned in a lexicographical order of the job index and the operation sequence of each job. The first chromosome segment in Fig. 1 refers to machine selection and is encoded as binary sets. A true bit (i.e., one) means that the machine is selected, and vice versa for a false bit (i.e., zero). Therefore, the machines selected for (O_{010}, $O_{001,}$ O_{131}, O_{102}, O_{230}, $O_{012,}$ O_{211}) are (2, 0, 2, 0, 1, 1, 2). The second segment represents the sequence in which operations of all jobs are placed on the time horizon of the corresponding machine. It is encoded as permutation with repetition [37], which naturally maintains the relative order among operations of a job and further facilitates the definition of crossover (Sect. 4.5) and mutation (Sect. 4.5). In Fig. 1, the operation sequence is (O_{120}, $O_{010,}$ O_{131}, O_{102}, O_{230}, O_{001}, O_{211}). The third segment indicates the operation insertion direction. It is encoded as a binary bit. A true bit (i.e., one) indicates the forward insertion of operations on the scheduling time horizon (Sect. 4.2.1), while a false bit (i.e., zero) implies the backward insertion of operations (Sect. 4.2.2).

The fourth chromosome segment in Fig. 1 indicates the schedule type, which is a decimal number varying from zero to two (0: original schedule, 1: semi-active schedule, 2: active schedule). Based on the primary classification of forward schedules in [9], this paper further classifies feasible non-preemptive forward and backward schedules in the following three categories: (1) *original schedule* (operations are placed on the time horizons of machines one after another by strictly following the assigned forward or backward operation sequence), (2) *semi-active schedule* (if no operation can be completed earlier or later for a forward or backward schedule, respectively, without changing the order of processing on any one of the machines), (3) *active schedule* (if no operation can start earlier without delaying at least one other operation in a forward schedule, or if no operation can end later without advancing at least one other operation in a backward schedule). Therefore, the schedule type determines the location of time horizon of the corresponding machine where an operation is placed and even the operation sequence on a machine for an active schedule. Details will be provided in Sect. 4.2 on decoding such a chromosome into a schedule.

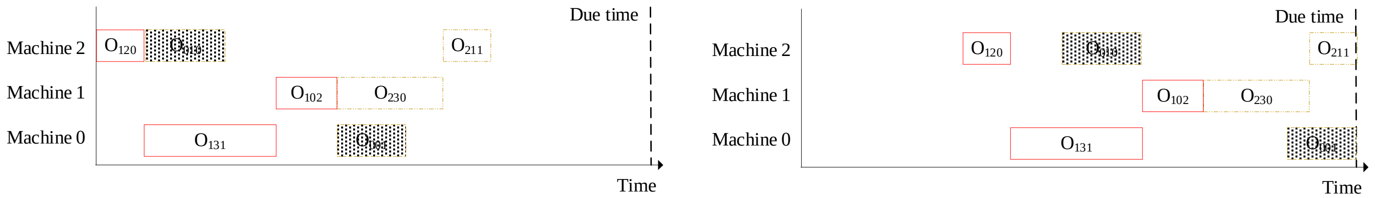
Despite having a longer makespan compared to active schedules, original and semi-active schedules are equally important. In the context of industrial demand response, more free durations in the latter two types of schedules allow for machine powering off and idling during peak electricity pricing periods. The combination of insertion direction and schedule type leads to overall six schedule types. Consequently, this diversifies the population to avoid premature convergence.

4.2 Scheduling solution decoding

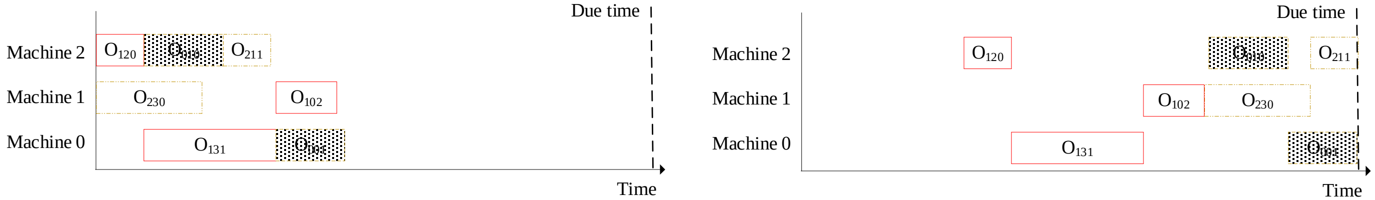
Fig. 2 demonstrates how to decode a scheduling solution from the heterogeneous chromosome illustrated in Fig. 1. Operations are assigned one after another to corresponding machines in the indicated direction and following the assigned sequence, while the final operation sequence on a machine may be altered in an active schedule. In an *original schedule*, an operation is simply inserted to the end (forward decoding in Fig.2a) or the start (backward decoding in Fig. 2b) of its precedent, which is indicated by the chromosome. In a *semi-active schedule*, an operation can start as early as possible (forward decoding in Fig. 2c) or end as late as possible (backward decoding in Fig. 2d) without changing the assigned operation sequence. In an *active schedule*, the operation sequence can be broken, and no left-shift can be further performed without right-shifting another operation (forward decoding in Fig. 2e), or no right-shift can be further performed without left-shifting another operation (backward decoding in Fig. 2f). Although there are overall six decoding methods, a specific decoding method is determined once the solution encoding is known.



(a) Original schedule (forward) (b) Original schedule (backward)



(c) Semi-active schedule (forward) (d) Semi-active schedule (backward)



(e) Active schedule (forward) (f) Active schedule (backward)

Figure 2 Example of forward and backward decoding an original, semi-active and active schedule from a chromosome with the same machine selection and operation sequence segments illustrated in Fig.1.

Fig. 2 does not illustrate job or changeover split for brevity. However, when actually inserting an operation, a weekend should be excluded from time allocation if weekend production is not allowed. Consequently, a job/changeover may have multiple start and end times, indicating that it is split by at least one weekend. For the concern of job/changeover split, the following two definitions are provided.

Definition 1: first start time refers to the start time of the first subpart of a job/changeover, considering its split by an event on the time horizon, e.g., prohibited weekend production.

Definition 2: last end time refers to the end time of the last subpart of a job/changeover, considering its split by an event on the time horizon, e.g., prohibited weekend production.

Note that if a job/changeover is not split into multiple subparts, it is considered to have only one subpart. Although the assigned operation sequence may be altered in decoding an active schedule, it rather refers to the operation sequence on a machine. The operation sequence of a job remains fixed, as a constant input for a scheduling algorithm, to ensure that a job is processed in the designed technological sequence. Analogously, even when backward inserting operations in a schedule, the operation sequence of a job should be fully respected by backward following the fixed operation sequence of a job.

The complexity of decoding an original or semi-active schedule is $O(n)$. Comparatively, the complexity of decoding active schedule is $O(n^2)$. While it is apparent to decode an original or semi-active schedule, Algorithms 1 & 2 are proposed as follows to forward and backward decode an active schedule, respectively.

4.2.1 Forward decoding an active schedule

Algorithm 1 describes how to forward insert operations to build an active schedule (illustrated in Fig. 2e). Operations are inserted to the time horizon of the corresponding machine following the forward operation sequence (lines 1-2, Algorithm 1). Each insertion should be performed as early as possible, while respecting the operation sequence of a job (lines 3-7, Algorithm 1) and machine availability (line 8, Algorithm 1). If an operation is the first one in a job, the operation sequential constraint does not exist, such that

Algorithm 1 Forward decoding an active flexible job shop production schedule

Input: *sequencedOperations* (with assigned machines)

Output: a production schedule π which ends as early as possible

```
for index $\leftarrow$ 1 : size(sequencedOperations)
    operation $\leftarrow$ sequencedOperations(index)
    if index == 0
        lowerBoundInAJob $\leftarrow$ startTimeSchedule + durationStartup
    else
        lowerBoundInAJob $\leftarrow$ operation.job.previousOperation.lastEndTime
    end if
    $\pi$. insertAnOperationOnAMachine(lowerBoundInAJob, operation, true) //Algorithm 3
end for
```

Algorithm 2 Backward decoding an active flexible job shop production schedule

Input: *sequencedOperations* (with assigned machines)

Output: a production schedule π which starts as late as possible

```
for index $\leftarrow$ size(sequencedOperations) : 1
    operation $\leftarrow$ sequencedOperations(index)
    if index == size(sequencedOperations)
        upperBoundInAJob $\leftarrow$ dueTime – durationShutdown
    else
        upperBoundInAJob $\leftarrow$ operation.job.nextOperation.firstStartTime
    end if
    $\pi$. insertAnOperationOnAMachine(upperBoundInAJob, operation, false) //Algorithm 3
end for
```

Algorithm 3 Operation insertion on a machine

Input: *boundInAJob*, *operationToBeInserted*, *isForwardInsertion*

Output: **true** if this insertion is successful, **false** otherwise

```
mach $\leftarrow$ operation.assignedMachine
if isForwardInsertion == true
    freePeriods $\leftarrow$ mach.getFreePeriodsLaterThan(boundInAJob)
    indexSet $\leftarrow$ 1 : size(freePeriods)
else
    freePeriods $\leftarrow$ mach.getFreePeriodsEarlierThan(boundInAJob)
    indexSet $\leftarrow$ size(freePeriods) : 1
end if
for index $\in$ indexSet
    freePeriod $\leftarrow$ freePeriods(index)
    if mach has no assigned operation
        requiredDuration $\leftarrow$ operation.duration
    else
        if period is before existing operations on mach
            requiredDuration $\leftarrow$ operation.duration + mach.getSetupTime(operation, mach.nextOperation)
        else if period is after existing operations on mach
            requiredDuration $\leftarrow$ operation.duration + mach.getSetupTime(mach,previousOperation, operation)
        else
            requiredDuration $\leftarrow$ operation.duration + mach.getSetupTime(mach,previousOperation, operation) + mach.getSetupTime(operation, mach.nextOperation)
        end if
    end if
    if period >= requiredDuration
        mach.insert(operation, period)
        return true
    end if
end for
return false
```

the *lowerBoundInAJob* is set as the start of scheduling span plus the time to start up the machine which will perform this operation (lines 3-4). Otherwise, the *lowerBoundInAJob* is the last end time of the previous operation in this job (lines 5-7, Algorithm 1).

The function insertAnOperationOnAMachine(*lowerBoundInAJob*, *operation*, true) in line 8 of Algorithm 1 is described in detail by Algorithm 3. It starts from the input *lowerBoundInAJob*, and forward looks for the earliest feasible free period on the target machine to accommodate the input *operationToBeInserted* (lines 2-4 & 9-26, Algorithm 3). If the target machine is not yet assigned any operations, a feasible free period should accommodate the duration of *operation* (lines 11-12, Algorithm 3). Otherwise, the sequence-dependent setup times should be additionally considered for accommodation (lines 14-21, Algorithm 3).

Generally, feasible free periods (*freePeriod(s)*, Algorithm 3) are categorized in three types, regarding the operation insertion position relative to the existing operations on the target machine: (1) leftmost of existing operations (lines 14-15, Algorithm 3), (2) between existing operations (lines 16-17, Algorithm 3), (3) rightmost of existing operations (lines 18-19, Algorithm 3). For the first type of *freePeriod*, a machine changeover occurs between the *operationToBeInserted* and the first existing operation (i.e., with the earliest start time among all existing operations). For the second type of *freePeriod*, two changeovers should be considered between the previous existing operation (relative to *operation*) and *operation*, as well as between *operation* and the next existing operation (relative to *operation*). For the third type of *freePeriod*, a changeover between the last existing operation (i.e., with the latest start time among all existing operations) and *operation* should be performed.

4.2.2 Backward decoding an active schedule

The existing FJSP research focuses on minimizing the makespan (Table 1). This paper additionally employs the as-late-as-possible philosophy to diversify the population. Although the makespan increases, this aims to create more trade-off scheduling solutions along the Pareto approximation frontier.

Algorithm 2 presents the method to backward insert operations to build an active schedule from a chromosome. Operations are iterated from the last operation toward the first one (lines 1-2, Algorithm 2). Each insertion should be performed as late as possible, while simultaneously satisfying the operation sequence in a job (lines 3-7, Algorithm 2) and machine availability (line 8, Algorithm 2). If an operation is the last one in a job, the *upperBoundInAJob* equals the due time minus the time to shut down the target machine (lines 3-4, Algorithm 2). Otherwise, the *upperBoundInAJob* is the first start time of the next operation in this job (lines 5-7, Algorithm 3).

Analogously, the function insertAnOperationOnAMachine(*upperBoundInAJob*, *operation*, false) is introduced in Algorithm 3. It starts from the input *upperBoundInAJob*, and backward looks for the latest feasible free period on the target machine to accommodate the input *operationToBeInserted* (lines 5-8 & 9-26, Algorithm 3).

4.2.3 Timing policy

When using Algorithms 1-3, the exact start time of each operation still needs to be assigned. This paper employs the directional timing policy while other timing policies may be alternatively used. In a directional timing policy, if the decoded insertion direction is forward (i.e., the binary insertion direction in Fig. 1 is one), the first start time of an operation to be inserted is the start time of the earliest feasible free period, and this operation is forward inserted on the time horizon; if the decoded insertion direction is backward (i.e., the binary insertion direction in Fig. 1 is zero), the last end time of an operation to be inserted is the end time of the latest feasible free period, and this operation is backward inserted. Note that the directional timing policy does not depend on the schedule type indicated in an encoded solution.

4.3 Crossover

The one-point and order crossovers are applied for the chromosome segments of machine selection and operation sequence, respectively. Fig. 3 provides an example based on the chromosome shown in Fig. 1. For the one-point crossover, the crossover locus is randomly generated between two adjacent operations. Then two parents swap their chromosome segments after this crossover locus, while keeping their chromosome segments before this crossover locus. The indicated crossover locus is randomly generated among

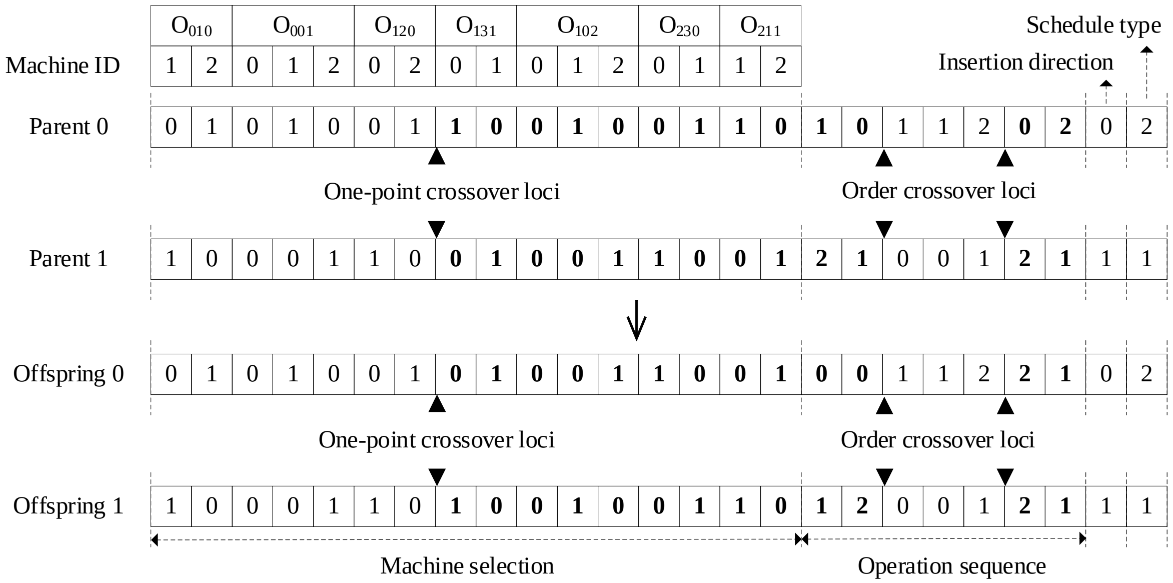

O_{010} O_{001} O_{120} O_{131} O_{102} O_{230} O_{211}
Machine ID 1 2 0 1 2 0 2 0 1 0 1 2 0 1 1 2
Schedule type
Insertion direction
Parent 0 0 1 0 1 0 0 1 1 0 0 1 0 0 1 1 0 1 0 1 1 2 0 2 0 2
One-point crossover loci
Order crossover loci
Parent 1 1 0 0 0 1 1 0 0 1 0 0 1 1 0 0 1 2 1 0 0 1 2 1 1 1
Offspring 0 0 1 0 1 0 0 1 0 1 0 0 1 1 0 0 1 0 0 1 1 2 2 1 0 2
One-point crossover loci
Order crossover loci
Offspring 1 1 0 0 0 1 1 0 1 0 0 1 0 0 1 1 0 1 2 0 0 1 2 1 1 1
Machine selection
Operation sequence

Figure 3 Example of one-point and order crossovers applied for the chromosome segments of machine selection and operation sequence, respectively. The genes in bold are exchanged from the other parent.

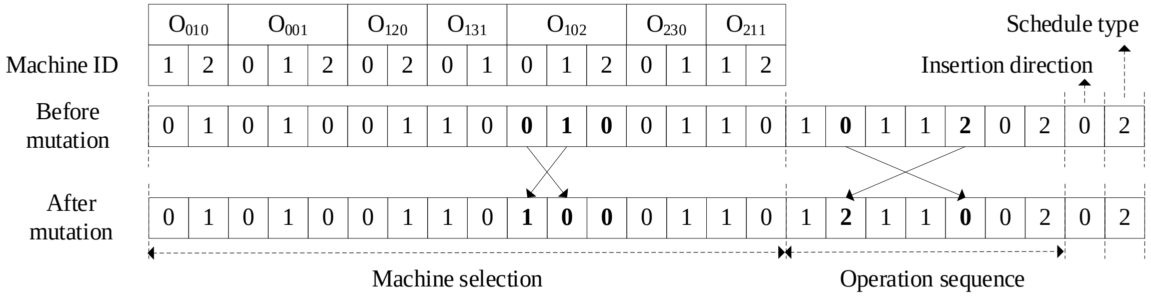

O_{010} O_{001} O_{120} O_{131} O_{102} O_{230} O_{211}
Machine ID 1 2 0 1 2 0 2 0 1 0 1 2 0 1 1 2
Schedule type
Insertion direction
Before mutation 0 1 0 1 0 0 1 1 0 0 1 0 0 1 1 0 1 0 1 1 2 0 2 0 2
After mutation 0 1 0 1 0 0 1 1 0 1 0 0 0 1 1 0 1 2 1 1 0 0 2 0 2
Machine selection
Operation sequence

Figure 4 Example of swap-based mutation of the chromosome parts of machine selection and operation sequence. The genes in bold numbers are exchanged.

six optional crossover loci. For the order crossover [38], two cut points are randomly selected and applied to both parents. Genes between these two cut points (inclusive) remain to the corresponding offspring. Starting from the second cut point, the rest genes are copied one by one from the other parent if the number of a certain job ID is not larger than the original number; otherwise, this copy operation is omitted. The former copy operation returns to the gene position of zero if it reaches the end of a chromosome. It continues until reaching the first cut point.

4.4 Mutation

The swap-based mutation is applied to both chromosome segments of machine selection and operation sequence. Fig. 4 demonstrates an example. For the former segment, an operation is randomly selected from all operations each of which can be processed on multiple machines. Then the unique true bit swaps with another randomly selected false bit of the same operation. For the latter chromosome segment, two loci with different job IDs are randomly generated and then swap these two job IDs.

4.5 Solution evaluation based on discrete-event simulation

A scheduling solution is evaluated in terms of objectives defined in Eqs. (2-6). As solution evaluation is frequently performed in population initialization and recombination of an evolutionary search (e.g., NSGA-III), the time step-based model formulated in Sect. 3 is extensively calculated in an evolutionary search. While a large number of time slots may be involved in a solution evaluation,

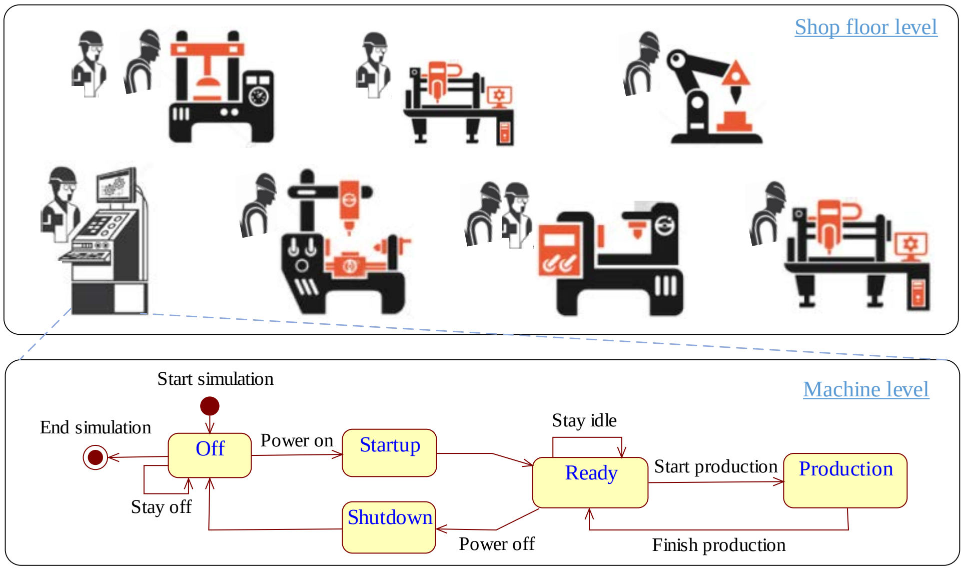
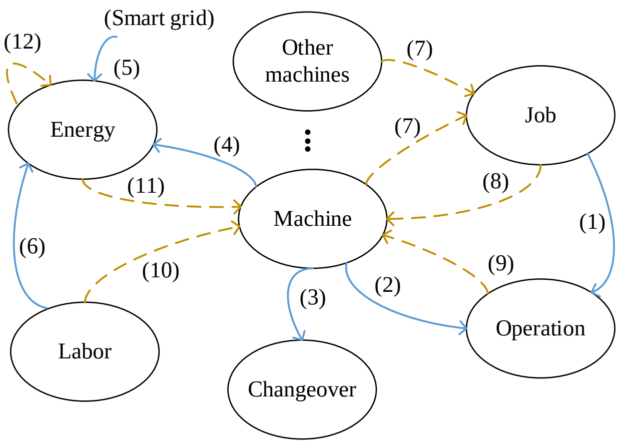


Figure 5 Multi-layer energy- and labor-aware shop floor production simulation framework



Input variables

(1) A set of jobs to be processed, a sequence of operations for each job
(2) A set of capable operations for each machine
(3) Sequence-dependent setup times for each machine
(4) State-based energy model for each machine
(5) Dynamic electricity prices
(6) Labor shifts per day, number and type of labor for each power state

Output variables

(7) A sequence of machines that have processed a job
(8) A sequence of processed jobs on a machine
(9) A sequence of processed operations on each machine
(10) Labor shift over time and labor cost at machine and shop floor levels
(11) Idling between adjacent jobs on a machine
(12) Energy consumption & cost at power state, machine, and shop floor levels, machine power consumption over time

Figure 6 Major input and output variables in energy- and labor-aware shop floor production simulation, as well as the interdependency of sustainable production aspects including machines, operations, changeovers, jobs, energy, and labor.

the slot-wise calculation will cause a high overhead. To reduce this calculation overhead, discrete-event simulation (DES) is used, as its runtime depends on the number of discrete events in a simulation instead of the number of time slots. It thus facilitates large-scale optimization, in contrast to the commercial off-the-shelf solver (e.g., IBM CPLEX) which is very sensitive to the problem size.

Fig. 5 presents the multi-layer framework for energy- and labor-aware discrete-event production simulation on the shop floor. The shop floor level comprises multiple machines and human workers. Machines of different types have distinct sets of operations that they can perform, e.g., grinding, milling, and cutting. Human workers of different types have distinct skill sets, such that operation assistance, quality checking, and packaging. A machine may be accompanied by a number of human workers, depending on the production configuration. Machines do not have any explicit connection among each other in a flexible job shop, as jobs may have different routes.

The machine level in Fig. 5 consists of five generic power states that mimic both energy consumption and production behaviors of a machine. The *Off* state is the location to start and end a simulation. It has a self-transition triggered by the command event “stay off” which indicates the duration for a machine to stay off. It transitions to *Startup* upon the command event “power on”. A machine automatically transitions from *Startup* to *Ready* when it completes the startup process. *Ready* state has three types of events: (1) “stay idle” upon which a machine stays idle for an assigned duration, (2) “start production” upon which a machine transitions to the *Production* state, (3) “power off” upon which a machine transitions to the *Shutdown* state for powering off. A machine returns from *Production* to *Ready* when it finishes processing the assigned job. When entering into the *Shutdown* state, it automatically transitions to *Off* after completing the shutdown procedure. Fig. 6 further introduces the key input and output variables for such a simulation, as well as the interdependency of machines, operations, changeovers, jobs, energy, and labor. During a DES process, the objective-related values are accumulated along with the chronological event evolution.

4.6 NSGA-III framework

The NSGA-III [36] follows the NSGA-II framework that emphasizes nondominated solutions in a population. However, for diversity preservation in the high-dimensional objective space, unlike in NSGA-II, it also emphasizes population members that are in some sense associated with each of the well-spread reference points, which are supplied upon the start of a NSGA-III instance and adaptively updated with the population evolution. To this end, the crowding distance operator in NSGA-II is replaced by the following sequential approach [36]: (1) determination of reference points on a hyper-plane, (2) adaptive normalization of population members, (3) association operation, and (4) niche-preservation operation.

5. Numerical experiments

5.1 Configurations

The numerical experiments were performed on a computer with Intel i5-3470 CPU @ 3.20 GHz and 8 GB RAM. A partial JFSSP instances was used: 8 (number of jobs) × 27 (number of operations) × 8 (number of machines) [39]. The machine processing power was randomly generated from the range [5 kW, 26 kW], based on the empirical machine processing power. The operation-sequence dependent machine setup times were randomly generated from the discrete set {10 s, 20 s, 30 m, 1 h, 1 h 30 m, 2 h}, which was statistically extracted from the changeover data in a Singaporean manufacturer in the precision engineering domain. The job quantity was set to 5000.

The 24-hour horizon is divided into three 8-hour labor shifts: morning shift (6 a.m. - 2 p.m.), late shift (2 p.m. - 10 p.m.), and night shift (10 p.m. - 6 a.m.). As the general energy model (Fig. 5) was used, the mapping between power states and type of personnel follows that in Table 1, where the number of workers per type of personnel was set as 1. The labor wage is the same in morning and late shifts, and increases by 10% in a night shift. Compared to a shift on a working day, the labor wage in the same shift but on a weekend day rises by 36%, if weekend production is allowed.

Besides NSGA-III, NSGA-II was used for the same many-objective EL-FJSSP as benchmarking. The hypervolume (HV) [40] and inverse generational distance (IGD) [41] were chosen as the performance metric for a many-objective evolutionary search, respectively, as each of them can simultaneously measure the convergence and diversity of the obtained nondominated solutions. A larger HV and a smaller IGD indicate an approximation set with better convergence and diversity. To enable the calculation of both metrics, the reference Pareto front approximation set was obtained by running each of the two algorithms twice and aggregating the obtained nondominated solutions. The stop criteria was set to 3 minutes. The crossover and mutation rates were 0.9 and 0.1, respectively. The population size was 212, which was approximately the number of reference points (6 outer divisions and 0 inner division) using the Das and Dennis’s approach [42].

5.2 Scheduling under real-time pricing (RTP)

The RTP data was taken from Belpex, the Belgium electricity spot market [43]. Under this pricing scheme, electricity prices vary in each hour; the prices for the next day are known after 3 p.m. on the current day. To get adapted to this pricing scheme and the labor shift configured in Sect. 5.1, the scheduling time span was set as 24 hours, lasting from 10 p.m. (included) on the current day to 10 p.m. (excluded) on the next day. NSGA-II and NSGA-III were run for 30 independent times, respectively, on the small-sized problem instance (Sect. 5.1)

5.2.1 Convergence of NSGA-II and NSGA-III

Fig. 7 depicts the statistical convergence curves of NSGA-II and NSGA-III, in terms of HV, IGD, and number of nondominated solutions. The number of generations in each algorithm is the minimal among 30 runs to ensure the statistical meaning of all these stochastic evolutionary searches. All the 6 curves in Fig. 7 are converged, except that the HV curve of NSGA-II still vibrates a bit near the end of a run. In contrast, the HV curve of NSGA-III remains stable after the 250th generation. The final median HV of NSGA-III (0.060) is higher than that of NSGA-II (0.045). The IGD curves of both algorithms rapidly converge at around the 100th generation. Compared to HV curves, IGD curves of both algorithms have much smaller variations in each generation. The final median IGD of NSGA-III (0.200) is comparable to that of NSGA-II (0.155). The number of nondominated solution of NSGA-II vibrates around 40 early starting from the 100th generation, while that of NSGA-III steadily rises to 55 in the first 140 generations, drops till the 250th generation, and remains around 30 afterward. The variations in the number of nondominated solution of both algorithms are notable, which are up to 100% compared to the median. Another observed phenomenon is that the number of generations in NSGA-II is as about 5 times large as that in NSGA-III.

Overall, NSGA-III is more effective than NSGA-II in solving a many-objective EL-FJSSP, as the former has a higher median HV and comparable IGD, and produces less nondominated solutions, which decreases the difficulty in solution selection for a production manager or a decision marker.

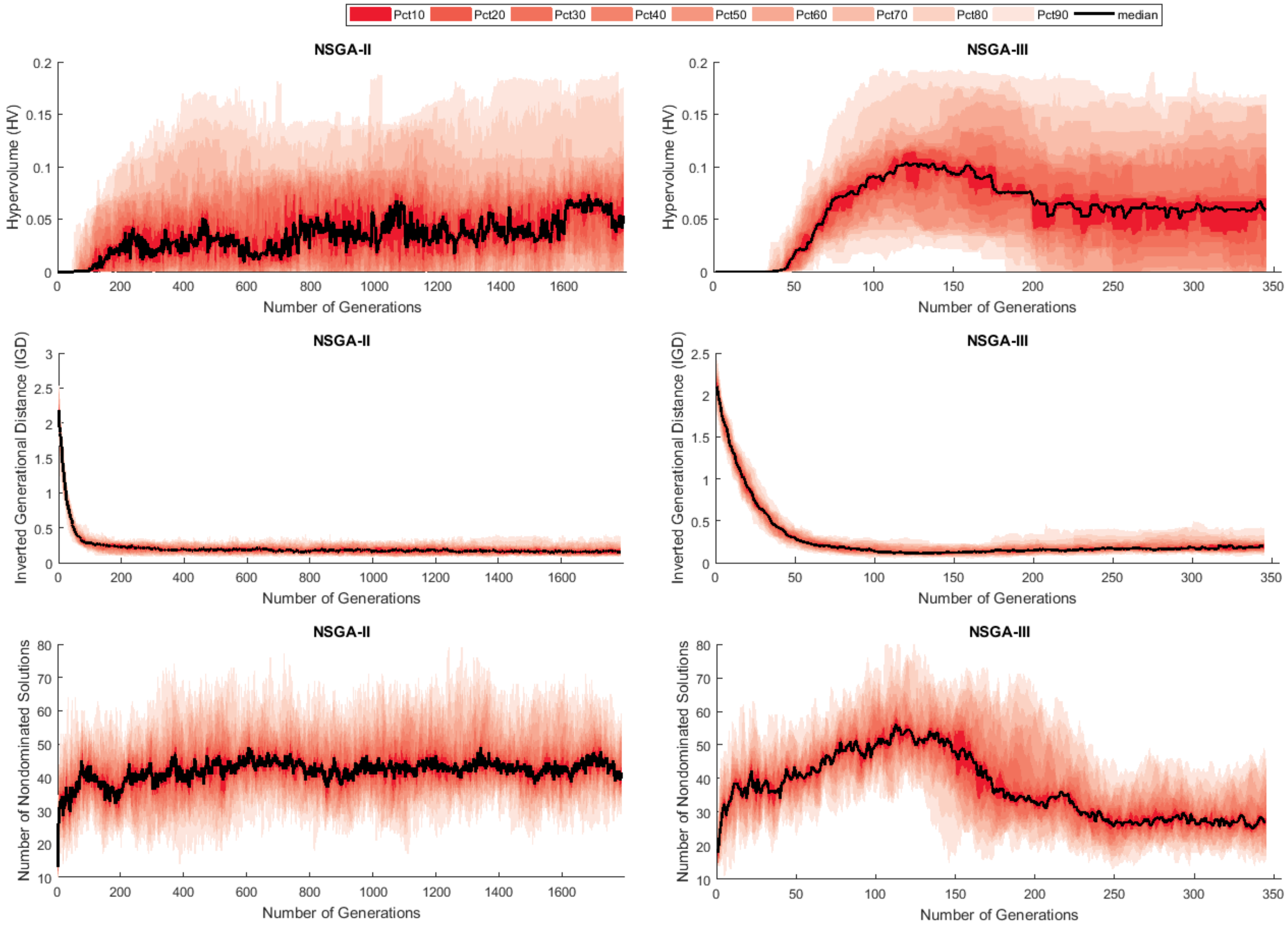


Figure 7 Statistical convergence curves of NSGA-II and NSGA-III for the 8×8×27 energy- and labor-aware flexible job shop scheduling problem (EL-FJSSP) under real-time pricing (RTP), in terms of hypervolume (HV), inverted generational distance (IGD), and number of nondominated solutions. These curves were obtained from 30 independent runs of NSGA-II and NSGA-III, respectively.

5.2.2 Relation among five production objectives

Fig. 8 illustrates the parallel coordinates plot of an approximation set provided by NSGA-III, where the objective values were normalized. In such a plot, a curve across the 5 objectives (Eqs. (2-6)) represents a nondominated solution. The superposition of all curves reveals the intrinsic relation among these 5 objectives. As clearly exhibited in Fig. 8, the makespan has a strong conflict with the total energy cost. This is explained by the fact that more and longer free durations will be created by postponing production jobs (thus raising the makespan). These free durations provide the evolutionary search with more opportunities to make use of lower-priced periods. The situation is vice versa if production jobs are fast processed: the total energy cost has to increase as a trade-off.

The total energy cost shows a relatively weak conflict with the total labor cost (Fig. 8). A lower total energy cost has to be slightly compromised by a higher total labor cost, and vice versa. This is explained by the contradiction of the electricity price and the labor wage: the former is usually higher during the day and lower in the night, and the other way around for the latter. Consequently, if an optimization process shifts production loads to the night for energy cost reduction, this has to be compensated by an increasing labor cost. However, this contradiction may decrease if production loads are shifted a bit to the night, such that the rising part of total labor cost is insignificant. This is why some curves have lower absolute slopes between the total energy cost and the total labor cost, although these slopes are not zero.

The total labor cost has an obvious trade-off relation with the maximum workload (Fig. 8). If the maximum workload is reduced, production jobs are more evenly distributed on machines in a flexible job shop. A rising number of labor shifts are thus more likely to be triggered, since even a piece of job on a machine will require at least one operator to work during the whole shift on that machine. On the contrary, if the maximum workload is raised, production is more concentrated on a limited number of machines. As a result, the production in each labor shift is more compact, leading to a lower total labor cost. Besides, the maximal workload has two clusters of values, which are at the two extremes of the whole range. This implies that it is a sensitive variable for an EL-FJSSP. A machine has either a very high workload or a very low workload, which is an important consideration factor when a production manager or a decision marker selects a final scheduling solution from an approximation set.

Nevertheless, the relation between the maximal workload and the total workload is nearly harmonious (Fig. 8). No cross, which indicates conflicts, is observed among curves between these two objectives. A low maximal workload corresponds to a low total workload. A high maximal workload leads to a high total workload, even though the corresponding range of the total workload is large (upper half of the whole range). This is explained by the reason that a high maximal workload means that a limited number of machines are highly used and could be the bottleneck of the whole production, as other jobs are more likely to wait before these machines until the current job is completed. On the contrary, if jobs are distributed on more machines, the maximal workload will significantly decrease and the total workload will also reduce as jobs are less likely to wait before a machine.

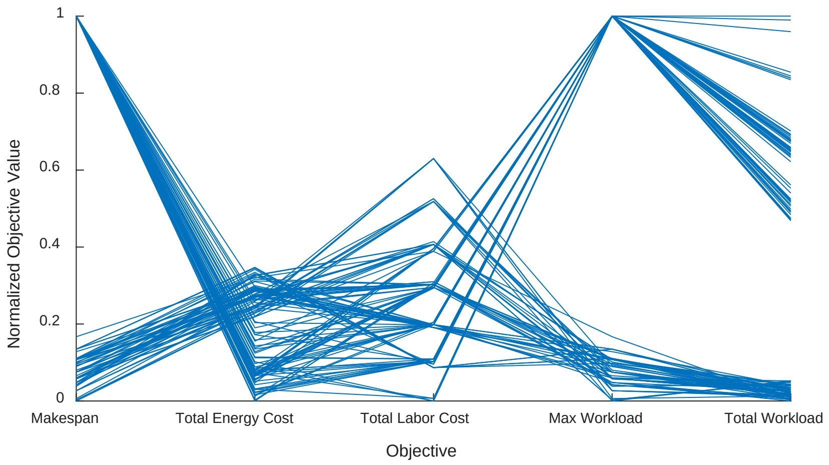


Figure 8 Parallel coordinates plot of an approximation set obtained by NSGA-III for an 8×8×27 energy- and labor-aware flexible job shop scheduling problem (EL-FJSSP) under real-time pricing (RTP). The scheduling time span is 24 hours. The values of each objective (makespan, total energy cost, total labor cost, maximum workload, and total workload) are normalized.

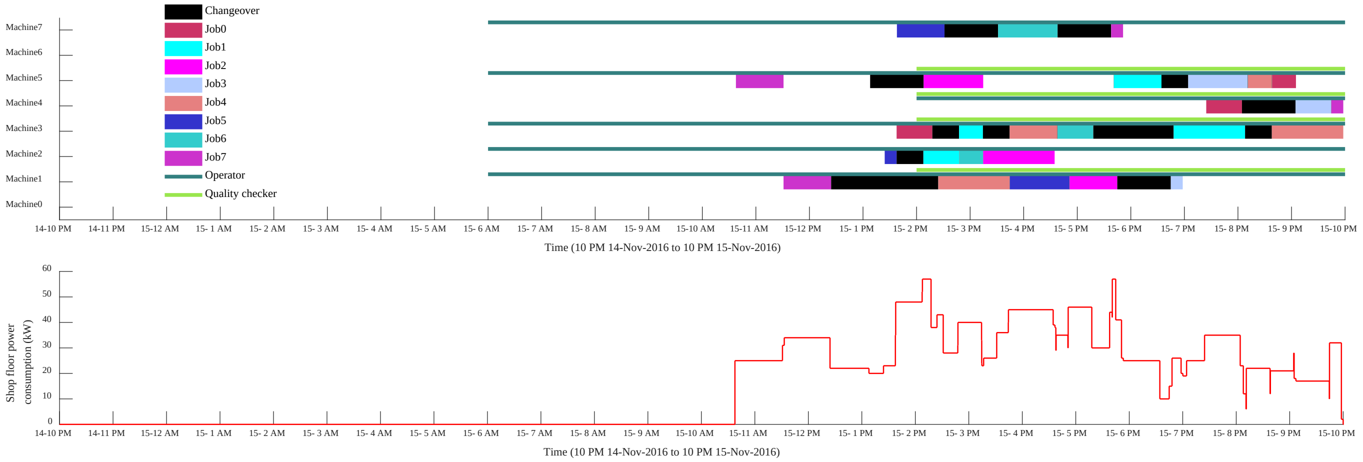


Figure 9 Visualization of an active backward production and labor schedule for an 8×8×27 energy- and labor-aware flexible job shop and the predicted overall power consumption over time.

Last but not least, both the total energy cost and the total labor cost remain in a limited range of small values. This proves the effectiveness and efficiency of the proposed model and tailored NSGA-III. In another word, the integration of energy and labor awareness in a conventional flexible job shop and the corresponding optimization are effective and efficient.

5.3 Visualization of an energy- and labor-aware flexible job shop production schedule

A backward active scheduling solution from the approximation set provided by NSGA-III in Sect. 5.2 is visualized in Fig. 9. Evidently, the operations of job0-job7 are backward placed on the time horizons of the corresponding machines, such that no operation can be further postponed without advancing other operations. The machine setup time exhibits its undeniable impact on the makespan of processing the 8 jobs. Machine3 turns out to be a bottleneck of the whole production, in the sense that it is under full utilization since its relatively early startup at 13:17:39 on 15 November 2016 until the due time. This high utilization could be explained by two reasons. Firstly, both job1 and job4 recirculate on machine3 such that machine3 has relatively more operations to process. Secondly, the makespan of many other machines is explicitly or implicitly affected by the full occupation of machine3, e.g., machine1, machine2, and machine5. Specifically, the job processing on machine5 highly depends on the operation processing of the same job on other machines. Consequently, it leads to the earliest start time in the schedule, directly influencing the makespan of the whole production.

Besides jobs and changeovers, the labor and the shift are also assigned in the schedule shown in Fig. 9. Both operator and quality checker are scheduled based on shifts. As the whole production are concentrated on the second half of the time span, all machines except machine0, machine4, and machine6 are assigned a morning shift and a late shift after 6 a.m. on 15 November 2016, during which an operator is required on each of these machines. Despite the light production load on these machines, the morning shift is triggered and obliged besides the late shift. This gives an illustration on the potential of total labor cost reduction by framing more compact production within less shifts. Machine4 is assigned a late shift, as it only has production close to the due time (10 p.m. on 15 November 2016). Less quality checker shifts are required compared to the operator shifts (Fig. 9), as a quality checker is only needed for inspecting the production quality at the last processing stage of a job.

Furthermore, the overall power consumption of all machines on the shop floor is predicted by the discrete-event simulation (Sect. 4.5) and visualized in Fig. 9. The power stays zero from the start of the scheduling time span until the earliest production (job7 on machine5). It then becomes a dynamic curve with different production loads on all machines over time. The highest power consumption occurs at around 2 p.m. and 6 p.m. on 15 November 2016, respectively. This is caused by the simultaneous job processing or setup of most machines during these two periods. In this way, the shop floor simulation framework (Fig. 5) provides production or factory managers the capability to notice the peak consumption before the actual production and proactively take the counter-measures, e.g., adjusting the production schedule to lower the power consumption. The predicted power consumption over time can also help a factory to achieve a better negotiation with power plants for a more economical energy contract.

5.4 Scheduling under time-of-use pricing (ToUP)

The proposed scheduling method is further demonstrated under ToUP, where the electricity pricing data was taken from a Belgium plastic bottle manufacturer. To ensure statistical significance, NSGA-II and NSGA-III were run 30 independent times, respectively. Fig. 10 presents the statistical convergence curves of both algorithms. All the 6 curves indicate the saturation of both algorithms within the given time budget, with two exceptions. The first gentle exception is the HV of NSGA-II. It steadily rises sine the 50th generation and reaches 0.149 at the end. Though this HV remains close to 0.149 after the 420th generation, the saturation trend is not evident enough. The second slight exception is the number of nondominated solutions of NSGA-II, which gently increases to 40 at the end of an evolutionary search. In contrast, the number of nondominated solution of NSGA-III has a rise to 38 in the first 1/3 generations, and steadily decreases in the latter 2/3 generations with a final value of 23. Analogous to the observations in Sect. 5.2.1, NSGA-III has a higher median HV (0.263). Its IGD (0.345) is comparable to that of NSGA-II (0.365).

Overall, two conclusion points can be drawn from Fig. 10, considering the observations in Sect. 5.2.1. Firstly, NSGA-III is superior to NSGA-II in solving a many-objective EL-FJSSP, due to its higher and more stable HV as well as comparable IGD on the one hand, and evidently less nondominated solutions that are output on the other hand. Secondly, HV is a more suitable metric in an EL-FJSSP. Though IGD provides performance evaluation with smaller variations, it cannot evidently differentiate the many-objective optimization performance of different evolutionary algorithms.

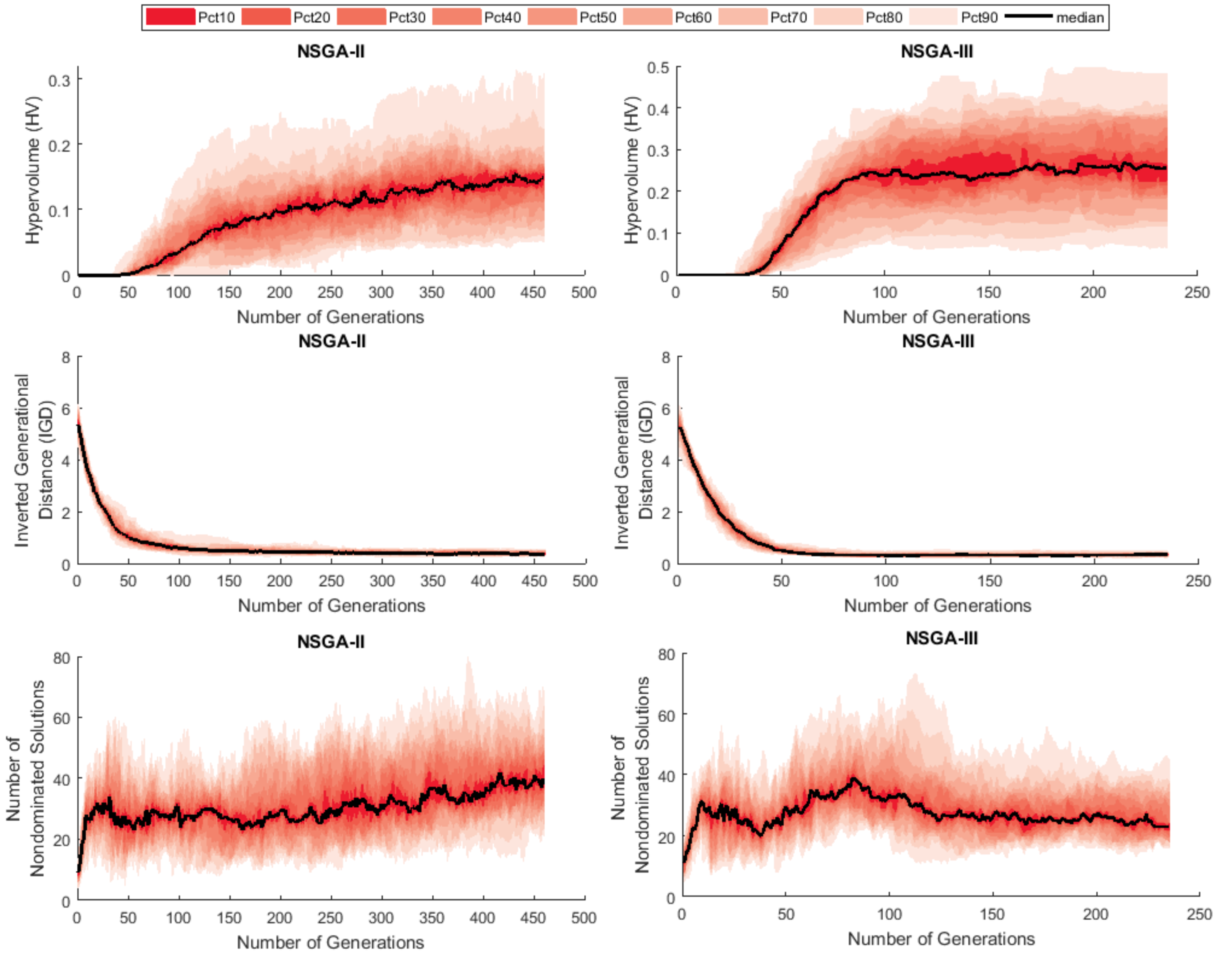


Figure 10 Statistical convergence curves of NSGA-II and NSGA-III for the 8×8×27 EL-FJSSP (energy- and labor-aware flexible job shop scheduling problem) under time-of-use pricing (ToUP), in terms of hypervolume (HV), inverted generational distance (IGD), and number of nondominated solutions. These curves were obtained from 30 independent runs of NSGA-II and NSGA-III, respectively.

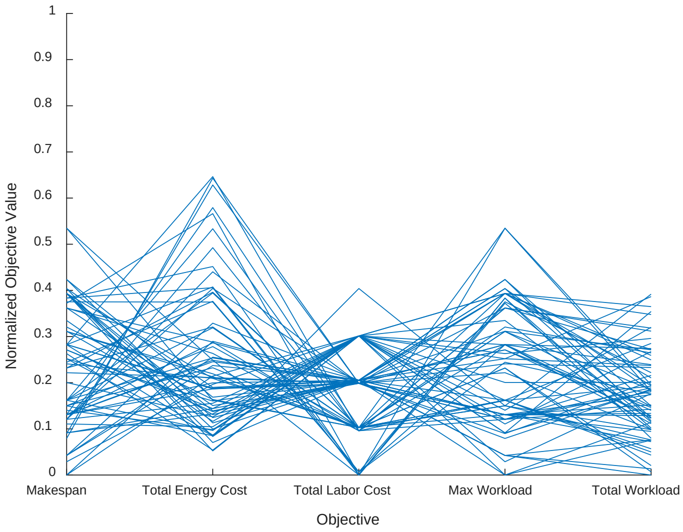


Figure 11 Parallel coordinates plot of an approximation set obtained by NSGA-III for an 8×8×27 energy- and labor-aware flexible job shop scheduling problem (EL-FJSSP) under time-of-use pricing (ToUP). The scheduling time span is two weeks while production is not allowed on weekends. The values of each objective (makespan, total energy cost, total labor cost, maximum workload, and total workload) are normalized.

The relation among the 5 objectives (Eqs. (2-6)) is demonstrated by the parallel coordinates in Fig. 11. Compared to the observations in Sect. 5.2.2, the trade-off between the makespan and the total energy cost still exists, though the conflict strength decreases a bit. The contradiction between the total energy cost and the total labor cost also holds. The range of total energy cost is nearly doubled. This is explained by the electricity price structure. The ToUP has only two pricing levels: on-peak and off-peak. Besides, the on-peak period is longer than the off-peak period. Both factors lead to less opportunities in reducing the total energy cost, compared to RTP. As a result, it is easier for the total energy cost to rise and have a larger range under ToUP than under RTP.

However, the range of total labor cost reduces by nearly a half in Fig. 11, compared to that in Fig. 8. As there is less production load shifting under ToUP, the assignment of labor shifts varies in a weaker manner. This consequently leads to the smaller variation in the total labor cost under ToUP. The conflict between the total labor cost and the maximal workload still remains in Fig. 11, though the strength decreases compared to that in Fig. 8. The range of maximal workload decreases by almost 50%. This is also because of the less production load shifting under ToUP, which facilities the NSGA-III to minimize in a general sense. The significantly reduced range is analogously observed in the total workload (Fig. 11) due to the same reason. Nonetheless, the nearly harmonious relation between the maximal workload and the total workload remains as in Fig. 8.

Last but not least, both the total energy cost and the total labor cost still remain in ranges with small values. This also demonstrates the effectiveness and efficiency of the proposed model and tailored NSGA-III.

5.5 Scalability

The size of the problem instance under ToUP increased from 8×27×8 (small size) to 200×600×20 (large size) to demonstrate the scalability of the proposed solution method. The HV metric was used to indicate the convergence of an NSGA-III instance. Multiple threads were utilized to speed up the convergence of an NSGA-III instance even in the large-sized problem instance. An NSGA-III instance was terminated by the end of the specified runtime. Different runtime was set, i.e., 3 min, 5 min, 10 min, 20 min, 30 min, and 40 min. For each configuration of thread number and runtime, the NSGA-III went through 30 independent runs such that an HV value was the average of these 30 runs for a specific configuration.

Table 4 lists the experiment results. The first observation is that the NSGA-III converged within reasonable runtime for both the small-sized problem instance (8×27×8) and the large-sized problem instance (200×600×20). For the small-sized problem instance, the NSGA-III converged at HV of 0.28 within 5 min using 4 threads. For the large-sized problem instance, it converged at HV of around 0.86 within 30 min and nearly converged at HV of around 0.96 within 20 min both using 8 threads. Therefore, the reasonable convergence time for the large-sized problem instance demonstrates the scalability of the proposed solution method.

The second observation of Table 4 is that the multithreading effectively contributes to the speedup of an NSGA-III search and thus the scalability of the solution method. For the large-sized problem instance, the convergence speed of the NSGA-III is stimulated by increasing the number of threads from 4 to 8. Consequently, the NSGA-III using 8 threads nearly converged within 20 min, while the NSGA-III using 4 threads nearly converged within 40 min. However, the convergence speed is less doubled as the number of threads increases from 4 to 8, e.g., the HV of using 4 threads at 10 min is 0.64 while the HV of using 8 threads at 5 min is 0.45. This is explained by the rising communication overhead caused by more threads, such that a higher speedup is increasingly compensated by this communication overhead.

Table 4

Convergence performance for algorithmic saturation in small- (8×27×8) and large-sized (200×600×20) problem instances

Problem instance	Number of threads	Runtime (min)	HV (hypervolume)	Did NSGA-III converge?
8×27×8	4	3	0.26	Nearly yes
8×27×8	4	5	0.28	Yes
8×27×8	4	10	0.28	Yes
200×600×20	4	3	0.21	No
200×600×20	4	5	0.33	No
200×600×20	4	10	0.64	No
200×600×20	4	20	0.87	No
200×600×20	4	30	0.91	No
200×600×20	4	40	0.93	Nearly yes
200×600×20	8	3	0.22	No
200×600×20	8	5	0.45	No
200×600×20	8	10	0.75	No
200×600×20	8	20	0.96	Nearly yes
200×600×20	8	30	0.98	Yes
200×600×20	8	40	0.98	Yes

5.6 Comparison of different algorithms for many-objective optimization

The tailored NSGA-III and seven other multiobjective evolutionary algorithms (MOEAs) were benchmarked to compare their many-objective optimization performance for the proposed problem (EL-FJSSP). The seven other MOEAs either have been widely used and highly cited in the past 10 years in the multiobjective optimization domain, or were recently proposed for many-objective optimization. These seven MOEAs include NSGA-II [44], ε-MOEA (epsilon-MOEA) [45], SPEA2 (strength-based evolutionary algorithm) [46], SMS-MOEA (S-metric Selection MOEA) [47], VEGA (vector evaluated genetic algorithm) [48], I-DBEA (improved decomposition-based evolutionary algorithm) [49], and RVEA (reference vector guided evolutionary algorithm) [50]. The generic operators for these benchmarking algorithms remain these in Sect. 4.3 and Sect. 4.4 to be consistent. The problem instance and other configurations remained these in Sect. 5.1. The experiment with each algorithm was performed under the two electricity pricing schemes RTP and ToUP, respectively. Both HV and IGD metrics were used to quantify the joint convergence and diversity

Table 5

Convergence performance comparison of different multiobjective optimization algorithms

Algorithm	HV (hypervolume)		IGD (inverted generational distance)		Number of nondominated solutions		Number of global nondominated solutions	
	RTP	ToUP	RTP	ToUP	RTP	ToUP	RTP	ToUP
NSGA-III	0.59	0.57	0.09	0.16	101	49	64	38
NSGA-II	0.52	0.46	0.13	0.15	150	71	110	14
ε-MOEA	0.18	0.27	0.27	0.23	112	70	0	65
SPEA2	0.50	0.51	0.15	0.14	100	80	47	39
SMS-MOEA	0.26	0.46	0.22	0.33	55	29	0	11
VEGA	0.16	2.5E-3	0.64	0.84	28	11	0	0
I-DBEA	0.56	0.01	0.10	0.76	112	14	103	0
RVEA	0.14	0	0.71	1.40	9	5	3	0

performance of these algorithms, respectively. The number of nondominated solutions was utilized to indicate the capability of an algorithm to differentiate solutions in a large-dimensional objective space. Besides, the number of global nondominated solutions was used to show how many nondominated solutions produced by an algorithm which are still nondominated among all the nondominated solutions provided by all benchmark algorithms. Finally, the runtime of each algorithm instance was set to 3 min.

Table 5 lists the benchmark results. The tailored NSGA-III exhibits superior convergence and diversity of NSGA-III for the proposed many-objective optimization problem. This is because it achieved the highest HV under RTP (0.59) and ToUP (0.57), the lowest IGD under RTP (0.09), and the near-lowest IGD under ToUP (0.16). More detailed discussions around Table 5 are given as follows.

For the problem instance under RTP (Table 5), the many-objective optimization performance (both HV and IGD) of NSGA-II, SPEA2, and I-DBEA is more or less close to that of NSGA-III. However, NSGA-II and I-DBEA output more nondominated solutions (150 and 112, respectively) than the tailored NSGA-III (101), exhibiting their weaker selection pressure of a population, such that they are less capable to distinguish solutions in the high-dimensional objective space compared to NSGA-III. On the one hand, this demonstrates the effectiveness of the reference direction-based diversity preservation scheme, which is introduced in NSGA-III to replace the crowding-distance-based density estimation scheme in NSGA-II [44]. On the other hand, this proves that I-DBEA is not as efficient as NSGA-III in balancing diversity and convergence, though both are designed based on the reference directions in the objective space. Furthermore, SPEA2 output less global nondominated solutions (47) compared to NSGA-III (64), implying its weaker convergence. This could be explained by the environment selection mechanism of SPEA2 (step 3 in [46]), which only employs the canonical dominance and may even include dominated solutions when selecting solutions as the next generation.

For the problem instance under ToUP (Table 5), the HV of SPEA2 (0.51) is close to that of NSGA-III (0.57) and the IGD of SPEA2 (0.14) is slightly lower than that of NSGA-III (0.16). Nonetheless, SPEA2 output evidently more nondominated solutions (80) than NSGA-III (49). This reveals its weaker selection pressure, as it does not employ additional measures to differentiate solutions besides the canonical dominance. Compared to the former instance under RTP where I-DBEA reached a high-performance level, I-DBEA obtained very poor performance in this instance under ToUP, i.e., HV = 0.01, IGD = 0.76. This illustrates that the performance of I-DBEA highly depends on the specific instance, in contrast to NSGA-III of which the superior performance widely holds.

For both problem instances in Table 5, the many-objective optimization performance of VEGA (HV = 2.5E-3, IGD = 0.84) and RVEA (HV = 0, IGD = 1.40) is evidently poor, respectively. The poor performance of VEGA would be due to its selection scheme [48] that was proposed for bi- or tri-objective optimization problems. The poor performance of RVEA would be explained by the fact that in the original experiments of RVEA [50], only the simulated binary crossover and the polynomial mutation were tested and explicitly recommended in the algorithm design, such that the actual performance of RVEA highly depends on the generic operators that are used.

6. Conclusion

This paper proposes an energy- and labor-aware flexible job shop scheduling problem (EL-FJSSP) under dynamic electricity pricing and tailors a NSGA-III for many-objective optimization of this problem. The contributions are threefold, covering problem modeling, solution algorithm, and analytics. (1) Machine energy consumption, human worker, and labor shift are jointly modeled and integrated in the EL-FJSSP. In this way, the intrinsic trade-off relation between the energy cost and the labor cost under dynamic electricity pricing is captured, instead of being ignored in the existing research on energy-aware FJSSP, and human workers can be matched to the scheduled production, instead of isolated rostering and production scheduling in literature. (2) For the first time, the NSGA-III is tailored for many-objective optimization of a FJSSP. Discrete-event simulation (DES) is used to build a digital twin of the energy- and labor-aware flexible job shop. DES is coupled with the tailored NSGA-III for efficient shop floor simulation and scheduling solution evaluation. (3) Through numerical experiments under different dynamic electricity pricing schemes, the underlying relations among the important production objectives are quantitatively revealed, i.e., makespan, total energy cost, total labor cost, maximal workload, and total workload. The effectiveness and efficiency of applying a NSGA-III in solving a many-objective EL-FJSSP are demonstrated by benchmarking with NSGA-II.

Some important conclusions are draws as follows. (1) It is of economic importance to model the labor aspect in an energy-aware FJSSP under dynamic electricity pricing, due to the demonstrated conflict between the energy cost and the labor cost. Although the portion of both cost parts in the overall production cost varies, the consideration of both energy and labor aspects in a FJSSP increases the flexibility of the model, when facing diverse industrial application cases. (2) In contrast to integer programming in a commercial off-the-shelf solver (e.g., IBM CPLEX) which has intrinsic problems of intractability and scalability, the DES is an effective and efficient approach to implement a complex model, e.g., the proposed EL-FJSSP. (3) NSGA-III is more effective and efficient than NSGA-II in solving a many-objective EL-FJSSP, due to a higher hypervolume (HV) and a lower number of nondominated solutions in all experiments. This conclusion may still hold for other variants of FJSSP or even other scheduling problems. (4) The HV is a more suitable metric in an EL-FJSSP, compared to the inverted generational distance (IGD), as the IGD has difficulty in differentiating multiobjective optimization performance of different algorithms. This conclusion may hold for other scheduling problems.

Future research can be performed in the following aspects. (1) Investigation on whether conclusions 3&4 still hold for other production scheduling problems or other scheduling problems. (2) Design of a novel evolutionary algorithm for many-objective optimization in the context of fast yet high-quality decision making. (3) Integration of intra-factory transportation in the current EL-FJSSP and its implementation in a DES environment.

Acknowledgement

This research was supported by the Belgium FWO Travel Grant (V416217N).

References

[1] R. Deng, Z. Yang, M. Y. Chow, and J. Chen, "A Survey on Demand Response in Smart Grids: Mathematical Models and Approaches," *IEEE Transactions on Industrial Informatics,* vol. 11, no. 3, pp. 570-582, 2015.

[2] X. Gong, T. De Pessemier, W. Joseph, and L. Martens, "A generic method for energy-efficient and energy-cost-effective production at the unit process level," *Journal of Cleaner Production,* vol. 113, no. Supplement C, pp. 508-522, 2016/02/01/ 2016.

[3] R. Zhang and R. Chiong, "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," *Journal of Cleaner Production,* vol. 112, no. Part 4, pp. 3361-3375, 2016/01/20/ 2016.

[4] Y. Zhang, J. Wang, and Y. Liu, "Game theory based real-time multi-objective flexible job shop scheduling considering environmental impact," *Journal of Cleaner Production,* vol. 167, no. Supplement C, pp. 665-679, 2017/11/20/ 2017.

[5] Y. Wang and L. Li, "Critical peak electricity pricing for sustainable manufacturing: Modeling and case studies," *Applied Energy,* vol. 175, pp. 40-53, 8/1/ 2016.

[6] S. Kim, C. Meng, and Y.-J. Son, "Simulation-based machine shop operations scheduling system for energy cost reduction," *Simulation Modelling Practice and Theory,* vol. 77, no. Supplement C, pp. 68-83, 2017/09/01/ 2017.

[7] X. Gong *et al.*, "Energy- and Labor-aware Production Scheduling for Sustainable Manufacturing: A Case Study on Plastic Bottle Manufacturing," *Procedia CIRP,* vol. 61, no. Supplement C, pp. 387-392, 2017/01/01/ 2017.

[8] X. Gong *et al.*, "Integrating labor awareness to energy-efficient production scheduling under real-time electricity pricing: An empirical study," *Journal of Cleaner Production,* vol. 168, no. Supplement C, pp. 239-253, 2017/12/01/ 2017.

[9] M. L. Pinedo, "Scheduling," in "Deterministic Models: Preliminaries," 2016.

[10] N. B. Ho and J. C. Tay, "GENACE: an efficient cultural algorithm for solving the flexible job-shop problem," in *Proceedings of the 2004 Congress on Evolutionary Computation*, 2004, vol. 2, pp. 1759-1766 Vol.2.

[11] Framinan J.M., Leisten R., and R. G. R., "Manufacturing Scheduling Systems," in "Overview of Manufacturing Scheduling," Springer, London2014.

[12] Hisao Ishibuchi, Naoya Akedo, and Y. Nojima, "Behavior of Multiobjective Evolutionary Algorithms on Many-Objective Knapsack Problems," *IEEE Transactions on Evolutionary Computation,* vol. 19, pp. 264-283, 2015.

[13] B. Li, J. Li, K. Tang, and X. Yao, "Many-Objective Evolutionary Algorithms: A Survey," *ACM Comput. Surv.,* vol. 48, no. 1, pp. 1-35, 2015.

[14] J. Salas and V. Yepes, "A discursive, many-objective approach for selecting more-evolved urban vulnerability assessment models," *Journal of Cleaner Production,* vol. 176, pp. 1231-1244, 2018/03/01/ 2018.

[15] H. Son and C. Kim, "Evolutionary many-objective optimization for retrofit planning in public buildings: A comparative study," *Journal of Cleaner Production,* vol. 190, pp. 403-410, 2018/07/20/ 2018.

[16] G. May, B. Stahl, M. Taisch, and V. Prabhu, "Multi-objective genetic algorithm for energy-efficient job shop scheduling," *International Journal of Production Research,* vol. 53, no. 23, pp. 7071-7089, 2015/12/02 2015.

[17] T. Stock and G. Seliger, "Multi-objective shop floor scheduling using monitored energy data," *Procedia CIRP,* vol. 26, no. Supplement C, pp. 510-515, 2015/01/01/ 2015.

[18] Y. Liu, H. Dong, N. Lohse, and S. Petrovic, "A multi-objective genetic algorithm for optimisation of energy consumption and shop floor production performance," *International Journal of Production Economics,* vol. 179, no. Supplement C, pp. 259-272, 2016/09/01/ 2016.

[19] M. Dai, D. Tang, Y. Xu, and W. Li, "Energy-aware integrated process planning and scheduling for job shops," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture,* vol. 229, no. 1_suppl, pp. 13-26, 2015/02/01 2014.

[20] J.-Y. Moon and J. Park, "Smart production scheduling with time-dependent and machine-dependent electricity cost by considering distributed energy resources and energy storage," *International Journal of Production Research,* vol. 52, no. 13, pp. 3922-3939, 2014/07/03 2014.

[21] A.-Q. Firas and G. Denis, "A multi-objective genetic method minimizing tardiness and energy consumption during idle times," *IFAC-PapersOnLine,* vol. 48, no. 3, pp. 1216-1223, 2015/01/01/ 2015.

[22] C. A. Garcia-Santiago, J. Del Ser, C. Upton, F. Quilligan, S. Gil-Lopez, and S. Salcedo-Sanz, "A random-key encoded harmony search approach for energy-efficient production scheduling with shared resources," *Engineering Optimization,* vol. 47, no. 11, pp. 1481-1496, 2015/11/02 2015.

[23] Y. He, Y. Li, T. Wu, and J. W. Sutherland, "An energy-responsive optimization method for machine tool selection and operation sequence in flexible machining job shops," *Journal of Cleaner Production,* vol. 87, no. Supplement C, pp. 245-254, 2015/01/15/ 2015.

[24] S. Kemmoé, D. Lamy, and N. Tchernev, "A job-shop with an energy threshold Issue considering operations with consumption peaks," *IFAC-PapersOnLine,* vol. 48, no. 3, pp. 788-793, 2015/01/01/ 2015.

[25] Y. Liu, H. Dong, N. Lohse, and S. Petrovic, "Reducing environmental impact of production during a Rolling Blackout policy – A multi-objective schedule optimisation approach," *Journal of Cleaner Production,* vol. 102, no. Supplement C, pp. 418-427, 2015/09/01/ 2015.

[26] D. Tang and M. Dai, "Energy-efficient approach to minimizing the energy consumption in an extended job-shop scheduling problem," *Chinese Journal of Mechanical Engineering,* vol. 28, no. 5, pp. 1048-1055, 2015/09/01 2015.

[27] M. A. Salido, J. Escamilla, A. Giret, and F. Barber, "A genetic algorithm for energy-efficiency in job-shop scheduling," *The International Journal of Advanced Manufacturing Technology,* vol. 85, no. 5, pp. 1303-1314, 2016/07/01 2016.

[28] W. Xu, L. Shao, B. Yao, Z. Zhou, and D. T. Pham, "Perception data-driven optimization of manufacturing equipment service scheduling in sustainable manufacturing," *Journal of Manufacturing Systems,* vol. 41, no. Supplement C, pp. 86-101, 2016/10/01/ 2016.

[29] D. Giglio, M. Paolucci, and A. Roshani, "Integrated lot sizing and energy-efficient job shop scheduling problem in manufacturing/remanufacturing systems," *Journal of Cleaner Production,* vol. 148, no. Supplement C, pp. 624-641, 2017/04/01/ 2017.

[30] H. Mokhtari and A. Hasani, "An energy-efficient multi-objective optimization for flexible job-shop scheduling problem," *Computers & Chemical Engineering,* vol. 104, no. Supplement C, pp. 339-352, 2017/09/02/ 2017.

[31] M. A. Salido, J. Escamilla, F. Barber, and A. Giret, "Rescheduling in job-shop problems for sustainable manufacturing systems," *Journal of Cleaner Production,* vol. 162, no. Supplement, pp. S121-S132, 2017/09/20/ 2017.

[32] L. Yin, X. Li, L. Gao, C. Lu, and Z. Zhang, "A novel mathematical model and multi-objective method for the low-carbon flexible job shop scheduling problem," *Sustainable Computing: Informatics and Systems,* vol. 13, no. Supplement C, pp. 15-30, 2017/03/01/ 2017.

[33] L. Zhang, Q. Tang, Z. Wu, and F. Wang, "Mathematical modeling and evolutionary generation of rule sets for energy-efficient flexible job shops," *Energy,* vol. 138, no. Supplement C, pp. 210-227, 2017/11/01/ 2017.

[34] M. L. Pinedo, *Scheduling*, 5 ed. Springer International Publishing, 2016.

[35] G. Zhang, X. Shao, P. Li, and L. Gao, "An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem," *Computers & Industrial Engineering,* vol. 56, no. 4, pp. 1309-1318, 2009/05/01/ 2009.

[36] K. Deb and H. Jain, "An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems With Box Constraints," *IEEE Transactions on Evolutionary Computation,* vol. 18, no. 4, pp. 577-601, 2014.

[37] C. Bierwirth, "A generalized permutation approach to job shop scheduling with genetic algorithms," *Operations-Research-Spektrum,* journal article vol. 17, no. 2, pp. 87-92, 1995.

[38] P. Larrañaga, C. M. H. Kuijpers, R. H. Murga, I. Inza, and S. Dizdarevic, "Genetic algorithms for the travelling salesman problem: A review of representations and operators," *Artificial Intelligence Review,* journal article vol. 13, no. 2, pp. 129-170, April 01 1999.

[39] I. Kacem, S. Hammadi, and P. Borne, "Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews),* vol. 32, no. 1, pp. 1-13, 2002.

[40] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach," *IEEE Transactions on Evolutionary Computation,* vol. 3, no. 4, pp. 257-271, 1999.

[41] Q. Zhang, A. Zhou, S. Z. Zhao, P. N. Suganthan, W. Liu, and S. Tiwari, "Multiobjective optimization test instances for the CEC-2009 special session and competition," in "Nanyang Technological University Technical Report," Singapore2008, Available: http://www.ntu.edu.sg/home/epnsugan/.

[42] I. Das and J. E. Dennis, "Normal-Boundary Intersection: A New Method for Generating the Pareto Surface in Nonlinear Multicriteria Optimization Problems," *SIAM Journal on Optimization,* vol. 8, no. 3, pp. 631-657, 1998.

[43] Belpex. (2017). *Belgium power exchange*. Available: http://www.belpex.be/

[44] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation,* vol. 6, no. 2, pp. 182-197, 2002.

[45] K. Deb, M. Mohan, and S. Mishra, "A fast multi-objective evolutionary algorithm for finding well-spread pareto-optimal solutions," KanGAL2003002, 2003.

[46] Z. E., M. Laumanns, and L. Thiele, "SPEA2: Improving the Strength Pareto Evolutionary Algorithm," 2001.

[47] N. Beume, B. Naujoks, and M. Emmerich, "SMS-EMOA: Multiobjective selection based on dominated hypervolume," *European Journal of Operational Research,* vol. 181, no. 3, pp. 1653-1669, 2007/09/16/ 2007.

[48] J. D. Schaffer, "Multiple Objective Optimization with Vector Evaluated Genetic Algorithms," presented at the Proceedings of the 1st International Conference on Genetic Algorithms, 1985.

[49] M. Asafuddoula, T. Ray, and R. Sarker, "A Decomposition-Based Evolutionary Algorithm for Many Objective Optimization," *IEEE Transactions on Evolutionary Computation,* vol. 19, no. 3, pp. 445-460, 2015.

[50] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "A Reference Vector Guided Evolutionary Algorithm for Many-Objective Optimization," *IEEE Transactions on Evolutionary Computation,* vol. 20, no. 5, pp. 773-791, 2016.