

An Approximation Algorithm for Unrelated Parallel Machine Scheduling Under TOU Electricity Tariffs

Zhi Pei¹, Member, IEEE, Mingzhong Wan, Zhong-Zhong Jiang, Member, IEEE, Ziteng Wang², and Xu Dai

Abstract—In an era of sustainable development, considerable emphasis has been put onto energy saving, environment friendly, and social welfare as well as productivity in the manufacturing sector. In this work, an unrelated parallel manufacturing setting with time-of-use (TOU) electricity price is explored, with an aim to reduce the electricity cost and increase productivity simultaneously. A nonlinear mathematical programming model is formulated to exploit the special structure of the scheduling problem, where the quadratic constraints are reformulated as second-order-cone (SOC) constraints, and several tailored cutting planes are introduced to further tighten the feasible region of the problem. Then, the original scheduling problem is transformed into several single-machine scheduling problems with TOU electricity price, which could be relaxed as a single-objective programming problem, and it could be solved rapidly via commercial solvers, such as CPLEX. Based on the optimal solution of the relaxed problem, an approximate algorithm is proposed, where a special rounding technique is employed to assign jobs to the unrelated parallel machines in a local search manner. Furthermore, a lower bound model is constructed by eliminating the nonpreemption constraint, and an iteration-based algorithm is devised to obtain the optimal solution of the lower bound problem. Meanwhile, a dispatch rule-based approach is proposed

to provide an upper bound of the scheduling problem with TOU constraint. In the numerical analysis section, the proposed approximate algorithm is validated through extensive testing on various scales of instances, different emphasis on productivity and electricity price, and under two typical TOU electricity pricing policies. It is observed that the gap between the proposed approximate algorithm and CPLEX is mostly within 4%, and the lower/upper bound methods could obtain a relaxed/feasible solution within 0.01 s.

Note to Practitioners—Energy saving together with productivity improvement in the manufacturing system becomes the focus of both academia and industry over the years due to the increasing deterioration of the environment and the advancement of commercialization. In order to maintain better social welfare, the government usually makes time-of-use (TOU) electricity pricing policies to rebalance energy demand. For each manufacturing company, a tradeoff analysis should be performed to adapt to the TOU electricity price, so as to achieve the most desirable outcome. During the decision-making process, the practitioners should assign weights to the productivity and the energy cost to match their anticipation. For a typical manufacturing setting where all the machines are unrelated parallel, the proposed approximate algorithm, also known as the F-SOC algorithm, could be employed to obtain a near-optimal schedule within a short period of time. The proposed approximate algorithm could also work well in situations where the TOU electricity pricing policies are switching, and the preferences for productivity versus energy cost are swaying. A near-optimal schedule could always be guaranteed within a 4% gap from the optimal solution for most cases.

Index Terms—Approximate algorithm, cutting plane, lower bound, second-order-cone (SOC), time-of-use (TOU), unrelated parallel machine scheduling.

I. INTRODUCTION

AS A result of the greenhouse effect and global climate warming, there is an increasing concern in manufacturing systems to reduce the energy costs, especially electricity costs. The unbalanced electricity demand increases the cost of an electricity company and the waste of electrical energy. The time-of-use (TOU) electricity pricing is a policy that could be used to rebalance the electricity demand. Higher electricity prices charged in the on-peak time period could help redistribute the electricity demand. In the TOU electricity pricing policy, the electricity price varies throughout the day.

For energy-saving production systems with productivity concerns, Che *et al.* [1] considered a single-machine scheduling problem under the TOU electricity policy. An efficient greedy insertion heuristic is proposed to minimize the total electricity cost. Fang *et al.* [2] also introduced scheduling

Manuscript received January 13, 2020; revised April 19, 2020; accepted May 12, 2020. This article was recommended for publication by Associate Editor Q. Chang and Editor Y. Tang upon evaluation of the reviewers' comments. This work was supported in part by the National Natural Science Foundation of China under Grant 71871203, Grant 51305400, Grant L1924063, Grant 71971052, and Grant 71671033, in part by the Fund for Innovative Research Groups of the National Natural Science Foundation of China under Grant 71621061, in part by the Major International Joint Research Project of the National Natural Science Foundation of China under Grant 71520107004, in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LY18G010017, in part by the Fundamental Research Funds for the Central Universities under Grant N2006006, in part by the Project of Promoting Talents in Liaoning Province under Grant XLYC1807252, and in part by the 111 Project under Grant B16009. (Corresponding author: Zhong-Zhong Jiang.)

Zhi Pei and Xu Dai are with the College of Mechanical Engineering, Zhejiang University of Technology, Hangzhou 310023, China (e-mail: peizhi@zjut.edu.cn).

Mingzhong Wan is with the School of Business Administration, Northeastern University, Shenyang 110167, China, and also with the Institute of Behavioral and Service Operations Management, Northeastern University, Shenyang 110167, China.

Zhong-Zhong Jiang is with School of Business Administration, Northeastern University, Shenyang 110167, China, also with the Institute of Behavioral and Service Operations Management, Northeastern University, Shenyang 110167, China, and also with the Key Laboratory of Data Analytics and Optimization for Smart Industry, Ministry of Education, Northeastern University, Shenyang 110167, China (e-mail: zzjiang@mail.neu.edu.cn).

Ziteng Wang is with the Department of Industrial and Systems Engineering, Northern Illinois University, DeKalb, IL 60115 USA (e-mail: zwang3@niu.edu).

Color versions of one or more of the figures in this article are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TASE.2020.2995078

1545-5955 © 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.

See <https://www.ieee.org/publications/rights/index.html> for more information.

problems under the TOU electricity pricing policy in a single-machine scenario, where jobs consume arbitrary power demands and must be processed with a uniform speed. For single-machine scheduling problems under a more complex setting, Wang *et al.* [3] presented a heuristic method to minimize the electricity cost and the makespan for the single-machine batch scheduling problem. A dynamic control approach for energy-efficient production scheduling on a single-machine under time-varying electricity price is proposed by Lee *et al.* [4]. Cheng *et al.* [5] formulated the single-machine batch scheduling problem with a machine ON/OFF switching, where a heuristic method is proposed to tackle large-size problems. Che *et al.* [6] introduced a single-machine scheduling problem with power-down mechanism. Zhang *et al.* [7] introduced an improved mixed-integer linear programming model (MILP) for single-machine batch scheduling problem. In their research, a heuristic method is proposed to minimize the total electricity cost and makespan. Chen and Zhang [8] considered a single-machine scheduling problem with TOU tariffs, in which a pseudo polynomial-time algorithm is provided to minimize one of the regular scheduling criteria.

In the more general setting, a flexible job shop scheduling problem with an aim to reduce the total electricity cost under the real-time electricity pricing is considered by Zhang *et al.* [9]. A two-stage flexible flow shop scheduling problem was considered to minimize the energy cost [10]. Wang *et al.* [11] considered the two-machine flow shop scheduling problem under TOU electricity tariffs. Two heuristic algorithms based on Johnson's rule and dynamic programming are proposed to minimize the total energy cost and makespan. Moon and Park [12] consider the scheduling problem with an aim to reduce the TOU energy cost in the distributed energy storage. Ding *et al.* [13] formulated the parallel machine scheduling problem under TOU electricity price to minimize the electricity cost. A differential evolution algorithm is proposed by Zhou *et al.* [14] to solve the parallel batch machine scheduling problem. Cheng *et al.* [15] developed a new model for parallel machine scheduling under TOU electricity price, in which an improved MILP model is introduced to minimize the total electricity cost. Wang *et al.* [16] considered an augmented ϵ -constraint method to solve the identical parallel machine scheduling problem. The parallel machine batch scheduling problem is studied by Tan *et al.* [17], where an MILP model is formulated to minimize the electricity cost in production with TOU price.

The scheduling problem becomes even more complex when the unrelated parallel machine setting is considered under the TOU electricity price. Che *et al.* [18] introduced an efficient two-stage heuristic for unrelated parallel machine scheduling to minimize the total electricity cost under TOU electricity tariffs. Abikarram *et al.* [19] considered the unrelated parallel machine scheduling problem under real time and demand charge pricing, which aims to reduce the total electricity cost. Unfortunately, only the electricity cost is considered in their works, which may cause a negative impact on productivity. To address that concern, this study focuses on unrelated parallel machine scheduling problem under the TOU electricity

price, with a biobjective to minimize the electricity cost and makespan simultaneously. One of the challenges in the unrelated parallel machine scheduling problems lies in how to handle the machine-dependent job processing time [20]. In addition, the two objectives, i.e., the makespan and the electricity cost, need to be coordinated during optimization [21]. In order to nondimensionalize the two objectives, the ratios of the makespan and the electricity cost with respect to their maximum values are employed to tackle the multiobjective dilemma. Moreover, the relative importance of the two criteria is considered in the objective. Due to its nonlinearity, a mixed quadratic integer programming (MQIP) model is formulated in this study. To reduce the computational complexity, the MQIP model is further reformulated via a series of tailored cutting planes. Then, a conic programming model is obtained, which is then handled by a designed approximate algorithm based on a local search approach. Meanwhile, a lower bound algorithm is introduced by skipping the nonpreemption constraint.

This article is structured as follows, in Section II, we describe the unrelated parallel machine scheduling problem with TOU electricity pricing policy, and an MQIP model is formulated. In Section III, the MQIP model is reformulated as an equivalent mixed-integer second-order-cone programming (MISOCP) model. Moreover, the continuous relaxation and cutting planes are considered in the MISOCP model. In Section IV, we propose an F-SOC approximate algorithm and a lower bound method for the unrelated parallel machine scheduling problem with TOU electricity cost. The computational results are reported in Section V, and we conclude this article in Section VI.

II. PROBLEM FORMULATION

In this section, the formulation for unrelated parallel machine scheduling problem with TOU electricity price is provided, also known as $R \mid \text{TOU} \mid C_{\max}, \text{TEC}$. Besides, an MQIP model in accordance with the scheduling formulation will be proposed. In this scheduling problem, there are n jobs to be processed on m machines at k time periods. The job set, machine set, and time period set are denoted as $N = \{1, 2, \dots, n\}$, $M = \{1, 2, \dots, m\}$, and $K = \{1, 2, \dots, k\}$, respectively. Each machine $i \in M$ can process at most one job at each time period, and preemption is not allowed, i.e., once the processing started, the job could not be stopped until it is finished. In this article, since we only consider the offline scheduling setting, each job $j \in N$ is available for processing at the start of the planning horizon. The processing time that job j spends on machine i is denoted by p_{ij} , which is machine dependent within the unrelated parallel machine scenario. The electricity price is related to the time period. For each time period $t \in K$ denoted by the half-closed interval $[PS_t, PS_t + PL_t)$, it maintains a constant electricity price e_t . PS_t denotes the starting point of the time period t , and PL_t is the length of time period t . Indeed, if job j is processed on machine i during time period t , it will incur an electricity cost $p_{ij}e_t$. In fact, one job can be processed at two or more time periods if this job has a sufficiently large processing time. The target of the present research is to minimize the total electricity

cost, which could be achieved by shifting the job's processing to the lower price time periods.

In the scheduling problem with the TOU electricity price, the decision variables are defined differently from the traditional ones. In the position-indexed formulation, the jobs' processing order is considered as a decision variable, but the electricity price charged before the job's completion time is difficult to express. In the time-indexed formulation, the decision variable is Boolean to indicate if one job is processed on any machine at any time slot, the electricity cost could be derived by summing up the electricity prices at all time slots. However, this formulation keeps track of the decision variables for all the time slots, which leads to a large number of binary variables.

In order to express the makespan and the TOU electricity cost, several decision variables are designed to build the mathematical model of the scheduling problem as follows: x_{ij} : $x_{ij} = 1$ if job j is processed on machine i , and $x_{ij} = 0$ otherwise; z_{jt} : $z_{jt} = 1$ if job j is processed at time period t , and $z_{jt} = 0$ otherwise; and y_{ijt} : the actual processing time of job j on machine i at time period t , where $i \in M$, $j \in N$ and $T \in K$.

Apparently, the abovementioned three decision variables are not independent. For instance, if $x_{ij} = 1$ and $z_{jt} = 1$, job j should be processed on machine i at time period t . Hence, y_{ijt} should be larger than or equal to 1. On the other hand, if $x_{ij} = 0$ and $z_{jt} = 0$, job j should not be processed on machine i at time period t . Thus, y_{ijt} should be equal to 0. Then, the following constraint is constructed to establish the relationship among the three decision variables:

$$x_{ij}z_{jt} \leq y_{ijt} \leq x_{ij}z_{jt}PL_t, \quad i \in M, j \in N, t \in K.$$

Since the aim is to minimize both the makespan and the electricity cost simultaneously, the following technique is employed to combine the two aforementioned objectives. Let ξ denote the actual makespan, and TWT denote the total working time within one production cycle. Then, the productivity is described as $\xi/\text{TWT} \times 100\%$. Moreover, let TEC denote the actual total electricity cost and MEC denote the maximal electricity cost, which is obtained by assuming that all the machines are always working without being idle or broken down in one production cycle. Then, the energy cost is denoted as $\text{TEC}/\text{MEC} \times 100\%$. Hence, the MQIP modeling of the unrelated parallel machine scheduling problem is as follows:

$$\text{MQIP: min } \alpha \cdot \frac{\xi}{\text{TWT}} + \beta \cdot \frac{\text{TEC}}{\text{MEC}} \quad (1)$$

$$\text{s.t. } \sum_{i \in M} x_{ij} = 1, \quad j \in N \quad (2)$$

$$\sum_{t \in K} y_{ijt} - x_{ij}PL_t = 0, \quad i \in M, j \in N \quad (3)$$

$$\sum_{j \in N} y_{ijt} \leq PL_t, \quad i \in M, t \in K \quad (4)$$

$$z_{jt} \geq z_{j,t-1} + z_{j,t+1} - 1, \quad j \in N, t \in K \setminus \{1, k\} \quad (5)$$

$$\sum_{i \in M} y_{ijt} \geq (z_{j,t-1} + z_{j,t+1} - 1)PL_t,$$

$$i \in M, j \in N, t \in K \setminus \{1, k\} \quad (6)$$

$$x_{ij}z_{jt} \leq y_{ijt} \leq x_{ij}z_{jt}PL_t, \quad j \in N, t \in K \quad (7)$$

$$x_{ij}(z_{jt} + z_{j,t+1}) + x_{ij'}(z_{j't} + z_{j',t+1}) \leq 3, \quad i \in M, j, j' \in N, t \in K \setminus \{k\} \quad (8)$$

$$\xi \geq PS_t z_{jt} + \sum_{j' \in N} y_{ij't}, \quad i \in M, j \in N, t \in K \quad (9)$$

$$\text{TEC} = \sum_{i \in M} \sum_{t \in K} y_{ijt} e_t, \quad (10)$$

$$x_{ij} \in \{0, 1\}, \quad i \in M, j \in N \quad (11)$$

$$z_{jt} \in \{0, 1\}, \quad j \in N, t \in K \quad (12)$$

$$y_{ijt} \geq 0, \quad i \in M, j \in N, t \in K \quad (13)$$

where α and β are the weights of makespan and electricity cost, respectively. In the MQIP model, the objective function (1) minimizes the sum of weighted makespan and electricity cost. Constraint (2) guarantees that each job is processed on only one machine. Constraint (3) ensures that each job completes its processing. In Constraint (4), the actual processing time of job j on machine i at the time period t should be no more than the length of the time period. Constraint (5) demonstrates that job j should be processed at time period t if job j has been processed at both time period $t - 1$ and $t + 1$. Constraint (6) guarantees that the actual processing time of job j at time period t equals to the length of time period t if job j is processed at both time period $t - 1$ and $t + 1$. In fact, if the actual processing time of job j fails to fill up the time period t , its processing must be interrupted during time period t , which contradicts the nonpreemption setting. In Constraint (7), the actual processing time of job j on machine i at time period t is no less than 1 if job j is processed on machine i at time period t , and 0 otherwise. Constraint (8) ensures that at most one job could be processed at both time periods t and $t + 1$ on each machine. Constraints (9) and (10) define the makespan and total electricity cost, respectively. For machine i , if t' is the last time period in which jobs are processed, then the makespan of the machine i could be stated as the sum of $PS_{t'}$ and $\sum_{j \in N} y_{ij't'}$. Constraints (11) and (12) denote the binary decision variables. Constraint (13) denotes the continuous variables. It is noteworthy that the jobs' actual processing order during every time period on each machine is not fixed via MQIP since the arrangement will not affect the value of makespan or electricity cost.

In the abovementioned MQIP model, the quadratic constraints (8) and (9) complicate the problem, which renders this MQIP unsolvable with the traditional mixed-integer programming algorithms. In Section III, a new formulation of the MQIP model based on nonlinear optimization techniques will be demonstrated to facilitate the computation process.

III. REFORMULATION AND CUTTING PLANES

Since the nonlinearity is the major concern in the MQIP modeling, in the present section, the quadratic constraints and the binary variables will be reformulated. Specifically, the quadratic constraints are transformed into the second-order-cone (SOC) constraints; the binary variables are relaxed

into continuous ones, which is, in general, regarded as a weak relaxation. In order to tighten the obtained relaxation bound, a series of cutting planes, which are designed based on the features of the scheduling problem, are proposed in this section.

A. Reformulation

To handle the nonlinearity of the original problem, we first consider the reformulation of Constraint (7). Basically, Constraint (7) could be decomposed into two quadratic constraints as follows:

$$x_{ij}z_{jt}PL_t \geq y_{ijt} \quad (14)$$

$$y_{ijt} \geq x_{ij}z_{jt}. \quad (15)$$

where $i \in M$, $j \in N$, and $t \in K$. Moreover, the quadratic Constraint (14) could be reformulated as an SOC constraint.

Lemma 1: In MQIP model, Constraint (14) is equivalent to the following SOC formulation:

$$z_{jt}PL_t^2 + x_{ij} \geq ||2y_{ijt}, z_{jt}PL_t^2 - x_{ij}|| \quad (16)$$

where $i \in M$, $j \in N$, and $t \in K$.

Proof: In the MQIP model, both x_{ij} and z_{jt} are 0-1 variables. It means that $x_{ij} = x_{ij}^2$ and $z_{jt} = z_{jt}^2$. Then, squaring both sides of Constraint (14), we have

$$x_{ij}z_{jt}PL_t^2 \geq y_{ijt}^2.$$

Moreover, $x_{ij}z_{jt}PL_t^2$ could be represented by the difference of two squared terms as follows:

$$(x_{ij}PL_t^2 + z_{ij})^2 - (x_{ij}PL_t^2 - z_{ij})^2 = 4x_{ij}z_{jt}PL_t^2.$$

Then, we have

$$(x_{ij}PL_t^2 + z_{ij})^2 \geq y_{ijt}^2 + (x_{ij}PL_t^2 - z_{ij})^2.$$

In the MQIP model, Constraint (13) guarantees $y_{ijt} \geq 0$. Hence, Constraint (16) is equivalent to Constraint (14). $\square$

For Constraint (15), it cannot be reformulated as an SOC constraint. However, this constraint can be transformed into a linear constraint based on the property of binary variables.

Lemma 2: In the MQIP model, Constraint (15) is equivalent to the following linear constraint:

$$x_{ij} + z_{jt} - 1 \leq y_{ijt} \quad (17)$$

where $i \in M$, $j \in N$, and $t \in K$.

Proof: In the MQIP model, both x_{ij} and z_{jt} are 0-1 variables. Then, there are only four possible outcomes for (x_{ij}, z_{jt}) : (1, 1), (1, 0), (0, 1), and (0, 0). When $(x_{ij}, z_{jt}) = (1, 1)$, (1, 0), or (0, 1), $x_{ij}z_{jt} = x_{ij} + z_{jt} - 1$. On the other hand, when $(x_{ij}, z_{jt}) = (0, 0)$, we have $-1 \leq y_{ijt}$, as indicated by Constraint (17), and $0 \leq y_{ijt}$, as indicated by Constraint (15). Moreover, since $0 \leq y_{ijt}$ holds in Constraint (13); thus, Constraint (17) is equivalent to Constraint (15) in the MQIP model. $\square$

Based on Lemmas 1 and 2, the quadratic Constraint (7) could be replaced by SOC Constraint (16) and linear Constraint (17). Then, we consider the technique for handling the quadratic Constraint (8).

In the MQIP model, a specific type of constraint named single-straddling constraint is designed to ensure that at most one job could be processed in both time periods t and $t + 1$ on each machine. Che *et al.* [18] proved that the scheduling plan with and without a single-straddling constraint could transform reciprocally for the scheduling problem with an aim to minimize the TOU electricity cost. Based on that conclusion, we find that it also works in the present setting with multiple objectives so as to minimize both the electricity cost and the makespan. In other words, the objective value of MQIP will not be affected if the single-straddling constraint is skipped. Equivalently, the optimal scheduling of MQIP could be obtained by rearranging the jobs' processing sequence, which is derived without considering the single-straddling constraint. For the MQIP formulation, Constraint (8) is a typical single-straddling constraint.

Lemma 3: In the MQIP model without a single-straddling constraint, the obtained scheduling outcome could be transformed into an equivalent feasible processing sequence.

Fig. 1 shows the reassignment for the scheduling without single-straddling constraint. Before reassignment, jobs J_1 and J_2 are both processed in k and $k + 1$ time periods. Without loss of generality, it is assumed that $y_{1k} = P_{1k}$, $y_{1k+1} = P_{1k+1}$, $y_{2k} = P_{2k}$, and $y_{2k+1} = P_{2k+1}$. After reassignment, the processing of job J_1 in time period $k + 1$ is shifted to time period k . Hence, we have the reassigned scheduling that $y'_{1k} = P_k + P_{1k}$, $y'_{1k+1} = 0$, $y'_{2k} = P_{2k} - P_{1k+1}$, and $y'_{2k+1} = P_{2k+1} + P_{1k+1}$. Apparently, the following equations hold after reassignment: $y_{1k} + y_{2k} + y_{1k+1} + y_{2k+1} = y'_{1k} + y'_{2k} + y'_{1k+1} + y'_{2k+1}$ and $(y_{1k} + y_{2k})e_k + (y_{1k+1} + y_{2k+1})e_{k+1} = (y'_{1k} + y'_{2k})e_k + (y'_{1k+1} + y'_{2k+1})e_{k+1}$. Therefore, the reassigned scheduling shares the same objective function value with the MQIP model without single-straddling constraint.

Based on Lemmas 1–3, the MQIP model transforms equivalently to the following MISOCP:

$$\begin{aligned} \text{MISOCP: } \min \quad & \alpha \cdot \frac{\xi}{\text{TWT}} + \beta \cdot \frac{\text{TEC}}{\text{MEC}} \\ \text{s.t. } \quad & (2) - (6), (9) - (13), (16), (17). \end{aligned}$$

Corollary 1: The solution of MISOCP could be equivalently transformed into the solution of MQIP.

In Corollary 1, it is illustrated that MQIP shares the same optimal schedule with MISOCP, which is a mixed integer SOC programming problem. Moreover, MISOCP could be further reformulated as an SOC programming problem by linear relaxation. The linear relaxation for Constraints (11) and (12) is described as follows:

$$0 \leq x_{ij} \leq 1, \quad i \in M, \quad j \in N \quad (18)$$

$$0 \leq z_{jt} \leq 1, \quad j \in N, \quad t \in K. \quad (19)$$

Then, we have the relaxed SOCP as follows:

$$\begin{aligned} \text{SOCP: } \min \quad & \alpha \cdot \frac{\xi}{\text{TWT}} + \beta \cdot \frac{\text{TEC}}{\text{MEC}} \\ \text{s.t. } \quad & (2) - (6), (9), (10), (13), (16) - (19). \end{aligned}$$

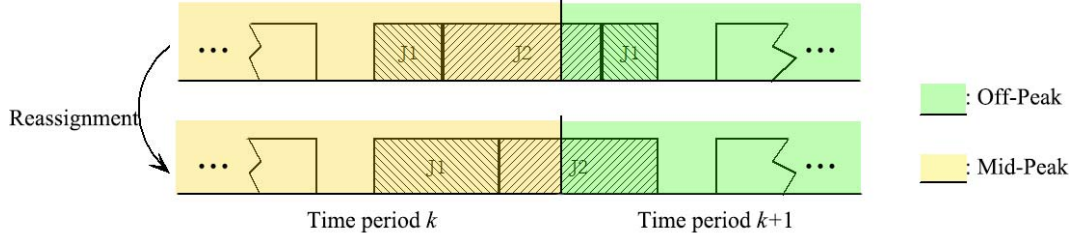


Fig. 1. Reassignment for the schedule without single-straddling constraint.

B. Cutting Planes

In general, linear relaxation is considered a relatively loose relaxation with a weak bound. It means that the solution of SOCP cannot guarantee efficient guidance to find the solution of MQIP based on branch-and-bound or other search methods. Hence, the following cutting planes are designed to tighten the bound set by SOCP.

1) *Single-Period Processing Time Cut*: In consideration of the continuous relaxation in SOCP model, the variables x_{ij} obtained are relatively tight because of the equality Constraints (2) and (3). In contrast, the variables z_{jt} derived in SOCP are not constrained tightly. For instance, in Constraint (7), if PL_t is substantially larger than y_{ijt} , z_{jt} tends to approach “0,” where $z_{jt} \in \{0, 1\}$ as defined in Constraint (12), even though z_{jt} should be equal to “1” as it is derived in the original unrelaxed problem. Therefore, the following cut is designed to tighten the obtained value of variable z_{jt} in SOCP:

$$x_{ij}z_{jt}p_{ij} \geq y_{ijt}, \quad j \in N. \quad (20)$$

Inequality (20) represents that the actual processing time of job j in time period t on machine i should be less than or equal to its total processing time p_{ij} . Hence, in MQIP, the single-period processing time (SPPT) cut is a valid inequality. Moreover, when the continuous relaxation in SOCP model is considered, $x_{ij}z_{jt}p_{ij} \geq y_{ijt}$ could provide even tighter bound than $x_{ij}z_{jt}PL_t \geq y_{ijt}$ if $p_{ij} \leq PL_t$.

2) *Improved Makespan Cuts*: In Constraint (9), the makespan is bounded from below by $PS_t z_{jt} + \sum_{j' \in N} y_{ij't}$. This constraint works well in MQIP, but the first term $PS_t z_{jt}$ renders the bound weaker as the continuous relaxation is considered in SOCP. Although the SPPT cut is added to tighten the obtained value of variable z_{jt} , its effect is limited. Hence, the structure of the scheduling problem is explored to design other cuts to improve the bound of the makespan

$$\xi \geq \sum_{j \in N} \sum_{t' \in K} y_{ij't'} + \sum_{j' \in N} y_{ij't}, \quad i \in M, \quad t \in K. \quad (21)$$

As it is indicated by Inequality (21), the makespan of machine i should be no less than the total processing time of the jobs on machine i . Moreover, the abovementioned inequality could also be reformulated as an SOC constraint based on Lemma 1

$$z_{jt}P_{ij}^2 + x_{ij} \geq ||z_{jt}P_{ij}^2 - x_{ij}, 2y_{ijt}||, \quad j \in N. \quad (22)$$

C. Improved SOCP Model

The SPPT cut in the SOCP model presents the variable z_{jt} with a tighter bound. The SPPT cut also provides a tighter bound of the makespan. Moreover, by employing the improved makespan (IM) cut, the bound on makespan could be further tightened. As a consequence, the improved SOCP formulation is shown as follows:

$$\begin{aligned} \text{RSOCP: min } & \alpha \cdot \frac{\xi}{\text{TWT}} + \beta \cdot \frac{\text{TEC}}{\text{MEC}} \\ \text{s.t. } & (2) - (6), (9), (10), (13), (16) - (19), (21), (22). \end{aligned}$$

IV. ALGORITHMS

In Section III, the RSOCP model is proposed to obtain the relaxed solution of the unrelated parallel machine scheduling problem with TOU electricity price. Since the goal of this study is to obtain a high-quality feasible solution to the original scheduling problem, the relaxation-based methods should be considered. For instance, the branch and bound (B&B) is a frequently used approach to obtain the optimal solution based on the relaxed formulation. However, one commonly known drawback of the B&B method is its time-consuming feature. In the real-life scheduling scenario, the high-quality feasible solution with acceptable computational time is usually more desirable than the optimal solution with huge calculation efforts. Hence, in this section, the aim is to find a feasible schedule with high quality based on the relaxed solution for RSOCP.

In this section, an approximate algorithm based on RSOCP is proposed to solve the unrelated parallel machine scheduling problem with TOU electricity price. As the RSOCP problem contains only SOC constraints besides linear ones, it could be solved to optimality via the CVX module. The outcome of the RSOCP determines the assignment of all the jobs to their corresponding machines; thus, the original scheduling problem could be divided into m single-machine scheduling problems with TOU electricity price, which is denoted as 1|TOU|Cmax, TEC. Fortunately, the exact solution of the single-machine scheduling problem could be solved via CPLEX quickly. Finally, the schedule of the original problem could be constructed by the m subproblems.

A. F-SOC Approximate Algorithm

In this section, the approximate algorithm is introduced to first assign all the jobs to the machines. Then, the actual

processing time of each job is fixed based on the feature of job-machine combinations. Thus, the original problem is decomposed into several single-machine scheduling problems with TOU electricity price. In particular, these single-machine scheduling problems are easy to formulate as MIP problems and solved via commercial solvers. Then, the feasible schedule of the original problem is constructed through the combination of all the single-machine scheduling problems with TOU electricity price.

1) *Rounding Technique*: After the RSOCP model is solved to optimality via commercial solvers, such as CVX, a new challenge emerges as to how to round the nonintegral solutions. Although the B&B method could be applied to tackle the rounding issue, the computational burden becomes unbearable due to the curse of dimensionality. Hence, in our algorithm, a new rounding technique is proposed to deal with relaxed variables.

In this section, the variable x_{ij} is considered as the rounding target for two reasons. For one thing, Constraint (2) is an equality constraint for x_{ij} ; thus, it is more tightly bounded than the variable z_{jt} , which is only constrained by inequalities. Second, the scheduling problem with rounded variable x_{ij} could be decomposed into several single-machine scheduling problems with the TOU electricity pricing policy, which could be easily formulated as an MIP model.

In the rounding process, each job $j \in N$ is assigned to the machine $i \in M$, where $x_{ij} = \max(x_{i'j} \mid \forall i' \in M)$. That is, the value of x_{ij} is rounded to 1 if $i = \arg \max_{i' \in M} x_{i'j}$. The value of x_{ij} is rounded to 0 otherwise. Moreover, to improve the obtained schedule, a local search-based approach is employed to find the better assignment in the neighborhood of the abovementioned rounded variable x_{ij} .

As it is described in the pseudocode of Algorithm 1, from Steps 3 to 8, the assigned machine of each job $j \in N$ is iteratively swapped. For each iteration, the objective value of the new assignment is obtained by solving several MIP_i models. If the new assignment leads to a better objective value, the assignment is updated, and the local search procedure continues for the next job; otherwise, just repeat the above-mentioned steps for another job. The local search approach terminates until no other jobs remain to be tested.

2) *Single-Machine Subproblem*: After the rounding procedure devised in Section IV-A1, all the jobs are assigned to their respective machines for processing. Then, the original scheduling problem is decomposed into several single-machine scheduling problems with TOU electricity price, also known as 1|TOU|Cmax, TEC. In addition, these single-machine subproblems with TOU electricity pricing policy share the same objective function as it is with the MISOCP model. The formulation for the single-machine scheduling subproblem with TOU electricity price is as follows:

$$\text{MIP}_i : \min \alpha \cdot \frac{\zeta}{\text{TWT}} + \beta \cdot \frac{\text{TEC}}{\text{MEC}} \quad (23)$$

$$\text{s.t.} \sum_{t \in K} y_{jt} - p_j = 0, \quad j \in N \quad (24)$$

$$\sum_{j \in N} y_{jt} \leq PL_t, \quad t \in K \quad (25)$$

$$z_{jt} \geq z_{j,t-1} + z_{j,t+1} - 1, \quad t \in K \setminus \{1, k\} \quad (26)$$

$$y_{jt} \geq (z_{j,t-1} + z_{j,t+1} - 1)PL_t, \\ j \in N, \quad t \in K \setminus \{1, k\} \quad (27)$$

$$z_{jt} \leq y_{jt} \leq z_{jt}PL_t, \quad j \in N, \quad t \in K \quad (28)$$

$$z_{jt} + z_{j,t+1} + z_{j',t} + z_{j',t+1} \leq 3, \\ j, j' \in N, \quad t \in K \setminus \{k\} \quad (29)$$

$$\zeta_i \geq PS_t z_{jt} + \sum_{j' \in N} y_{j't}, \quad j, j' \in N, \quad t \in K \quad (30)$$

$$\text{TEC} = \sum_{t \in K} y_{jt} e_t, \quad (31)$$

$$z_{jt} \in \{0, 1\}, \quad j \in N, \quad t \in K \quad (32)$$

$$y_{jt} \geq 0, \quad j \in N, \quad t \in K. \quad (33)$$

In the MIP_i model, the objective function (23) minimizes the sum of weighted productivity and electricity usage ratio. Constraint (24) ensures that each job completes its processing. In Constraint (25), the actual processing of job j at time period t should be shorter than or equal to the length of time period t . Constraint (26) guarantees that job j is processed at time period t if job j is processed at both time periods $t - 1$ and $t + 1$. Constraint (27) illustrates that the actual processing time of job j at time period t equals the length of time period t if job j is processed at both time periods $t - 1$ and $t + 1$. In Constraint (28), the actual processing time of job j at time period t is larger than or equal to 1 if job j is processed at time period t , and the actual processing time of job j at time period t equals 0 otherwise. Constraint (29) ensures that there is at most one job that could be processed at both time periods t and $t + 1$ on each machine. Constraints (30) and (31) define the makespan and total electricity cost, respectively. Constraint (32) denotes the binary decision variables. Constraint (33) denotes the continuous variables.

In the RSOCP model, the makespan is denoted via $\zeta = \max \zeta_i \forall i \in M$, where ζ_i is the makespan of machine i . It means that the value of ζ would change if and only if the value of $\max \zeta_i$ changes. In MIP_i, given that the variable x_{ij} has been fixed as a binary integer value, the objective makespan $\hat{\zeta}$ would be greater than it is of RSOCP. Then, the processing of each job is considered to be moved to the other time period with a lower electricity price to reduce the electricity cost on each machine i , where $i \neq \hat{i}$ and $\hat{i} = \arg \max \zeta_i$. Apparently, the makespan of each machine i is bounded by $\hat{\zeta}$ to maintain the value of maximum makespan. Then, the improved model for single-machine scheduling problem with TOU electricity cost on machine $i \neq \hat{i}$ is formulated as follows:

$$\text{RMIP}_i : \min \beta \cdot \frac{\text{TEC}}{\text{MEC}} \quad (34)$$

$$\text{s.t.} \hat{\zeta} \geq PS_t z_{jt} + \sum_{t' \in K} y_{j't}, \quad j \in N, \quad t \in K \\ (24) - (29), (31) - (33) \quad (35)$$

where $\zeta = \max \zeta_i$ for $\forall i \in M$. In RMIP_i, the objective function (34) is aimed to minimize the electricity cost only. Constraint (35) guarantees that the makespan should not exceed the value of $\hat{\zeta}$.

Algorithm 1 F-SOC Approximate Algorithm

```

1 Initialization,  $m, n, k$  is the number of machines, jobs
  and time periods respectively.
2 Obtain the relaxed solution  $x_{ij}$  by solving RSOCP model.
3 for  $j=1:n$  do                                /*Local search*/
4   for  $i=1:m$  do
5     Let  $x_{ij} = 1$ , and  $x_{i'j} = 0 \forall i' \in M \setminus \{i\}$ ;
6     Solve the  $MIP_i$  models based on the new assignment
       plan, and let  $v'$  be the sum of objective values by
       solving  $MIP_i$  model;
7   if  $v' < v^*$  then
8     Replace the assignment plan by this new
       assignment plan;
9 Let  $\xi$  be the makespan of the current scheduling plan,
  and solve the  $RMIP_i$  model  $\forall i \in M$ .
10 Output the assignment plan and the scheduling plan for
    each machine.

```

Fortunately, the MIP_i and $RMIP_i$ models are mixed-integer linear programming problems with $n(i) \cdot k$ binary variables, where $n(i)$ indicates the number of jobs assigned to the i th machine. It is observed that the scales of $n(i)$ and k are both in a manageable range, which makes the two single-machine scheduling models solvable via CPLEX in a short period of time, usually less than 300 s. Hence, the scheduling plan of each machine can be obtained by solving its respective MIP_i and $RMIP_i$ in sequence. In other words, the solution of MIP_i for $i = \arg \max \xi_i$ is employed as an upper bound to guide the optimization model of $RMIP_{i'}$ for $i' \neq i$ and $i' \in M$, which then serves to constitute the schedule of the original problem.

3) *Pseudocode for F-SOC Approximate Algorithm:* To sum up, in the proposed F-SOC approximate algorithm, the RSOCP model is tackled by CVX in the first place. Whereafter, the relaxed decision variable x_{ij} is rounded to an integer by the rounding method introduced in Section IV-A1. Based on the rounded decision variable x_{ij} , all jobs are assigned to their respective machines. Hence, the original scheduling problem could be easily decomposed into several single-machine scheduling problems with TOU electricity cost. To attack the decomposed subproblems, the MIP_i model is proposed to solve the i th single-machine scheduling problem. Moreover, the $RMIP_i$ model with only the electricity cost in the objective function is proposed to improve the quality of scheduling by arranging a job sequence on each machine. In detail, the pseudocode of the F-SOC approximate algorithm is shown as follows.

B. Lower Bound Algorithm and Dispatch Rule

In Section IV-A, an approximate algorithm is proposed to obtain the approximate solution of the unrelated parallel machine scheduling problem with TOU electricity cost. Due to the complexity of the original problem, in order to measure the quality of the solution in our approximate algorithm, a lower bound algorithm is proposed by relaxing several constraints of the original problem.

In the lower bound problem, the preemption is allowed for each job. Moreover, it is assumed that each job $j \in N$ incurs the same processing time on each machine, as if it is processed on the fastest machine, that is, $\tilde{p}_j = \min_{i \in M} p_{ij}$. For each job to be processed, the scheduling problem is now reduced to $P|TOU|C_{max}, TEC$. Therefore, the optimal value of the relaxed problem provides a lower bound, which is strictly less than the optimal value of the original problem.

In order to obtain the optimal objective value of the lower bound problem, a dedicated method is designed based on dynamic programming and a greedy algorithm. The aim is to move the jobs' processing to the time periods with the lowest electricity cost, so as to decrease the objective value. In order to justify the existence of such a lower bound solution satisfying the parallel machine scheduling setting, the following theorem is introduced.

Theorem 1: In the lower bound model, there always exists an optimal solution where all the machines share an identical scheduling plan.

Proof: Assume that there is a single machine i that involves a different scheduling plan in time period t and shares the same schedule in other time periods $t' \neq t$ with all the other machines, where $i \in M$ and $t, t' \in K$. Let φ_{it} denote the total processing time (workload) of machine i in time period t ; if $\varphi_{it} \neq \varphi_{i't'} \forall i' \in M$, then one better solution would be to redistribute $|\varphi_{it} - \varphi_{i't'}|$, so as to balance the workload on each machine. Thus, it will not increase the objective value by assigning the average workload $\sum_{i \in M} \varphi_{it}/m$ to each machine equally. The following two different scenarios are considered.

- 1) If time period t is not the last time period in which jobs complete processing, the makespan and electricity cost will not be changed by assigning the workload $\sum_{i \in M} \varphi_{it}/m$ to each machine equally.
- 2) If time period t is the last time period in which jobs finish processing, the makespan will be reduced by assigning the average workload $\sum_{i \in M} \varphi_{it}/m$ to each machine, where the current workload assignment is not an optimal solution. However, the electricity cost will stay unchanged because the total processing time of jobs on all the machines in time period t is constant. Therefore, under the optimal solution assumption, any optimal workload schedule having different job-machine couplings in the last time period of processing could be equivalently transformed to the balanced workload assignment via redistributing $\sum_{i \in M} \varphi_{it}/m$ to each machine.

Therefore, for each time period, the workload could be assigned to each machine following the same procedure. Since the initial nonidentical scheduling plan is also optimal for the relaxed problem, there exists an optimal solution where all the machines share the same scheduling plan for the lower bound problem. $\square$

Based on Theorem 1, only one machine needs to be considered for the optimal scheduling in the lower bound problem, and all the other machines maintain exactly the same scheduling plan. Let p^q denote the actual processing time of jobs in time period q . In order to improve the solution, the jobs' processing in the q th time period is moved to other

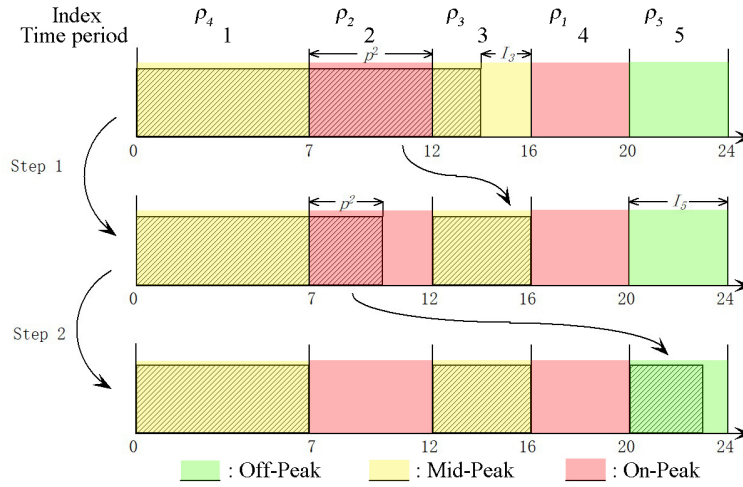


Fig. 2. Instance of a moving step in the lower bound algorithm.

time periods with less expensive electricity costs when the machine is available.

During the design of the workload moving algorithm, straightforward thinking would be to spread the workload of time period q to all the other time periods and choose the best outcome based on the respective objective value. However, there is a drawback on this method that will possibly cause the workload to be moved back and forth to the time period q , and it could cause a significant time delay in deriving the best solution.

In order to handle this problem, all the time periods are ordered according to their electricity price, and $\rho_1, \rho_2, \dots, \rho_k$ are used to denote the index for the decreasing order of time periods according to their electricity prices. Specifically, if $e_{t_1} = e_{t_2}$ and $t_1 < t_2$, then we have $\rho_q = t_1$ and $\rho_{q+1} = t_2$. In the initial solution of the scheduling, all the jobs are processed to minimize the makespan, i.e., each machine will process jobs within the time interval $[0, P/m]$, where $P = \sum_{j \in N} p_j$. Whereafter, we will introduce the workload moving method based on the above-ordered time periods. While implementing the method, we could only consider the schedule on one machine since all the machines synchronize, as it is indicated in Theorem 1.

In the workload moving procedure, the time periods are considered from the most expensive one ρ_1 to the cheapest ρ_k . If there exists workload in time period ρ_q , then let p^{ρ_q} denote the jobs' processing time in time period ρ_q , and let $I^{\rho_{q'}}$ be the machine idle time in time period $\rho_{q'}$, $\forall q' \in q+1, \dots, k$. Then, the workload moving strategy is as follows.

- 1) If $p^{\rho_q} \leq I^{\rho_{q'}}$, move the workload p^{ρ_q} to the time period $\rho_{q'}$.
- 2) If $p^{\rho_q} > I^{\rho_{q'}}$, move a part of the workload $I^{\rho_{q'}}$ to the time period $\rho_{q'}$, and the remaining workload $p^{\rho_q} - I^{\rho_{q'}}$ will be moved to the future time period.

An iteration-based algorithm is then designed to find the optimal schedule for the lower bound problem. Let D_{ρ_q} , $\forall q \in K$ denote the best schedule after moving the workload of time period ρ_q to periods $\rho_{q+1}, \rho_{q+2}, \dots, \rho_k$. As for the

boundary condition, let D_{ρ_0} denote the initial scheduling plan, and $V(D_{\rho_q})$ is used to represent the objective value for the scheduling plan D_{ρ_q} . Hence, the iteration formula is demonstrated as follows:

$$V(D_{\rho_q}) = \min\{V(D_{\rho_{q-1}}), V(D_{\rho_{q-1}}) - V(\gamma_{\rho_q, \rho_\pi})\}, \quad \forall q \in K \quad (36)$$

where $\gamma_{\rho_q, \rho_\pi}$ denotes the moving of workload in time period ρ_q to the time period set ρ_π , and $V(\gamma_{\rho_q, \rho_\pi})$ stands for the decreased objective value after the moving operation $\gamma_{\rho_q, \rho_\pi}$. In particular, π is a subset of $\{q+1, \dots, k\}$, which means that the workload in time period q will not be moved to an earlier time period. In fact, it could be proved that the objective value will not decrease by moving the workload to an earlier time period, which is illustrated in Theorem 2. An instance to implement this search algorithm is shown as follows.

As it is shown in Fig. 2, an instance of a typical workload moving in the lower bound algorithm is demonstrated. The index of time periods is labeled on the top of each colored square, and the time periods with different electricity prices are marked by different colors, i.e., the red squares represent time periods with on-peak electricity price, the yellow time periods have mid-peak electricity price, and the green squares indicate regions with off-peak electricity price. In order to demonstrate the improvement of both steps more explicitly, it is assumed that $\alpha \cdot m \cdot (e_2 - e_3)/\text{MEC} \geq \beta/\text{MP}$ and $\alpha \cdot m \cdot (e_2 - e_5)/\text{MEC} \geq \beta/\text{MP}$.

In Step 1 of Fig. 2, the workload of time period 2 (on-peak) is moved to time period 3 (mid-peak), which has a larger index and idle time. However, the idle time of time period 3 is less than the workload of time period 2. Then, in Step 2, the remaining workload from Step 1 is moved to another time period with an even larger index and idle time, which is now sufficient to contain the rest of the workload. Fig. 2 showcases a typical job reallocation procedure implemented by the lower bound algorithm. In Algorithm 2, the nonpreemption constraint has been relaxed, thus making the obtained optimal solution better than that of the original problem. In real cases, the production could be under the preemption setting; however,

Algorithm 2 Lower Bound Algorithm

```

1 Initialization, let  $P/m$  be the total workload that the jobs
  should be processed on one machine.
2 Let the jobs be processed on time period 0 to  $P/m$ , and
  its objective value is  $v_{\text{opt}}$ .
3 Let  $\rho_1, \rho_2, \dots, \rho_k$  be the index of time periods with the
  decreasing order of electricity price. Besides, if  $e_{t_1} = e_{t_2}$ 
  and  $t_1 < t_2$ , then  $\rho_q = t_1, \rho_{q+1} = t_2$ .
4 for  $q=1:k$  do
5   Let  $p^{\rho_q}$  be the actual workload in time period  $\rho_q$ .
6   for  $s = q : k$  do
7     if  $I^{\rho_s} \neq 0$  then
8       if  $\alpha \cdot m \cdot (e_q - e_s)/\text{MEC} \geq \beta/\text{MP}$  then
9         if  $I^{\rho_{q'}} \geq p^{\rho_q}$  then
10          Move the processing  $p^{\rho_q}$  to time periods
             $\rho_s$ , and let  $v'$  be the objective value of the
            derived schedule
11        else
12          Move the amount of workload  $I^{\rho_{q'}}$  to
            time periods  $\rho_s$ , and move the rest
            workload to other time periods which
            obey the inequality
             $\alpha \cdot m \cdot (e_{\rho_q} - e_{\rho_s})/\text{MEC} \geq \beta/\text{MP}$ . Let  $v'$  be
            the objective value of the derived schedule
13        if  $v_{\text{opt}} \geq v'$  then
14          Save this scheduling plan as the current
            scheduling plan;

```

since the processing sequence is altered, the installation and accurate positioning of the job on mechanical working should be taken into account. To facilitate the demonstration of the workload moving method, the pseudocode of the lower bound algorithm is listed as follows.

Let e_{ρ_s} denote the electricity price of the time period ρ_s . In the lower bound algorithm, the amount of workload to be moved equals $\min\{p^{\rho_q}, I^{\rho_s}\}$. Let $\gamma_{\rho_q, \rho_s, t}$ stand for the moving of workload t from time period ρ_q to ρ_s , and $V(\gamma_{\rho_q, \rho_s, t})$ represents the improved objective value. Then, the following lemma could be obtained.

Lemma 4: Given time periods $q, s \in K$, if $V(\gamma_{\rho_q, \rho_s, 1}) \geq 0$, then $\forall t \in [0, \min\{p^{\rho_q}, I^{\rho_s}\}]$, and the inequality $V(\gamma_{\rho_q, \rho_s, \min\{p^{\rho_q}, I^{\rho_s}\}}) \geq V(\gamma_{\rho_q, \rho_s, t})$ holds.

Proof: In the lower bound algorithm, if $V(\gamma_{\rho_q, \rho_s, 1}) \geq 0$, the inequality holds $\alpha \cdot m \cdot (e_{\rho_q} - e_{\rho_s})/\text{MEC} \geq \beta/\text{MP}$. It means that by moving one more unit of workload from time period ρ_q to ρ_s , the objective value decreases, i.e., $V(\gamma_{\rho_q, \rho_s, 2}) - V(\gamma_{\rho_q, \rho_s, 1}) \geq 0$. Thus, the objective value $V(\cdot)$ is monotonically increasing with respect to t , which is equivalent to state that $\forall t \in [0, \min\{p^{\rho_q}, I^{\rho_s}\}]$, $V(\gamma_{\rho_q, \rho_s, \min\{p^{\rho_q}, I^{\rho_s}\}}) \geq V(\gamma_{\rho_q, \rho_s, t})$. $\square$

Based on Lemma 4, the number of workload moving could be reduced significantly since the workload is now moved based on the time period instead of the time slot. For each time period $t \in K$, the workload is redistributed to the other $k-1$ time periods following a determined order. Hence, it could be

easily proved that the computational complexity of the lower bound algorithm is $O(k^2)$, where k is the number of time periods.

Theorem 2: The lower bound algorithm constructs an optimal schedule for the lower bound problem.

Proof: For two different time periods ρ_t and $\rho_{t'}$, let $\rho_t > \rho_{t'}$, where $t, t' \in K$. Then, we aim to prove that if moving the workload from time period ρ_t to $\rho_{t'}$ in the lower bound algorithm, the objective value will not decrease.

Assume that there is an instance where by moving the workload from time period ρ_t to $\rho_{t'}$, the objective value could be reduced. Hence, the moving has to cause reduction of the makespan because $e_{\rho_t} \geq e_{\rho_{t'}}$. Let w be the last time period where there is a positive workload. The makespan could be reduced only by moving the workload in time period w to w' , where $w' < w \forall w' \in K$.

In the lower bound algorithm, the initial schedule is proposed with the minimum makespan. Hence, in the first step of moving, the makespan would not decrease. In the later steps, the makespan could be reduced only by moving the workload in time period w to w' , where $w' < w \forall w' \in K$. However, if this situation is true, it means that the workload could be moved to a better location in the earlier moving steps. This situation violates the rule of a lower bound method, which requires the workload to be moved to the location where it could reduce the objective value to the most extent.

Therefore, by moving the workload from time period ρ_t to $\rho_{t'}$, it could not reduce the objective value. Moreover, it also means that the lower bound method has searched all the moving possibilities, and an optimal solution is guaranteed. $\square$

After introducing the lower bound method, a dispatch rule-based algorithm is proposed to obtain the upper bound of the optimal objective value for the unrelated parallel machine scheduling problem with TOU electricity price. In the dispatch rule, we consider a straightforward approach to assign all the jobs to the machines with the minimum job-machine processing time. Thereafter, without considering the effect of TOU electricity cost, the jobs will be processed on their corresponding machines one by one. The pseudocode of the upper bound algorithm is shown as follows.

V. COMPUTATIONAL RESULTS

A reported survey of 43 TOU pricing programs targeting industrial customers is offered by U.S. utility [22]. In their research, the TOU policies can be reduced to two models based on the number of on-peak time periods. That is, Mid-Peak $\rightarrow$ On-Peak $\rightarrow$ Mid-Peak $\rightarrow$ Off-Peak and Mid-Peak $\rightarrow$ On-Peak $\rightarrow$ Mid-Peak $\rightarrow$ On-Peak $\rightarrow$ Off-Peak. Hence, in Section V, these two typical TOU electricity pricing policies are considered. To elaborate, the TOU electricity prices of our computational examples are listed as follows.

In Table I, the electricity price of the off-peak, mid-peak, and on-peak scenarios are assumed as 4, 2, and 1 for the sake of simplicity. In Policy A, there are one on-peak time period, one off-peak time period, and two mid-peak time periods. In comparison, there is one additional on-peak time period

Algorithm 3 Dispatch Rule-Based Algorithm

```

1 Initialization,  $m, n, k$  is the number of machine, job and
  time period respectively.  $P$  is the set of processing time
  and  $PL$  is the set of length of each time period.
2 Obtain the relaxed solution  $x_{ij}$ .
3 for  $j=1:n$  do
4   Find the index  $\hat{i}$  of maximal  $x_{ij}$ , let  $x_{\hat{i}j} = 1$  and
    $x_{ij} = 0$  for all  $i \in M \setminus \hat{i}$ .
5 Scheduling the jobs one by one on each machine without
  idle time, let  $v_{opt}$  be the objective value of this
  scheduling.
6 for  $j=1:n$  do
7   for  $i=1:m$  do
8     Let  $x_{ij} = 1$  and  $x_{i'j} = 0$  for all  $i' \in M \setminus i$ , let  $v'$  be
     the objective value of this scheduling.
9     if  $v' \leq v_{opt}$  then
10      Saving the new scheduling plan
11    else
12      Return to the last scheduling plan
13 Calculate the objective value of the original scheduling
  problem.

```

TABLE I
TOU ELECTRICITY PRICING POLICY

	On-peak		Mid-peak		Off-peak	
	Time period(h)	Electricity price	Time period(h)	Electricity price	Time period(h)	Electricity price
Policy A	13-19	4	7-13 19-23	2 2	23-7	1
Policy B	7-12 16-20	4 4	0-7 12-16	2 2	20-24	1

in Policy B. These two TOU electricity pricing policies would be employed to test our algorithms.

In this section, we present a series of computational examples to test the performance of the proposed method. All the algorithms listed here are implemented by MATLAB R2016a. The numerical experiments are performed on a laptop with a 3.1-GHz CPU and 8-GB memory. The RSOCP model would be solved by CVX, and the MIP_i and $RMIP_i$ models would be solved by CPLEX 12.6 with a stopping time of 300 s. In order to compare with the optimal solution, the RMIP model is solved via CPLEX, where the variable x_{ijt} is used to replace x_{ij} and z_{jt} , with a stopping time of 3600 s. Although solving the RMIP model is supposed to reach optimality, due to the curse of dimensionality, the computation time is huge as the size of the problem grows.

In this section, we compare the performance of the F-SOC algorithm with the lower bound method, CPLEX, and the dispatch rule-based algorithm. The numerical instances are categorized in terms of the number of jobs, i.e., $n \in \{8, 10, 12, 14\}$ (small-scale instances), $n \in \{20, 25, 30, 35\}$ (medium-scale instances), and $n \in \{40, 50, 60, 70\}$ (large-scale instances). Moreover, the instances are also categorized by the number of

TABLE II
PERFORMANCE COMPARISON FOR SMALL-SCALE INSTANCES

m	n	CPLEX	Lower bound		F-SOC			Dispatch rule	
		CPU time (sec)	CPU time (sec)	Gap LC (%)	CPU time (sec)	Gap C (%)	Gap L (%)	CPU time (sec)	Gap LD (%)
3	6	0.35	0.01	3.32	0.10	3.79	6.98	0.01	10.20
	8	0.35	0.01	3.00	0.13	3.45	6.35	0.01	6.40
	10	0.46	0.01	2.67	0.17	3.37	5.95	0.01	6.00
	12	0.70	0.01	2.15	0.63	2.48	4.58	0.01	4.50
5	6	0.35	0.01	5.25	9.85	5.96	10.89	0.01	8.34
	8	0.32	0.01	4.46	12.51	4.59	8.84	0.01	8.97
	10	0.48	0.01	5.74	15.28	4.08	9.59	0.01	9.27
	12	0.95	0.01	7.86	18.39	3.59	11.17	0.01	10.34
7	6	0.32	0.01	7.16	14.20	7.77	14.37	0.01	14.37
	8	0.60	0.01	6.31	17.89	6.33	12.23	0.01	12.76
	10	1.10	0.01	6.25	22.52	5.66	11.56	0.01	11.07
	12	1.12	0.01	6.48	27.17	5.41	11.54	0.01	11.62

machines, $m \in \{3, 5, 7\}$. In the objective function, the weights of productivity and electricity cost are designated as $\alpha = 0.7$ and $\beta = 0.3$. Besides, to simulate the actual variations of manufacturing processes, the processing times of jobs are uniformly distributed with respect to $U[20,50]$.

In Tables II–IV, the average performances of the proposed algorithm over 30 randomly generated instances for small to large scale are reported, respectively. Each instance is solved by CPLEX, the lower bound method, the F-SOC algorithm, and the dispatch rule-based algorithm.

Let $V(\text{algorithm})$ represent the objective value obtained by an algorithm. In small-large instances, the following performance indicators are to describe the effectiveness of the algorithms. Gap LC is defined as the difference between lower bound method and CPLEX solution, and it is obtained via $[V(\text{LB}) - V(\text{CPLEX})]/V(\text{LB}) \times 100\%$. This indicator could be used to illustrate the closeness of the solutions obtained via a lower bound algorithm and the optimal solution. Gap L is to measure the difference between the proposed F-SOC algorithm and the lower bound method, and it is calculated as $[V(\text{F-SOC}) - V(\text{LB})]/V(\text{F-SOC}) \times 100\%$. Gap C is to demonstrate the difference between the proposed F-SOC method and CPLEX solution; it could be obtained via $[V(\text{F-SOC}) - V(\text{CPLEX})]/V(\text{F-SOC}) \times 100\%$. Gap LD illustrates the difference between the upper bound and the lower bound, which is derived as $[V(\text{Dispatch rule}) - V(\text{LB})]/V(\text{Dispatch rule}) \times 100\%$. Since CPLEX could not obtain an optimum solution for medium to large size problems in 1 h, the performance indicators Gap LC and Gap C may not be accurate to describe the performance of the proposed F-SOC and lower bound algorithms. Although the two indicators could be applied to obtain the closeness of the proposed methods and the optimum solutions in small-size problems, in medium and large problems, Gap LD together with Gap L could pinpoint the relative gap between the proposed F-SOC and the optimum solution, which is located in between the obtained two bounds.

In Table II, the average results of over 30 trials for small-scale instances are reported. For these cases, the scheduling problem could be solved by CPLEX quickly. It should be pointed out that the small-case instances always can be solved in seconds via CPLEX. However, the CPU time of CPLEX would increase sharply when the number of

TABLE III
PERFORMANCE COMPARISON FOR MEDIUM-SCALE INSTANCES

m	n	CPLEX	Lower bound		F-SOC			Dispatch rule	
		CPU time (sec)	CPU time (sec)	Gap LC (%)	CPU time (sec)	Gap C (%)	Gap L (%)	CPU time (sec)	Gap LD (%)
3	15	1.42	0.01	3.45	11.87	1.84	5.32	0.01	5.70
	20	51.62	0.01	1.71	15.82	1.42	3.11	0.01	4.50
	25	785.56	0.01	0.43	19.72	1.25	1.68	0.01	5.30
	30	3600	0.01	0.81	24.44	1.11	1.91	0.01	7.10
5	15	12.87	0.01	5.95	23.21	3.40	9.16	0.01	9.23
	20	354.38	0.01	5.09	31.53	2.80	7.74	0.01	8.74
	25	3600	0.01	5.56	40.48	2.75	8.21	0.01	9.13
	30	3600	0.01	4.04	50.57	2.63	6.56	0.01	8.72
7	15	2.12	0.01	7.52	34.43	4.19	11.40	0.01	11.94
	20	488.21	0.01	5.16	47.65	3.69	8.66	0.01	10.73
	25	3600	0.01	7.92	63.84	2.71	10.42	0.01	12.17
	30	3600	0.01	5.55	87.13	2.49	7.91	0.01	11.32

variables up to the determine lever. This phenomenon can be observed in Table III. When the number of machines is 3, Gap LC is always less than 4%, and Gap C is also less than 4%. Although Gap LC and Gap C would increase with the number of machines, the values of Gap LC is always less than 8%, and the value of Gap C is always less than 6% and 8% for the instances where $m = 5$ and $m = 7$, respectively. It means that both the solutions of the lower bound method and the F-SOC algorithm are quite close to the optimal solution. Interestingly, Gap C would decrease with the increasing number of jobs. In these small-scale instances, the dispatch rule-based method and F-SOC algorithm have similar gaps between the lower bound approach.

The computation result of medium-scale instances is reported in Table III. The CPU time of CPLEX increases rapidly in comparison with the small-scale instances. In detail, the F-SOC algorithm has significantly shorter CPU time than CPLEX when the number of jobs is large than 20. It should be pointed out that CPLEX does not always provide an optimal solution, given the stopping time is preset as 3600 s. In the medium-scale instances, both the lower bound method and F-SOC algorithm have smaller gaps than the small-scale instances. The values of Gap C are less than 2%, 4%, and 5% for the instance where $m = 3$, $m = 5$, and $m = 7$, respectively. Moreover, the Gap LD between the lower bound and the upper bound becomes significantly larger than the Gap C, which measures the difference between our proposed algorithm and CPLEX solution.

Table IV demonstrates the computation result of large-scale instances. In these instances, all the CPU times of CPLEX are equal to 3600, which means that CPLEX cannot solve to optimum under this problem scale setting. However, the solution of CPLEX could still be regarded as quite close to the optimal solution, because Gap LC, the gaps between lower bound and CPLEX, is less than 7%, 5%, and 8% for the instance where $m = 3$, $m = 5$, and $m = 7$, respectively. Moreover, the values of Gap C, the gaps between F-SOC and CPLEX solutions, are less than 1%, 3%, and 3% for the instances where $m = 3$, $m = 5$, and $m = 7$, respectively. In large-scale instances, the value of Gap LD increases dramatically, which indicates that the upper bound may not be tight without considering the effect of TOU electricity cost. In the large-scale instances, the F-SOC algorithm has significantly shorter CPU time than

TABLE IV
PERFORMANCE COMPARISON FOR LARGE-SCALE INSTANCES

m	n	CPLEX	Lower bound		F-SOC			Dispatch rule	
		CPU time (sec)	CPU time (sec)	Gap LC (%)	CPU time (sec)	Gap C (%)	Gap L (%)	CPU time (sec)	Gap LD (%)
3	40	3600	0.01	3.50	34.86	0.24	3.73	0.01	15.36
	50	3600	0.01	4.27	47.28	-0.34	3.95	0.01	21.13
	60	3600	0.01	6.27	79.34	-0.28	6.01	0.01	20.03
	70	3600	0.01	2.07	108.55	0.14	2.21	0.01	24.71
5	40	3600	0.01	2.97	88.86	2.13	5.04	0.01	9.28
	50	3600	0.01	3.75	132.41	1.72	5.41	0.01	10.37
	60	3600	0.01	3.75	193.69	1.01	4.72	0.01	12.45
	70	3600	0.01	4.23	262.82	1.24	5.42	0.01	18.42
7	40	3600	0.01	6.32	153.07	2.05	8.25	0.01	13.28
	50	3600	0.01	5.63	236.42	1.59	7.14	0.01	14.26
	60	3600	0.01	5.56	338.57	2.17	7.61	0.01	17.17
	70	3600	0.01	7.61	472.08	1.09	8.62	0.01	21.25

TABLE V
PERFORMANCE COMPARISON FOR DIFFERENT WEIGHTS

m	n	Gap C (%)				
		$\alpha, \beta = (0.9, 0.1)$	(0.7, 0.3)	(0.5, 0.5)	(0.3, 0.7)	(0.1, 0.9)
3	20	1.17	1.42	1.82	2.07	2.96
	50	0.27	-0.34	0.63	1.24	1.81
5	20	2.56	2.80	3.02	3.41	3.92
	50	1.47	1.72	1.98	2.27	3.17
7	20	3.45	3.69	3.98	4.13	4.41
	50	1.37	1.59	1.72	2.51	2.82

CPLEX and nearly objective value with CPLEX. Specially, the F-SOC algorithm has better objective value than CPLEX for several instance, i.e., $m = 3$, $n = 50$ and $m = 3$, $n = 60$.

Tables II–IV represent the trend that the F-SOC algorithm has gradually less gap of objective value than CPLEX. Meanwhile, the CPU time of the F-SOC algorithm is increased significantly slower than CPLEX. In industry applications, a large number of jobs are considered, but the solving time is limited. Hence, the F-SOC algorithm can play an important role in industry application.

Since, in reality, different levels of emphasis could be assigned to productivity and electricity cost, we then test the instances for different weights of terms in the objective function. In the instances mentioned earlier, the number of machines and jobs are assumed as $n = \{20, 50\}$ for 3, 5, and 7 machines. The computation results are shown as follows.

In Table V, the performance comparison of F-SOC algorithm for different weights is illustrated. The gap employed here is Gap C, which could be calculated via $(V(\text{F-SOC}) - V(\text{CPLEX}))/V(\text{F-SOC}) \times 100\%$. This gap increases slightly with the decrease of α/β ratio. In particular, this gap reaches its largest value when $\alpha = 0.1$ and $\beta = 0.9$. However, under each weight setting, the value of the gap will not exceed 3%, 4%, and 5% for the instances where $m = 3$, $m = 5$, and $m = 7$, respectively.

Meanwhile, the instances under various electricity pricing policies are also examined to validate the performance of the F-SOC approximate algorithm, and the performance analysis is shown in Table VI.

In Table VI, the gap used is also Gap C, which could be calculated as $(V(\text{F-SOC}) - V(\text{CPLEX}))/V(\text{F-SOC}) \times 100\%$. It is observed that the gaps for Policy A and Policy B have

TABLE VI
PERFORMANCE COMPARISON FOR DIFFERENT POLICIES

Policy	M #	Gap C (%)						
		n=10	n=20	n=30	n=40	n=50	n=60	n=70
A	m=3	3.37	1.42	1.11	0.24	-0.34	-0.28	0.14
	m=5	4.58	2.80	2.36	2.13	1.72	1.01	1.24
	m=7	5.66	3.69	2.49	2.05	1.59	2.17	1.09
B	m=3	3.44	2.72	2.02	1.33	0.72	0.59	0.21
	m=5	4.40	3.01	2.54	2.07	1.82	1.71	1.20
	m=7	5.72	4.10	2.58	2.02	2.10	1.94	1.59

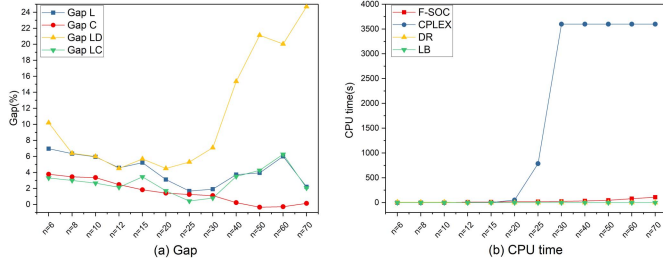


Fig. 3. Algorithm performance comparison for the instances when $m = 3$. (a) Gap. (b) CPU time.

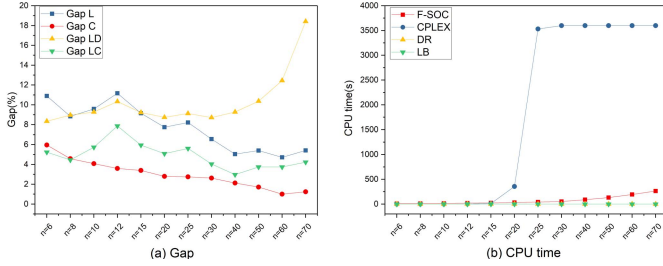


Fig. 4. Algorithm performance comparison for the instances when $m = 5$. (a) Gap. (b) CPU time.

a similar trend, which, in general, increases as the number of machines increases and decreases as the number of jobs increases. Besides, the gaps for different pricing policies with the same job-machine numbers are rather close.

To better depict the relationship between various settings of job-machine pairs, the curves of the above-defined gaps are presented in Figs. 3–5. In Fig. 3, the average algorithm performance over ten trials for the instances where $m = 3$ are portrayed. It also shows that the value of Gap C would decrease with the increasing number of jobs. Moreover, the value of Gap C is always less than 4% when the number of machines is equal to 3. Gap L and Gap LD have similar values when $n \leq 15$, but the value of Gap L becomes smaller than the value of Gap LD to a great extent when $n > 15$. The CPU times of the F-SOC algorithm, lower bound method, and dispatch rule-based approach increase slowly with the increase of the number of jobs, while the CPU time of CPLEX is skyrocketing when $n > 15$.

Similarly, the average results over ten trials for the instances where $m = 5$ are depicted in Fig. 4. In general, the gaps emerge a similar trend, as shown in Fig. 3. However, the values of Gap C are always less than 6%. Specifically, when the number of jobs is equal to 6, 10, or 12, the value of Gap LD

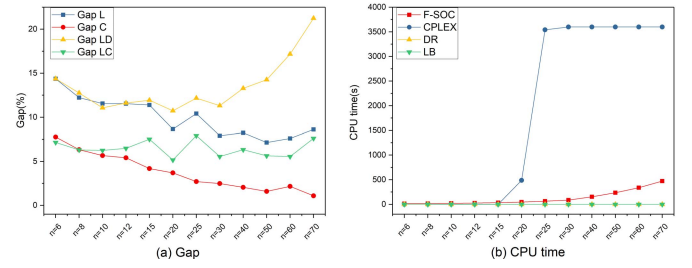


Fig. 5. Algorithm performance comparison for the instances when $m = 7$. (a) Gap. (b) CPU time.

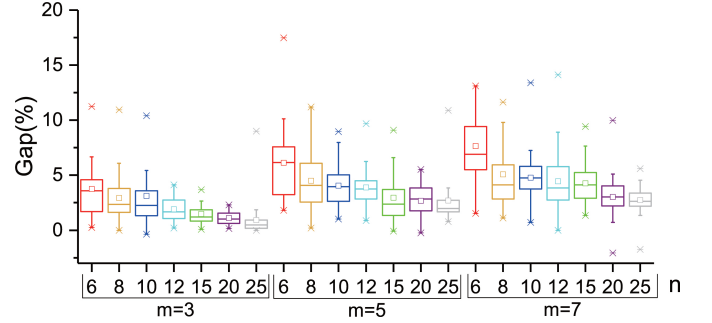


Fig. 6. Gaps between F-SOC and CPLEX results for various values of n and m .

is less than the value of Gap L. However, when the number of jobs grows larger than 15, the values of Gap C become significantly less than the values of Gap LD.

In Fig. 5, the average computation outcomes over ten trials for the instances where $m = 7$ are portrayed. It could be observed that the values of Gap C maintain a decreasing trend with the increasing number of jobs, and as the number of jobs exceeds 30, the value of Gap C is always less than 2.5%.

In order to reflect the statistical properties of the proposed F-SOC algorithm, as shown in Fig. 6, the box plot depicts the gap between F-SOC and CPLEX solution over 21 combinations in terms of job and machine numbers. Each of the job-machine combinations consists of 30 randomly generated instances with their statistics drawn. For most of the instances, Gap C would not exceed 5%, 7%, and 10% when $m = 3$, $m = 5$, and $m = 7$, respectively. Although there are several outliers marked in the figure, the gap values of most extreme outliers tend not to exceed 15% in the statistical analysis outcome. Furthermore, there is a clear decreasing trend of gaps in each group with a fixed number of machines as the number of jobs increases. There is a slight increase in gap value for the same amount of jobs with the increasing of machines.

VI. CONCLUSION

In this study, the unrelated parallel machine scheduling problem with the TOU electricity pricing policy is explored. With a multiobjective MQIP model, an equivalent MISOCP is formulated with a group of SOC constraints and some special treatments of the binary integer programming. A series of cutting planes are designed based on the unique structure of the problem. It is proven that the optimal scheduling plan of MISOCP could be rescheduled as the optimal scheduling

plan of MQIP. However, the optimal solution of the MISOCP model is still difficult to obtain due to the existence of binary variables. Hence, the continuous relaxation is implemented to handle the SOCP model. Then, the RSOCP model, the relaxed version of SOCP, is solved by the local search-based F-SOC algorithm to obtain a high-quality solution with fast speed. In comparison, the optimal solution of MISOCP could also be derived via the B&B method based on the SOCP model, which would require a large amount of computation time. Moreover, we also design a lower bound algorithm to obtain the lower bound of MQIP. It is proven that the proposed lower bound method could always find an optimal solution for the relaxed scheduling problem without nonpreemption constraint. Meanwhile, a dispatch rule-based approach is introduced to provide an upper bound of MQIP by ignoring the TOU electricity cost. In the numerical analysis section, an extensive experiment is conducted for various sizes of instances randomly generated. Statistically, it is observed that the proposed F-SOC algorithm could obtain a high-quality solution, which is, in general, within the 4% gap from CPLEX solution, in a short period of time (7 min for 70 jobs). It is also found that the proposed lower bound approach could maintain a narrow gap from the CPLEX solution, with an even faster computation speed (approximately 0.01 s). To further demonstrate our algorithm, the original scheduling problem with various weights on productivity and electricity cost is analyzed. Besides, different outcomes due to two types of typical TOU electricity pricing policies are presented. For industrial applications, the proposed approximate algorithm could be employed to generate near-optimal scheduling plans for factories with unrelated parallel machines and distribution strategy for the distributed manufacturing network with unrelated parallel production centers. Taking into account various electricity pricing regulations enforced by the local government, and different emphasis put on the makespan as well as the energy cost, the F-SOC as well as the lower bound algorithms could yield desirable scheduling results in a timely fashion.

ACKNOWLEDGMENT

The authors are grateful for the insightful comments from the guest editors and the two respected referees, especially during this tough period fighting a global battle against COVID-19.

REFERENCES

- [1] A. Che, Y. Zeng, and K. Lyu, "An efficient greedy insertion heuristic for energy-conscious single machine scheduling problem under time-of-use electricity tariffs," *J. Cleaner Prod.*, vol. 129, pp. 565–577, Aug. 2016.
- [2] K. Fang, N. A. Uhan, F. Zhao, and J. W. Sutherland, "Scheduling on a single machine under time-of-use electricity tariffs," *Ann. Oper. Res.*, vol. 238, nos. 1–2, pp. 199–227, Mar. 2016.
- [3] S. Wang, M. Liu, F. Chu, and C. Chu, "Bi-objective optimization of a single machine batch scheduling problem with energy cost consideration," *J. Cleaner Prod.*, vol. 137, pp. 1205–1215, Nov. 2016.
- [4] S. Lee, B. Do Chung, H. W. Jeon, and J. Chang, "A dynamic control approach for energy-efficient production scheduling on a single machine under time-varying electricity pricing," *J. Cleaner Prod.*, vol. 165, pp. 552–563, Nov. 2017.

- [5] J. Cheng, F. Chu, M. Liu, P. Wu, and W. Xia, "Bi-criteria single-machine batch scheduling with machine on/off switching under time-of-use tariffs," *Comput. Ind. Eng.*, vol. 112, pp. 721–734, Oct. 2017.
- [6] A. Che, X. Wu, J. Peng, and P. Yan, "Energy-efficient bi-objective single-machine scheduling with power-down mechanism," *Comput. Oper. Res.*, vol. 85, pp. 172–183, Sep. 2017.
- [7] S. Zhang, A. Che, X. Wu, and C. Chu, "Improved mixed-integer linear programming model and heuristics for bi-objective single-machine batch scheduling with energy cost consideration," *Eng. Optim.*, vol. 50, no. 8, pp. 1380–1394, Aug. 2018.
- [8] B. Chen and X. Zhang, "Scheduling with time-of-use costs," *Eur. J. Oper. Res.*, vol. 274, no. 3, pp. 900–908, May 2019.
- [9] H. Zhang, Z. Dai, W. Zhang, S. Zhang, Y. Wang, and R. Liu, "A new energy-aware flexible job shop scheduling method using modified biogeography-based optimization," *Math. Problems Eng.*, vol. 2017, Aug. 2017.
- [10] Z. Pei, X. Zhang, L. Zheng, and M. Wan, "A column generation-based approach for proportionate flexible two-stage no-wait job shop scheduling," *Int. J. Prod. Res.*, vol. 58, no. 2, pp. 487–508, Jan. 2020.
- [11] S. Wang, Z. Zhu, K. Fang, F. Chu, and C. Chu, "Scheduling on a two-machine permutation flow shop under time-of-use electricity tariffs," *Int. J. Prod. Res.*, vol. 56, no. 9, pp. 3173–3187, May 2018.
- [12] 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," *Int. J. Prod. Res.*, vol. 52, no. 13, pp. 3922–3939, Jul. 2014.
- [13] J.-Y. Ding, S. Song, R. Zhang, R. Chiong, and C. Wu, "Parallel machine scheduling under time-of-use electricity prices: New models and optimization approaches," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 1138–1154, Apr. 2016.
- [14] S. Zhou, X. Li, N. Du, Y. Pang, and H. Chen, "A multi-objective differential evolution algorithm for parallel batch processing machine scheduling considering electricity consumption cost," *Comput. Oper. Res.*, vol. 96, pp. 55–68, Aug. 2018.
- [15] J. Cheng, F. Chu, and M. Zhou, "An improved model for parallel machine scheduling under Time-of-Use electricity price," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 2, pp. 896–899, Apr. 2018.
- [16] S. Wang, X. Wang, J. Yu, S. Ma, and M. Liu, "Bi-objective identical parallel machine scheduling to minimize total energy consumption and makespan," *J. Cleaner Prod.*, vol. 193, pp. 424–440, Aug. 2018.
- [17] M. Tan, B. Duan, and Y. Su, "Economic batch sizing and scheduling on parallel machines under time-of-use electricity pricing," *Oper. Res.*, vol. 18, no. 1, pp. 105–122, Apr. 2018.
- [18] A. Che, S. Zhang, and X. Wu, "Energy-conscious unrelated parallel machine scheduling under time-of-use electricity tariffs," *J. Cleaner Prod.*, vol. 156, pp. 688–697, Jul. 2017.
- [19] J. Batista Abikarram, K. McConky, and R. Proano, "Energy cost minimization for unrelated parallel machine scheduling under real time and demand charge pricing," *J. Cleaner Prod.*, vol. 208, pp. 232–242, Jan. 2019.
- [20] Z. Pei, M. Wan, and Z. Wang, "A new approximation algorithm for unrelated parallel machine scheduling with release dates," *Ann. Oper. Res.*, vol. 285, nos. 1–2, pp. 397–425, Feb. 2020.
- [21] W. Yi *et al.*, "ε constrained differential evolution using half-space partition for optimization problems," *J. Intell. Manuf.*, 2020, doi: [10.1007/s10845-020-01565-2](https://doi.org/10.1007/s10845-020-01565-2).
- [22] Y. Wang and L. Li, "Time-of-use electricity pricing for industrial customers: A survey of U.S. utilities," *Appl. Energy*, vol. 149, pp. 89–103, Jul. 2015.



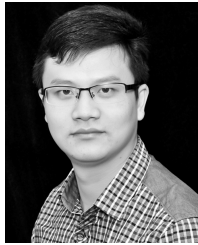
Zhi Pei (Member, IEEE) received the B.S. and Ph.D. degrees in industrial engineering from Tsinghua University, Beijing, China, in 2005 and 2011, respectively.

He was a Visiting Professor with North Carolina State University, Raleigh, NC, USA, in 2015. He is currently an Associate Professor with the Department of Industrial Engineering, College of Mechanical Engineering, Zhejiang University of Technology, Hangzhou, China. He has published academic articles in many journals, such as *Annals of Operations Research*, the *International Journal of Production Research*, *Computers & Industrial Engineering*, *Applied Soft Computing*, and *Information Sciences*. His current research interests include manufacturing system modeling, machine scheduling, nonlinear optimization, and queuing theory.



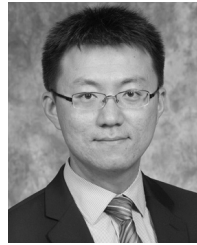
Mingzhong Wan received the M.S. degree in engineering from the Zhejiang University of Technology, Hangzhou, China, in 2019. He is currently pursuing the Ph.D. degree with the School of Business Administration, Northeastern University, Shenyang, China.

His current research interests include scheduling and revenue management.



Zhong-Zhong Jiang (Member, IEEE) received the B.S. and M.S. degrees in physics and the Ph.D. degree in systems engineering from Northeastern University, Shenyang, China, in 2000, 2003, and 2006, respectively.

He was a Senior Research Associate with The Hong Kong Polytechnic University, Hong Kong, and North Carolina State University, Raleigh, NC, USA, in 2008, 2010, and 2011, and a Visiting Professor with the University of Minnesota, Minneapolis, MN, USA, in 2015. He is currently a Professor, the Dean of the School of Business Administration, and the Director of the Institute of Behavioral and Service Operations Management, Northeastern University. He has published academic articles in many journals, such as *Manufacturing & Service Operations Management*, the *European Journal of Operational Research*, *Transportation Research Part B*, *Decision Support Systems*, and the *IEEE TRANSACTIONS ON FUZZY SYSTEMS*. His current research interests include behavioral and service operations management, logistics, and supply chain optimization.



Ziteng Wang received the B.S. degree in mathematics from Tsinghua University, Beijing, China, in 2010, and the M.S. and Ph.D. degrees in industrial engineering from North Carolina State University, Raleigh, NC, USA, in 2011 and 2015, respectively.

He is currently an Assistant Professor with Northern Illinois University, DeKalb, IL, USA. He has published academic articles in many journals, such as the *European Journal of Operational Research*, *Annals of Operations Research*, *Annals of Data Science*, and the *Journal of Industrial and Management*

Optimization. His current research interests include data analytics, operations management, optimization, and applications.



Xu Dai received the B.S. degree in logistics management from Zhejiang Gongshang University, Hangzhou, China, in 2016. He is currently pursuing the master's degree with the College of Mechanical Engineering, Zhejiang University of Technology, Hangzhou.

His current research interests include stochastic modeling and revenue management.