A memetic algorithm for energy-efficient scheduling of integrated production and shipping

Jian Chen, Tong Ning, Gangyan Xu & Yang Liu

Published online: 08 Feb 2022.

Submit your article to this journal

Article views: 879

View related articles

View Crossmark data

Taylor & Francis
Taylor & Francis Group

A memetic algorithm for energy-efficient scheduling of integrated production and shipping

Jian Chen [a], Tong Ning[a], Gangyan Xu [b] and Yang Liu [c,d]

aCollege of Economics and Management, Nanjing University of Aeronautics and Astronautics, Nanjing, P. R. China; bDepartment of Aeronautical and Aviation Engineering, Faculty of Engineering, The Hong Kong Polytechnic University, Kowloon, Hong Kong SAR, P. R. China; cDepartment of Management and Engineering, Linköping University, Linköping, Sweden; dDepartment of Production, University of Vaasa, Vaasa, Finland

ABSTRACT

Energy-efficient manufacturing is critical as the industrial sector accounts for a substantial portion of global energy consumption. This research aims to address an energy-efficient scheduling problem of production and shipping for minimizing both makespan and energy consumption. It contributes to an integrated energy-efficient production and shipping system, which is separately studied in most existing research. The production stage allocates jobs onto unrelated parallel machines that can be shut off and adjust their cutting speed to save energy. The shipping stage aims to allocate jobs to vehicles of various sizes with varied unit energy consumption. The problem is modelled as a mixed-integer quadratic program. Considering its complexity, a memetic algorithm (MA) is proposed to incorporate a knowledge-driven local search strategy considering the balance between exploration and exploitation. Two dominance rules are derived from the characteristics of the specific problem and embedded into the proposed MA to enhance its performance. Experimental results demonstrate that the proposed MA outperforms two other population-based algorithms, genetic algorithm and traditional MA, in terms of performance and computing time. This research practically contributes to improving productivity and energy efficiency for the production-shipping supply chain of make-to-order products.

1. Introduction

With the rapid development of industrialization and urbanization, energy demand significantly increases in these years. Energy consumption has increased by 300% in the last 50 years (Park et al. 2009). A recent report from the International Energy Agency anticipated that the global energy need by 2040 would rise by 37% (Che et al. 2017). Excessive energy consumption, especially non-renewable resources (e.g. coal, gas and oil), incurs many environmental issues, such as air pollution and global climate change. Based on the statistical data, nearly 47% of Germany's total national electricity is consumed by industrial sectors (Dai et al. 2013). Thus, industrial sectors are primarily responsible for energy savings and reducing environmental impacts (Liu et al. 2020; Zeng et al. 2018).

Industrial sectors have adopted many strategies to approach energy-saving targets, e.g. using low-energy machines or vehicles. From the operational management perspective, energy-efficient scheduling could serve as an effective and powerful method

for energy savings without extra investment in equipment or technologies. Energy-efficient scheduling aims to optimally allocate limited resources to the given tasks considering the energy savings goal in addition to other traditional machine scheduling goals, such as minimization of makespan or operational cost. Recently, many efforts have been devoted to energy-efficient scheduling problems in the literature (Gahm et al. 2016; Mouzon and Yildirim 2008).

Power-down and speed-scaling mechanisms are widely used for energy-efficient scheduling in the literature. The earliest research on the power-down mechanism was investigated by Mouzon, Yildirim, and Twomey (2007), who turned off the non-bottleneck machines during their idle time to reduce energy consumption. Their illustrated example showed that a machine stays idle on average 16% of an 8-hour shift. Using the power-down mechanism can save at least 13% of energy. Yao et al. (1995) first introduced the speed-scaling mechanism when scheduling jobs in computer operating systems. Their

model balanced the minimization of completion time of jobs and energy consumption, considering that the faster a processor runs, the more electricity it takes per unit of work.

Besides considering scheduling within manufacturing shop floors, the authors jointly consider downstream shipping scheduling for make-to-order business models. By doing so, customized products can be directly delivered from manufacturing companies to customers, leading to (near) zero finished product inventory (Chen, Huang, and Wang 2019; Leung and Chen 2013). The research on integrated production and shipping at the detailed scheduling level is relatively recent (Chen et al. 2015; Chen 2010). Most of these studies attempt to optimize detailed production and shipping scheduling by considering delivery costs, customer service and makespan. However, limited efforts have been devoted to energy-efficient scheduling of integrated production and shipping.

This research considers an integrated system and investigates an *energy-efficient production and shipping scheduling (EPSS) problem*. The production part is comprised of unrelated parallel machines using power-down and speed-scaling mechanisms. The shipping part consists of heterogeneous vehicles which are associated with different sizes and energy consumption rates. The objective is to minimize the total cost of makespan and energy consumption. To the best of the authors' knowledge, this research is among the first to consider EPSS with power-down and speed-scaling mechanisms.

The contribution is threefold. First, the research is the pioneer to study EPSS considering power-down and speed-scaling mechanisms. A new mixed-integer quadratic model is developed for an integrated production and shipping system to minimize both energy consumption and makespan. To some extent, the research practically contributes to energy and lead time reduction for the supply chain of make-to-order products. Second, an MA is proposed to incorporate a newly designed knowledge-driven local search strategy, which efficiently solves the complex problem. Third, two problem-specific dominance rules are derived from the characteristics of the specific problem. The performance of the proposed MA is further enhanced after incorporating a rule-based dominance adjustment, which is different from other evolutionary algorithms. This research benefits

increasing productivity while reducing energy consumption for the production-shipping supply chain of make-to-order products.

The reminders are organized as follows. Section 2 is a literature review of relevant research to show the research gap explicitly. Section 3 formulates and models the problem and derives the properties of the EPSS problem. Section 4 presents the details of the proposed MA. Section 5 conducts a series of experimental experiments and analyses the results. Section 6 concludes the research and suggests possible future directions.

2. Literature review

Energy-efficient scheduling has attracted increasing attention from manufacturing companies due to rising energy prices and growing environmental concerns (Rager, Gahm, and Denz 2015; Zeng et al. 2018). To the best of the authors' knowledge, the pioneer study on energy-related production scheduling was conducted by Subai, Baptiste, and Niel (2006). Since then, more and more research has been done on energy-efficient scheduling in various manufacturing environments, i.e. single/parallel machine and flow shop.

The first study on energy-efficient scheduling of a single machine was conducted by Mouzon and Yildirim (2008). They considered two objectives, i.e. total energy consumption and total tardiness, to minimize. They proposed a meta-heuristic to generate multiple Pareto solutions, from which the analytical hierarchy process method was used to select the best alternative. Fang et al. (2016) studied a single machine to minimize the total electricity cost, considering the policy of time-of-use (TOU) electricity tariffs. The uniform-speed and speed-scalable cases were considered and addressed by approximation algorithms. Mouzon, Yildirim, and Twomey (2007) were among the first to investigate the power-down mechanism that machines could be turned off instead of keeping running during idle time. Shrouf et al. (2014) further considered the power-down mechanism with fluctuating energy prices. They generated near-optimal solutions by a genetic algorithm (GA) and optimal analytical solutions for a single machine, respectively. Che et al. (2017) adopted the power-down mechanism to schedule a single machine to minimize tardiness and energy consumption. To address the bi-

objective scheduling problem, an ε-constraint method incorporated with local search was presented to successfully find the exact Pareto front. Rubaiee and Yildirim (2019) attempted to schedule a single machine scheduling with pre-emption under the TOU mechanism to minimize energy consumption and total completion time. Ant colony optimization with a dominance ranking procedure was adapted to address the problem.

Another stream of energy-efficient scheduling research focused on the parallel machine environment. The speed-scaling mechanism that the dynamically adjusted processing speed of machines was often used in these works. Generally, a higher cutting speed will lead to less processing time but more energy consumption. Fang and Lin (2013) studied how to schedule jobs to parallel processors and adjust the computing speed of processors to optimize the weighted tardiness and energy consumption. They proposed a particle swarm optimization algorithm with a new encoding operator to generate approximate solutions. Wang, Wang, and Lin (2017) investigated how to assign jobs to parallel machines and determine machines' cutting speed to minimize the makespan of jobs considering the bound of electricity load. Shioura et al. (2018) considered a speed-scaling mechanism in an immediate start scheduling problem that jobs must be exactly started at their release time. The cost functions included the quality of service as well as the cost of running. Single and parallel machine cases were both analysed. Wu and Che (2019) considered the environment of unrelated parallel machines with a speed-scaling mechanism. They proposed an effective memetic differential evolution algorithm with an adaptive learning strategy-driven local search to minimize both makespan and energy consumption.

The energy-efficient flow shop scheduling research has been gradually emerging recently (Ding, Song, and Wu 2016). Mansouri, Aktas, and Besikci (2016) adopted the speed-scale mechanism in a two-machine permutation flow shop. Considering the conflict of makespan and energy consumption, they built a bi-objective mixed-integer programming (MIP) model and developed a constructive heuristic to address the problem in a reasonable time. Lu et al. (2017) considered energy consumption, sequence-related setup time and belt transportation time in a permutation flow shop. A backtracking search algorithm was presented to search Pareto front scheduling solutions. Zheng et al. (2020) studied a two-stage permutation flow shop considering the blocking constraint, the TOU and speeding scaling machines. They developed a MIP model, which was addressed by an ant colony optimisation algorithm. Öztop et al. (2020) focused on a multi-stage permutation flow shop with similar settings, i.e. speeding scaling machines. They developed three heuristics to obtain the Pareto schedule considering both energy conservation and production efficiency. Yüksel et al. (2020) adopted a speed-scale mechanism to minimize energy consumption and tardiness for a permutation flow shop with no-wait characteristics. They proposed three multi-objective heuristics based on the artificial bee colony algorithm (ABC) and GA.

This research is somewhat related to the energy-efficient scheduling of a hybrid flow shop (HFS). Luo et al. (2013) incorporated the consideration of electric power consumption with TOU prices into traditional HFS scheduling. Ant colony optimization was adopted due to the complexity of the problem. Yan et al. (2016) studied an integrated optimization of cutting parameters and scheduling in an HFS. Several machine states, such as warm-up, standby setup, tool change and machining, were considered for cutting parameters optimization. A GA was used to minimize makespan and energy consumption simultaneously. Lei, Gao, and Zheng (2017) adopted a speed-scale mechanism to schedule an HFS to minimize total tardiness and energy consumption. They adopted a teaching-learning-based algorithm to capture the characteristics of human learning. Li et al. (2018) considered an HFS with the setup energy consideration and proposed a multi-objective heuristic with an adaptive neighbourhood operator based on the problem structure. Through adopting the power-down mechanism, the energy-efficient scheduling problem of HFS with unrelated parallel machines was studied (Meng et al. 2019). They built five MIP-based models, which were addressed by an improved GA. Wang et al. (2020) considered both TOU and power-down mechanisms to schedule an HFS. The system consists of two stages, in which stage 1 has parallel, and stage 2 has batch machines. They proposed an augmented ε-constraint method for small-scale problems, a constructive heuristic for medium-scale problems, and an ant colony optimization algorithm for large-scale problems. Gong et al. (2020a)

studied energy reduction in scheduling a flexible flow shop considering both worker flexibility and machine flexibility, in which the worker costs are incorporated into the makespan and energy-related indicators. They proposed a hybrid evolutionary algorithm with newly designed operators and neighbourhood search. Zhang et al. (2021) investigated TOU in a two-stage hybrid flow shop that the first stage has speed-scaling parallel machines and the second stage has unrelated parallel machines.

Besides, there are a few studies on energy-efficient scheduling in other environments, such as job shop (May et al. 2015; Mokhtari and Hasani 2017; Zhang, Wang, and Liu 2017), distributed manufacturing (Gong et al. 2020b), dynamic scheduling (Wang et al. 2018; Nouiri, Bekrar, and Trentesaux 2020) and batching scheduling (Liu 2014), which are not detailed in this paper.

A review paper by Gao et al. (2020) indicates that processing product, machine idle, machine setup and on/off, and product or component transportation are considered most in energy demand. To achieve energy-efficient production, more than 30% of reviewed literature consider idle energy reduction, more than 30% consider processing energy reduction and more than 20% consider setup energy reduction. This is in line with the data in the review paper of Gahm et al. (2016), which reports that most of the research focuses on machine idle, machine on, machine off in non-processing energy demand and machine-related in processing energy demand. Thus, the authors attempt to reduce machine idle energy consumption through the power-down mechanism. Considering that the power-down mechanism may not be practical for some machines, the proposed mixed solution incorporates the speed-scale mechanism.

It is observed from Table 1 that most of the energy-efficient scheduling related research focuses on production environments, i.e. single (S), parallel (P) or flow shop-related (F) shop floor. Most research also adopts a single energy mechanism, such as either power-down (On/Off), speed-scaling (Speed) or TOU mechanisms. Most research considers multi-objective, such as total energy consumption and production-related cost, i.e. makespan, total completion time or total tardiness. The research gap is identified. First, this

paper considers a two-stage system of production and shipping. The first stage consists of unrelated parallel machines with power-down and speed-scaling mechanisms, and the second stage considers heterogeneous vehicles with distinct sizes and energy consumption rates. Most of the research related to energy-efficient scheduling focus on the manufacturing domain. There is limited energy-efficient scheduling research considering both production and shipping. Only a few papers (Nouiri, Bekrar, and Trentesaux 2020; Zhang et al. 2021) are somewhat related, but the shipping scheduling problems they considered are within the production shop floor rather than outside shipping like ours. To the best of the authors' knowledge, it is the first attempt at energy-efficient scheduling of integrated production and shipping considering power-down and speed-scaling mechanisms. Second, the authors propose a new memetic algorithm with a novel knowledge-driven local search and two dominance rules derived from the characteristics of the proposed EPSS.

3. Problem formulation, modelling and properties

This section first mathematically formulates the energy-efficient production and shipping scheduling problem, then presents its mathematical model and finally derives the properties of the problem.

3.1. Problem formulation

Consider an integrated system comprised of production and shipping. The production stage has a set M of unrelated machines. Machines can be shut off at idle time but requiring extra reset energy to restart. Each machine $i \in M$ can run at a different speed mode $v \in V$. Usually, higher cutting speed contributes to better machining productivity while it requires higher energy consumption. Denote u_v as processing speed factor for different processing speeds of machines. There are a set N of jobs arriving at the system at its release time r_j, $j \in N$. The processing time of job j is given as t_j. Thus, the actual time used for processing job j at speed mode v is equal to t_j/u_v.

Table 1. Classification of related energy-efficient scheduling studies.

| References	Production environments			With shipping	Energy mechanisms				Objectives	Methods
	S	P	F		On/Off	Speed	TOU	Others		
Mouzon, Yildirim, and Twomey (2007)	✓				✓				Energy+completion time	Dispatching rules
Mouzon and Yildirim (2008)	✓				✓				Energy + tardiness	Adaptive search metaheuristic
Shrouf et al. (2014)	✓				✓		✓		Energy	Genetic algorithm
Fang et al. (2016)	✓					✓	✓		Energy	Approximation algorithm
Che et al. (2017)	✓				✓				Energy + tardiness	ε−constraint method
Rubaiee and Yildirim (2019)	✓						✓		Energy+ completion time	Ant colony optimization
Fang and Lin (2013)		✓				✓			Energy + tardiness	Particle swarm optimization
Wang, Wang, and Lin (2017)		✓						✓	Makespan	Heuristics
Shioura et al. (2018)	✓	✓				✓			Energy+ completion time	Theoretical analysis
Wu and Che (2019)		✓				✓			Energy + makespan	Memetic differential evolution algorithm
Ding, Song, and Wu (2016)					✓	✓			Carbon emissions + makespan	Multi-objective NEH algorithm and iterated greedy algorithm
Mansouri, Aktas, and Besikci (2016)			✓			✓			Energy + makespan	Constructive heuristic
Luo et al. (2013)			✓				✓		Energy + makespan	Ant colony optimization
Yan et al. (2016)			✓					✓	Energy + makespan	Genetic Algorithm
Lei, Gao, and Zheng (2017)			✓			✓			Energy and tardiness	Teaching-learning-based algorithm
Lu et al. (2017)			✓		✓				Energy + makespan	Backtracking search algorithm
Li et al. (2018)			✓					✓	Energy + makespan	Multi-objective optimization algorithm
Meng et al. (2019)			✓		✓				Energy	Improved genetic algorithm
Zheng et al. (2020)			✓				✓		Energy + makespan	Multi-objective hybrid ant colony optimisation
Wang et al. (2020)			✓		✓		✓		Energy + makespan	ε−constraint method and meta-heuristics
Gong et al. (2020a)			✓						Energy+ makespan + worker cost	Hybrid evolutionary algorithm
Öztop et al. (2020)			✓			✓			Energy + flowtime	Augmented epsilon-constraint approach
Zhang et al. (2021)			✓			✓	✓		Energy	Tabu search-greedy insertion hybrid algorithm
Yüksel et al. (2020)			✓			✓			Energy + tardiness	Artificial bee colony, genetic algorithm
This research		✓		✓	✓				Energy + makespan	Memetic algorithm
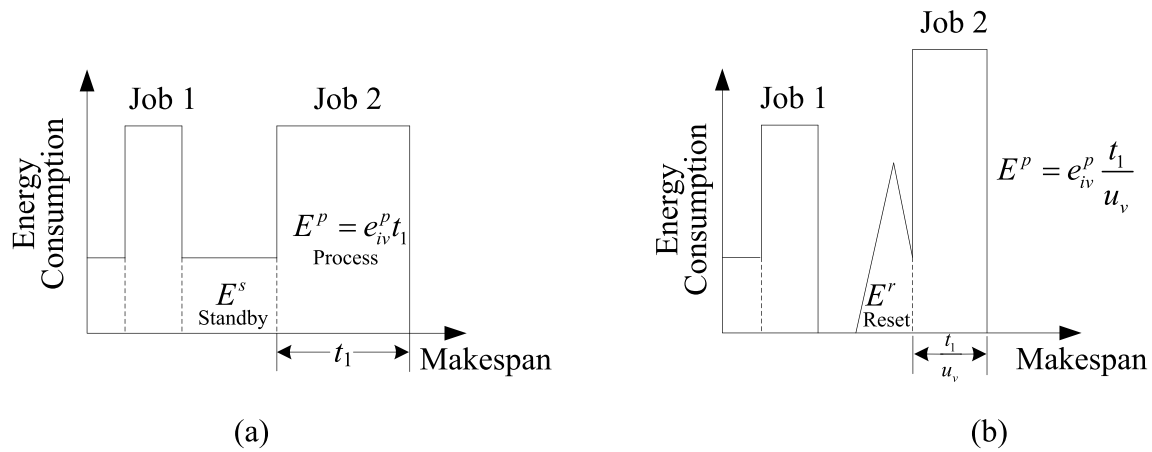
Figure 1. Schematic diagram of the energy-efficient scheduling scheme.

There are a set H of heterogeneous vehicles for shipping jobs to customers. Each job cannot be delivered to its customer in a separate vehicle. Once a vehicle is assigned to a job, it cannot be released until the delivery is finished. The round shipping time of job j is denoted as t'_j. Each vehicle has its capacity, Q_h, $h \in H$. Each job has its size q_j. The capacity constraint $q_j \leq Q_h$ satisfies if vehicle h is used to deliver job j. This assumption fits make-to-order production, especially for one-of-a-kind production (OKP), such as die and mould. OKP products are delivered in dedicated vehicles to customers so that the capacity constraint should be satisfied. Usually, the parameter q_j can be size or weight that dominates the capacity constraint of shipping. It prefers size for shipping large-size products, weight for heavy products. For OKP products, manufacturing starts only after a customer order is received. Thus, order fulfilment is lead time-sensitive and needs an integrated optimization of production and shipping scheduling.

Three types of production energy consumption are commonly considered: process energy, reset energy and standby energy (Dai et al. 2013; Zhang, Wang, and Liu 2017). The shipping energy is considered due to the fuel consumption of vehicles for shipping. In detail, the types of energy consumption to be minimized are listed as follows.

- The process energy E^p accounts for the total energy consumed during the cutting process, such as the energy consumed in the spindle drive unit. The process energy consumed per unit time for machine i at speed mode v is denoted as e^p_{iv}.
- The reset energy E^r is consumed when machines switch from off-state back to work-state. The reset energy consumed per unit time for machine i is e^r_i. The reset time each time for machine i is denoted as t^r_i.
- The standby energy refers to the energy consumed by machines on-state when they are not working, for instance, waiting for the next job. The standby energy consumed per unit time for machine i is denoted as e^s_i.
- The shipping energy E^l is considered due to the fuel consumption of vehicles during shipping. The fuel consumption depends on vehicle features as well as shipping time. The shipping energy consumed per unit time for vehicle h is denoted as e'_h.

In addition to energy consumption, the objective is also to minimize makespan. Minimizing makespan is widely adopted in many scheduling literatures, which equals maximizing machine/vehicle utilization (Pinedo 2012). Specifically, the makespan is defined as the duration between the production's starting time and the shipping's completion time $C'_{\max} = \max_{1 \leq j \leq n}(C'_j) - \min_{1 \leq j \leq n}(r_j)$.

Therefore, the studied problem is to find an integrated production and shipping schedule to minimize the total cost of makespan and energy consumption. The studied problem is named *the energy-efficient production and shipping scheduling (EPSS) problem*.

Specifically, the integrated schedule contains two parts: the production schedule and the shipping schedule. The production schedule determines the allocation and sequencing of a set of jobs onto the machines, simultaneously making the *machine state* decisions, i.e. speed mode for processing jobs and on-off for idle time. The shipping schedule determines which vehicle is used to deliver the jobs to customers in what sequence. Note that routing-related decisions are not considered in this problem. Inner transportation is not considered because its transportation time is usually much shorter than that of outside shipping.

As shown in Figure 1, take job 2 for example. Two machine state decisions should be made. First, machine on or off decisions will incur different energy consumptions. For example, if the machine keeps on at the idle time, it consumes E^s on standby in Figure 1 (a), while the energy consumption is E^r on the setup in Figure 1(b). Second, machine speed will also lead to different energy consumptions as well as completion time of jobs. For example, faster speed leads to earlier completion but larger energy consumption in Figure 1 (b) than in Figure 1(a).

Some basic assumptions for scheduling problems are also considered: 1) Setup of different types of jobs is not considered. 2) Each machine is turned on at time 0. 3) Each machine can process only one job at a time. 4) Pre-emption of jobs is not permitted. 5) Each job can be processed on only one machine at a time. 6) Each job is available for shipping once its production is completed.

3.2. Mathematical modelling

Before introducing the mathematical model for the EPSS problem, the used notation is defined as follows.

Sets	
M	Set of unrelated machines, $i \in M$
N	Set of jobs, $j, k \in N$
V	Set of processing speed modes, $v \in V$
H	Set of vehicles, $h \in H$
Parameters	
r_j	Release time of job j, $j \in N$
t_j	Processing time of job j, $j \in N$
t'_j	Shipping time of job j from manufacturer to customer
u_v	Processing speed factors for different processing speeds, $v \in V$
t_i^r	Reset time for machine i per time
e_{iv}^p	Processing energy consumed per unit time for machine i at speed mode v
e_i^r	Reset energy consumed per unit time for machine i
e_i^s	Standby energy consumed per unit time for machine i
E^p	Processing energy consumption

(Continued)

Sets	
M	Set of unrelated machines, $i \in M$
N	Set of jobs, $j, k \in N$
V	Set of processing speed modes, $v \in V$
H	Set of vehicles, $h \in H$
Parameters	
r_j	Release time of job j, $j \in N$
t_j	Processing time of job j, $j \in N$
t'_j	Shipping time of job j from manufacturer to customer
u_v	Processing speed factors for different processing speeds, $v \in V$
t_i^r	Reset time for machine i per time
e_{iv}^p	Processing energy consumed per unit time for machine i at speed mode v
e_i^r	Reset energy consumed per unit time for machine i
e_i^s	Standby energy consumed per unit time for machine i
E^p	Processing energy consumption
E^r	Reset energy consumption
E^s	Standby energy consumption
B	A very large number
q_j	Size of job j
Q_h	Capacity of vehicle h
e'_h	Shipping energy consumed per unit time for vehicle h
E'	Shipping energy consumption
Decision variables	
C_j	Completion time of job j in production
χ_{ijk}	1 if job j immediately precedes job k on machine i, and 0 otherwise
η_{ijv}	1 if job j is processed on machine i at speed v, and 0 otherwise
θ_{ij}	1 if machine i is reset before processing job j, and 0 otherwise
C'_j	Completion time of job j for shipping
C'_{max}	The makespan
U_{hjk}	1 if job j is immediately dispatched before job k on vehicle h, and 0 otherwise
W_{hj}	1 if job j is assigned on vehicle h, and 0 otherwise

The bi-objective, i.e. makespan and energy consumption, are incorporated into a weighted single-objective. Thus, the objective of minimizing the total cost of makespan and energy consumption can be formulated as

$$C_{ME} = \alpha C'_{max} + \beta\left(E^p + E^r + E^s + E'\right) \qquad (1)$$

where α and β are the cost coefficients of makespan and energy consumption.

A weighted single-objective method is adopted instead of the bi-objective optimization considering several aspects. First, the cost coefficients are obtainable, such as from historical data or expert experiences. Second, it is straightforward and convenient for shopfloor managers to find a solution with minimum total cost rather than searching from a large amount of non-dominated Pareto solutions. Third, the cost coefficients could also be regarded as the weights of makespan and energy consumption, which can be used as a control tool by shopfloor managers according to realistic situations or

a normalization role for the different dimensional criteria. The weighted single-objective method is also widely used in many existing scheduling studies, such as Chen and Vairaktarakis (2005), Fang and Lin (2013), Gawiejnowicz and Suwalski (2014).

Finally, the EPSS problem is modelled as

$$\text{Min} C_{ME} \qquad (2)$$

Subject to

$$E^p = \sum_{i \in M} \sum_{j \in N} \sum_{v \in V} e^p_{iv} \frac{t_j}{u_v} \eta_{ijv}, \, i \in M, \, j \in N \qquad (3)$$

$$E^r = \sum_{i \in M} \sum_{j \in N} e^r_i t^r_i \theta_{ij}, \, i \in M, \, j \in N \qquad (4)$$

$$E^s = \sum_{i \in M} \sum_{k \in N} e^s_i \left(C_k - \sum_{v \in V} \frac{t_k}{u_v} \eta_{ikv} - \sum_{j \in N \cup \{0\}} C_j \chi_{ijk} \right) (1 - \theta_{ik}), \, j \neq k, \, i$$
$$\in M, \, j \in N \cup \{0\}, \, k \in N \qquad (5)$$

$$\sum_{i \in M} \sum_{v \in V} \eta_{ijv} = 1, \, j \in N \qquad (6)$$

$$\sum_{k \in N \cup \{0\}} \chi_{ijk} = \sum_{v \in V} \eta_{ijv}, \, j \neq k, \, i \in M, \, j \in N \cup \{0\} \qquad (7)$$

$$\sum_{j \in N \cup \{0\}} \chi_{ijk} = \sum_{v \in V} \eta_{ikv}, \, j \neq k, \, i \in M, \, k \in N \qquad (8)$$

$$C_k - C_j - B(\chi_{ijk} - 1) \geq \sum_{v \in V} \frac{t_k}{u_v} \eta_{ikv} + t^r_i \theta_{ik}, \, j \neq k, \, i$$
$$\in M, \, j \in N \cup \{0\}, \, k \in N \qquad (9)$$

$$C_j - \sum_{v \in V} \frac{t_j}{u_v} \eta_{ijv} - t^r_i \theta_{ij} \geq r_j, \, i \in M, \, j \in N \qquad (10)$$

$$\sum_{v \in V} \eta_{i0v} = 1, \, i \in M \qquad (11)$$

$$E' = \sum_{h \in H} \sum_{j \in N} e'_h t'_k W_{hj}, \, h \in H, \, j \in N \qquad (12)$$

$$\sum_{h \in H} W_{hj} = 1, \, j \in N \qquad (13)$$

$$\sum_{k \in N \cup \{0\}} U_{hjk} = W_{hj}, \, j \neq k, \, h \in H, \, j \in N \cup \{0\} \qquad (14)$$

$$\sum_{j \in N \cup \{0\}} U_{hjk} = W_{hk}, \, j \neq k, \, h \in H, \, k \in N \qquad (15)$$

$$C'_k - C'_j - B(U_{hjk} - 1) \geq t'_k, \, j \neq k, \, h \in H, \, j \in N \cup \{0\}, \, k \in N \qquad (16)$$

$$C'_j - W_{hj} t'_j \geq C_j, \, h \in H, \, j \in N \qquad (17)$$

$$Q_h - B(W_{hj} - 1) \geq q_j, \, h \in H, \, j \in N \qquad (18)$$

$$W_{h0} = 1, \, h \in H \qquad (19)$$

$$C'_{\max} \geq C'_j, \, j \in N \qquad (20)$$

$$\chi_{ijk} \in \{0, 1\}, \, j \neq k, \, i \in M, \, j \in N \cup \{0\}, \, k \in N \qquad (21)$$

$$\eta_{ijv} \in \{0, 1\}, \, i \in M, \, j \in N \qquad (22)$$

$$\theta_{ij} \in \{0, 1\}, \, i \in M, \, j \in N \qquad (23)$$

$$U_{hjk} \in \{0, 1\}, \, j \neq k, \, h \in H, \, j \in N \cup \{0\}, \, k \in N \qquad (24)$$

$$W_{hj} \in \{0, 1\}, \, h \in H, \, j \in N \qquad (25)$$

Note that an additional dummy job is defined indexed by 0 for modelling. The dummy job is placed on each machine and vehicle with zero processing time and shipping time.

Constraints (3)–(10) belong to the production part. Constraints (3)–(5) define the process energy, reset energy and standby energy consumptions, respectively. Constraint (6) enforces that each job can be assigned to only one machine with one-speed mode. Constraints (7) and (8) ensure that for a job j, there should be exactly one job before it and one after it on the same machine. Constraint (9) requires that job k should complete after its preceding job j plus its processing time and reset time if needed. This constraint is used to avoid the overlap of jobs on the same machine. Constraint (10) defines that production cannot start until the job is released. Constraint (11) defines a dummy job processed at the first and last place of all jobs assigned to a machine.

Constraints (12)–(20) belong to the shipping part. Constraint (12) defines shipping energy consumption. Constraint (13) ensures that each job can be assigned to only one vehicle for shipping. Constraints (14) and (15) ensure that for a job j, there should be exactly one job before it and one after it on the same vehicle. Constraint (16) enforces that job k should finish after

its preceding job j on the same vehicle plus its shipping time. Constraint (17) imposes that a job cannot be shipped until its production is finished. Constraint (18) imposes that each job cannot exceed the size of its assigned vehicle. Constraint (19) similarly defines a dummy job shipped at the first and last place of all jobs assigned to a vehicle. Constraint (20) imposes that the makespan should be greater than the completion time of all jobs. Constraints (21)–(25) define all the binary decision variables.

The proposed model is a mixed-integer model with quadratic terms in the constraints, called a mixed-integer quadratic program (MIQP). In the model, the objective is not convex and contains binary variables and linear variables. According to ILOG CPLEX 12.9.0 (IBM 2019) handbook, CPLEX cannot solve the proposed model because it cannot satisfy the prescribed conditions on the objective function. In order to address the MIQP model, a memetic algorithm is proposed in Section 4. Before introducing this algorithm, the authors analyze the problem complexity and propose two important dominance rules, which will be used in algorithm design.

3.3. Problem complexity

The EPSS problem proves to be an NP-hard problem.

Theorem 1. *The EPSS problem is NP-hard in the strong sense.*

Proof. Only consider the production scheduling part without energy-state decisions. The problem is treated as a parallel machine scheduling problem of minimizing the makespan, i.e. $P_{|M|}||C_{max}$, which is still an NP-Hard problem in the strong sense (Pinedo 2012). In addition to the energy-state decisions and the shipping scheduling integration, the EPSS problem is an NP-Hard problem in a strong sense.

3.4. Dominance rules

Then, two optimal properties are established as *dominance rules* for the latter proposed MA to remove the inferior solutions during the evolution of the proposed MA. Let $C_{i[j-1]}$ denote the completion time of the job immediately preceding job j on machine i.

Property 1. *If $r_j \leq C_{i[j-1]}$, thus $\theta_{ij}=0$ for job j on machine i, $j \in N$.*

Proof. Because job j is released before the completion of job $[j-1]$, the job j is available for processing at any time when the machine i is available. Because the unit reset energy consumption e_i^r is larger than the unit standby energy consumption e_i^s, thus the machine is inclined to prefer standby mode.

Property 2. *If $e_i^s(r_j - C_{i[j-1]}) \leq e_i^r t_i^r$, thus $\theta_{ij}=0$ for job j on machine i, $j \in N$.*

Proof. If the standby energy consumption $e_i^s(r_j - C_{i[j-1]})$ is no more than the reset energy consumption $e_i^r t_i^r$, the machine i is inclined to prefer standby mode considering the makespan is not affected on the matter the machine is standby or reset.

4. A memetic algorithm

As mentioned above, the EPSS problem is an NP-Hard problem. Most of the complex scheduling problems in the literature were addressed by meta-heuristics, considering the combinatorial complexity and computing time requirement (Gao et al. 2019a, 2019b, 2020). In this section, an MA is proposed to search for high quality heuristically but not necessarily optimal solutions within acceptable computing time. MA performs exceptionally well in terms of exploration and exploitation by combining the evolutionary search and the problem-specific local search (Deng and Wang 2017). In recent years, MAs have been successfully used for addressing the complex optimization problems, for example, graph colouring problem (Lü and Hao 2010), flow shop scheduling problems (Wang and Wang 2015), vehicle routing problems(Cattaruzza et al. 2014; Nalepa and Blocho 2016), integrated production and transportation scheduling problems (Guo et al. 2017).

More specifically, the proposed MA employs the population-based search framework of GA and incorporates a new knowledge-driven local search strategy. Thus, the proposed MA has remarkable exploration ability inherited from the global search of GA and excellent exploitation ability from the local search strategy. The proposed MA also incorporates
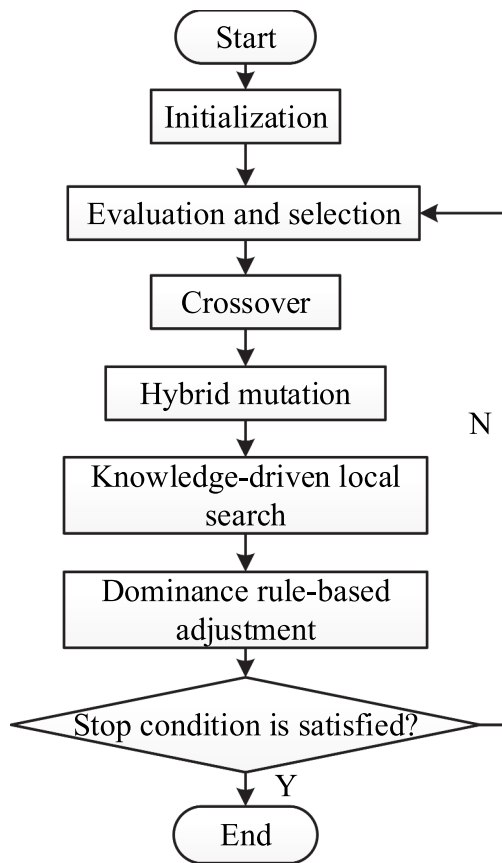
Figure 2. Flowchart of the proposed MA.

the dominance rules proposed in Section 3.4, which fully utilize the special characteristics of the considered problem. The computational experiments have demonstrated the performance of incorporating knowledge-driven local search strategy and dominance rules in Section 5.3.

4.1. The procedure of MA

The flowchart of the proposed MA is illustrated in Figure 2. First, the energy-efficient production scheduling decisions are encoded for generating an initial population. Each chromosome is decoded into a production schedule. Then, a greedy heuristic is proposed to generate a shipping schedule when the production completion time of each job is known. At each generation, crossover, mutation, knowledge-driven local search and dominance rule-based adjustment are conducted in sequence to generate new chromosomes. The performance of a chromosome is evaluated by the fitness value using the decoding operator. The selection operator then generates the new population at the next generation.

The MA has several key parameters, namely population size ($Psize$), mutation probability (Pm) and crossover probability (Pc). The initial population is randomly generated according to the given population size. Population size is kept constant during evolution. The MA will not stop until the stop condition is satisfied. The stop condition used in MA is the maximum generation ($Mgen$). The MA search is also stopped to avoid an ineffective search if no improvement can be achieved in a certain number of successive generations named search depth ($Sdep$).

4.2. Encoding representation and decoding

The energy-efficient production scheduling needs to make the following decisions: 1) Partition jobs for machines, 2) Sequence the assigned jobs for each machine, 3) Determine the standby/reset states of machines for processing each job, 4) Determine the speed modes of machines for processing each job. A representation scheme is proposed for encoding these decisions into a chromosome. The chromosome is a $3 \times (n + m - 1)$ matrix for a m-machine and n-job problem. The first row consists of job permutations and partitioning symbols, i.e. asterisk. In the list, integers represent all possible jobs sequence, and asterisks denote the partition of jobs to machines. The second row designates the standby/reset states of the machine before processing the job in the same column. Without loss of generality, integer 1 represents standby state, while 2 represents reset state. The third row consists of different integers that stand for the speed modes when processing the jobs in the same column. Consider an example with nine jobs and three machines. A chromosome is represented in Figure 3.

The decoding operator can convert a chromosome into a feasible schedule, which is described in Figure 4.

4.3. Evaluation and selection

The quality of a chromosome is evaluated by its fitness value. As the EPSS problem is for minimization, the fitness of each chromosome is defined as the reciprocal of the objective function value. Thus, a better chromosome owns a larger fitness. Note that the objective function value is obtained, considering the production schedule decoded by the decoding operator in Section 4.2 and the shipping schedule obtained in Section 4.6.

3	2	5	*	7	4	8	1	*	6	9
1	2	2	*	1	2	1	1	*	2	1
3	2	1	*	1	2	3	1	*	2	3

Machine 1		Machine 2		Machine 3

Figure 3. Chromosome representation.

The procedure of decoding

Input: a chromosome

Output: a feasible schedule

Begin

   For each job at the chromosome **do**

   Check the standby/reset states of the current job j on machine i

     If it is the standby state **then**

     Machine i remains a standby state, and the starting time of the standby is $C_{i,[j-1]}$.

     Obtain the speed mode of job j as u_v. Then the completion time of job j is computed as $C_j = \max(r_j, C_{i,[j-1]}) + t_j/u_v$, where $C_{i,[j-1]}$ is the completion time of the job immediately before job j on the same machine i.

     Else

       If the reset time t_i^r is less than $r_j - C_{i,[j-1]}$ **then**

       Obtain the speed mode of job j as u_v. Then the completion time of job j is computed as $C_j = \max(r_j, C_{i,[j-1]}) + t_j/u_v$.

       Machine i chooses reset state (turn off and reset), and the starting time of the reset is $r_j - t_i^r$.

       Else

       Set machine i as standby state that the machine remains standby in-between $C_{i,[j-1]}$ and the starting time of job. The completion of job j is calculated as $C_j = \max(r_j, C_{i,[j-1]}) + t_j/u_v$.

       End if

     End if

   End for

End

Figure 4. Pseudocode for decoding.

After obtaining the fitness values of all chromosomes, the selection operator is used to generate a new population. The goal of the selection operator is to ensure that chromosomes with higher quality will have a larger probability of being selected into the mating pool. In the MA, unbiased tournament selection is employed, taking advantage of a small-time complexity (Yu and Gen 2010).
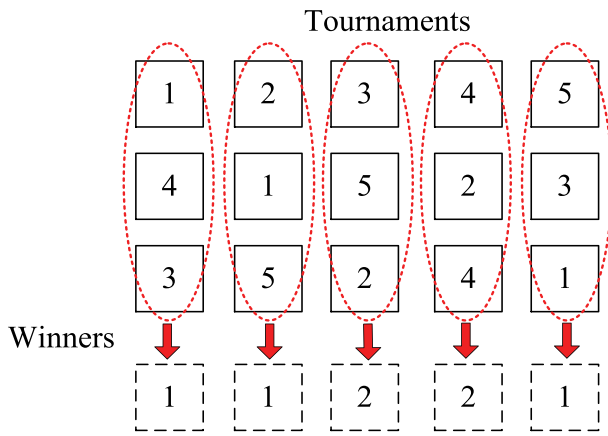
Tournaments

Figure 5. Unbiased tournament selection.

To implement the unbiased tournament selection, some permutations regarding the indexes of chromosomes are randomly generated. The number of permutations equals the tournament size. In each tournament, the chromosome with the best fitness value is selected to generate a new population comprise good-quality chromosomes.

For instance, in Figure 5, suppose there is a population of five chromosomes with the fitness values {1.0, 2.2, 3.1, 4.5, 5.6}. If the tournament size is given as three, then three permutations of the five chromosomes are randomly generated, such as {1, 2, 3, 4, 5}, {4, 1, 5, 2, 3} and {3, 5, 2, 4, 1}. The chromosome

with the smallest fitness value (minimization problem) is selected as the winner for each tournament. After five times selection, a new population {1, 1, 2, 2, 1} is formed.

4.4. Crossover

Crossover aims to reserve the good gene segments of parents to create a new child/offspring. The essential issue of the crossover operator is how to make use of gene information of parents as much as possible to create a high-quality child. The sub-schedule preservation crossover operator is adopted in the proposed MA. A single child is created by two parents using the crossover operator through inheriting the partitioning structure and a sub-schedule from one parent and deriving the remaining job sequence from another parent. The basic procedure is given below.

Step 1: Reserve the asterisk positions from one parent.

Step 2: Randomly inherit a sub-schedule from the same parent.

Step 3: Complete the child with the remaining jobs with the same sequence from another parent.

Consider the example of two parents in Figure 6. The partitioning structure is inherited from the first parent to the child. The sub-schedule in the third machine of the first parent is reserved. The remaining
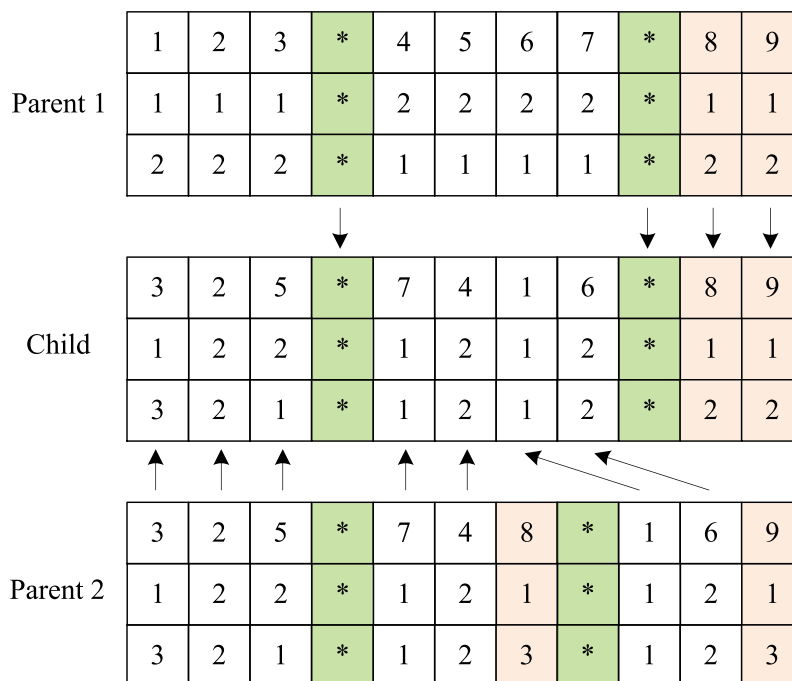

Figure 6. Crossover operator.

jobs, i.e. job 3, 2, 5, 7, 4, 1 and 6, are inserted to complete the child following the sequence of these jobs at the second parent.

4.5. Hybrid mutation

It is noted that the proposed crossover operator could alter the job partition and job sequence of chromosomes. However, variations of standby/reset states and speed modes of machines are minimal. To enhance variability and diversity of chromosomes, a hybrid mutation operator is proposed.

The hybrid mutation operator combines three mutation operators considering characteristics of three rows of the chromosome. The first mutation operator is the swapping mutation that randomly selects two positions and swaps the genes of the two columns of a chromosome. The randomly swapped genes can be a job or asterisk. The swapping mutation, taking advantage of the characteristics of the first row of the proposed encoding operator, can incur meaningful gene mutations. The swapping mutation can alter the job sequence if two jobs at the same machine are selected. Job partition and job sequence can be altered if two jobs at different machines are selected. Asterisk positions can be altered if a job and an asterisk are selected.

The second one is the bit-flip mutation designed for changing the standby/reset state. The bit-flip mutation randomly selects a position and makes the bit-flip change of the gene at the second row, i.e. 1 to 2, or 2 to 1 (Yu and Gen 2010).
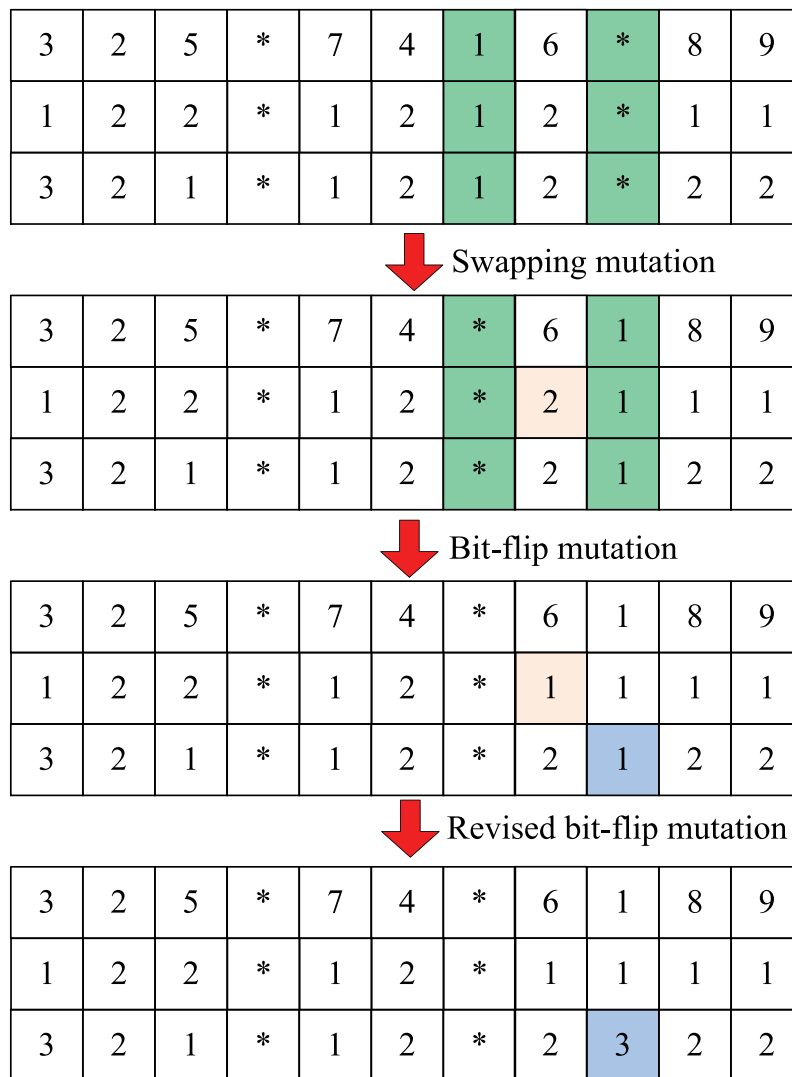

Figure 7. Hybrid mutation operator.

Finally, the third mutation operator is a variant of bit-flip mutation revised for changing the speed modes of machines. It similarly selects a random position and changes the current speed mode to the other two modes with the same probability for the third row. For instance, the randomly selected gene is speed mode 1, which can be changed to speed mode 2 or speed mode 3 with equal probability. The basic procedure is described as follows.

Step 1: Employ the swapping mutation that randomly selects two positions and then swaps the genes of the two columns of a chromosome.

Step 2: Employ the bit-flip mutation that randomly selects a position at the second row of the chromosome and makes the bit-flip change to the gene of the position.

Step 3: Employ the revised bit-flip mutation that randomly selects a position at the third row of the chromosome and makes the flip change to the gene of the position.

Consider a chromosome for the mutation in Figure 7. First, two randomly selected columns, i.e. the green columns, are exchanged by swapping mutation. Then, a randomly selected gene at the second row, i.e. the yellow one, is changed from the reset state to the standby state. Finally, the speed mode is changed from 1 to 3 by making a flip change to the randomly selected gene at the third row, i.e. the blue one.

4.6. A greedy heuristic for shipping scheduling

Regarding the complexity of the EPSS problem, the shipping scheduling is separately investigated, which is described as $P_{|H|}|r'_j, q_j \leq Q_h|\alpha C'_{max} + \beta E'$ using the three-field notation (Pinedo 2012). Shipping scheduling is to allocate n jobs onto b vehicles to minimize the total cost of makespan and energy consumption. A constraint is imposed that the release time of each job is equal to its completion time of production, i.e. $r'_j = C_j$. Another constraint is that the job size should be no larger than the size of the allocated vehicle, i.e. $q_j \leq Q_h$.

Given the framework of the proposed MA, the shipping scheduling is embedded in each evolutionary generation of the MA. Computing time of generating the shipping schedule has a considerable effect on the efficiency of the whole MA. Considering the complexity of downstream shipping scheduling problem, optimal exact algorithms are unaffordable due to the exponential growth of computing time. Thus,

Heuristic H

Input: parameters of jobs and vehicles r'_j, t'_j, q_j, e'_h, Q_h, α, β;

Output: a feasible shipping schedule S_h;

Begin

    Sort all the jobs so that $r'_1 \leq r'_2 \leq ... \leq r'_n$;

    For $h \in H$

    $S_h = \varnothing$;

    End for

    For $j \in N$

    $h = \underset{h \in \{h | Q_h \geq q_j\}}{\arg\min} \{\alpha(\max(C(\pi_h), r'_j) + t'_j) + \beta e'_h t'_j\}$;

    $S_h = S_h \cup \{j\}$;

    End for

End

Figure 8. Pseudocode for heuristic H.

an approximation algorithm named greedy heuristic H is devised to obtain the shipping schedule in Figure 8 efficiently.

Heuristic H sorts jobs in the non-decreasing sequence of $r'_j, j \in N$. According to the sequence, jobs are assigned one by one to the exact machine with the least objective value. Let S_h denote a sequence of jobs assigned on vehicle h. Denote $C(S_h)$ as the completion time of the shipping schedule $\pi_h, h \in H$.

4.7. Knowledge-driven local search

Usually, GA is good at global search but easily stuck in local optimum (Yu and Gen 2010). A knowledge-driven local search is proposed and embedded into GA to enhance exploitation ability and avoid premature convergence.

The knowledge-driven local search employs simulated annealing (SA) as the local search framework, which has successfully addressed extensive optimization problems (Goh, Kendall, and Sabar 2017; Yuan et al. 2019). Based on SA, the MA could avoid trapping into the local optimum by accepting worse solutions in small probability during iterations.

Besides, a new knowledge-driven neighbour generation operator is developed to generate a high-quality neighbourhood. The knowledge-driven neighbour generation operator uses the existing knowledge using good dispatching rules for the variants of the EPSS problem. The proposed neighbour generation operator is different from many traditional neighbour generation operators in SAs, such as SWAP, taking no account of the specific characteristics of the target problem.

The knowledge-driven neighbour generation operator comprises three dispatching rules: Johnson's rule, first-come-first-served (FCFS) and large-size job first (LSJF). As a variant of the EPSS problem, Johnson's rule can optimally solve the two-stage flow shop scheduling problem (Ruiz and Maroto 2005). FCFC means that the earlier a job is released, the earlier the job is assigned. Consider that the jobs arrive at the production part with different release time. It is well known that FCFC works well for many scheduling problems to minimize the makespan, especially for situations with large intervals of release time.

Finally, LSJF means that the larger the job size is, the earlier the job is assigned. Small jobs can be shipped by either large or small vehicles. In contrast, large jobs can only be shipped by large vehicles due to the size constraint. It is more likely that large jobs have higher priority than small jobs to be assigned. The hybrid features of the EPSS problem lead the authors to design a new neighbour generation operator based on the three dispatching rules, i.e. Johnson's rule, FCFS and LSJF.

At each generation of the proposed MA, a local search is performed on the second-best chromosome of the current population to generate a new neighbouring chromosome. The procedure of the knowledge-driven local search is described as follows.

Step 1: Choose the second-best chromosome Y of the current population. Randomly select one machine of the chromosome. Randomly employ one of the three dispatching rules with equal probability to re-sequence the jobs on the selected machine.

Step 2: Employ the bit-flip mutation and the revised bit-flip mutation to change the standby/reset machine mode and the machine speed mode, respectively. Then, a new neighbouring chromosome namely Y_x is generated.

Step 3: Adopt a decoding operator to evaluate the fitness value $f(Y_x)$ of the new neighbouring chromosome.

Step 4: Accept the neighbouring chromosome Y_x to replace the original chromosome Y using the following condition. This step enables local search to avoid trapping into a local optimum through accepting non-locally optimal chromosomes at an adaptive probability.

- If the $f(Y) - f(Y_x) \geq 0$, replace the original chromosome Y with the new chromosome Y_x.
- If the $f(Y) - f(Y_x) < 0$, replace the original chromosome Y with the new chromosome Y_x at the adaptive probability, i.e. $exp\left(\frac{f(Y)-f(Y_x)}{T_x}\right)$. In particular, generate a random number, e.g. $rand$, $0 < rand < 1$. If $exp\left(\frac{f(Y)-f(Y_x)}{T_x}\right) > rand$, accept new chromosome Y_x, otherwise preserve the original chromosome Y.

Step 5: Update the temperature T_x. The temperature at generation x-th, T_x is updated by the geometric cooling scheme $T_x = \zeta T_{x-1}$, where ζ is the prescribed cooling speed factor.

4.8. Dominance rule-based adjustment

After using the crossover, the hybrid mutation and the knowledge-driven local search, the population at each generation is updated by the following adjustment based on the dominance rules in Section 4.2.

Step 1: Check jobs of each chromosome one by one whether Property 1 is satisfied. If yes, set the machine standby, otherwise do nothing.

Step 2: Check jobs of each chromosome one by one whether Property 2 is satisfied. If yes, set the machine standby, otherwise do nothing.

The chromosome adjustment is embedded into the chromosome evaluation step not to take much computing time. The latter experimental results also show that the rule-based dominance adjustment does not increase the computing time of the whole MA. In many cases, however, it can narrow the ineffective search space, resulting in a decrease in the computing time of the MA.

5. Computational experiments and results

This section conducts a series of computational experiments to evaluate the performance of the proposed MA using randomly generated instances. First, the key parameters of the MA are tuned. Then, the MA is compared with two other population-based algorithms. Finally, the energy-saving effect of the proposed energy-efficient scheduling approach is identified.

5.1. Experiment settings

There are few related works on the EPSS problem whose benchmark instances are available now. The values of parameters in the proposed model are varied to generate testing instances in this article. The parameter settings such as the number of jobs/speed, processing time and energy consumption are similar to the previous energy-efficient scheduling research (Mansouri, Aktas, and Besikci 2016; Wu and Che 2019).

In detail, different numbers of jobs, i.e. $|N| = 25, 50, 75, 100, 150, 300$, are considered. The processing time and shipping time of jobs, namely t_j and t'_j, are randomly generated according to the uniform distribution [1, 99] minutes. The processing energy per unit time for machine i, denoted as e_i^p, is randomly generated according to the uniform distribution [30, 70] kilowatts. The standby energy per unit time e^s is set as 10 kilowatts. The reset energy per unit time e_i^r is set as 20 kilowatts and let $e_i^r \geq e^s$. The reset time t_i^r is randomly generated according to the uniform distribution [1, 25] minutes. The processing speed mode u_v is set as 0.75, 1, 1.25. For each speed mode, set the processing energy parameter $e_{iv}^p = e_i^p \cdot (u_v)^2$.

Besides, to test the various bottleneck situations of the integrated system, different machine-vehicle combinations, i.e. $(|M|, |H|)$ are considered as (2, 4), (4, 4) and (8, 4). The size of jobs q_j and vehicle capacity Q_h are randomly generated according to the uniform distribution [2, 5]. Then four different sizes of jobs and vehicles are considered based on practical experience. It simulates major size correlations of job and vehicle with the ten size combinations, i.e. [2, 2], [2, 3], [2, 4], [2,5], [3,3], [3,4], [3, 5], [4, 4], [4, 5], [5, 5]. At least one vehicle is larger than the largest job in each instance so that the instance is solvable. The shipping energy per unit time e'_h associated with the capacity of the vehicle is set as $20 \times Q_h$. Besides, jobs arrive at different release times, which are randomly generated according to the uniform distribution [0, 800].

Table 2. MA parameters and their levels in the orthogonal method.

	Factor levels		
Factors	1	2	3
$Psize$	50	100	200
Pc	0.5	0.7	0.9
Pm	0.1	0.3	0.5

Table 3. Orthogonal array and results.

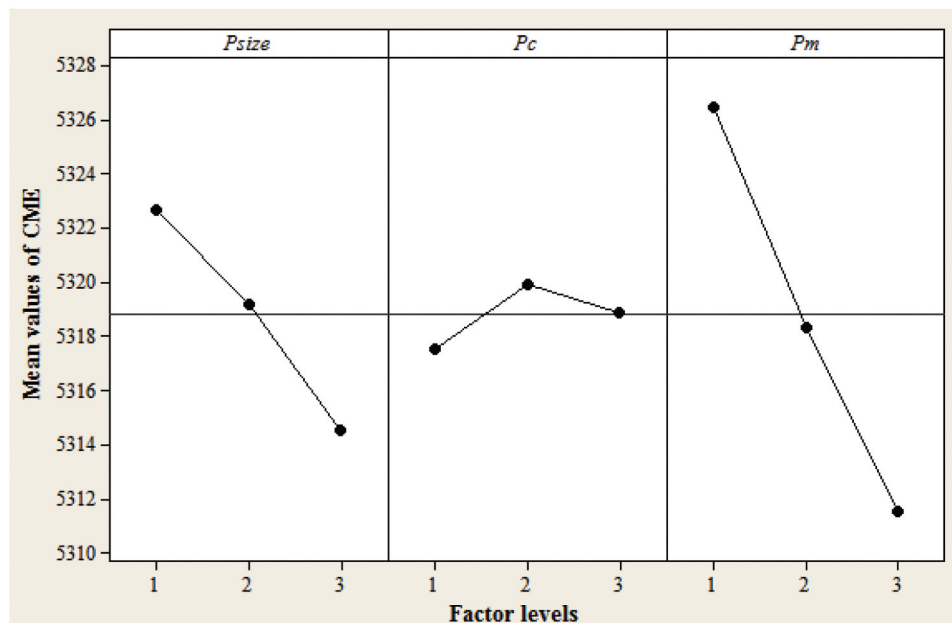|	Factor levels			
Run No.	$Psize$	Pc	Pm	C_{ME}
1	1	1	1	5328.8
2	1	2	2	5322.9
3	1	3	3	5316.5
4	2	1	2	5318.4
5	2	2	3	5312.7
6	2	3	1	5326.4
7	3	1	3	5305.4
8	3	2	1	5324.3
9	3	3	2	5313.8

Figure 9. Main effect plot of the MA parameters.

The experiments are run on a Lenovo PC Tianyi 510Pro with Intel i7-9700 CPU 3.0 GHz and 16 GB RAM. The experiments are implemented by MATLAB R2014b. Each instance is run three times in all experiments to obtain its mean objective value used for evaluation.

5.2. Parameters tuning

It is well known that the parameters of MAs have a dramatic impact on their performance. In this section, the key parameters, i.e. population size (*Psize*), mutation probability (*Pm*) and crossover probability (*Pc*) are tuned by experiments. The stop condition, i.e. maximum generation, *Mgen* is set as 1000. To avoid an ineffective search, the search is stopped if no improvement can be found in the successive 50 generations. Besides, in the local search, the initial temperature T_0 is set as 1000, and the cooling speed factor ζ was set as 0.99.

The three key parameters are determined by the orthogonal method of design-of-experiment (DOE). Each parameter is treated as a factor, and three levels are considered for each factor in Table 2. An orthogonal array $L_9(3^3)$ is selected. All instances generated in Section 5.1 are considered. For each row in Table 3, each C_{ME} is calculated by averaging the objective values of all instances.

First of all, the normal distribution test and Bartlett's test are conducted. The results illustrate that the samples are normally and independently distributed with a common variance for each treatment.

Then, the main effect plot of the MA parameters is given in Figure 9. It is observed that the effect of *Psize* and *Pm* on C_{ME} are more significant than *Pc*. The C_{ME} drops with the increase of *Psize* and *Pm*, indicating the optimal setting of *Psize* and *Pm* at the endpoint. Besides, *Pc* has a slight impact on C_{ME}. Thus, the parameters of MA are set as follows: population size *Psize* = 200, mutation probability *Pm* = 0.5 and crossover probability *Pc* = 0.5 for the subsequent experiments.

5.3. Algorithm comparison and analysis

It is well known that MA is a population-based metaheuristic. To evaluate the performance of the proposed MA, two famous population-based algorithms, i.e. GA and MA, are adapted according to the studied problem following the experimental design rules in the relevant research by (Guo et al. 2017). Besides, the authors justify the effectiveness of the proposed knowledge-driven local search and dominance rule-based adjustment here. Thus, four algorithms for comparisons are 1) the adapted GA by Yu and Gen (2010) (GA); 2) the adapted MA developed by Moscato and Cotta (2003) (AMA); 3) the

Table 4. Algorithms comparison.

$|N|$	$(|M|, |H|)$	GA C_{ME}	GA Time (s)	AMA C_{ME}	AMA Time (s)	MA_L C_{ME}	MA_L Time (s)	MA_L_D C_{ME}	MA_L_D Time (s)
25	(2, 4)	1848.4	14.9	**1840.5**	15.5	1847.7	10.7	1846.9	2.8
	(4, 4)	1683.0	15.7	1682.0	15.1	1672.9	14.4	**1671.7**	7.0
	(8, 4)	1574.5	12.0	1577.5	17.8	1578.0	14.4	**1573.2**	7.5
50	(2, 4)	3423.3	20.0	3421.4	20.1	3419.0	17.5	**3418.8**	3.8
	(4, 4)	3028.2	24.6	3034.7	24.7	3011.8	23.0	**2986.9**	12.3
	(8, 4)	2819.3	28.0	2807.0	28.2	2786.2	24.9	**2767.9**	21.7
75	(2, 4)	4668.7	26.3	4658.7	30.1	4645.6	23.9	**4638.0**	5.5
	(4, 4)	4127.6	34.5	4129.1	34.6	4097.5	25.8	**4059.3**	16.3
	(8, 4)	3840.9	38.6	3826.3	39.2	**3788.3**	33.3	3792.5	28.0
100	(2, 4)	6449.9	39.0	6452.8	35.1	6442.8	33.0	**6440.9**	6.6
	(4, 4)	5507.6	44.5	5495.7	44.2	5494.5	39.8	**5468.0**	23.2
	(8, 4)	5004.6	48.8	4978.1	49.3	4932.3	41.8	**4910.1**	47.4
150	(2, 4)	10293.6	62.5	10295.0	58.8	10283.1	56.9	**10272.4**	12.1
	(4, 4)	8991.0	64.7	9018.4	65.0	8923.2	55.7	**8904.0**	33.2
	(8, 4)	8265.7	68.7	8264.9	69.1	**8210.4**	59.7	8221.1	65.1
300	(2, 4)	20676.0	124.0	20695.2	124.4	20671.3	110.7	**20648.9**	19.2
	(4, 4)	17911.1	125.9	17881.2	124.7	17964.9	109.1	**17773.9**	54.5
	(8, 4)	16,339.8	127.3	16306.1	126.6	16218.0	111.8	**16184.4**	143.5
Average		7025.2	51.1	7020.3	51.2	6999.3	44.8	**6976.6**	28.3

proposed MA with the knowledge-driven local search (MA_L); 4) the proposed MA with both knowledge-driven local search and dominance rule-based adjustment (MA_L_D).

Besides, tight lower bounds for the studied EPSS problems are difficult to obtain considering the weighted bi-objectives of makespan and energy consumption as well as multiple complex decisions. The authors went through the energy-efficient scheduling publications considering bi-objectives of makespan and energy consumption. Most of these papers did not consider lower bound comparisons; for example, Wang et al. (2020), Wu and Che (2019), and Zhang et al. (2021) conducted algorithm comparisons to evaluate the performance of their proposed algorithms instead.

In the GA and AMA, the parameters, i.e. population size, mutation probability, crossover probability and maximum generation, are identical to those of MA_L and MA_L_D. Small-scale instances, i.e. $|N|$ =25, 50, 75 as well as large-scale instances, i.e. $|N|$ =100, 150, 300 are considered. The results are given in Table 4. The columns 'C_{ME}' and 'Time' show the objective and the computing time (seconds), respectively. The details of each instance are listed in the table. The row 'average' shows the average value of all instances.

It is observed that the average C_{ME} values of the proposed MA_L_D and MA_L are noticeably better than those of the AMA and the GA, especially for large instances. It is reasonable that the AMA obtains a better average C_{ME} value than the GA due to the existence of the local search. The MA_L obtains an even better

average C_{ME} value than the AMA. This verifies the performance of the proposed knowledge-driven local search. Furthermore, the MA_L_D obtains the best average C_{ME} value, demonstrating the effectiveness of the embedded dominance rule-based adjustment.

In detail, the **bold values** represent the best C_{ME} of the four algorithms for each instance. The MA_L_D obtains most of the best C_{ME} values of the instances compared to the other three algorithms. Although the MA_L_D has not dominated in all instances, it is highly competitive in terms of the average value and performance in most cases.

Besides, the average computing time of the MA_L_D is less than those of the other three algorithms. It is majorly attributed to the problem-specific dominance rules that shrink the unmeaningful search space for more effective and high-quality searches.

5.4. Energy reduction evaluation and analysis

The proposed energy-efficient scheduling approach incorporated energy consumption minimization into makespan minimization of the traditional scheduling. In this section, the effect of energy reduction is

Table 5. ANOVA for different problem sizes.

$|N|$	f_0	P-value
25	80.12	0.000
50	69.81	0.000
75	73.89	0.000
100	53.77	0.000
150	59.25	0.000
300	101.08	0.000

Table 6. Energy savings by energy-efficient scheduling with different mechanisms.

|N|	Traditional scheduling (Energy consumption) No energy optimization	Energy-efficient scheduling (Energy consumption/Energy savings)		
		Power-down	Speed-scaling	Both
25	2835.9	2478.1/12.6%	2384.3/16.0%	2308.7/18.6%
50	5352.1	4952.3/7.5%	4654.1/13.0%	4630.3/13.5%
75	8065.1	7511.9/6.9%	7021.8/12.9%	7011.6/13.1%
100	10774.6	10221.8/5.1%	9503.7/11.8%	9500.4/11.8%
150	16164.5	15445.8/4.5%	14422.7/10.8%	14402.7/10.9%
300	21297.4	20602.9/3.3%	18963.0/11%	18947.5/11%
Average energy savings		6.7%	12.6%	13.2%

assessed by comparing energy-efficient scheduling, specifically, using both power-down and speed-scaling mechanisms, only power-down mechanism and only speed-scaling mechanism to traditional scheduling (no energy optimization).

Firstly, a single-factor analysis of variance (ANOVA) experiment is conducted to verify whether energy-efficient scheduling considers energy optimization significantly impacting energy reduction. Two different levels of the single factor, i.e. energy-efficient scheduling
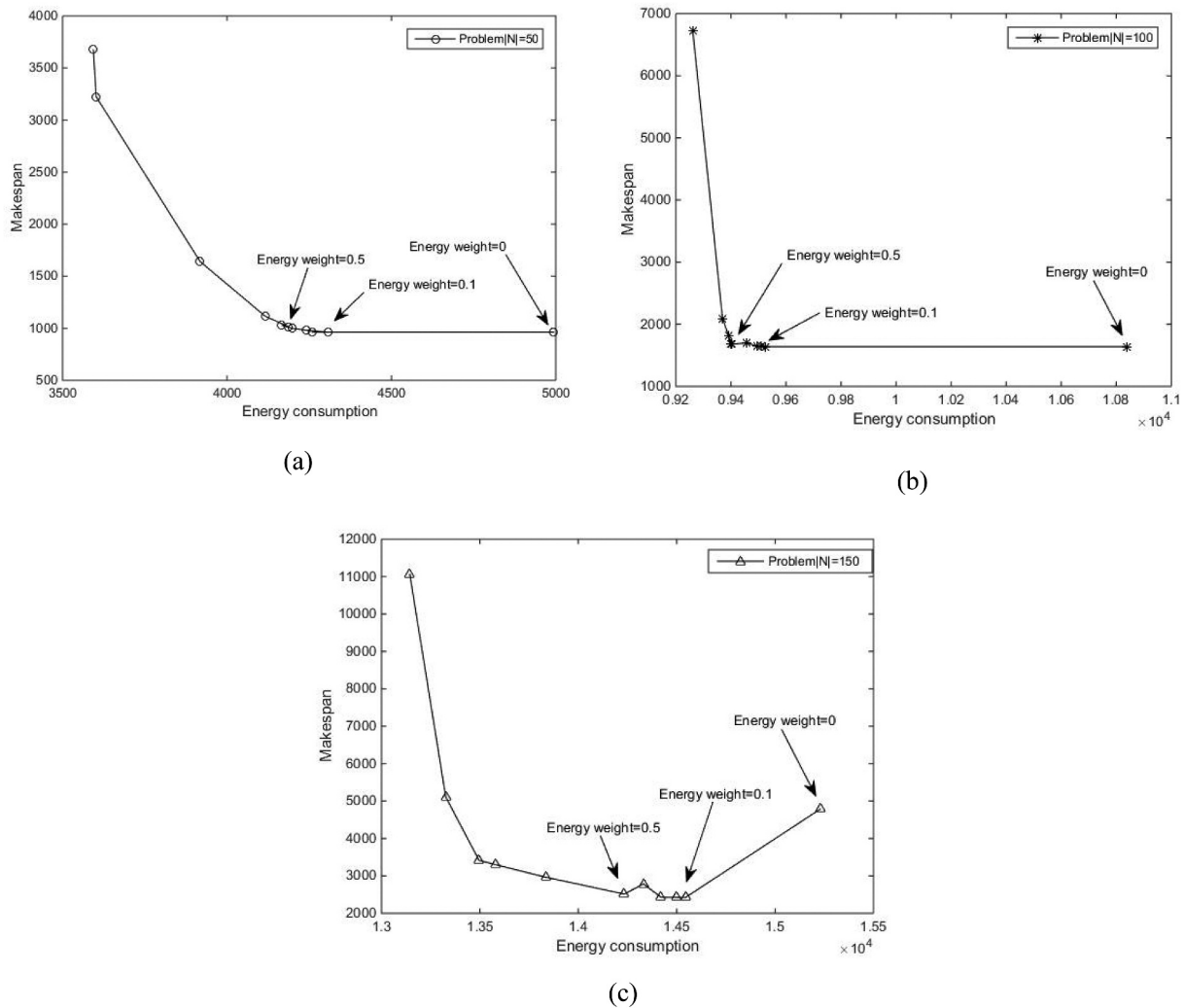
Figure 10. Pareto frontier with different weight combinations: (a) Problem |N| = 50, (b) Problem |N| = 100 and (c) Problem |N| = 150.

with energy optimization (using both power-down and speed-scaling mechanisms) and traditional scheduling with no energy optimization, are considered. For doing so, the weight combinations of energy consumption and makespan are set as (0.5, 0.5) and (0, 1) for energy-efficient scheduling with energy optimization and traditional scheduling with no energy optimization, respectively. Although setting different weight combinations, i.e. (0.1, 0.9), (0.2, 0.8), will generate different specific energy consumption values. Similar findings are successfully obtained, which are not repeatedly listed here. For each problem size, 50 instances are randomly generated, and their mean value is used.

Table 5 shows the results of the problems, $|N| = 25$, 50, 75, 100, 150 and 300, respectively (with $|M| = 4$, $|H| = 4$). It is observed that the P-values of all problems are far less than 0.05, indicating that energy-efficient scheduling has a significant impact on energy reduction.

Specifically, it is observed in Table 6 that compared to traditional scheduling, energy consumptions by energy-efficient scheduling using power-down mechanism only, the speed-scaling mechanism only and the two mechanisms both are averagely reduced by 6.7%, 12.6% and 13.2%, respectively. This indicates that using both power-down and speed-scaling mechanisms is better than using only one of the two mechanisms.

Secondly, the effect of the weight combination is investigated. Weight combinations ranging from (0, 1) to (1, 0) with 0.1 increment are considered. For length and clarity, Figure 10 shows the obtained Pareto frontier of problem $|N| = 50$, problem $|N| = 100$ and problem $|N| = 150$. Note that the Pareto frontier is incomplete due to the approximate search of the weighted objective method.

It is easily found the conflicted relationship between makespan and energy consumption. Energy can be saved at the expense of a longer makespan. Take problem $|N| = 150$ for example. The energy consumption sharply drops as energy weight increases from 0 at first. Nearly 6% energy consumption saving is obtained, incurring almost no makespan deterioration when energy weight varies from 0 to 0.1. However, as the energy weight increases to some extent, for example, from 0.6 to 1, makespan increases more quickly than energy consumption reduces. A similar tendency is found for the other problems that energy consumption quickly reduces when taking energy-saving optimization into account, for example, energy weight increases

from 0 to 0.1. Then, makespan increases sharply if the increase of energy weight continues (such as after 0.8). Thus, determining an appropriate weight combination is highly useful for successfully applying the proposed energy-efficient scheduling approach.

Besides, it is observed that the Pareto frontier shapes of the three problems differ from each other. This indicates that the trade-off between makespan and energy consumption is varied from different problems. A similar finding can be found in the research (Mansouri, Aktas, and Besikci 2016), which was explained using the shadow price of the respective objectives. The finding provides an excellent way to identify the impact of modifying weight combinations by checking the shadow prices of Pareto front schedules in terms of makespan and energy consumption.

6. Conclusions

This research investigates an integrated production and shipping scheduling problem with energy-efficient considerations. Two effective energy-saving mechanisms, power-down and speed-scaling mechanisms, are considered simultaneously. The problem is modelled as a mixed-integer quadratic program, and a GA-based MA incorporating a knowledge-driven local search strategy is proposed to address the problem. Besides, two dominance rules are derived according to the problem characteristics embedded in the MA to enhance its search capability. Experimental results also demonstrate that the proposed MA performs better than two other population-based algorithms, i.e. GA and the traditional MA, in terms of performance and computing time.

The proposed energy-efficient scheduling approach incorporates energy optimization into makespan-oriented scheduling of the integrated production and shipping system in a weighted bi-objective way. The experimental results show that 13.2% of energy consumption can be averagely saved after considering energy-efficient optimization. It is found that energy-saving can be fairly obtained, incurring almost no makespan deterioration when energy weight is set from 0 to 0.1. The effect of various weight combinations is evaluated on respective energy consumption and makespan, suggesting how to leverage weight combinations of makespan and energy consumption in practical application. The findings indicate that the proposed energy-

efficient approach can benefit sustainable production for the whole production-shipping supply chain of make-to-order products. By leveraging the two objectives of optimization, increasing productivity and reducing energy consumption can be balanced.

This work can be further extended from the following aspects. First, considering the uncertainties in production and shipping, the stochastic version of the proposed model can be studied. Second, to fit more situations, the proposed model can be extended considering other energy-saving mechanisms, such as the time of use mechanism. Third, it can be studied how weight combination can be self-adaptive regarding different situations. Fourth, the shipping scheduling part can be realistically extended by including vehicle routing and 3D container packing decisions.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

ORCID

Jian Chen 🔵 http://orcid.org/0000-0001-8882-0000
Gangyan Xu 🔵 http://orcid.org/0000-0001-9537-9006
Yang Liu 🔵 http://orcid.org/0000-0001-8006-3236

References

Cattaruzza, D., N. Absi, D. Feillet, and T. Vidal. 2014. "A Memetic Algorithm for the Multi Trip Vehicle Routing Problem." *European Journal of Operational Research* 236 (3): 833–848. doi:10.1016/j.ejor.2013.06.012.

Che, A., X. Wu, J. Peng, and P. Yan. 2017. "Energy-efficient Bi-objective Single-machine Scheduling with Power-down Mechanism." *Computers & Operations Research* 85: 172–183. doi:10.1016/j.cor.2017.04.004.

Chen, J., G. Q. Huang, H. Luo, and J. Wang. 2015. "Synchronisation of Production Scheduling and Shipment in an Assembly Flowshop." *International Journal of Production Research* 53 (9): 2787–2802. doi:10.1080/00207543.2014.994075.

Chen, J., G. Q. Huang, and J.-Q. Wang. 2019. "Synchronized Scheduling of Production and Outbound Shipping Using Bilevel-based Simulated Annealing Algorithm." *Computers & Industrial Engineering* 137: 106050. doi:10.1016/j.cie.2019.106050.

Chen, Z.-L., and G. L. Vairaktarakis. 2005. "Integrated Scheduling of Production and Distribution Operations." *Management Science* 51 (4): 614–628. doi:10.1287/mnsc.1040.0325.

Chen, Z.-L. 2010. "Integrated Production and Outbound Distribution Scheduling: Review and Extensions." *Operations Research* 58 (1): 130–148. doi:10.1287/opre.1080.0688.

Dai, M., D. Tang, A. Giret, M. A. Salido, and W. D. Li. 2013. "Energy-efficient Scheduling for a Flexible Flow Shop Using an Improved Genetic-simulated Annealing Algorithm." *Robotics and Computer-Integrated Manufacturing* 29 (5): 418–429. doi:10.1016/j.rcim.2013.04.001.

Deng, J., and L. Wang. 2017. "A Competitive Memetic Algorithm for Multi-objective Distributed Permutation Flow Shop Scheduling Problem." *Swarm and Evolutionary Computation* 32: 121–131. doi:10.1016/j.swevo.2016.06.002.

Ding, J.-Y., S. Song, and C. Wu. 2016. "Carbon-efficient Scheduling of Flow Shops by Multi-objective Optimization." *European Journal of Operational Research* 248 (3): 758–771. doi:10.1016/j.ejor.2015.05.019.

Fang, K.-T., and B. M. Lin. 2013. "Parallel-machine Scheduling to Minimize Tardiness Penalty and Power Cost." *Computers & Industrial Engineering* 64 (1): 224–234. doi:10.1016/j.cie.2012.10.002.

Fang, K., N. A. Uhan, F. Zhao, and J. W. Sutherland. 2016. "Scheduling on a Single Machine under Time-of-use Electricity Tariffs." *Annals of Operations Research* 238 (1–2): 199–227. doi:10.1007/s10479-015-2003-5.

Gahm, C., F. Denz, M. Dirr, and A. Tuma. 2016. "Energy-efficient Scheduling in Manufacturing Companies: A Review and Research Framework." *European Journal of Operational Research* 248 (3): 744–757. doi:10.1016/j.ejor.2015.07.017.

Gao, K. Z., Z. M. He, Y. Huang, P. Y. Duan, and P. N. Suganthan. 2020. "A Survey on Meta-heuristics for Solving Disassembly Line Balancing, Planning and Scheduling Problems in Remanufacturing." *Swarm and Evolutionary Computation* 57: 100719. doi:10.1016/j.swevo.2020.100719.

Gao, K., F. Yang, M. C. Zhou, Q. Pan, and P. N. Suganthan. 2019b. "Flexible Job-Shop Rescheduling for New Job Insertion by Using Discrete Jaya Algorithm." *IEEE Transactions on Cybernetics* 49 (5): 1944–1955. doi:10.1109/TCYB.2018.2817240.

Gao, K., Z. Cao, L. Zhang, Z. Chen, Y. Han, and Q. Pan. 2019a. "A Review on Swarm Intelligence and Evolutionary Algorithms for Solving Flexible Job Shop Scheduling Problems." *IEEE/CAA Journal of Automatica Sinica* 6 (4): 904–916. doi:10.1109/JAS.2019.1911540.

Gawiejnowicz, S., and C. Suwalski. 2014. "Scheduling Linearly Deteriorating Jobs by Two Agents to Minimize the Weighted Sum of Two Criteria." *Computers & Operations Research* 52: 135–146. doi:10.1016/j.cor.2014.06.020.

Goh, S. L., G. Kendall, and N. R. Sabar. 2017. "Improved Local Search Approaches to Solve the Post Enrolment Course Timetabling Problem." *European Journal of Operational Research* 261 (1): 17–29. doi:10.1016/j.ejor.2017.01.040.

Gong, G., R. Chiong, Q. Deng, and Q. Luo. 2020b. "A Memetic Algorithm for Multi-objective Distributed Production Scheduling: Minimizing the Makespan and Total Energy Consumption." *Journal of Intelligent Manufacturing* 31 (6): 1443–1466. doi:10.1007/s10845-019-01521-9.

Gong, G., R. Chiong, Q. Deng, W. Han, L. Zhang, W. Lin, and K. Li. 2020a. "Energy-efficient Flexible Flow Shop Scheduling with Worker Flexibility." *Expert Systems with Applications* 141: 112902. doi:10.1016/j.eswa.2019.112902.

Guo, Z., L. Shi, L. Chen, and Y. Liang. 2017. "A Harmony Search-based Memetic Optimization Model for Integrated Production and Transportation Scheduling in MTO Manufacturing." *Omega* 66: 327–343. doi:10.1016/j.omega.2015.10.012.

IBM. 2019. "User's manual for CPLEX of ILOG CPLEX Optimization Studio 12.9.0.". https://www.ibm.com/docs/en/icos/12.9.0?topic=smippqt-miqcp-mixed-integer-programs-quadratic-terms-in-constraints

Lei, D., L. Gao, and Y. Zheng. 2017. "A Novel Teaching-learning-based Optimization Algorithm for Energy-efficient Scheduling in Hybrid Flow Shop." *IEEE Transactions on Engineering Management* 65 (2): 330–340. doi:10.1109/TEM.2017.2774281.

Leung, J. Y.-T., and Z.-L. Chen. 2013. "Integrated Production and Distribution with Fixed Delivery Departure Dates." *Operations Research Letters* 41 (3): 290–293. doi:10.1016/j.orl.2013.02.006.

Li, J., H. Sang, Y. Han, C. Wang, and K. Gao. 2018. "Efficient Multi-objective Optimization Algorithm for Hybrid Flow Shop Scheduling Problems with Setup Energy Consumptions." *Journal of Cleaner Production* 181: 584–598. doi:10.1016/j.jclepro.2018.02.004.

Liu, C.-H. 2014. "Approximate Trade-off between Minimisation of Total Weighted Tardiness and Minimisation of Carbon Dioxide (CO2) Emissions in Bi-criteria Batch Scheduling Problem." *International Journal of Computer Integrated Manufacturing* 27 (8): 759–771. doi:10.1080/0951192X.2013.834479.

Liu, Y., Y. Zhang, S. Ren, M. Yang, Y. Wang, and D. Huisingh. 2020. "How Can Smart Technologies Contribute to Sustainable Product Lifecycle Management?" *Journal of Cleaner Production* 249: 119423. doi:10.1016/j.jclepro.2019.119423.

Lu, C., L. Gao, X. Li, Q. Pan, and Q. Wang. 2017. "Energy-efficient Permutation Flow Shop Scheduling Problem Using a Hybrid Multi-objective Backtracking Search Algorithm." *Journal of Cleaner Production* 144: 228–238. doi:10.1016/j.jclepro.2017.01.011.

Lü, Z., and J.-K. Hao. 2010. "A Memetic Algorithm for Graph Coloring." *European Journal of Operational Research* 203 (1): 241–250. doi:10.1016/j.ejor.2009.07.016.

Luo, H., B. Du, G. Q. Huang, H. Chen, and X. Li. 2013. "Hybrid Flow Shop Scheduling considering Machine Electricity Consumption Cost." *International Journal of Production Economics* 146 (2): 423–439. doi:10.1016/j.ijpe.2013.01.028.

Mansouri, S. A., E. Aktas, and U. Besikci. 2016. "Green Scheduling of a Two-machine Flowshop: Trade-off between Makespan and Energy Consumption." *European Journal of Operational Research* 248 (3): 772–788. doi:10.1016/j.ejor.2015.08.064.

May, G., B. Stahl, M. Taisch, and V. Prabhu. 2015. "Multi-objective Genetic Algorithm for Energy-efficient Job Shop Scheduling." *International Journal of Production Research* 53 (23): 7071–7089. doi:10.1080/00207543.2015.1005248.

Meng, L., C. Zhang, X. Shao, Y. Ren, and C. Ren. 2019. "Mathematical Modelling and Optimisation of Energy-conscious Hybrid Flow Shop Scheduling Problem with Unrelated Parallel Machines." *International Journal of Production Research* 57 (4): 1119–1145. doi:10.1080/00207543.2018.1501166.

Mokhtari, H., and A. Hasani. 2017. "An Energy-efficient Multi-objective Optimization for Flexible Job-shop Scheduling Problem." *Computers & Chemical Engineering* 104: 339–352. doi:10.1016/j.compchemeng.2017.05.004.

Moscato, P., and C. Cotta. 2003. "A Gentle Introduction to Memetic Algorithms." In *Handbook of Metaheuristics*, edited by F. Glover, and G. A. Kochenberger. International Series in Operations Research & Management Science, 105–144.

Mouzon, G., and M. B. Yildirim. 2008. "A Framework to Minimise Total Energy Consumption and Total Tardiness on A Single Machine." *International Journal of Sustainable Engineering* 1 (2): 105–116. doi:10.1080/19397030802257236.

Mouzon, G., M. B. Yildirim, and J. Twomey. 2007. "Operational Methods for Minimization of Energy Consumption of Manufacturing Equipment." *International Journal of Production Research* 45 (18–19): 4247–4271. doi:10.1080/00207540701450013.

Nalepa, J., and M. Blocho. 2016. "Adaptive Memetic Algorithm for Minimizing Distance in the Vehicle Routing Problem with Time Windows." *Soft Computing* 20 (6): 2309–2327. doi:10.1007/s00500-015-1642-4.

Nouiri, M., A. Bekrar, and D. Trentesaux. 2020. "An Energy-efficient Scheduling and Rescheduling Method for Production and Logistics Systems [†]." *International Journal of Production Research* 58 (11): 3263–3283. doi:10.1080/00207543.2019.1660826.

Öztop, H., M. F. Tasgetiren, D. T. Eliiyi, Q.-K. Pan, and L. Kandiller. 2020. "An Energy-efficient Permutation Flowshop Scheduling Problem." *Expert Systems with Applications* 150: 113279. doi:10.1016/j.eswa.2020.113279.

Park, C.-W., K.-S. Kwon, W.-B. Kim, B.-K. Min, S.-J. Park, I.-H. Sung, Y. S. Yoon, K.-S. Lee, J.-H. Lee, and J. Seok. 2009. "Energy Consumption Reduction Technology in Manufacturing — A Selective Review of Policies, Standards, and Research." *International Journal of Precision Engineering and Manufacturing* 10 (5): 151–173. doi:10.1007/s12541-009-0107-z.

Pinedo, M. 2012. *Scheduling*. New York: Springer.

Rager, M., C. Gahm, and F. Denz. 2015. "Energy-oriented Scheduling Based on Evolutionary Algorithms." *Computers & Operations Research* 54: 218–231. doi:10.1016/j.cor.2014.05.002.

Rubaiee, S., and M. B. Yildirim. 2019. "An Energy-aware Multi-objective Ant Colony Algorithm to Minimize Total Completion Time and Energy Cost on a Single-machine Preemptive Scheduling." *Computers & Industrial Engineering* 127: 240–252. doi:10.1016/j.cie.2018.12.020.

Ruiz, R., and C. Maroto. 2005. "A Comprehensive Review and Evaluation of Permutation Flowshop Heuristics." *European Journal of Operational Research, Project Management and Scheduling* 165 (2): 479–494. doi:10.1016/j.ejor.2004.04.017.

Shioura, A., N. V. Shakhlevich, V. A. Strusevich, and B. Primas. 2018. "Models and Algorithms for Energy-efficient Scheduling with Immediate Start of Jobs." *Journal of Scheduling* 21 (5): 505–516. doi:10.1007/s10951-017-0552-y.

Shrouf, F., J. Ordieres-Meré, A. García-Sánchez, and M. Ortega-Mier. 2014. "Optimizing the Production Scheduling of a Single Machine to Minimize Total Energy Consumption Costs." *Journal of Cleaner Production* 67: 197–207. doi:10.1016/j.jclepro.2013.12.024.

Subai, C., P. Baptiste, and E. Niel. 2006. "Scheduling Issues for Environmentally Responsible Manufacturing: The Case of Hoist Scheduling in an Electroplating Line." *International Journal of Production Economics* 99 (1–2): 74–87. doi:10.1016/j.ijpe.2004.12.008.

Wang, S.-Y., and L. Wang. 2015. "An Estimation of Distribution Algorithm-based Memetic Algorithm for the Distributed Assembly Permutation Flow-shop Scheduling Problem." *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 46 (1): 139–149. doi:10.1109/TSMC.2015.2416127.

Wang, S., X. Wang, F. Chu, and J. Yu. 2020. "An Energy-efficient Two-stage Hybrid Flow Shop Scheduling Problem in a Glass Production." *International Journal of Production Research* 58 (8): 2283–2314. doi:10.1080/00207543.2019.1624857.

Wang, W., H. Yang, Y. Zhang, and J. Xu. 2018. "IoT-enabled Real-time Energy Efficiency Optimisation Method for Energy-intensive Manufacturing Enterprises." *International Journal of Computer Integrated Manufacturing* 31 (4–5): 362–379. doi:10.1080/0951192X.2017.1337929.

Wang, Y.-C., M.-J. Wang, and S.-C. Lin. 2017. "Selection of Cutting Conditions for Power Constrained Parallel Machine Scheduling." *Robotics and Computer-Integrated Manufacturing* 43: 105–110. doi:10.1016/j.rcim.2015.10.010.

Wu, X., and A. Che. 2019. "A Memetic Differential Evolution Algorithm for Energy-efficient Parallel Machine Scheduling." *Omega* 82: 155–165. doi:10.1016/j.omega.2018.01.001.

Yan, J., L. Li, F. Zhao, F. Zhang, and Q. Zhao. 2016. "A Multi-level Optimization Approach for Energy-efficient Flexible Flow Shop Scheduling." *Journal of Cleaner Production* 137: 1543–1552. doi:10.1016/j.jclepro.2016.06.161.

Yao, F., A. Demers, and S. Shenker. 1995. "A scheduling model for reduced CPU energy." *Proceedings of IEEE 36th Annual Foundations of Computer Science*, 374–382.

Yu, X., and M. Gen. 2010. *Introduction to Evolutionary Algorithms*. London: Springer Science & Business Media.

Yuan, M., H. Yu, J. Huang, and A. Ji. 2019. "Reconfigurable Assembly Line Balancing for Cloud Manufacturing." *Journal of Intelligent Manufacturing* 30 (6): 2391–2405. doi:10.1007/s10845-018-1398-7.

Yüksel, D., M. F. Taşgetiren, L. Kandiller, and L. Gao. 2020. "An Energy-efficient Bi-objective No-wait Permutation Flowshop Scheduling Problem to Minimize Total Tardiness and Total Energy Consumption." *Computers & Industrial Engineering* 145: 106431. doi:10.1016/j.cie.2020.106431.

Zeng, Z., M. Hong, J. Li, Y. Man, H. Liu, Z. Li, and H. Zhang. 2018. "Integrating Process Optimization with Energy-efficiency Scheduling to Save Energy for Paper Mills." *Applied Energy* 225: 542–558. doi:10.1016/j.apenergy.2018.05.051.

Zhang, H., Y. Wu, R. Pan, and G. Xu. 2021. "Two-stage Parallel Speed-scaling Machine Scheduling under Time-of-use Tariffs." *Journal of Intelligent Manufacturing* 32 (1): 91–112.

Zhang, Y., J. Wang, and Y. Liu. 2017. "Game Theory Based Real-time Multi-objective Flexible Job Shop Scheduling considering Environmental Impact." *Journal of Cleaner Production* 167: 665–679. doi:10.1016/j.jclepro.2017.08.068.

Zheng, X., S. Zhou, R. Xu, and H. Chen. 2020. "Energy-efficient Scheduling for Multi-objective Two-stage Flow Shop Using a Hybrid Ant Colony Optimisation Algorithm." *International Journal of Production Research* 58 (13): 4103–4120.