Multi-criteria Batch Scheduling under Time-of-Use Tariffs

Junheng Cheng

NNT : 2017SACLE035

Ordonnancement multicritère par lots avec tarifs d'électricité différenciés

Thèse de doctorat de l'Université Paris-Saclay
préparée à l'Université d'Evry-Val-d'Essonne
et Université Polytechnique du Nord-Ouest

École doctorale N°580 - Sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat: Mathématiques et Informatique

Thèse présentée et soutenue à Evry de soutenance, le 07 Décembre 2017, par

Junheng Cheng

Composition du Jury :

Xiaolan XIE Professeur des Université, Ecole des Mines de Saint Etienne	Président
Lyes BENYOUCEF Professeur des Université, Aix-Marseille University	Rapporteur
Antoine JOUGLET Maître de Conférences, Université de Technologie de Compiègne	Rapporteur
Imed KACEM Professeur des Université, Université de Lorraine	Rapporteur
Eric ANGEL Professeur des Université, Université Evry Val d'Essonne	Examinateur
Feng CHU Professeur des Université, Université d'Evry-Val-d'Essonne	Directrice de thèse
Weili XIA Professeur, Northwestern Polytechnical University	Co-Directeur de thèse
Ming LIU Professeur associé, Tongji university	Co-Directeur de thèse

Thèse de doctorat

THÈSE DE DOCTORAT
DE
L'UNIVERSITÉ PARIS SACLAY

Spécialité: Mathématique et Informatique

préentée et soutenue par

Junheng CHENG

pour obtenir le grade de

Docteur de l'Université Paris Saclay

Titre de la thèse :

Ordonnancement multicritère par lots avec tarifs d'électricité différenciés

soutenue le 07 Décembre 2017

JURY:

M. L. BENYOUCEF	Professeur des Universités	Rapporteur
M. A. JOUGLET	Maître des Conférences - HDR	Rapporteur
M. I. KACEM	Professeur des Universités	Rapporteur
M. E. ANGEL	Professeur des Universités	Examinateur
M. X. XIE	Professeur des Universités	Examinateur
Mme. F. CHU	Professeur des Universités	Directrice de thèse
M. W. XIA	Professeur	Co-encadrant de thèse
M. M. LIU	Professeur associé	Co-encadrant de thèse

Multi-criteria Batch Scheduling under Time-of-Use Tariffs

by

Junheng CHENG

Laboratoire d'Informatique, Biologie Intègrative et Systèmes Complexes (IBISC)

Université d'Evry Val d'Essonne, France

Supervisors: Prof. Feng CHU, Prof. Weili XIA and Prof. Ming LIU

December 07, 2017

Acknowledgements

I am honored to have this opportunity to show my appreciation to those who have helped me over the years of my thesis.

Firstly, I would like to express my sincerest gratitude to Professors Lyes Benyoucef, Antoine Jouglet, Imed Kacem, Eric Angel and Xiaolan Xie who kindly agree to spend time on evaluation of my work and participation in my defense.

My deepest gratitude goes to my supervisor Professor Feng Chu who led me into the operations research area and helped me throughout all the stages of completing this thesis. During the past three years, Professor Chu spent tremendous time guiding me how to identify, analyse and solve specific scientific research issues and how to better demonstrate the ideas and results. When I encountered barriers, she was always there to provide me insightful suggestions. She read the manuscript carefully and corrected the errors word by word. Her rigorous working styles and contagious enthusiasm towards science research influenced me deeply, and these treasured characteristics will accompany me in the future.

I am very grateful to my co-supervisors Professors Weili Xia and Ming Liu for their precious encouragements and helpful advice. Though they work in China, they always kept in touch with me and gave me continuous help, guidance and support whenever I needed.

Special gratitude is extended to Professors Chengbin Chu and Mengchu Zhou for their valuable suggestions and comments on improving a part of this thesis.

I would like to particularly thank Professors Franck Delaplace, Hanna Klaudel, Jean-Marc Delosme, Jean-Christophe Janodet, Madame Murielle Bourgeois, Doctor Laurent Poligny, Monsieur Thierry Millant in Lab. IBISC for their enthusiastic help during my study in France. Many thanks to all my friends and colleagues for their unceasing help and for sharing the happy time in France.

Finally, I would like to owe my special appreciation to my beloved family for their unwavering support and great confidence in me. My parents and sisters give me all their unselfish love, and always stand by me whatever I am facing. I am especially grateful to my husband and comrade Doctor Peng Wu who gave me unconditional love, meticulous caring, and continuous support on the thesis with numerous helpful discussions and useful suggestions.

Résumé

L'industrie est le plus grand consommateur d'énergie dans le monde et la majeure partie de sa consommation est électrique. Pour moduler la consommation et équilibrer les périodes creuses et de pic, les producteurs d'électricité dans de nombreux pays pratiquent une tarification différenciée, en anglais "time-of-use (TOU) policy", afin d'encourager les industriels et les particuliers à adapter leur consommation. Cette stratégie incite les gros consommateurs industriels, en particulier le secteur semi-conducteur où la fabrication se fait souvent par lots, à réduire leurs factures d'électricité en adaptant leur production. Dans ce travail, nous étudions plusieurs problèmes d'ordonnancement de production par lots avec tarification différenciée d'électricité. Nous nous intéressons d'abord à l'ordonnancement d'une machine par lots pour minimiser le coût total d'électricité et le makespan. Le deuxième problème étudié généralise le premier en considérant le coût d'électricité pendant les périodes inactives de la machine telles que les périodes de réglage ou d'attente. Enfin, nous traitons l'ordonnancement sur machines parallèles par lots avec des pièces non identiques. Pour chacun de ces problèmes, nous construisons des modèles mathématiques appropriés, et évaluons sa complexité. Pour la résolution, nous proposons plusieurs méthodes de ε-contrainte dans lesquelles des sous-problèmes sont transformés en problèmes de sac-à-doc, de sacs-à-doc multiples et ou de bin packing. Nous développons aussi une méthode itérative à deux étapes. Les performances des méthodes développées sont évaluées à l'aide d'un grand nombre d'instances représentatives générées au hasard. Les résultats numériques montrent l'efficacité de ces méthodes par rapport au logiciel commercial CPLEX.

Mots clés: Optimisation bi-objectif; Ordonnancement par lots; Tarification d'électricité différenciée; Programmation mathématique; Méthodes de ε-contrainte; Heuristiques.

Abstract

The industrial sector is the largest consumer of the world's total energy and most of its consumption form is electricity. To strengthen the grid's peak load regulation ability, time-of-use (TOU) electricity pricing policy has been implemented in many countries to encourage electricity users to shift their consumption from on-peak periods to off-peak periods. This strategy provides a good opportunity for manufacturers to reduce their energy bills, especially for energy-intensive ones, where batch scheduling is often involved. In this thesis, several bi-objective batch scheduling problems under TOU tariffs are studied. We first investigate a single machine batch scheduling problem under TOU tariffs with the objectives of minimizing total electricity cost and makespan. This primary work is extended by further considering machine on/off switching. Finally, a parallel batch machines scheduling problem under TOU tariffs with non-identical job sizes to minimize total electricity cost and number of enabled machines is studied. For each of the considered problems, appropriate mathematical models are established, their complexities are demonstrated. Different bi-objective resolution methods are developed, including knapsack heuristic based ε-constraint method, multiple knapsack heuristic based ε-constraint method, bin packing heuristic based ε-constraint method and two-stage heuristic based iterative search algorithm. The performance of the proposed methods is evaluated by randomly generated instances. Extensive numerical results show that the proposed algorithms are more efficient and/or effective for the studied problems than the commercial software CPLEX.

Keywords: Bi-objective optimization; Batch scheduling; Time-of-use electricity tariffs; Mathematical programming; ε-constraint method; Heuristics.

Contents

Notation

j: index of jobs

b: index of batches

m: index of machines

i: index of time intervals

J: set of all jobs, i.e., $J = \{1, 2, ..., |J|\}$

B: set of all batches, i.e., $B = \{1, 2, ..., |B|\}$

M: set of all machines, i.e., $M = \{1, 2, ..., |M|\}$

I: set of time intervals on the planning horizon, i.e., $I = \{1, 2, ..., |I|\}$

p_j: processing time of job j, $\forall j \in J$

c_j: size of job j, $\forall j \in J$

C: capacity of each batch

v_m: processing speed of machine m, $\forall m \in M$

q_m: power rate of machine m when processing, $\forall m \in M$

e_i: electricity cost of time interval i, $\forall i \in I$

s_i: starting time of time interval i, $1 \leq i \leq |I| + 1$

S_i: duration of time interval i, $\forall i \in I$

L: a sufficiently large integer

List of Figures

List of Tables

Chapter 1

Introduction

This thesis investigates a new class of scheduling problem: batch scheduling under time-of-use electricity tariffs (BS-TOU). It mainly concerns designing production plans for batch processing machines under variable electricity prices environment to create economic benefits for manufactures and promote sustainable development. In this chapter, the research background is firstly introduced, and then the contributions and outline of the thesis are stated.

1.1 Background

In accordance of the Energy Information Administration [3], the total world's energy consumption is forecasted to expand from 575 quadrillion Btu in 2015 to 736 quadrillion Btu in 2040. Meanwhile, majority of them is generated from nonrenewable natural resources, such as coal, fossil oil, and natural gas. Facing the pressure of increasing energy demand and diminishing nonrenewable natural resources, saving energy has come into a public interest.

Globally, the industrial sector is the largest energy consumer, which contributes about 50% of the world's total energy consumption [53]. Electricity plays an important role in industrial production with rapidly economic development. For example, electricity contributes approximately 30% to total industrial energy consumption in APEC area [9]. However, electricity cannot be efficiently stored, such that it has to be generated, transmitted and consumed instantly. Moreover, electricity demand and supply are imbalanced over time. To meet the customer demand during peak periods, the electricity suppliers have to construct and enable costly backup facilities that are usually less efficient thereby more greenhouse gases can be generated [139]. To reduce the backup facilities investments as well as carbon emissions, demand response (DR) strategy has been carried out by many electricity suppliers due to its flexibility and

inexpensiveness [110]. Time-of-use electricity (TOU) pricing is the most prevalent DR strategy [139], which offers variable electricity prices over time, i.e., high price during on-peak periods and low price during off-peak periods. This provides a good opportunity for electricity consumers to save their electricity cost by shifting their use from on-peak periods to off-peak periods.

Batch processing manufacturing system representing a typical production environment has been widely encountered in modern manufacturing industries, such as steel manufacturing, semiconductor manufacturing, and aircraft industry, and most of them are energy-intensive ones. A specific feature of batch processing is that a processing machine can process multiple jobs at a time. As a result, batch scheduling is usually more complex than traditional production scheduling, because it needs to optimally group the jobs into batches and schedule the formed batches. Majority of batch scheduling problems have been proved to be NP-hard, even under single-machine environments. The existing researches on batch scheduling problems are mainly concerned with improving production efficiency such as optimizing makespan, total completion time or maximum tardiness, while energy cost optimization is usually ignored.

In light of the above analysis, production efficiency is a primary target that decision makers have always been pursuing, while minimizing total electricity cost can also be economically and environmentally beneficial in the context of sustainable development. However, production efficiency optimization and electricity cost minimization are usually conflicting with each other, since the former requires processing "as soon as possible" and the later prefers processing during low-price periods. In other words, there is a trade-off between the total electricity cost and the production efficiency. Batch scheduling under TOU tariffs can be considered as multi-criteria combinatorial optimization problems and their optimization can be achieved by advanced technology and optimization techniques. To the best of our knowledge, there are very few studies on multi-objective batch scheduling under TOU tariffs. This thesis aims to develop new mathematical models and solution approaches for the multi-objective BS-TOU.

1.2 Content and contribution

This thesis mainly focuses on batch scheduling problem under TOU tariffs to minimize total electricity cost and production efficiency. Three BS-TOU problems are investigated successively. We firstly concentrate on single batch processing machine

scheduling with identical job size under TOU tariffs. Then we extend it by considering machine on/off switching to further save the electricity cost when the machine is idling. Finally, we devote our attention to parallel batch machines scheduling with non-identical job sizes under TOU tariffs. For each investigated problem, we develop the mathematical model, analyse the problem property and complexity, and design solution approach based on the characteristics of the problem.

The main contributions of the thesis are summarized as follows.

1) Study three new bi-objective batch scheduling problems under TOU tariffs: a single batch machine scheduling problem with identical job size; a single batch machine scheduling problem with identical job size and machine on/off switching strategy; and a parallel batch machine scheduling problem with non-identical job sizes.

2) For all three problems, effective mathematical models are established, their properties and complexities are analysed, then efficient heuristics are developed based on the characteristics of the problems. The results from computational experiments on extensive randomly generated instances exhibit the well performance of the proposed algorithms.

The remainder of this thesis is organized as below.

Chapter 2 is devoted to literature review on batch scheduling under TOU tariffs and multi-objective optimization methods. We first review the research on batch scheduling, and important research works on scheduling considering energy saving and TOU pricing policy. Then the basic concepts, main solution approaches and performance metrics of multi-objective optimization are explained.

Chapter 3 addresses a bi-objective single batch machine scheduling problem with identical job size under TOU tariffs to minimize total electricity cost and makespan. The problem is first formulated as a bi-objective mixed integer linear programming (MILP) model and is proved to be NP-hard. Then a tighter MILP model is proposed based on property analysis. Two efficient heuristic based ε-constraint methods (ECM), i.e., knapsack heuristic based ECM and multiple knapsack heuristic based ECM (KH-ECM and MKH-ECM in brief), are respectively developed for the problem. The two methods both transform the bi-objective problem into a series of single-objective problems (SOPs). Then knapsack heuristic and multiple knapsack heuristic are respectively developed for each SOP. Finally, the computational results of numerical experiments are presented to evaluate their performance.

In Chapter 4, we extend the studied problem in Chapter 3 to a single batch machine scheduling problem with identical job size and machine on/off switching under TOU tariffs, which aims to simultaneously minimize total electricity cost and makespan. For the problem, we first develop a bi-objective MILP model. Based on optimal batch rule analysis, an improved model is further provided which greatly reduces a Pareto optimal solution search space. A bin-packing heuristic based ε-constraint method (BPH-ECM) is developed for the problem, which first transform the bi-objective problem into a series of SOPs then resolve each SOP through heuristics inspired by assignment rules for bin packing problems. The results of extensive computational experiments confirm the effectiveness and efficiency of the proposed model and algorithm.

Chapter 5 investigates a bi-objective parallel batch processing machine scheduling problem with non-identical job sizes under TOU tariffs to minimize total electricity cost and number of enabled machines. We first establish a bi-objective MILP model for the problem. Then by reformulating the constraints of non-preemption requirement, an improved MILP model is developed. To fast solve the problem, a two-stage heuristic approach is designed to quickly obtain an approximation Pareto front, where the first stage focus on grouping jobs into batches and the second stage allocates the formed batches to parallel machines under TOU tariffs. The computational results on a case study and randomly generated instances show the well performance of the proposed methods.

Chapter 6 concludes this thesis and discusses perspectives for future work.

Chapter 2

Literature review

In this chapter, we first review batch scheduling models and their research progress. Then, energy-saving scheduling studies and time-of-use (TOU) tariffs strategy are recalled, respectively. Subsequently, we present multi-objective combinatorial optimization methods and their performance evaluation metrics.

2.1 Batch scheduling

Scheduling, which distributes and orders tasks to scarce resources, is widely applied in many different areas, e.g. production systems [1], personnel management [17], hospital services [118] and project management [84]. The scheduling for a production system is to make a production plan for a set of jobs (tasks) to be processed on machines (resources), such that one or more production efficiency metrics are achieved on respecting all conditions [60]. Scheduling plays an important role in manufacturing since it can improve the efficiency of resource utilization and reduce the production cost.

Processing jobs in a batch may be cheaper or faster than to process them individually [114], since batching can help to avoid setups or facilitate material handling. There are two types of batching in the literature: serial batching and parallel batching [143]. In serial batching, the processing time of a batch equals the total processing times of its jobs, and a setup time that reflects tool changing or machine cleaning is required for each batch [14],[114]. While in parallel batching, a machine can simultaneously processing multiple jobs at a time, and the length of a batch is its largest job processing time. This thesis focus on the latter one, which is extensively involved in manufacturing industries, e.g. semiconductor industry [101], steel industry [115], aircraft industry [131], shoe manufacturing industry [56]. Most of them are high energy

consuming ones, where the reduction of energy cost makes important contribution to the enterprise's profit [63], [97].

2.1.1 Variety of batch scheduling problems

A classic scheduling problem can be described by a triplet $\alpha|\beta|\gamma$ [59]. α states the scheduling environments, β indicates the job characteristics and the restrictive requirements, and γ expresses the objective functions to be optimized. A batch scheduling problem can be described with the triplet notation by adding B in the β field. For example, $1|B|C_{max}$ means scheduling a single ($\alpha = 1$) batch machine ($\beta = B$) with the objective of minimizing makespan ($\gamma = C_{max}$). Next, we introduce the three fields more specifically.

(1) α describing the the scheduling environments as follows:

Single machine: denoted by 1. It is the simplest machine environment and a basis of more complicated machine environments.

Identical parallel machines (P): There are M identical machines. A job has to be processed on one of the M identical machines with the same processing time.

Uniform parallel machines (Q): There are M machines with different machine speed rates v_m, where $m = 1, 2, ..., M$. The processing time of nonpreemptive job j on machine m is $p_{m,j} = p_j/v_m$, where p_j denotes the processing time of job j on machine with $v_m = 1$. If all the machines have the same speed, the environment reduces to identical parallel machines.

Unrelated parallel machines (R): There are M parallel machines, where machine m can process job j with speed rate $v_{m,j}$. The processing time of job j on machine m equals to $p_j/v_{m,j}$. If the speed of each machine is independent on the jobs, then the unrelated parallel machines can be reduced to the uniform parallel machines.

Flow shop (F): There are M machines in series. Each job has to be sequentially processed on each machine. The processing route of all jobs on the machines are the same.

Job shop (J): There are M machines. Each job has to be processed on M' machines ($1 \leq M' \leq M$), and the processing route of job on the M' machines is pre-known.

Open shop (O): There are M machines. Each job has to be processed on M' machines ($1 \leq M' \leq M$), and there is no restriction on the processing route of each job. In addition, a job can be processed on a machine once or multiple times. Note

that in the previous machine environments, each job is processed at most once on a given machine.

Except the machine environment, **time of use tariffs** provide a scheduling environment with variable prices. It is denoted as TOU in the α field. For example, $TOU, 1|B|E$ denotes scheduling a single batch machine under TOU tariffs to minimize total electricity cost ($\alpha = TOU, 1$; $\beta = B$; $\gamma = E$).

(2) The β field specifying the following processing constraints

Processing time ($p_{m,j}$): The processing time of job j on machine m. If the processing times of job j are identical on different machines, $p_{m,j}$ can be replaced by p_j.

Preemption ($prmp$): The processing of jobs is allowed to be interrupted at any time, and then continued later or processed on another machine. The jobs are preemptive when $prmp$ appears in the β field. Otherwise, the jobs are nonpreemptive.

Release time (r_j): the earliest available time of job j for processing.

Due date (d_j): the date that job j is promised to the customer, or to say the desired date that the processing of job j is completed.

Batch processing (B): Multiple jobs can be simultaneously processed on a machine. We record the capacity of a batch as C.

Job size (c_j): c_j denotes the space occupied by job j in a batch with capacity C. It is assumed that total size of the jobs in a batch, i.e., $\sum_{j \in b} c_j$, is not larger than C. If job sizes are identical, c_j can be equal to 1 and C corresponds to the number of jobs.

(3) The optimized objectives in the γ field are usually **regular objective functions** that often meet two features: a) they are in minimization form; b) objectives are a non-decreasing function of jobs' completion times. Some commonly regular objective functions include:

Makespan (C_{max}): It states the completion time of the last job, i.e., $\max_{j \in J}\{C_j\}$, where C_j denotes the completion time of job j. In batch scheduling, C_{max} refers to the completion time of the last batch. This objective indicates the machine utilization or throughput of the schedule.

Total completion time ($\sum C_j$): the total completion times of all jobs. This objective implies the inventory cost incurred by the schedule.

Maximum lateness (L_{max}): the maximum lateness of all jobs, i.e., $\max_{j \in J}\{L_j\}$, where $L_j = C_j - d_j$.

Number of tardy jobs ($\sum U_j$): U_j is a binary, which equals to 1 if $C_j > d_j$, 0 otherwise.

Total tardiness ($\sum T_j$): the sum of the tardiness of all jobs, where $T_j = \max\{L_j, 0\}$. It reflects the customer dissatisfaction.

Total earliness/tardiness ($\sum |C_j - d_j|$): the total deviation of job completion times from their due dates. A large earliness implies a high inventory cost.

Besides, in a multiple machine environment, the **number of enabled machines** may be minimized when the machines are scarce resources. We denote the number of enabled machines as N in the γ field.

In the context of energy saving, **total energy cost** E minimization has been involved in many works, which reflects economical and ecological consideration of enterprises.

Based on the above notations, the studies on batch scheduling can be classified from three perspectives: machine environment, processing properties or objective functions. This thesis studies single and parallel batch machine scheduling, which will be reviewed in the following subsections.

2.1.2 Single batch machine scheduling

For a single batch machine with identical job size, the first study was probably done by Ikura and Gimple [67]. They investigated makespan minimization and provided a greedy-adjusted-bunching algorithm to determine whether a feasible schedule exists for the case with agreeable release times and due dates (i.e., $r_j \leq r_{j'}$ indicates $d_j \leq d_{j'}$). The algorithm can give optimal makespan if a feasible schedule exists. Lee *et al.* [81] studied a batch problem with agreeable job release times and due dates to respectively minimize maximum lateness and the number of tardy jobs. Efficient dynamic programming-based algorithms were proposed for the problems. Later, Li and Lee [85] proved the problems are strongly NP-hard.

Considering jobs with identical size and different processing times, Chandru *et al.* [21] presented an exact method and several heuristics for a batch machine to minimize total completion time. Then they extended their own work by considering different job families [22]. Sung and Choung [123] studied a batch machine scheduling with release times and presented several heuristics to minimize makespan. For the same problem, Li *et al.* [88] proposed an approximation algorithm with worst-case ratio $2 + \varepsilon$, where ε can be made arbitrarily small.

For batch scheduling with non-identical job size, Uzoy [129] was the first researcher who investigated C_{max} and $\sum C_j$ minimization, respectively. Both problems were proved to be NP-hard and efficient heuristics were provided. Zhang *et al.* [148] demonstrated that the worst-case ratio of first fit longest processing time (FFLPT)

rule proposed by Uzoy [129] is not larger than 2. Furthermore, they presented an approximation algorithm with worst-case ratio 7/4. Later, Uzoy and Yang [130] proposed a branch-and-bound method for a single batch scheduling problem with objective of minimizing total weighted completion time. For the same problem, Azizoglu and Webster [12] proposed another branch-and-bound method which is able to find optimal solutions for instances up to 25 jobs within reasonable time. To minimize makespan, Dupont and Ghazvini [51] presented a successive knapsack problems (SKP) based heuristic and a best fit longest processing time (BFLPT) heuristic. Dupont and Dhaenens-Flipo [50] proposed a branch-and-bound method based on some dominance properties. Kashan *et al.* [74] proposed an algorithm with asymptotic worst-case ratio 4/3 for the instances with agreeable job sizes and job processing times. A simulated annealing (SA) approach, a genetic algorithm (GA) and a hybrid GA were respectively developed for the same problem by Melouk *et al.* [103], Damodaran *et al.* [41] and Kashan *et al.* [73]. Later, Chen *et al.* [30] proposed a clustering algorithm for the problem based on the definition of waste ratio of batch (WAB). Xu *et al.* [143] further developed an ant colony optimization (ACO) based on the definition of WAB. Zhou *et al.* [153] proposed some constructive heuristics for a single batch scheduling problem with release times. To minimize earliness-tardiness, Li *et al.* [92] developed a hybrid genetic algorithm.

In addition, a number of scholars addressed single machine batch scheduling problems with some specific factors, such as incompatible job families [72], [78], penalty of rejected jobs [66], [96], multiple agents [126], integration of manufacture and transportation [31], [79], etc.

2.1.3 Parallel batch machine scheduling

In a parallel machine environment, a job has to be processed by one available machine [89]. If the machines are batch processing ones, the scheduling requires to batch the jobs and determine the batch processing sequence on different machines.

For identical parallel batch scheduling problem, Chang and Damodaran [23] proposed a simulated annealing (SA) approach to minimize makespan for identical job size and release time. Chandru *et al.* [21] presented several heuristics to minimize $\sum C_j$. Considering non-identical job sizes, Lu and Yuan [95] addressed makespan optimization on unbounded parallel batch machines and proposed a heuristic method. Damodaran and Chang [38] proposed several two-step heuristics to minimize makespan. For the same problem, Kashan *et al.* [75] and Damodaran *et al.* [40] developed a hybrid genetic algorithm (HGA) and a genetic algorithm, respectively. Considering job

release times, mixed-integer linear programming (MILP) approaches, heuristics or meta-heuristics (e.g. GA, ACO, SA) were developed to optimize makespan (Mathirajan *et al.* [100], Chung *et al.* [33], Chen and Du [29], Damodaran *et al* **?**, **?**, Chen *et al.* [29], Zhou *et al.* [152]). Considering incompatible job families, where jobs from different families cannot be processed in the same batch, Koh *et al.* [77] proposed constructive heuristics and genetic algorithm to optimize C_{max} and $\sum C_j$, respectively. Malve and Uzsoy [99] proposed a genetic algorithm to minimize maximum lateness. For parallel machines scheduling with non-identical capacities, random-key genetic algorithm (RKGA), particle swarm optimization algorithm (PSO), max-min ant system (MMAS) heuristic and polynomial time approximation scheme were proposed to minimize makespan (Xu and Bean [144], Damodaran *et al.* [39], Jia *et al.* [69] and Li [87]).

For uniform parallel batch scheduling problems, Mor and Mosheiov [107] addressed it with identical jobs and setup times to minimize total completion time. For non-identical job sizes, Suhaimi *et al.* [122] presented a Lagrangian Relaxation (LR) approach to minimize makespan. Further considering non-identical machine capacities, Zhou *et al.* [154] proposed a discrete differential evolution algorithm to optimize the makespan. Li *et al.* [90] investigated the same objective by considering non-identical job release times.

Relatively few works have addressed problems with unrelated parallel batch scheduling. The optimization criterion of the existing works is makespan, and different processing properties are considered. Specifically, Li *et al.* [91] addressed non-identical job sizes and Zhang *et al.* [151] considered non-identical job release times. Arroyo and Leung [10], [11] and Shahidi-Zadeh *et al.* [120] simultaneously took non-identical job sizes and release times into account.

Most of the early research focused on the single and parallel batch machine environment with conventional production efficiency objectives, such as makespan, total completion time. Batch scheduling problems closely related to environmental impacts have not been studied.

2.2 Energy-saving scheduling

Recent years, energy-saving scheduling has attracted public's growing attention due to energy shortage, greenhouse effect and increasing energy cost [37], [98]. Statistical data shows that the total energy demand has been doubled within the last 40 years and it will be double again till 2030 [104]. Moreover, industrial sector contributes

about one-half of the world's total energy consumption [53]. Hence, improving the ratio between energy input and economic output during the industrial manufacturing process, namely, improving energy efficiency of manufacturing systems, plays an important role for sustainable development [57] and can be achieved by technological or organizational measures [116]. The technological measures concentrate on efficiency improvements through machine or product innovations [4],[37],[48],[61],[82],[94], and their main drawbacks are the considerable investment of financial and human resources. The organizational measures aim to improve the energy efficiency via more rationalized planning and scheduling strategies, which are cheaper, faster and more flexible than technological measures. These scheduling strategies can be grouped into saving energy consumption or saving energy cost, which are respectively presented in subsection 2.2.1 and 2.2.2.

2.2.1 Scheduling considering energy consumption saving

Energy saving scheduling can be divided into saving non-processing energy (NPE) consumption and saving processing energy consumption [57].

NPE consumption arises when energy is consumed but is not directly related to the production operation. For example, the energy consumed by machine being turned on, turned off or idling. The most famous strategy for saving NPE consumption is to power down the machine when it is idle, which was firstly arose in computer systems [5],[124]. In production systems, Mouzon *et al.* [109] observed that a large amount of energy is consumed for non-processing operations and proposed a machine turn-on and turn-off strategy (on-off strategy in brief) to minimize total energy consumption. In this strategy, if the turn-on/turn-off cost is lower than idle machine running cost, a machine will be turned off and then turned on when the next processing starts. Later, Mouzon and Yildirim [108] extended the work of [109] to find a trade-off between the total energy consumption and total tardiness. Yildirim and Mouzon [145] proposed to better sequence processing orders to reduce energy-intensive setups. Recently, Che *et al.* [27] proposed to simultaneously optimize energy consumption and maximum tardiness with the power down strategy. For flow shop scheduling, Dai *et al.* [37] applied the on-off strategy to simultaneously optimize total energy consumption and makespan. Bruzzone *et al.* [19] considered peak power consumption saving by adjusting the timetable for a given schedule with pre-determined job processing sequence. Mokhtari and Hasani [104] developed a mathematical model and an evolutionary algorithm to minimize total energy consumption for production and maintenance activities.

Processing energy consumption occurs when energy input directly impacts on the desired output. The processing energy can be saved by speed scaling or job allocation. Speed scaling saves energy by dynamically changing the speed/frequency of the machine for processing, which has been widely investigated in computer systems to extend the battery life [6]–[8], [13]. In production scheduling area, Che *et al.* [26] minimized energy consumption on a single machine with speed scaling strategy and formulated two mixed-integer linear programming models. For parallel machines, Ji *et al.* [68] scheduled jobs with the consideration of their different energy requirements to minimize total resource consumption. For flow shops environment, Fang *et al.* [54] proposed a speed scaling framework to minimize energy cost. Liu *et al.* [93] and Ding *et al.* [48] considered energy consumption saving via selecting more efficient machines in flexible flow shops. Halim and Srinivasan [64] proposed to improve energy efficiency for a serial batch processing machine by exploiting energy recovery potential.

2.2.2 Scheduling considering energy cost saving

Nowadays, electricity plays an increasingly important role in human's society. According to Energy Information Administration of US [2], the electricity consumption in U.S. is expected to keep increasing at an average rate of 0.8% each year through 2040, among which the industrial sector accounts for approximate 25% of the total consumption [86]. However, the distribution of the electricity demand is unbalanced over time. To alleviate the peak load of power grid, demand response (DR) strategy has been carried out in many coutries due to its flexibility and inexpensiveness [110], such as the United States, Canada, France and China [76]. DR strategy aims to guide the electricity users in changing their consumption patterns via providing variable prices, so that reducing the demand when generation cost is high or generation system is jeopardized [57], [76]. Time-of-use (TOU) electricity pricing policy (see Fig. 2.1) is one of the most important and popular DR strategies [139]. It provides high electricity price in on-peak periods and low price in off-peak periods, thus to balance grid's demand and improve peak load regulation ability [49]. Under TOU policy, manufacturing industries, especially power-intensive ones, are motivated to improve their competitiveness by reducing energy cost, for example, processing high energy consumption operations in off-peak periods. Certainly, considering electricity cost together with production efficiency criteria will be of huge significance for manufacturing industry.

In recent years, there has been a growing number of studies on production scheduling to save energy cost under time-of-use tariffs. For single machine environment,

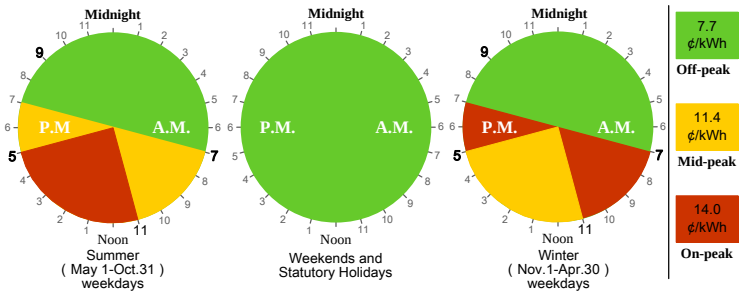


Fig. 2.1: An example of Time-of-Use tariffs (Source: Ontario Energy Board)

Shrouf *et al.* [121] proposed a genetic algorithm for jobs with a given processing order to optimize total electricity cost with turn-on/off strategy. Fang *et al.* [55] considered energy cost saving for uniform-speed and speed-scalable machines and proposed several heuristics to obtain near-optimal solution. Che *et al.* [28] developed a mixed-integer linear programming (MILP) model and a greedy insertion heuristic to minimize total electricity cost. For parallel machines, Moon *et al.* [106] developed a hybrid inserted GA to minimize weighted sum of the makespan and time-dependent electricity cost. Ding *et al.* [49] presented a MILP model and a column generation based heuristic to minimize total electricity cost respecting a given makepsan. For flow shops, Luo *et al.* [98] developed a novel ant colony optimization based meta-heuristic to minimize both electricity cost and makespan. Zhang *et al.* [149] formulated a time-indexed MILP model to minimize total electricity cost and carbon emissions while ensuring the production throughput at the same time.

It can be found that the existing research taking into account TOU pricing is mainly concerned with classical shop scheduling, e.g., single machine [55], [121], parallel machine [49], [105], and flow shop [98], [149], and most of them mainly focus on single-objective optimization, i.e., minimizing the total energy cost.

2.3 Multi-objective combinatorial optimization

Facing the complicated processing environments, a modern production system has to be evaluated by several criteria, such as maximum machine utilization, minimum inventory cost, high customer service and energy saving. To help decision makers find a balance among the multiple targets, the corresponding problems should be modelled and solved by the multi-objective optimization models and methods. In the following subsections, we state the principles of multi-objective optimization and their solution approaches and evaluation measures.

2.3.1 Principles of multi-objective optimization

In general, a multi-objective optimization problem (MOP) can be stated as follows:

$$(MOP): \quad \min\{f_1(x), f_2(x), \ldots, f_n(x)\}, \text{s.t.} x \in \chi$$

where n is the number of the objectives to be optimized simultaneously. x denote the vector of decision variables and $f(x)$ is the objective vector. χ represents the *feasible solution space*.

Generally, due to the conflicting nature of the objectives, a MOP does not have a unique optimal solution that is better than all other solutions as a single objective optimization problem. Thus, in order to define the preferred solutions, the principles of *Pareto dominance* is widely endorsed in the multi-objective optimization area.

Definition 1 *(Pareto Dominance) A solution x is said to weakly dominate ($\succeq$) another solution x' if $f_i(x) \leq f_i(x')$, $i = \{1, 2, ..., n\}$. solution x dominate ($\succ$) solution x' if and only if $f_i(x) \leq f_i(x')$, $i = \{1, 2, ..., n\}$ with at least one inequality is strict; solution x strongly dominate ($\succ\succ$) solution x' if $f_i(x) < f_i(x')$, $i = \{1, 2, ..., n\}$.*

Definition 2 *(Weakly Pareto Optimal solution) x^* is a weakly Pareto optimal solution if and only if no $x \in \chi$ exists such that $f_i(x) < f_i(x^*)$ for $i \in \{1, 2, \ldots, n\}$.*

Definition 3 *(Pareto Optimal solution) x^* is a Pareto optimal solution (non-dominated solution or efficient solution) if and only if no $x \in \chi$ exists such that $f_i(x) \leq f_i(x^*)$ for $i \in \{1, 2, \ldots, n\}$ with at least one inequality being strict. And $f(x^*)$ in the objective space is called a Pareto optimal objective vector or a non-dominated point.*

Definition 4 *(Pareto optimal set) All Pareto optimal solutions in the feasible solution space constitute the Pareto optimal set.*

Definition 5 *(Pareto front) All non-dominated points in the objective space form the Pareto front, denoted by $\mathcal{F}$.*

Fig 2.2 provides an example of Pareto front for a bi-objective minimization problem. To define the range of the Pareto front, two special objective vectors are often used in multi-objective optimization methods, which are *ideal point* and *nadir point*. The ideal point representing the lower bound of each objective in the entire feasible solution space, can be expressed as follows for the n conflicting objectives:

Definition 6 *The vector $f^I = (f_1^I, f_2^I, ..., f_n^I)^T$ with $f_i^I = \min f_i(x), s.t. x \in \chi, i = \{1, 2, ..., n\}$ represents the ideal point.*

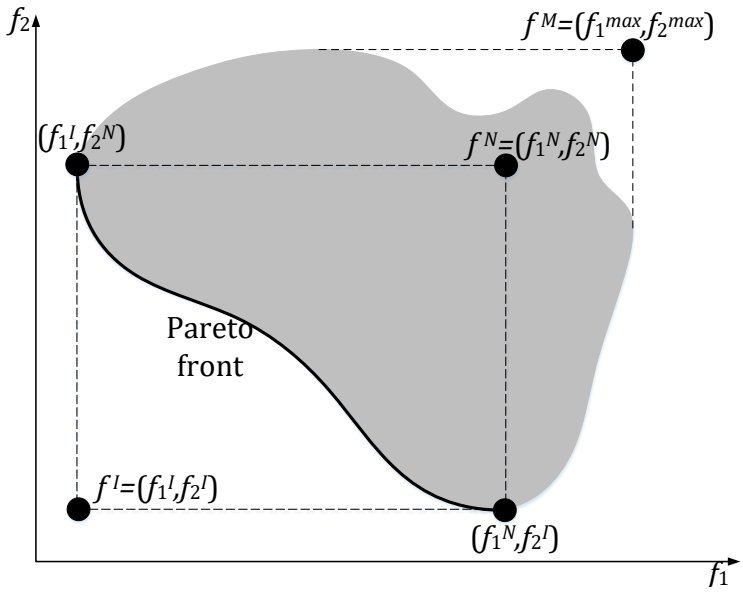


Fig. 2.2: An example of Pareto front, ideal and nadir points

According to the definition, the ideal objective vector consists of the individual optimal objective values. Since the objectives are conflicting in general, their corresponding minimum solutions are often not the same. Thus, the ideal point is a non-existent solution in the feasible solution space.

The nadir point f^N is constructed from the upper bound of each objective in the Pareto optimal set. f^N must not be confused with another point f^M (shown in Fig 2.2) formed by the worst function value of each objective. The nadir point may represent an existent or non-existent solution, which depends on the convexity and continuity of the Pareto optimal set [44]. Unlike the ideal point which can be easily obtained by individually optimizing each objective over the feasible solution space, the computation of a nadir point is much more complex [15]. However, for a bi-objective optimization problem (BOP), the ideal and nadir points can be calculated according to the following definition (see Fig 2.2) [18].

Definition 7 *The vector* (f_1^I, f_2^I) *with* $f_1^I = min\{f_1(x), x \in \chi\}$, *and* $f_2^I = min\{f_2(x), x \in \chi\}$, *denotes the ideal point; and the vector* (f_1^N, f_2^N) *with* $f_1^N = min\{f_1(x) : f_2(x) = f_2^I, x \in \chi\}$, *and* $f_2^N = min\{f_2(x) : f_1(x) = f_1^I, x \in \chi\}$, *denotes the nadir point.*

Definition 8 *(Extreme points) The Vector* (f_1^I, f_2^N) *and* (f_1^N, f_2^I) *are two extreme points on the Pareto front for a bi-objective optimization problem.*

In many cases, due to the high complexity of the underlying application, generating the Pareto optimal set can be very time-consuming or even infeasible. A good approximtion of the Pareto front is an alternative.

Definition 9 *(Approximation set) Let $\mathcal{A}$ be a set of objective vectors in the objective space. $\mathcal{A}$ is called an approximation set if no element of $\mathcal{A}$ is weakly dominated by any other objective vector in $\mathcal{A}$.*

Up to now, an optimization problem mainly can be solved by analytical method or numerical method. The analytical method that can reach the exact optimal solution requires strict mathematical proofs and deduction. It can only be effective for some special problems. The numerical method involves appropriate formulas and multiple iterations to calculate the optimal or near-optimal objective values. Over the past few decades, many numerical methods have been proposed to solve MOPs in literature. Based on the different techniques of handling the multiple objectives, these methods mainly can be grouped into two categories: intelligent optimization algorithms and scalarization methods. The former one employs appropriate heuristic search rules to achieve different objective optimization simultaneously; and the latter one generally transforms a MOP into a series of single objective optimization problems and solve them successively to obtain the exact or approximate Pareto front.

2.3.2 Intelligent optimization algorithms

The intelligent optimization algorithms, also known as meta-heurstics, are a class of search methods that inspired by animal or human behaviour patterns, reaction modes or communication mechanisms [35]. Most multi-objective intelligent optimization algorithms employ Pareto-based ranking schemes to classify the individuals in the evolution population and use a mechanism to allocate appropriate fitness to promote the individual dispersion in the population [71], [141]. Two most well-known multi-objective intelligent optimization algorithms are *non-dominated sorting genetic algorithm-II* (NSGA-II) [16], [43], [47] and *strength Pareto evolutionary algorithm 2* (SPEA-2) [158]. Other multi-objective intelligent optimization algorithms include multi-objective ant colony system algorithm [58] and multi-objective particle swarm optimization algorithm [34], etc.

The major advantage of the intelligent optimization algorithms for MOPs is their high efficiencies in generating approximative Pareto front, since multiple solutions can be simultaneously obtained in one iteration. However, the optimality of the solutions

are not guaranteed, and the performance of the solution quality often highly depends on the parameters setting and the initial population generation. In addition, these algorithms usually perform no better than random blind search when they are directly applied without considering problem characteristics [48].

2.3.3 Scalarization methods

Scalarization methods are usually applied to find efficient solutions for MOPs. In this subsection, three most widely and prevalent scalarization techniques: weighted sum method, goal programming approach and ϵ-constraint method are presented, respectively.

2.3.3.1 Weighted sum method

A MOP is often solved by combining the multiple objective functions into a single one. The most popular and straightforward one is *weighted sum method* [147]. More specifically, this approach minimizes a positively weighted convex sum of the objectives:

$$\mathcal{P}_\omega : \ y = \min \sum_{i=1}^{n} \omega_i f_i(x), s.t. x \in \chi$$

where $\omega_i \geq 0, i = \{1, 2, ..., n\}$ is a set of weighted coefficients, and $\sum_{i=1}^{n} \omega_i = 1$. Then an efficient solution of the original MOP can be obtained by optimally solving the single objective problem $\mathcal{P}_\omega$ [62]. By appropriately changing the weight parameters, a set of tangent points of the single objective function line and the Pareto curve can be obtained by minimizing the single objective problems. That is, a set of efficient solutions can be acquired.

Although the weighted-sum method is easy to implement, it has several technical shortcomings. Firstly, it is a hard task to determine appropriate weights of objectives, since setting a uniform spread of weight parameters generally cannot produce a uniform spread of solutions on the Pareto front [20]. Secondly, a part of non-dominated points cannot be obtained when the Pareto curve is non-convex [42]. Thirdly, the weighted-sum method is not appropriate to the case where the linear combination is not suitable for integrating different objectives into a single one.

2.3.3.2 Goal programming

Goal programming was first introduced by Charnes *et al.* [24], [25]. In this approach, decision makers firstly set the a desired value $v_i, \ i = \{1, 2, ..., n\}$ for each objective,

and then find a solution whose objective vector is as closer as possible to the desired value vector. This can be achieved by formulating a new single-objective problem, which is to minimize the deviation of the original objectives from their desired values, and subjects to original constraints as well as new constraints transformed from the original objectives. Take a three objective optimization problem for an example.

$$\min f_i(x), i = 1, 2, 3$$
$$s.t.\ x \in \chi$$

The desired values of the objectives are as follows.

$$f_1(x) \leq v_1$$
$$f_2(x) = v_2$$
$$f_3(x) \geq v_3$$

To find the solution that is closest to desired value vector $(v_1, v_2, v_3)^T$, slack variables $d_i^-, i = \{1, 2, 3\}$ (resp. surplus variables $d_i^+, i = \{1, 2, 3\}$) are introduced to calculate the negative (resp. positive) deviation from desired objective value $v_i, i = \{1, 2, 3\}$. For each constraint of the type 1

$$f_1(x) \leq v_1,$$

a positive deviation variable d_1^+ is introduced to form the following constraints:

$$f_1(x) - d_1^+ \leq v_1.$$

For each constraint of the type 2

$$f_2(x) = v_2,$$

both positive and negative deviation variables are introduced form the new constraints as follows:

$$f_2(x) - d_2^+ + d_2^- = v_2.$$

For the constraint of the type 3

$$f_3(x) \geq v_3,$$

a negative deviation variable d_3^- is introduced, and the above constraint becomes

$$f_3(x) + d_3^- \geq v_3.$$

There are several approaches to optimize the introduced variables, Archimedean goal programming is one of the most popular. It is also called as weighted goal programming, where the slack and surplus variables are assigned weights based on their relative importance to the decision maker and minimized as an Archimedian sum. Let ω_i^- and ω_i^+ be the weights of d_i^- and d_i^+, respectively. The problem to be optimized can be formulated as follows.

$$\begin{aligned} \min \quad & \omega_1^+ d_1^+ + \omega_2^+ d_2^+ + \omega_2^- d_2^- + \omega_3^- d_3^- \\ s.t. \quad & f_1(x) - d_1^+ \leq v_1 \\ & f_2(x) - d_2^+ + d_2^- = v_2 \\ & f_3(x) + d_3^- \geq v_3 \\ & d_1^+, d_2^+, d_2^-, d_3^- \geq 0 \\ & x \in \chi \end{aligned}$$

For more formulating methods, such as lexicographical goal programming, the reference goal programming and the interactive goal programming, please refer to [127].

For linear programming problem, goal programming approach has been proved to be effective to obtain satisfactory solution. However, it is difficult in setting appropriate desired objective values without preference informaton of decision makers [35].

2.3.3.3 ε-constraint method

The ε-constraint method is a well-known technique for solving multi-criteria optimization problems. It aims to optimize a single preferred objective function while formulating the other objectives as constraints, called ε-constraints. For the bi-objective case, the ε-constrained problem can be illustrated as follows if the first objective is considered as the preferred one:

$$\mathcal{P}(\varepsilon) : \min f_1(x), \text{s.t.} f_2(x) \leq \varepsilon, x \in \chi$$

where ε progressively varies from f_2^N to f_2^I. Based on the different ways of modifying ε, the ε-constraint methods can be classified into two subsets: *equidistant ε-constraint method* and *exact ε-constraint method* [142].

In the equidistant ε-constraint method, the range of ε is uniformly divided into a number of K subintervals and each subinterval's upper limit is taken as the value of ε [83], [117], [155]. Mathematically, we first calculate the iteration step of ε by the following formula:

$$\delta = (f_2^N - f_2^I)/K,$$

where K is given by the decision maker in advance. Then the value of ε_k for the k-th ε-constraint problem $\mathcal{P}(\varepsilon_k)$ is defined as follows:

$$\varepsilon_k = \varepsilon_{k-1} - \delta, k \in \{1, ..., K\}.$$

Especially, ε_0 is initialized as f_2^N. Thus, the K ε-constraint problems are determined. This method cannot guarantee to obtain the Pareto front but a set of non-dominated solutions can be efficiently obtained by successively solving the K problems. The performance of this method has been widely examined by many works [70],[128],[155].

In the exact ε-constraint method, the value of ε at iteration k' equals to the optimal value of $f_2(x)$ at iteration $k'-1$ minus parameter δ', i.e.,

$$\varepsilon_{k'} = f_2(x^{k'-1}) - \delta',$$

where $x^{k'}$ denotes the solution vector at iteration k', $f_2(x^0) = f_2^N$. The iteration of the method terminates if $\varepsilon_{k'} = f_2^I$. In the work of Bérubé *et al.* [18], δ' is set as 1 for the problems with integer objective values. Then Wu *et al.* [142] proposed setting δ' as the minimum unit value of $f_2(x)$ for the problems with integer or fractional objective values. By progressively reducing ε and solving a series of ε-constraint problems, then removing dominated solutions from the obtained solution set, exact ε-constraint method has been proved to be able to generate the Pareto front for BOPs for travelling-salesman problem [18], vehicle routing problem [117], lane reservation problem [142] and prize-collecting Steiner tree problem [83], etc.

2.3.4 Performance metrics

Unlike the single-objective optimization whose effectiveness can be directly evaluated by comparing the obtained solution with the lower or upper bounds, the solution quality evaluation of multi-objective optimization is less straightforward for the approximation sets. Generally, the quality of an approximation set $\mathcal{A}$ can be evaluated from three aspects [111]. The first is cardinality, which is the number of the obtained non-dominated solutions; the second is accuracy, which reflects the closeness of the obtained frontier to the Pareto front or a reference set (see Definition 10); the third is distribution and spread, which indicates the evenness and covering breadth of the solutions distribute along the obtained front.

Definition 10 *(Reference set) It is an artificial or desired reference solution set in the feasible solution space [111], denoted by $\mathcal{R}$. It is used to evaluate the obtained solution set for the cases where the Pareto optimal set is unknown.*

To evaluate the obtained frontier, diverse performance indices (PIs) have been proposed [111], [157], [161] and they can be divided into the following three categories: cardinality-based PIs, distance-based PIs, volume-based PIs. A PI from the three categories may reflect one or more aspects of the solution set quality. For example, e-dominance indicator, which is a distance-based PI, can indicate the accuracy and distribution of a solution set. In the following subsections, we detail some widely used PIs of the three categories.

2.3.4.1 Cardinality-based PIs

A basic index of cardinality is *overall non − dominated solutions* (Q) in appromimation set $\mathcal{A}$ [132], [133].

$$Q = |\mathcal{A}| \tag{2.1}$$

Q is closer to the number of Pareto optimal solutions $|\mathcal{F}|$, the better the solution set. Furthermore, *overall non-dominated solutions ratio* (R_Q) is proposed by Veldhuizen *et al*. [133] as follows.

$$R_Q = |\mathcal{A}|/|\mathcal{F}| \tag{2.2}$$

Based on the cardinality, *error ratio* [133], [134] and *ratio of the reference points found* [36], [65] were proposed to compare the obtained solution set to the Pareto optimal set or reference set. *Error ratio* indicates the number of the points that belong to set $\mathcal{A}$ but are not in set $\mathcal{P}$. Conversely, *ratio of the reference points found* implies the points that exist in both $\mathcal{A}$ and $\mathcal{R}$. To compare two solution sets obtained by different algorithms, *coverage of two sets* [156], [159], [160] is often involved. It reflects the dominance relationship between two solution sets.

More cardinality-based PIs details that include *generational non-dominated vector generation*, *generational non-dominated vector generation ratio* and *non-dominated vector addition*, etc., can be referred to the works of [111], [156], [161]. The PIs in this category are easy to be implemented. However, they provide very limited information about the distribution of the solutions.

2.3.4.2 Distance-based PIs

Here, distance-based PIs are shown, which include distribution and spread PIs and accuracy PIs. *Spacing* (SP) is a widely used PI for describing the distribution of an obtained solution set [119].

$$SP(\mathcal{A}) = \sqrt{\frac{1}{|\mathcal{A}-1|}\sum_{k=1}^{|\mathcal{A}|}(d_k - \bar{d})^2} \tag{2.3}$$

$$d_k = \min_{x^j \in \mathcal{A}, x^j \neq x^k} \sum_{i=1}^{n} |f_i(x^k) - f_i(x^j)| \tag{2.4}$$

where $\bar{d}$ means the average value of $d_k, k = 1, ..., |\mathcal{A}|$, and x^k is the k-th solution vector in set $\mathcal{A}$. $SP(\mathcal{A})$ aims to indicate distributing evenness of approximation set $\mathcal{A}$. However, the distance d_k is computed by the sum of absolute difference along each objective, and only the shortest distance is used for each objective vector without sorting.

Later, metric $\triangle'$ is proposed based on sorted solutions by Deb *et al.* [45]. Moreover, d_k is calculated by the Euclidean distance between consecutive solutions in $\mathcal{A}$.

$$\triangle'(\mathcal{A}) = \sum_{k=1}^{|\mathcal{A}|-1} \frac{|d_k - \bar{d}|}{|\mathcal{A}| - 1} \tag{2.5}$$

$$d_k = \sqrt{\sum_{i=1}^{n} [f_i(x^k) - f_i(x^{k+1})]^2} \tag{2.6}$$

Further taking spread quality into account, the $\triangle'$ has been extended to $\triangle$ PI as follows [43], [44].

$$\triangle(\mathcal{A}) = \frac{d_f + d_l + \sum_{k=1}^{|\mathcal{A}|-1} |d_k - \bar{d}|}{d_f + d_l + (|\mathcal{A}| - 1)\bar{d}} \tag{2.7}$$

where d_f and d_l represent the Euclidean distance between the extreme solutions of Pareto front and the boundary solutions of $\mathcal{A}$, shown as Fig.2.3.

Considering the quality of both distribution and accuracy, the e-dominance indicator [161] is proposed, which measures the average distance from $\mathcal{A}$ to $\mathcal{R}$. An illustration of e-dominance indicator in the bi-objective case is depicted in Fig.2.4. A point $(f_1(x), f_2(x)) \in \mathcal{A}$ is said to e-dominate a point $(f_1(x^*), f_2(x^*)) \in \mathcal{R}$ if $f_1(x) \leq e(x^*) \cdot f_1(x^*)$ and $f_2(x) \leq e(x^*) \cdot f_2(x^*)$. Then, the e-dominance indicator for a given point $(f_1(x^*), f_2(x^*)) \in \mathcal{R}$ can be calculated as:

$$e(x^*) = \min_{(f_1(x), f_2(x)) \in \mathcal{A}} \max \left\{ \frac{f_1(x)}{f_1(x^*)}, \frac{f_2(x)}{f_2(x^*)} \right\} \tag{2.8}$$

$e(x^*) < 1$ indicates that $(f_1(x^*), f_2(x^*))$ is dominated by $(f_1(x), f_2(x))$. $e(x^*) = 1$ reveals that at least one dimension of the two points are overlapping. With the above definition, each point in the reference set has a value of e-dominance indicator. To illustrate the comparison results of the entire frontiers, the average, maximum and

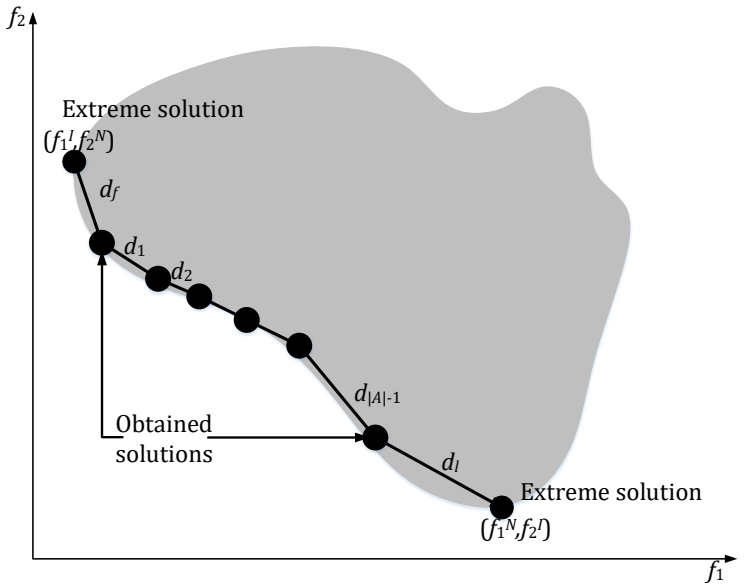


Fig. 2.3: Distribution and spacing metric $\triangle$ [43]

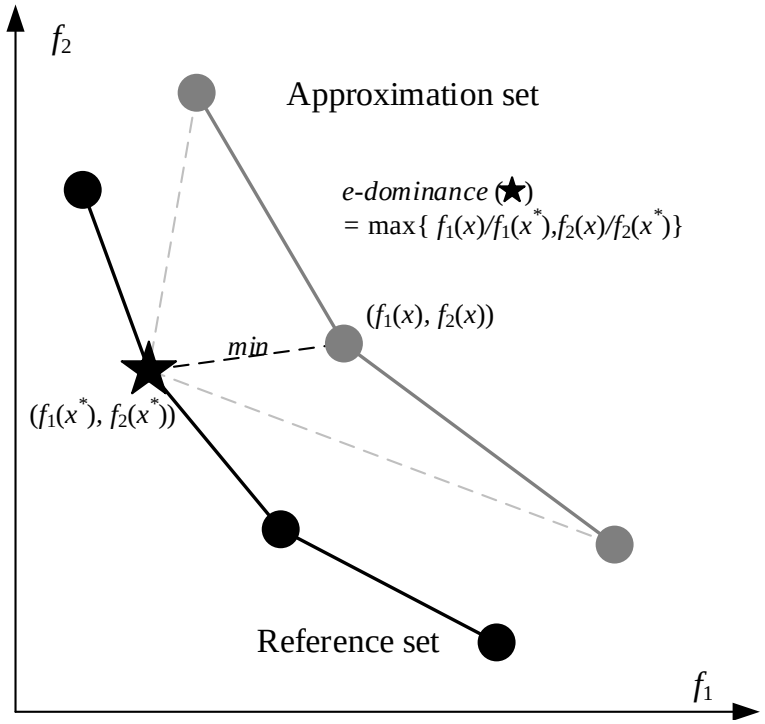


Fig. 2.4: *e*-dominance metric [46]

minimum e-dominance indicators (D_{av}, D_{max} and D_{min}) are selectively used. The closer the metrics getting to 1 suggests $\mathcal{A}$ being closer to $\mathcal{R}$.

$$D_{av} = \frac{1}{|\mathcal{R}|} \cdot \sum_{(f_1(x^*), f_2(x^*)) \in \mathcal{R}} e(x^*) \tag{2.9}$$

$$D_{max} = \max_{(f_1(x^*), f_2(x^*)) \in \mathcal{R}} e(x^*) \tag{2.10}$$

$$D_{min} = \min_{(f_1(x^*), f_2(x^*)) \in \mathcal{R}} e(x^*) \tag{2.11}$$

More distance-based PIs such as *generational distance, seven points average distance, maximum spread, overall Pareto spread,* can be referred to [36], [111], [125].

2.3.4.3 Volume-based PIs

The most famous volume-based PIs is *hypervolume*, which computes the area in the objective space that the solution set can dominate [156], [159]. Such area is formed by all points in the solution set and a reference point (generally the Nadir point). As depicted in Fig.2.5, the hypervolume is composed of all shaded rectangles. The larger the hypervolume value, the better the derived solution set.

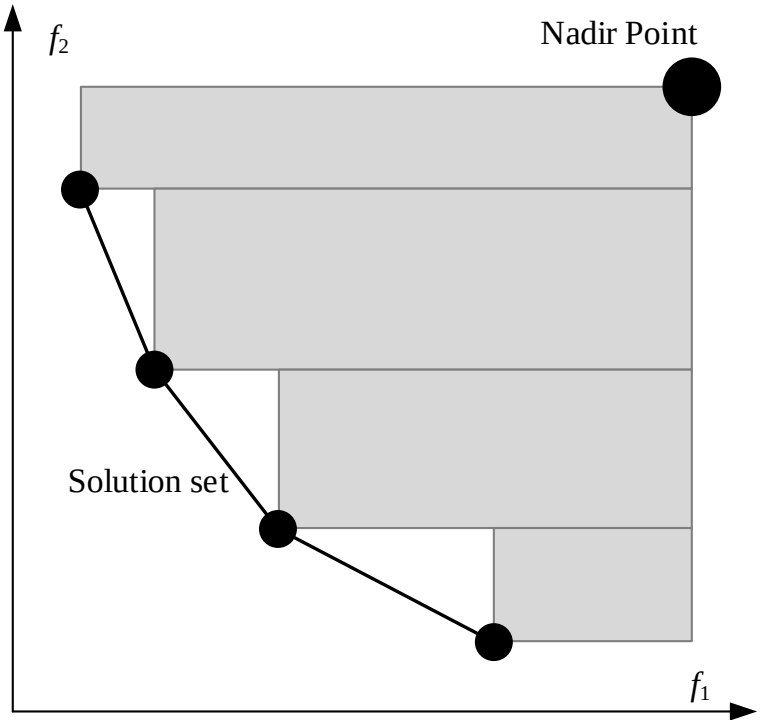


Fig. 2.5: Hypervolume indicator [46]

The hypervolume ratio, denoted by H, is the ratio of the hypervolumes of $\mathcal{A}$ and

$\mathcal{R}$, denoted by $HV_{\mathcal{A}}$ and $HV_{\mathcal{R}}$, respectively [140]. It is calculated as:

$$H = HV_{\mathcal{A}}/HV_{\mathcal{R}} \tag{2.12}$$

Obviously, $H > 1$ (resp. $H < 1$) indicates that $\mathcal{A}$ is better (resp. worse) than $\mathcal{R}$; $H = 1$ implies they have the same quality.

In summary, each PI has its advantages and disadvantages, no existing metric can account for all aspects of the quality of solution sets for MOPs. Therefore, multiple PIs are usually simultaneously employed to appropriately evaluate and compare the performance of multi-objective optimization algorithms. In this thesis, overall non-dominated solutions (Q), average e-dominance indicator (D_{av}) and hypervolume ratio (H) that represent the three different aspects of PIs will be used to evaluate the proposed methods.

2.4 Conclusions

In this chapter, we first introduced batch processing machine that is widely applied in modern manufacturing and many of them are energy intensive ones. By reviewing variety of batch scheduling problems and the literature on single and parallel batch machine environments, it can be found that considerable studies have been conducted on optimizing production efficiency objectives, such as makespan, total completion time or maximum tardiness. Batch scheduling problems concentrate on environmental impacts have not been studied. Then, we reviewed the literature on scheduling considering energy consumption saving and energy cost saving under time-of-use(TOU) tariffs. However, the existing scheduling works under TOU tariffs mainly concentrate on classical machine environment, such as single machine, parallel machines or flow shop, and most of them are single objective optimization oriented. It is economical and ecological to consider production efficiency optimization and electricity cost minimization together for batch processing machines. This is the motivation of this thesis. Finally, we presented some multi-objective optimization methods and their performance metrics. In this thesis, overall non-dominated solutions (Q), e-dominance indicator (D) and hypervolume ratio (H) that respectively reflects cardinality-, distance- and volume-based PIs are employed to evaluate the proposed bi-objective optimization methods.

Chapter 3

Single batch machine scheduling under TOU tariffs

3.1 Introduction

As reviewed in Chapter 2, majority of batch scheduling problems have been proved to be NP-hard, even for single batch machine environments. Under TOU tariffs, the complexity of the batch scheduling problems will further increase due to variable electricity cost on the scheduling horizon. Single machine is a basis of more complicated machine systems and has been widely encountered in real production environments, such as shoe manufacturing industry [56], semiconductor manufacturing industry [81], [135]. As a beginning work, this chapter focuses on single batch processing machine scheduling under TOU tariffs with bi-objective of total electricity cost and makespan minimization (called SBS-TOU in short after).

The considered problem SBS-TOU aims to group the jobs into batches, sequence the formed batches on the horizon, such that the two objectives are simultaneously minimized. For this new problem, we first provide a mixed-integer linear programming (MILP) model. Then the problem complexity and an simplified MILP model based on the analytic property are proposed. Subsequently, two fast effective heuristics based on exact ε-constraint method are developed for the problem, in which the transformed single objective problem in each iteration is considered as a series of knapsack problems and multiple knapsack problems to construct the two heuristics, respectively. Finally, numerical experiments are presented to evaluate the performance of the proposed methods.

The rest of the chapter is structured as follows. Section 3.2 gives the problem description and mathematical formulation. In Section 3.3, the problem property and complexity are analysed and then an simplified model is proposed. In Section 3.4,

two efficient heuristic based ε-constraint methods, i.e., KH-ECM and MKH-ECM, are developed for the problem. Computational results are presented in Section 3.5. Section 3.6 is devoted to the conclusion.

3.2 Problem formulation

A bi-objective single batch machine scheduling problem under TOU tariffs can be represented as $TOU, 1|B|E, C_{max}$ by using the triplet notation proposed by Graham [59]. The problem can be described as follows:

A given set $J = \{1, 2, \ldots, |J|\}$ jobs are to be processed on a single batch processing machine within a horizon $I = \{1, 2, \ldots, |I|\}$. The duration of period $i \in I$ is denoted as S_i. Job $j \in J$ is nonpreemptive and has a processing time p_j. Any p_j is less than the duration of any period i; i.e., $S_i \gg p_j, \forall j \in J, \forall i \in I$. Without loss of generality, we assume that the jobs are numbered in nonincreasing order of the processing times; i.e.,

$$p_1 \geq p_2 \geq \cdots \geq p_{|J|}.$$

The jobs can be regrouped to $|B|$ (to be optimized) batches and each batch can contain at most C jobs. Therefore, we must have $\lceil |J|/C \rceil \leq |B| \leq |J|$ [67]. The processing time of a batch is determined by the longest processing time of the jobs in the batch.

The processing of a batch should be completed before the end of a period or it must wait the beginning of another period. The periods can be defined by electricity prices or work shifts. If the periods are defined by electricity prices, that means a batch may need to wait a new electricity price period to be processed. This is not appropriate and appreciated in manufacturing industry. While a job has to be completed in a single work shift exists in some production environments, for example, French Atomic Energy Industry, CEA. Therefore, a work shift is regarded as a period in this thesis and its average unit electricity cost is calculated according to electricity prices and work shifts. One work day is often composed of two or three work shifts according to the types of products, so-called two-shift and three-shift, respectively. The average unit electricity cost $e_i, \forall i \in I$ for each work shift can be calculated according to the tariffs information. Take the TOU tariffs of Ontario (see Fig 2.1) for an example, for a three-shift in a work day: 8h-16h, 16h-0h, 0h-8h, the corresponding unit electricity costs are as follows, $e_{8h-16h} = (11.4 * 3 + 14.0 * 5)/8$ ¢/kWh $*$ 1kWh/h $= 13.0250$ ¢/h, $e_{16h-0h} = 9.4125$ ¢/h, and $e_{0h-8h} = 8.1625$ ¢/h.

Before formulating the problem, the parameters and decision variables are summarized as follows.

Indices:

j: index of jobs; $j \in J = \{1, 2, \ldots, |J|\}$

b: index of batches; $b \in B = \{1, 2, \ldots, |B|\}$

i: index of periods; $i \in I = \{1, 2, \ldots, |I|\}$

Parameters:

J: set of all jobs, i.e., $J = \{1, 2, ..., |J|\}$;

B: set of batches, i.e., $B = \{1, 2, ..., |B|\}$;

I: set of time periods on the planning horizon, i.e., $I = \{1, 2, ..., |I|\}$;

C: capacity of a batch;

p_j: processing time of job j, $\forall j \in J$;

s_i: starting time of period i, $\forall i \in I$;

S_i: duration of period i, $\forall i \in I$, in which $S_i = s_{i+1} - s_i$;

e_i: unit electricity cost of period i, $\forall i \in I$;

Decision Variables:

$|B|$: number of the batches;

$x_{j,b,i}$: $x_{j,b,i} = 1$, if job j is assigned to batch b and processed in period i; otherwise $x_{j,b,i} = 0$; $\forall j \in J$, $\forall b \in B$, $\forall i \in I$;

$y_{b,i}$: $y_{b,i} = 1$, if batch b is assigned to period i; otherwise $y_{b,i} = 0$; $\forall b \in B$, $\forall i \in I$;

z_i: $z_i = 1$, if at least one batch is assigned to period i; otherwise $z_i = 0$; $\forall i \in I$;

$P_{b,i}$: $P_{b,i} = P_b = \max\{p_j \mid j \in b\}$, if batch b is processed in period i, where P_b is the processing time of batch b; and otherwise $P_{b,i} = 0$; $\forall b \in B$, $\forall i \in I$;

E: total electricity cost for completing all jobs;

C_{max}: completion time of the last job.

In the work, the number of batches $|B|$ is initially set as $|J|$. A batch is opened if there is at least one job allocated to the batch. On the contrary, a batch is closed without any job and its corresponding processing time equals to 0, i.e., $P_b = 0$. The considered problem can be formulated as the following bi-objective MILP model $\mathcal{P}'_s$.

$$\mathcal{P}'_s: \quad \min E \tag{3.1}$$

$$\min \ C_{max} \tag{3.2}$$

$$s.t. \quad \sum_{i\in I}\sum_{b\in B} x_{j,b,i} = 1, \forall j \in J \tag{3.3}$$

$$\sum_{i\in I} y_{b,i} = 1, \forall b \in B \tag{3.4}$$

$$\sum_{j\in J} x_{j,b,i} \leq C y_{b,i}, \forall b \in B, \forall i \in I \tag{3.5}$$

$$x_{j,b,i} p_j \leq P_{b,i}, \forall j \in J, \forall b \in B, \forall i \in I \tag{3.6}$$

$$\sum_{b\in B} P_{b,i} \leq S_i z_i, \forall i \in I \tag{3.7}$$

$$\sum_{i\in I} e_i \sum_{b\in B} P_{bi} \leq E \tag{3.8}$$

$$s_i z_i + \sum_{b\in B} P_{b,i} \leq C_{max}, \forall i \in I \tag{3.9}$$

$$x_{j,b,i}, y_{b,i}, z_i \in \{0,1\}, \forall j \in J, \forall b \in B, \forall i \in I \tag{3.10}$$

$$P_{b,i} \geq 0, E \geq 0, C_{max} \geq 0, \forall b \in B, \forall i \in I \tag{3.11}$$

Objective (3.1) is to minimize the total electricity cost E on the horizon I. Objective (3.2) is to minimize the makespan C_{max}, which is the completion time of the last batch. Constraint (3.3) ensures that job j, $\forall j \in J$, is assigned to only one batch. Constraint (3.4) guarantees that each batch $b, \forall b \in B$, is processed in only one period. Constraint (3.5) assumes that the number of jobs assigned to any batch should not exceed the batch capacity C, and any job $j, \forall j \in J$, cannot be assigned to period i if its corresponding batch is not processed in this period. Constraint (3.6) determines the batch processing time. Constraint (3.7) ensures that the total processing time of batches in period $i, \forall i \in I$, should not exceed its duration, and $z_i = 1$ if there is at least one batch assigned to period i. Constraint (3.8) calculates the total electricity cost. Constraint (3.9) defines the makespan C_{max}. Constraint (3.10) -(3.11) enforce the restrictions on decision variables.

3.3 Property analysis and problem complexity

As mentioned above, the number of batch $|B|$ equals to the number of jobs $|J|$ in model $\mathcal{P}'_s$. According to the preliminary results, it is very time consuming to directly solve model $\mathcal{P}'_s$. This section is devoted to reducing the search space via problem property analysis. We show that the formation of batches can be solved independent of the scheduling of batches, with two objectives we consider in this paper.

A solution of the problem is uniquely defined by $(|B|, \{J_b, 1 \leq b \leq |B|\}, \{\tau_b, 1 \leq b \leq |B|\})$, where $|B|$ is the number of batches, J_b and τ_b are the set of jobs involved in the batch b and the period the batch is processed in, respectively.

We consider in particular those solutions where the batches are formed with a so-called LPT-based method. In this method, any job j with $(b-1)C < j \leq bC$ and $1 \leq b \leq \lceil |J|/C \rceil - 1$ is put into batch b, and the remaining jobs, to batch $\lceil |J|/C \rceil$, where $\lceil x \rceil$ denotes the smallest integer greater than or equal to x. Thus, the processing time of batch b equals to that of job $(b-1)C+1$. Figure 3.1 gives a simple example to illustrate the method.

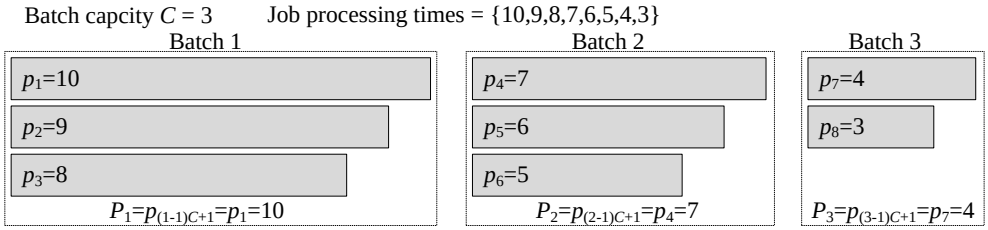


Fig. 3.1: An example of batches formed with the LPT-based method

The following theorem shows that we only need to consider such solutions in order to find the Pareto front.

Theorem 1 *Any solution in which the batches are different from those formed with the LPT-based method is (at least weakly) dominated.*

Proof: To facilitate the proof, the following notations are used.

J_b: the set of jobs contained in batch b, $J_b \subseteq J$;
$n(J_b)$: the serial number of the least indexed job (thus with the largest processing time) in set J_b, i.e., $n(J_b) = \min\{j | j \in J_b\}$;
$P(J_b)$: the processing time of batch b (the processing time of the least indexed job in J_b), i.e., $P(J_b) = \max_{j \in J_b} p_j = p_{n(J_b)}$.

Let $|B^*|$ and J_b^* represent the number of batches formed with LPT-based method and the set of jobs involved in batch b $(1 \leq b \leq |B^*|)$, respectively. We have

$$|B^*| \quad = \quad \lceil |J|/C \rceil,$$

$$
\begin{aligned}
J_b^* &= \{(b-1)C+1, (b-1)C+2, \ldots, bC\}, \quad b = 1, 2, \ldots, |B^*| - 1, \\
J_{|B^*|}^* &= \{(|B^*|-1)C+1, (|B^*|-1)C+2, \ldots, |J|\}.
\end{aligned}
$$

With the above construction, we have the following equations for batch b, $1 \leq b \leq |B^*|$:

$$
\begin{aligned}
n(J_b^*) &= (b-1)C+1, && (3.12) \\
P(J_b^*) &= p_{(b-1)C+1}. && (3.13)
\end{aligned}
$$

Consider a feasible solution $\hat{S}$ with $(|\hat{B}|, \{\hat{J}_b, 1 \leq b \leq |\hat{B}|\}, \{\hat{\tau}_b, 1 \leq b \leq |\hat{B}|\})$ in which the batches are different from those formed with the LPT-based method. Obviously, we must have

$$
|\hat{B}| \geq \lceil |J|/C \rceil = |B^*| \tag{3.14}
$$

In other words, there are at least as many batches as those formed with the LPT-based method. Without loss of generality, we renumber the batches in an increasing order of $n(\hat{J}_b)$'s; i.e.,

$$
n(\hat{J}_1) < n(\hat{J}_2) < \cdots < n(\hat{J}_{|\hat{B}|}) \tag{3.15}
$$

Then for any batch b such that $1 \leq b \leq |\hat{B}|$, there exists

$$
n(\hat{J}_b) = n(\widetilde{J}), \text{where } \widetilde{J} = \hat{J}_b \cup \hat{J}_{b+1} \cup \cdots \cup \hat{J}_{|\hat{B}|}. \tag{3.16}
$$

Owing to the fact that for any subset $\widetilde{J}$ belonging to set $J = \{1, 2, ..., |J|\}$, we have

$$
n(\widetilde{J}) \leq |J| - |\widetilde{J}| + 1, \quad \forall \widetilde{J} \subseteq J, \tag{3.17}
$$

where $|J|$ denotes the number of jobs contained in set J. As for schedule $\hat{S}$, in which the job set has $|\hat{B}|$ disjoint subsets that are sorted as (3.15). According to (3.16) and (3.17), we must have:

$$
n(\hat{J}_b) = n(\widetilde{J}) \leq |J| - \sum_{\beta=b}^{|\hat{B}|} |\hat{J}_\beta| + 1, \quad \widetilde{J} \subseteq J. \tag{3.18}
$$

Due to $J/\widetilde{J} = J/\{\hat{J}_b \cup \hat{J}_{b+1} \cup \cdots \cup \hat{J}_{|\hat{B}|}\} = \hat{J}_1 \cup \hat{J}_2 \cup \cdots \cup \hat{J}_{b-1}$, Equation (3.18) can further written as:

$$
n(\hat{J}_b) \leq \sum_{\beta=1}^{b-1} |\hat{J}_\beta| + 1 \leq \sum_{\beta=1}^{b-1} C + 1 = (b-1)C + 1. \tag{3.19}
$$

Since the jobs are indexed in nonincreasing order of their processing times, thus Equation (3.19) implies that

$$
P(\hat{J}_b) = p_{n(\hat{J}_b)} \geq p_{(b-1)C+1} = P(J_b^*), 1 \leq b \leq |B^*|. \tag{3.20}
$$

In other words, the processing time of the batches are at least as long as those formed with the LPT-based method.

Construct a new solution by removing batches $|B^*|+1, \ldots, |\hat{B}|$, if any, and replacing each batch $\hat{J}_b$ $(1 \leq b \leq |B^*|)$ by the corresponding one formed with the LPT-based method (i.e., batch J_b^*), without changing starting time. In other words, consider solution $(|B^*|, \{J_b^*, 1 \leq b \leq |B^*|\}, \{\hat{\tau}_b, 1 \leq b \leq |B^*|\})$. Relation (3.20) implies that this new solution is also feasible. Furthermore, due to the fact that some batches are removed and the processing times of the remaining batches are reduced, neither the electrical consumption cost nor the makespan is increased, which means that the initial solution is (at least weakly) dominated by the new one. □

As a consequence, by considering batches formed with the LPT-method as new jobs, the problem is transformed into a classical production scheduling problem without batching machine. Due to the fact that each batch (new job) should be entirely executed in one period, these new jobs are nonpreemptive. There is an (infinitely short) unavailability period between two successive periods. This latter problem has been proved to be NP-hard in the strong sense, even when the single objective is to minimize the makespan. Hence, we have the following theorem.

Theorem 2 *The TOU,* $1|B|E, C_{max}$ *is strongly NP-hard.*

3.3.1 An equivalent simplified model

According to Theorem 1, we can focus on scheduling problems of the batches. Therefore, the decision variables can be restricted to $y_{b,i}$' and z_i's. And $P_b = p_{(b-1)C+1}$ with the LPT-based method. The initial model can be simplified into following model $\mathcal{P}_s$.

$$
\begin{aligned}
\mathcal{P}_s: \quad & \min E \\
& \min C_{max} \\
s.t. \quad & \sum_{i \in I} y_{b,i} = 1, \forall b \in B^* && (3.21)\\
& \sum_{b \in B^*} P_b y_{b,i} \leq S_i z_i, \forall i \in I && (3.22)\\
& \sum_{i \in I} \sum_{b \in B^*} e_i P_b y_{b,i} \leq E && (3.23)\\
& s_i z_i + \sum_{b \in B^*} P_b y_{b,i} \leq C_{max}, \forall i \in I && (3.24)\\
& y_{b,i}, z_i \in \{0,1\}, \forall b \in B, \forall i \in I, E \geq 0, C_{max} \geq 0 && (3.25)
\end{aligned}
$$

where $B^* = \{1, 2, \ldots, |B^*|\}$ denotes the set of batches formed with the LPT-based method. Constraint (3.21) ensures that a formed batch $b, \forall b \in B^*$ is allocated into

exactly one period. Constraint (3.22) guarantees that the total processing time of the batches in period i does not exceed its duration and $z_i = 1$ if there is at least one batch allocated to period i, $\forall i \in I$. Constraints (3.23) and (3.24) restrict the total electricity cost and makespan, respectively. Constraint (3.25) specifies the restrictions on the variables.

Remark 1 *Compared with the initial model $\mathcal{P}'_s$, the improved model $\mathcal{P}_s$ reduces $|I| \cdot |J|^2 + (|J| - \lceil |J|/C \rceil) \cdot |I|$ binary variables and $|I| \cdot |J|$ real variables as well as $(2 + |I| \cdot |J| + |I|) \cdot |J| - (|J| \cdot |I| + |I| + 1) \cdot \lceil |J|/C \rceil$ constraints.*

Taking an instance with $|J| = 100, C = 10$ and $|I| = 10$ as example, $\mathcal{P}_s$ can reduce 100900 binary variables, 1000 real variables and 91090 constraints compared with $\mathcal{P}'_s$. Owing to such reduction of variables and constrains, the search space for Pareto optimal solutions via the improved model $\mathcal{P}_s$ is significantly reduced.

3.4 Solution approaches

Since the two objectives E and C_{max} of the SBS-TOU are conflicting in general, there may not exist a single optimum that simultaneously optimizes both objectives but a set of Pareto optimal solutions. As mentioned in Chapter 2, ε-constraint method is one of the most prevalent methods and achieves good performance for solving bi-objective problems [18], [102], [155]. Thus, ε-constraint method is selected to solve the SBS-TOU in this chapter. In what follows, two efficient heuristics based on the framework of ε-constraint method are developed for the studied problem.

3.4.1 The ε-constraint method based framework

The core idea of ε-constraint method is to transform a bi-objective problem into a series of single objective problems and limiting the another by some given values ε's, known as ε-constraints. Since this thesis focuses on energy-saving, total electricity cost E is selected as the preferred objective. Then, the ε-constraint problem of SBS-TOU can be formulated as follows.

$$\begin{aligned} \mathcal{P}_s(\varepsilon^k) : &\min E \\ &\text{s.t. Constraints (3.21) - (3.25), and } C_{max} \leq \varepsilon^k \end{aligned}$$

where ε^k is a parameter bounded by Ideal point (E^I, C^I_{max}) and Nadir point (E^N, C^N_{max}), which are obtained by optimally solving the following four single objective problems.

$$\mathcal{P}^1_s : E^I = \min E \text{ s.t. Constraints (3.21) - (3.25)}$$

$$\mathcal{P}_s^2 : C_{max}^I = \min C_{max} \text{ s.t. Constraints (3.21) - (3.25)}$$
$$\mathcal{P}_s^3 : E^N = \min E \text{ s.t. Constraints (3.21) - (3.25), and } C_{max} = C_{max}^I$$
$$\mathcal{P}_s^4 : C_{max}^N = \min C_{max} \text{ s.t. Constraints (3.21) - (3.21), and } E = E^I$$

where the constraint $C_{max} = C_{max}^I$ in $\mathcal{P}_s^3$ (resp. $E = E^I$ in $\mathcal{P}_s^4$) ensures that the nadir value of total electricity cost (resp. makespan) is obtained with restricting the makespan (resp. total electricity cost) as its ideal value. Here, the iteration way of the exact ε-constraint method is implemented. That is, ε is initialized as C_{max}^N in the first iteration. Then the value of ε in k-th ($k > 1$) iteration ε^k is updated as $C_{max}^{k-1} - \triangle$, where C_{max}^{k-1} is obtained from $(k-1)$-th iteration, and $\triangle$ is a constant that is set as 1 in this work [142].

Due to the NP-hardness of the problem, the computational time will increase exponentially if we exactly solve $\mathcal{P}_s(\varepsilon^k)$ and $\mathcal{P}_s^1$ to $\mathcal{P}_s^4$. According to our preliminary results, the commercial solver CPLEX cannot generate even a feasible solution for any above single objective problem within 3600s for instances with 80 batches. Meanwhile, decision makers may desire to rapidly obtain a set of near-optimal solutions that approximates the Pareto front $\mathcal{F}$ [136], [137], [146]. Let $\mathcal{A}$ denote the approximative Pareto solution set. Thus, this work resorts to developing heuristics to efficiently and effectively solve each single objective problem $\mathcal{P}_s(\varepsilon^k)$. The complete framework of the bi-objective method for the SBS-TOU can be summarized as Fig.3.2. In the framework, KH is the first proposed heuristic for the single objective problems. Replacing KH by MKH, we can obtain the framework of solving SBS-TOU with the second proposed heuristic. In what follows, we present the details of the two heuristics.

3.4.2 Design of KH

As mentioned in Section 3.2, the scheduling for the SBS-TOU needs to assign each batch to one period without exceeding the period duration. That is, we can regard the periods as a series of independent knapsacks and the batches as items. The batch scheduling is to allocate all items to available knapsacks where ε^k is set as a scheduling horizon for each $\mathcal{P}_s(\varepsilon^k)$. Its number of available knapsacks n^k can be determined by $n^k = i$, where i satisfies $s_i \leq \varepsilon^k \leq s_{i+1}$, and the capacity of n^k-th knapsack equals to $\varepsilon^k - s_i$. The capacity of knapsack $i, 1 \leq i \leq n^k - 1$, is the duration of period i. A knapsack problem can be formulated as:

$$\mathcal{P}_{KP} : \max \sum_{b \in B} P_b y_b$$

The ε-constraint method based framework for the SBS-TOU

1: set $\mathcal{A}' = \emptyset,\ k = 1$ and $C_{max} = +\infty$;
2: solve $\mathcal{P}_s^1, \mathcal{P}_s^2$ by KH and KH$'$, respectively, to obtain their near optimal values $E^{I\prime}$ and $C_{max}^{I\prime}$; replace E^I and the decision variables of $\mathcal{P}_s^4$ by $E^{I\prime}$ and the corresponding solution to obtain $C_{max}^{N\prime}$;
3: set $\mathcal{A}' = \mathcal{A}' \cup (E^{I\prime}, C_{max}^{N\prime})$, and $\varepsilon^k = C_{max}^{N\prime} - \triangle$ ($\triangle = 1$);
4: **while** $\varepsilon^k \geq C_{max}^{I\prime}$ **do**
5: solve $\mathcal{P}_s(\varepsilon^k)$ by KH to obtain E^k, calculate C_{max}^k with the solution of $\mathcal{P}_s(\varepsilon^k)$, and set $\mathcal{A}' = \mathcal{A}' \cup (E^k, C_{max}^k)$;
6: reset $k = k + 1$, $\varepsilon^k = C_{max}^{k-1} - \triangle$;
7: **end while**
8: remove dominated points from the $\mathcal{A}'$ if any and obtain the approximation Pareto front $\mathcal{A}$.

Fig. 3.2: The ε-constraint method based framework for the SBS-TOU

$$s.t.\ P_b y_b \leq S, \forall b \in B$$
$$y_b \in \{0, 1\}, \forall b \in B.$$

where $y_b = 1$ if item b is selected to fill in the knapsack, otherwise $y_b = 0$; and S denotes the capacity of the knapsack; the weight and value of each item both equal to the batch processing time P_b.

For solving problem $\mathcal{P}_s(\varepsilon^k)$, an intuitive idea is to fill in the cheaper knapsacks as full as possible. This inspires us to develop KH as Fig.3.3.

Note that $\mathcal{P}_s^1$ can be regarded as a special ε-constraint problem with $\varepsilon = s_{|I|+1}$. That is, $\mathcal{P}_s^1$ can be solved by directly implementing KH. $\mathcal{P}_s^2$ is solved by KH$'$.

3.4.3 Design of MKH

This heuristic is also developed based on regarding the periods as knapsacks and batches as items. Different from the previous heuristic KH, this heuristic aims to fill the most expensive available knapsack at first with as less total processing time items as possible for a $\mathcal{P}_s(\varepsilon^k)$. It can be achieved by solving a series of multiple knapsack problems, denoted by MKH. A general multiple knapsack problem can be stated as below:

$$\mathcal{P}_{MKP} : \max \sum_{i=1}^{\lambda} \sum_{b \in B} P_b y_{b,i}$$

KH: solution method for a given ε-constraint problem $\mathcal{P}_s(\varepsilon^k)$

1: consider batch tasks to be scheduled $b \in B$ as items, and a horizon ε^k with the corresponding periods $i = 1, ..., n^k$ as available knapsacks. Set the total energy consumption cost as $E^k = 0$;
2: order the n^k knapsacks (periods) with its non-decreasing unit electricity cost e_i in a list L, denote sequence number of period i in list L as l_i, and then we have $l_i = \{1, ..., n^k\}$;
3: select the knapsack with $l_i = 1$;
4: if $B = \emptyset$, go to Step 7; otherwise, fill in the available knapsack l_i by items in B, which corresponds to exactly solving an above $\mathcal{P}_{KP}$ problem; obtain the batches with $y_b = 1$, set $y_{b,i} = 1$;
5: calculate the energy consumption cost of knapsack l_i in list L (period i) as E_i; update $E^k = E^k + E_i$;
6: remove scheduled tasks from B, update $l_i = l_i + 1$, go to Step 4;
7: calculate the corresponding makespan C^k_{max} with $y_{b,i}$.

Fig. 3.3: KH: solution method for a given ε-constraint problem $\mathcal{P}_s(\varepsilon^k)$

KH′: solution method for $\mathcal{P}^2_s$

1: consider the batches $b \in B$ as items, and the entire horizon with $|I|$ periods as available knapsacks. Keep the $|I|$ knapsacks in increasing order of their starting times as list L, and denote the sequence number of period i as l_i.
2: implement Steps 3, 4, 6 and 7 of KH.

Fig. 3.4: KH′: solution method for $\mathcal{P}^2_s$

MKH: solution method for a given ε-constraint problem $\mathcal{P}_s(\varepsilon^k)$

1: implement Steps 1 and 2 of KH;
2: determine the number of candidate periods λ with First-fit rule: sequentially assign batch $b \in B$ to the first period in list L where it is fit until all the batches are assigned, then record the number of enabled periods as λ;
3: set $l_i = \lambda$, $\lambda = \lambda - 1$;
4: if $\lambda = 0$, go to Step 7; otherwise, select the first λ periods in list L as multiple knapsacks, and fill in them by items in B, which is equivalent to exactly solving an above $\mathcal{P}_{MKP}$ problem; obtain the batches with $\sum_{l=1}^{\lambda} y_{b,l} = 0$, and set $y_{b,i} = 1$, where i is the l_i-th period in list L;
5: calculate electricity cost of period i as E_i with $y_{b,i}$, set $E^k = E^k + E_i$;
6: remove the scheduled batches (with $y_{b,i} = 1$) from B, go to Step 3;
7: calculate the corresponding makespan C^k_{max} with $y_{b,i}$.

Fig. 3.5: MKH: solution method for a given ε-constraint problem $\mathcal{P}_s(\varepsilon^k)$

MKH′: solution method for $\mathcal{P}_s^2$

1: implement Step 1 of KH′;
2: implement Steps 2, 3, 4, 6 and 7 of MKH.

Fig. 3.6: MKH′: solution method for $\mathcal{P}_s^2$

$$
\begin{aligned}
s.t. & \sum_{b \in B} P_b y_{b,i} \leq S_i, 1 \leq i \leq \lambda \\
& \sum_{i=1}^{\lambda} y_{b,i} \leq 1, \forall b \in B \\
& y_{b,i} \in \{0,1\}, \forall b \in B, 1 \leq i \leq \lambda
\end{aligned}
$$

where i is the index of knapsacks; $y_{b,i} = 1$ if item b is selected to fill in knapsack i, otherwise $y_{b,i} = 0$; and λ denotes the number of knapsacks. MKH is detailed in Fig.3.5.

Similarly, $\mathcal{P}_s^1$ is directly solved with MKH via setting ε as $s_{|I|+1}$. Then, $\mathcal{P}_s^2$ is solved with MKH′.

3.4.4 Select the most preferable Pareto optimal solution

Among all the obtained nondominated points, a decision maker may desire to select a preferable one. In this subsection, the *fuzzy-logic-based* approach [52] is employed

to recommend a preferable solution, since it can not only take into account the preferences of the decision maker, but also indicate the optimality degree each obtained nondominated point for each objective.

With the fuzzy-logic-based approach, the *membership function* $\delta_i(f_i^s)$, which represents the optimality degree of the s-th Pareto optimal solution for the i-th objective function, is presented as follows [52]:

$$\delta_i(f_i^s) = \begin{cases} 1, & \text{if } f_i^s \leq f_i^I \\ \frac{f_i^N - f_i^s}{f_i^N - f_i^I}, & \text{if } f_i^I < f_i^s < f_i^N,\ 1 \leq i \leq n, 1 \leq s \leq S \\ 0, & \text{if } f_i^s \geq f_i^N \end{cases} \tag{3.26}$$

where f_i^I and f_i^N represent the lower and upper limits of the ith objective function, respectively, and f_i^s expresses the value of the i-th objective of the s-th Pareto solution. S denotes the total number of Pareto solutions.

On the basis of the membership functions $\delta_i(f_i^s)$, the *membership degree* δ^s of the s-th solution can be calculated as follows [52]:

$$\delta^s = \frac{\sum_{i=1}^{n} \omega_i \delta_i(f_i^s)}{\sum_{i=1}^{n} \omega_i} \tag{3.27}$$

where ω_i denotes the weight of objective i. It can be determined according to the preferences about the objectives of the decision maker. The most preferable solution is the one giving the maximum value of δ^s.

3.5 Computational results

In this section, computational results of 510 instances (102 sets $\times$ 5 instances) are reported and analysed to show the performance of the proposed models and algorithms. The algorithms were coded in C++ and the experiments were implemented on a PC with 1.7 GHz Intel i5-3317U CPU and 3.12 GB RAM under windows 7 operating system. The bi-objective models are solved with ε-constraint framework using commercial optimization software CPLEX 12.4, which is achieved with the algorithm in Fig. 3.2 and replacing KH by CPLEX. For each instance, the total completion time and the computational time for each single-objective problem are limited to 18000s and 3600s, respectively.

The instances are randomly generated as follows. Two generation schemes for job processing times are considered, namely $p_j \in (50, 100]$ and $p_j \in (100, 200], \forall j \in J$, to evaluate the performance of the proposed method in different scenarios. Batch

capacity C is set as 10 [80]. The duration of each period $S_i, \forall i \in I$, is taken as 480, which corresponds to one period of three work shifts in each day, i.e., 8*60 minutes. The number of periods $|I|$ is set as$|I| = \alpha \max_{j \in J}\{p_j \times |B^*|/S_i\}$, where α representing the production urgency extent is randomly generated from the interval [0.6, 1.0] in the default case to avoid overmuch idle periods in scheduling horizon and to make it closer to reality. Since the the gap of the electricity prices in on-peak and off-peak period can be 5 to 7 times (such as TOU tariffs in Italy and Beijing [49]), combined with the machine power rate, we set the electricity cost of the three work shifts as 30, 15, 5, respectively. Considering the both objectives are very important in industrial production, the weights in fuzzy-logic-based method are set to be equal, i.e., $\omega_1 = \omega_2 = 0.5$, to select the most preferable solution.

The computational results are presented in Tables 3.1-3.8. We successively evaluate the performance of the models and the developed heuristics, then the sensitivity analysis for input parameters is reported. The solution quality of the near optimal solution set is evaluated by three performance metrics that has been reviewed in Chapter 2: number of solutions Q, hypervolume ratio H and the average e-dominance indicator D. The solution time of each method is denoted by CT.

3.5.1 Comparison results of the models

We first compare the performance of models P_s and P'_s with small sized instances in Table 3.1, where the number of jobs varies from 20 to 40 and number of intervals varies from 3 to 5. The computational results show that the solutions obtained by the two models are the same, i.e., both models have found the Pareto front with the exact ε-constraint method. In terms of computational time, we can see that CT of $\mathcal{P}'_s$ varies from 0.369s to 892.205s, while CT of $\mathcal{P}_s$ varies from 0.25 to 2.17s. Averagely, the computational time spent by $\mathcal{P}_s$ is only 17% of that by $\mathcal{P}'_s$, which indicates that model $\mathcal{P}_s$ is much more efficient than $\mathcal{P}'_s$. This is because the derived properties significantly reduce the search space for Pareto optimal solutions. In summary, model $\mathcal{P}_s$ can obtain the Pareto front and its solution efficiency dramatically outperforms $\mathcal{P}'_s$. Thus, we only use the computational results of $\mathcal{P}_s$ to evaluate the performance of the proposed heuristics and implement the sensitivity analysis of the input parameters.

3.5.2 Performance of KH-ECM and MKH-ECM

Now we compare the computational results of the proposed algorithms, i.e., KH-ECM and MKH-ECM, with that of model $\mathcal{P}_s$ directly solved by the commercial software

Table 3.1: Comparison results of models $\mathcal{P}'_s$ and $\mathcal{P}_s$

Set	$\|J\|$	$\|I\|$	$\mathcal{P}'_s$		$\mathcal{P}_s$		CT($\mathcal{P}_s$)/CT($\mathcal{P}'_s$)
			Q	CT	Q	CT	
1	20		3.0	0.369	3.0	0.246	0.667
2	25		6.2	1.898	6.2	0.347	0.183
3	30	3	4.6	2.475	4.6	0.328	0.133
4	35		7.0	7.905	7.0	0.658	0.083
5	40		9.0	87.552	9.0	0.797	0.009
6	20		3.0	0.724	3.0	0.278	0.384
7	25		6.2	2.344	6.2	0.449	0.192
8	30	4	11.2	2.531	11.2	0.747	0.295
9	35		16.2	24.627	16.2	1.320	0.054
10	40		18.2	489.426	18.2	1.526	0.003
11	20		8.6	2.294	8.6	0.446	0.194
12	25		14.8	6.196	14.8	0.983	0.159
13	30	5	16.0	8.155	16.0	1.161	0.142
14	35		24.2	37.346	24.2	1.912	0.051
15	40		26.4	892.205	26.4	2.172	0.002
Average			11.640	104.403	11.640	0.891	0.170

CPLEX 12.4. Since the jobs can be pre-formed as batches in model $\mathcal{P}_s$, we use $|B^*|$ instead of $|J|$ to state the scale of the instances. Four scenarios of instances are involved, which have different sizes (middle and large) with different range of job processing times ($p_j \in (100, 200]$ and $p_j \in (50, 100]$). Many thanks to David Pisinger, who developed fast exact algorithms for KP [112] and MKP [113], by which a large problem can be solved in very short time. In this section, we will call their codes to solve the KPs and MKPs embedded in our algorithms.

Tables 3.2 and 3.3 present the comparison results for middle sized instances. Since $\mathcal{P}_s$ obtained the Pareto front, we select its solutions as the reference set. It is understandable that both H and D of the reference set itself equal to 1, thus the two indicators of $\mathcal{P}_s$ are omitted in the tables.

Table 3.2 reports the computational results of the instances with job processing time ranging from 100 to 200. By comparing the number of non-dominated solutions Q derived by the three methods, it can be seen that the performance of KH-ECM is a bit worse than that of $\mathcal{P}_s$, but MKH-ECM has found almost the same Q as $\mathcal{P}_s$. Meanwhile, the hyper volume ratio H of both KH-ECM and MKH-ECM is very close to 1, which states the two fronts have similar quality as Pareto front in terms of dominated space. Moreover, the average e-dominance value D of both proposed methods almost equals to 1, which indicates that the solutions sets are very near to Pareto front. It is worth pointing out that in several sets, D equals to 1, which means

Table 3.2: Comparison results for middle sized instances with $p_j \in (100, 200]$

Set	$\lvert B^* \rvert$	$\lvert I \rvert$	$\mathcal{P}_s$		KH-ECM				MKH-ECM			
			Q	CT	Q	H	D	CT	Q	H	D	CT
16	5	2.0	12.0	1.122	12.0	1.000	1.000	0.244	12.0	1.000	1.000	0.140
17	6	2.6	16.0	2.114	15.8	0.998	1.000	0.411	13.0	0.990	1.000	0.190
18	7	3.0	34.8	4.942	35.4	0.994	1.040	0.829	33.8	1.000	1.000	0.391
19	8	3.0	43.4	7.197	43.4	0.998	1.003	1.008	40.8	1.000	1.000	0.506
20	9	3.8	45.2	8.569	32.0	0.985	1.035	0.749	43.4	0.999	1.000	0.618
21	10	4.0	61.0	10.127	38.2	0.960	1.058	0.952	60.4	1.000	1.000	0.744
22	11	4.2	89.6	14.730	54.2	0.987	1.013	1.301	82.8	0.999	1.001	1.165
23	12	4.6	76.2	15.242	50.6	0.932	1.031	1.370	66.4	0.997	1.001	0.983
24	13	5.0	120.2	27.363	73.0	0.956	1.034	1.906	107.8	1.000	1.001	1.494
25	14	5.2	148.2	43.454	88.2	0.978	1.021	2.231	130.4	1.000	1.000	2.011
26	15	6.0	211.4	80.817	150.0	0.996	1.004	3.588	184.8	0.998	1.003	3.354
27	16	6.6	246.6	215.775	120.0	0.974	1.042	2.977	204.2	0.998	1.002	7.374
28	17	6.6	253.0	1461.130	121.2	0.968	1.044	3.138	246.0	0.999	1.002	7.055
29	18	7.0	320.0	3169.854	153.6	0.987	1.015	3.739	279.4	0.998	1.002	9.478
30	19	7.6	304.2	6996.100	154.4	0.968	1.033	3.944	269.2	0.998	1.001	56.598
31	20	7.8	353.4	16390.878	169.0	0.988	1.021	4.469	325.4	1.000	1.001	49.094
Average		4.9	145.950	1778.088	81.938	0.979	1.025	2.053	131.238	0.998	1.001	8.825

the obtained solutions all lie in the Pareto front. Considering the three metrics, it can be concluded that $\mathcal{P}_s$ performs best in terms of solution quality, and MKH-ECM have very similar performance as $\mathcal{P}_s$. In terms of solution efficiency, we can find that computational time CT of $\mathcal{P}_s$ ranges from 1.122s to 16390.878s with its average value being 1778.088s, which grows 14608 times (16390.878/1.122) from set 16 to 31. While KH-ECM (resp. MKH-ECM) spent less than 5 seconds (resp. 60 seconds) for any set, which demonstrates that the developed heuristics are much more efficient than $\mathcal{P}_s$.

Table 3.3: Comparison results for middle sized instances with $p_j \in (50, 100]$

Set	$\lvert B^* \rvert$	$\lvert I \rvert$	$\mathcal{P}_s$		KH-ECM				MKH-ECM			
			Q	CT	Q	H	D	CT	Q	H	D	CT
32	5	1.2	6.8	0.494	6.8	1.000	1.000	0.005	6.8	1.000	1.000	0.006
33	10	2.2	229.4	21.576	229.2	1.000	1.002	4.240	229.4	1.000	1.000	2.408
34	15	3.0	280.6	43.826	277.8	1.000	1.001	5.772	280.6	1.000	1.000	2.896
35	20	4.0	370.8	207.734	302.8	0.991	1.029	6.052	370.8	1.000	1.000	4.192
36	25	4.2	101.8	205.057	87.6	0.994	1.004	2.046	101.8	1.000	1.000	1.508
37	30	5.0	104.2	55.265	102.6	0.999	1.002	2.609	104.2	1.000	1.000	1.478
38	35	7.0	617.0	505.571	580.2	1.000	1.010	12.680	617.0	1.000	1.000	9.641
39	40	7.4	502.2	1283.992	452.6	0.997	1.005	10.041	501.6	1.000	1.000	8.132
40	45	8.6	660.8	1424.866	612.0	0.999	1.001	13.033	660.6	1.000	1.000	11.599
41	50	9.4	554.4	2357.996	549.4	1.007	1.004	12.270	554.4	1.000	1.000	11.799
42	55	9.8	386.8	4066.906	402.2	1.022	1.001	9.305	386.8	1.000	1.000	8.977
43	60	11.4	676.4	11323.212	631.2	1.001	1.002	13.640	676.4	1.000	1.000	14.637
44	65	12.0	659.8	16353.410	570.6	0.971	1.002	13.075	659.8	1.000	1.000	13.979
Average		6.6	396.231	2911.531	369.615	0.999	1.005	8.059	396.169	1.000	1.000	7.019

In Table 3.3, we report the results on instances with shorter p_j that varies from 50 to 100. It can be seen that $\mathcal{P}_s$ averagely found 396.231 non-dominated solutions, and analogously, KH-ECM and MKH-ECM obtained 369.615 and 396.169 non-dominated solutions on average, respectively. Moreover, both H and D of KH-ECM are very close to 1, which indicates all the obtained solution points are very near to the Pareto front. Furthermore, this two metrics of MKH-ECM both equal to 1 over all sets 32 to 44, which implies the obtained solutions are on the Pareto front. That is, the MKH-ECM has found the Pareto front of the set where its number of obtained solutions equals to that of $\mathcal{P}_s$, i.e., sets 32-38 and 41-44. While by observing CT, we can see that the proposed methods spent far less time than $\mathcal{P}_s$. Meanwhile, CT of $\mathcal{P}_s$ exponentially grows with the problem size, whereas CT of both KH-ECM and MKH-ECM increases very slightly. This shows the significant efficiency of the proposed methods.

In addition, we can find that the three solution quality metrics as well as CT in Table 3.3 are better than that in Table 3.2. This indicates that the proposed methods are more effective and efficient for the instances with shorter job processing time.

Table 3.4 and 3.5 give the computational results for large sized instances. Since it is very time consuming to obtain the Pareto front, to evaluate the performance of the proposed method in reasonable time, we limit the CT of each instance to 3600 CPUs and the CT of each iteration of $\mathcal{P}_s$ to 30 CPUs. According to our pretest, KH-ECM is more efficient and can obtain more solutions, thus we set its results as the reference solution set. In addition, due to all the methods provide approximative Pareto fronts, metric D is omitted in the two tables. Besides, H of reference set obtained by KH-ECM equals to 1, which is also omitted. In the tables, "-" expresses that no feasible solution is generated by the method within the time limit for the five instances.

Table 3.4: Comparison results for large-size instances with $p_j \in (100, 200]$

Set	$\lvert B^* \rvert$	$\lvert I \rvert$	$\mathcal{P}_s$			KH-ECM		MKH-ECM		
			Q	H	CT	Q	CT	Q	H	CT
45	25	9.8	126.6	0.427	2891.720	216.6	2.786	20.0	0.396	3600.000
46	30	11.4	93.2	0.493	2166.440	227.8	2.957	18.4	0.217	3600.000
47	35	13.6	166.4	0.671	2807.090	369.8	4.678	-	-	-
48	40	15.4	49.2	0.163	3600.000	397.4	5.119	-	-	-
49	50	19.4	46.2	0.232	2184.060	514.2	7.068	-	-	-
50	80	31.6	-	-	-	1016.0	14.948	-	-	-
51	100	39.6	-	-	-	1105.4	19.319	-	-	-
52	200	79.8	-	-	-	2778.6	66.550	-	-	-
53	300	114.4	-	-	-	3772.8	114.545	-	-	-
54	400	143.0	-	-	-	2797.0	116.733	-	-	-
55	500	170.4	-	-	-	2121.2	105.225	-	-	-
Average		58.945	-	-	-	1392.4	41.812	-	-	-

Table 3.4 illustrates the results for instances with $p_j \in (100, 200]$ and number of batches $|B^*|$ varying from 25 to 500. It can be found that $\mathcal{P}_s$ can obtain a set of solutions for set 45 to 49 within the time limit. Meanwhile, MKH-ECM can only yield solutions for set 45 and 46. This is because that MKP is also NP-hard, it becomes difficult to solve a series of large-size MKPs. Fortunately, KH-ECM averagely found 1392.4 solutions for instances with batches up to 500. Both number of solutions and hyper volume ratio of KH-ECM are better than that of $\mathcal{P}_s$, which indicates that KH-ECM has better performance on solution quality than $\mathcal{P}_s$. Moreover, KH-ECM only consumes less than 120s to find the solutions for all the instances, which shows great time efficiency.

Table 3.5: Comparison results for large-size instances with $p_j \in (50, 100]$

Set	$\|B^*\|$	$\|I\|$	$\mathcal{P}_s$			KH-ECM		MKH-ECM		
			Q	H	CT	Q	CT	Q	H	CT
56	70	13.6	330.8	0.767	3006.000	475.6	6.117	609.4	1.013	8.918
57	80	14.8	244.2	0.614	3600.000	652.0	7.621	697.6	1.002	10.736
58	90	17.0	179.6	0.521	3600.000	656.4	10.829	738.8	1.008	44.291
59	100	19.8	123.8	0.403	3600.000	558.0	7.071	614.2	1.003	166.375
60	120	22.4	38.0	0.565	3600.000	1031.2	22.575	732.8	0.830	2168.747
61	140	23.0	32.2	0.353	3600.000	355.4	6.421	144.0	0.592	2882.333
62	150	28.0	31.8	0.320	3600.000	1402.0	20.033	321.6	0.390	2887.449
63	160	30.4	26.0	0.108	3600.000	1577.6	31.072	241.2	0.214	3600.000
64	200	35.6	-	-	-	1229.8	19.459	-	-	-
65	300	56.0	-	-	-	2480.2	48.476	-	-	-
66	500	87.0	-	-	-	2251.6	66.737	-	-	-
Average		31.600	-	-	-	1151.8	22.401	-	-	-

Table 3.5 provides the computational results for instances with $p_j \in (50, 100]$ and $|B^*|$ increasing from 70 to 500. We find that both $\mathcal{P}_s$ and MKH-ECM can obtain a set of non-dominated solutions for the instances with $|B^*| \leq 160$ within 3600s. While KH-ECM can generate the solution sets for instances with batches up to 500 within 70s. This shows KH-ECM is the most time efficient one. In terms of solution quality, we can observe that both MKH-ECM and KH-ECM obtain better Q and H than $\mathcal{P}_s$, which implies that the two proposed methods have better performance than $\mathcal{P}_s$. Moreover, it can be seen from set 56 to 59, MKH-ECM performs better than KH-ECM since it gets best Q and H.

By comparing Table 3.4 and 3.5, similar to Table 3.2 and 3.3, we can find that the proposed methods perform better on the instances with shorter processing time, especially MKH-ECM. This is because that the called MKP code is more efficient for the instances where the quotient $|B^*|/\lambda$ is relatively large [113].

Summing up, both proposed heuristic methods can find a set of solutions that very close to the exact Pareto front. Especially, MKH-ECM almost found the Pareto front for the middle sized instances. While the heuristics are much more efficient than $\mathcal{P}_s$. Moreover, KH-ECM outperforms $\mathcal{P}_s$ in terms of solution quality for larger sized instances.

3.5.3 Sensitivity analysis

This subsection aims to test the impact of different input parameters on the problem via analysing the information of Pareto front. The sensitive analysis experiments of α and e_i are conducted with middle size instances solving with model $\mathcal{P}_s$. The computational results are reported in Tables 3.6 - 3.8 and Fig.'s 3.7 and 3.8. In the tables, $\overline{CT}$ denote the average computational time of each non-dominated point, i.e., $\overline{CT}$ =CT/Q. Let (C^S_{max}, EC^S) represent the objective vector of the selected preferable solution calculated by fuzzy-logic-based approach. To evaluate the electricity cost reducing effectiveness of the selected solution, we will compare its objective value vector with the vector (C^I_{max}, EC^N), where C^I_{max} can be regarded as the optimal makespan without constraint of electricity cost and EC^N is the corresponding cost. Thus, we calculate the reduced electricity cost by the proposed approach R_{EC} as $(EC^N - EC^S)/EC^N$. Similarly, the increased makespan is calculated as $(C^S_{max} - C^I_{max})/C^I_{max}$, denoted by $IN_{C_{max}}$.

Table 3.6 reports the computational results for three scenarios regrading number of periods $|I|$. $|I|$ is defined as $\alpha \max_{j\in J}\{p_j \times |B^*|/S_i\}$. Parameter α is generated from [0.6, 1.0], which is regarded as the baseline. Other two cases α are generated from $[1.1, 1.5]$ and $[1.6, 2.0]$, respectively, and the other parameters remain unchanged.

From Table 3.6, we can see that for each type of α, the computational time CT and number of nondominated points Q increase with $|B^*|$ and $|I|$. Moreover, we can see in Fig. 3.7 that the changing trends of CT and Q are more obvious for larger α. More precisely, we can observe that given the number of batches $|B^*|$, the computational time increases with the number of periods $|I|$. Take sets 80 and 87 as an example, both sets have 15 batches but different periods, while CT of set 87 is greater than that of set 80, which is mainly because that Q increases from 322.6 to 413.6. This shows that the increasing of $|I|$ adds the complexity of the problem, since more periods result in more non-dominated points. On the other hand, we can also find that given $|I|$, CT increases with $|B^*|$. Take sets 76 and 71 for example, both of them have five periods, but CT are 21.40 and 29.60s, respectively. This implies that the complexity of the problem increases with the number of jobs.

Table 3.6: Comparison for sensitivity analysis of $|I|, |I| = \alpha \max_{1 \leq j \leq J}\{p_j \times |B^*|/S_i\}$

Set	α	$\lvert B^*\rvert$	$\lvert I\rvert$	Q	CT	$\overline{CT}$	$R_{EC}(\%)$	$IN_{C_{max}}(\%)$
67		9	3.6	45.0	8.321	0.183	31.03	21.73
68		10	4.0	65.0	12.942	0.200	33.50	20.79
69		11	4.2	91.0	19.332	0.210	23.81	15.44
70	[0.6, 1.0]	12	4.8	116.4	22.567	0.196	31.49	21.63
71		13	5.0	119.0	29.596	0.249	27.39	16.49
72		14	5.4	166.2	62.309	0.367	24.74	19.28
73		15	5.8	193.6	102.540	0.527	25.71	20.54
Average				113.743	36.801	0.276	28.24	19.41
74		9	4.0	64.4	9.578	0.149	49.55	34.36
75		10	4.8	106.8	17.086	0.161	32.89	28.87
76		11	5.0	134.0	21.403	0.160	35.27	28.05
77	[1.1,1.5]	12	5.8	197.0	34.427	0.175	46.17	46.50
78		13	7.0	220.6	54.315	0.252	35.40	33.26
79		14	7.4	248.4	67.485	0.273	26.07	22.69
80		15	8.4	322.6	104.918	0.321	16.05	11.93
Average				184.829	44.173	0.213	34.49	29.38
81		9	5.0	85.4	12.966	0.152	25.83	31.17
82		10	5.2	119.4	19.031	0.159	32.89	37.00
83		11	6.0	140.8	24.971	0.179	30.72	26.98
84	[1.6,2.0]	12	6.8	165.2	24.532	0.151	13.17	12.90
85		13	7.8	222.8	52.482	0.239	37.18	32.02
86		14	8.6	345.4	79.176	0.235	30.51	25.65
87		15	9.4	413.6	145.823	0.355	37.01	32.65
Average				213.229	51.283	0.210	29.62	28.34

In addition, we can see that R_{EC} distributes between 13.17% and 49.55% with its average value being 30.78%. In other words, the total electricity cost under the TOU policy can reduced from 13.17% up to 49.55% with appropriate scheduling. This shows that appropriate scheduling under TOU policy can offer great benefits to reduce energy cost for industrial users. $IN_{C_{max}}$ varies from 11.93% to 46.50% with its average value being 25.71%, which means that the total electricity cost and the makespan are two conflicting objectives. However, note that R_{EC} is greater than $IN_{C_{max}}$ for almost all the problem sets, which indicates that industrial users can benefit more than loss with the selected preferable solution. Besides, by comparing R_{EC} and $IN_{C_{max}}$ under three different scenarios, we can find that they slightly increases when α increases. This shows that more periods may result in more electricity cost reduction, but may incur a longer makespan.

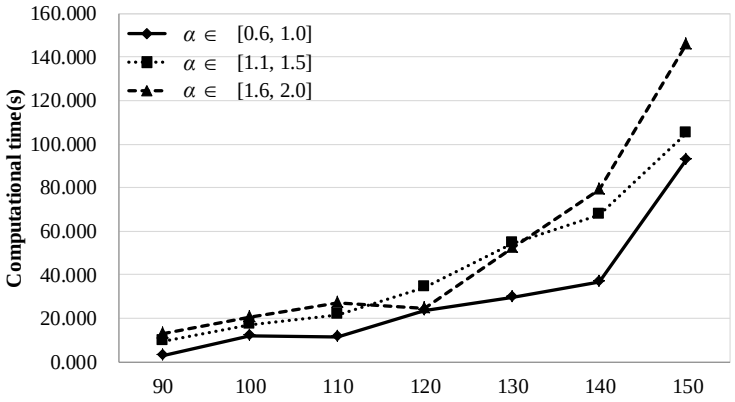


Fig. 3.7: Computational time for sensitivity analysis of $|I|, |I| = \alpha \max_{1 \leq j \leq J}\{p_j \times |B^*|/S_i\}$

Table 3.7: Comparison for sensitivity analysis of e_i

Set	e_i	$\|B^*\|$	$\|I\|$	Q	CT	$\overline{CT}$	$R_{EC}(\%)$	$IN_{C_{max}}(\%)$
75		9	3.0	32.2	4.274	0.153	5.00	4.35
76		10	4.0	86.0	12.858	0.150	33.30	27.38
77		11	4.0	57.0	9.752	0.168	6.20	4.69
78	{20, 12, 5}	120	5.0	92.8	16.382	0.180	22.64	19.36
79		13	5.0	109.8	25.377	0.230	13.55	11.55
80		14	5.0	99.4	43.517	0.360	12.10	9.21
81		15	6.0	148.4	64.410	0.409	18.28	16.33
Average				89.371	25.224	0.236	15.87	13.27
82		9	3.0	21.0	2.705	0.129	3.82	2.58
83		10	4.0	68.4	11.928	0.175	35.21	21.58
84		11	4.0	64.0	11.353	0.178	17.62	10.35
85	{30, 15, 5}	12	5.0	113.0	23.687	0.210	37.85	25.62
86		13	5.0	124.4	29.671	0.239	28.09	17.25
87		14	5.0	123.0	36.473	0.296	17.77	10.59
88		15	6.0	217.2	92.663	0.441	29.06	23.40
Average				104.429	29.783	0.238	24.20	15.91
89		9	3.0	21.0	2.824	0.135	4.29	2.58
90		10	4.0	68.6	11.148	0.163	40.02	21.58
91		11	4.0	64.4	10.477	0.166	20.47	10.47
92	{50, 25, 5}	12	5.0	114.0	20.486	0.280	42.03	25.62
93		13	5.0	124.2	27.214	0.219	31.53	17.25
94		14	5.0	124.5	35.885	0.287	20.51	10.85
95		15	6.0	218.0	92.543	0.439	31.89	23.40
Average				104.957	28.654	0.227	27.25	15.96

Table 3.7 presents the results of sensitivity analysis of unit electricity cost e_i, which are set as {20, 12, 5}, {30, 15, 5} and {50, 25, 5}, respectively. The computational time of the three scenarios range between 4.27 and 64.41s, 2.71 and 92.66s, 2.82

and 92.54s, respectively. Moreover, it can be found in Fig. 3.8 that the changing trends of CT for the three scenarios are almost the same. Furthermore, the average computational time of the proposed algorithm for all scenarios are 25.22s, 29.78s and 28.65s, respectively, which are almost the same. These results indicate that our proposed algorithm is insensitive to the changes of unit electricity cost.

Besides, by comparing R_{EC} of the three scenarios, we can see that given the number of jobs and periods, in general the greater difference of unit electricity cost among different periods, the more electricity cost can be reduced. Take sets 81 and 95 for example, both problem sets have 15 batches and six periods, set 81 reduces 18.28% electricity cost while the electricity cost in the latter set is reduced up to 31.89%. This demonstrates that industrial users may benefit more from the TOU policy when the differences of electricity prices among different periods are bigger.

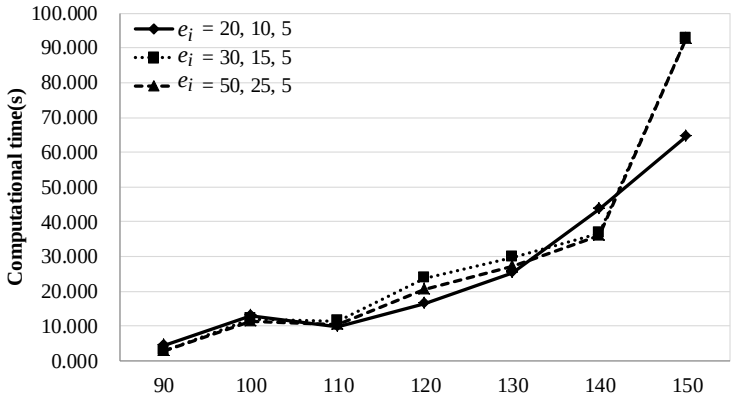


Fig. 3.8: Computational time for sensitivity analysis of e_i

In addition, it would be significative to observe the impact of different batch capacity on the problem. Thus we set the batch capacity as 5 and other parameters in Table 3.3 remain unchanged. Note that the optimal number of batches $|B^*| = \lceil |J|/C \rceil$ has increased due to the reduction of batch capacity C, which leads that it is necessary to increase the number of periods $|I|$ such that all the jobs can be completed in the scheduling horizon. Thus, $|I|$ is regenerated according to $\alpha \max_{j \in J}\{p_j \times |B^*|/S_i\}$. The computational results are presented in Table 3.8. By comparing the two Tables, it can be observed that for the instances with the same number of jobs, CT increases when the batch capacity reduces. This is because the number of batches increases. Take sets 16 and 96 for example, both sets have 50 jobs while their computational time are 1.122s and 11.616s, respectively. Moreover, we can find from Tables 3.3 and 3.8

that computational time of most instances with same number of batches are almost the same, such as sets 21 and 96, sets 27 and 99, etc.

Table 3.8: Computational results for the instances with $E_i = \{30, 15, 5\}$, $r_m \in [0.6, 1.0]$, $C = 5$, $p_j \in (100, 200]$

Set	$\|J\|$	$\|I\|$	Q	CT	$\overline{CT}$	$R_{EC}(\%)$	$IN_{C_{max}}(\%)$
96	50	4.0	74.4	11.616	0.156	35.51	21.52
97	60	5.6	136.8	24.231	0.181	35.17	31.28
98	70	6.0	277.8	63.971	0.229	38.74	35.34
99	80	6.2	239.8	226.190	1.003	28.47	22.00
100	90	6.8	276.6	2488.884	8.638	25.94	18.70
101	100	7.8	436.4	16954.372	38.851	20.46	17.09
102	110	8.2	–	–	–	–	–
Average			240.300	3294.877	8.176	30.71	24.32

3.6 Conclusions

This chapter investigates a bi-objective NP-hard problem: single batch processing machine scheduling considering time-of-use tariffs problem. Two multi-objective heuristic algorithms were developed for the problem. Their core idea is to transform the bi-objective problem into a series of single objective problems with the framework of ε-constraint method. Then, each single objective problem was efficiently resolved by solving successive knapsack problems (KP) or multiple knapsack problems (MKP). Denote the two methods as KH-ECM and MKH-ECM, respectively. The computational results demonstrate that the proposed heuristics can efficiently solve large-size problems with up to 500 batches. The computational results demonstrate that the proposed approaches can obtain a set of good quality solutions and are much more efficient and effective than the commercial solver $\mathcal{P}_s$ for the studied problem. To be more specific, MKH-ECM is almost able to find the Pareto front for a middle sized instance within one minute and KH-ECM is more effective and efficient than $\mathcal{P}_s$ in solving larger sized instances. The corresponding work has been published in the following paper.

J. Cheng, F. Chu, C. Chu, and W. Xia. Bi–objective optimization of single–machine batch scheduling under time–of–use electricity prices. *RAIRO – Operations Research*, 50(4–5):715–732, 2016.
J. Cheng, F. Chu, M. Liu, and W. Xia. Effective heuristics for single machine batch scheduling under time-of-use tariffs. *Sustainability*, under review.

Chapter 4

Single machine batch scheduling with machine on/off switching under TOU tariffs

4.1 Introduction

We can observe that an idle duration may exist in some periods for the single batch processing machine scheduling problem studied in Chapter 3 in which machine turn-on and -off are assumed to not consume energy. Consequently, the machine is always turned off when finishing processing in each period and is restarted when needed. However, turning on machines can consume a great amount of energy in some manufacturing environments, e.g., steel manufacturing. As indicated by [109], the resulted non-processing energy (NPE) consumption related to machine turn-on, turn-off and idling constitutes a significant part (over 30%) of the total energy consumption for certain scheduling environments. Remarkably, it has been shown in [108],[109],[145], the NPE consumption can be significantly reduced by rationalized machine turn-on/off. The above statement motives us to optimize the whole electricity consumption cost considering machine on/off switching that is a natural extension of the work in Chapter 3.

In this chapter, we investigate a new bi-objective single machine batch scheduling problem under TOU tariffs with machine on/off strategy. The objectives are to minimize total electricity cost and makespan. Compared with the existing work, we assume that the machine has different energy consumption states: processing, running idle, and turning on. For each idling duration, a decision that leaving machine running idle or turning it off has to be made. The work in this chapter has to determine batch composition, batch sequencing and machine on/off status, which is more complex

than that in Chapter 3. In the following, we first describe the problem and formulate it as an MILP model. Then an improved model is further proposed based on optimal batch rule analysis, which greatly reduces Pareto optimal solution space. A bin-packing heuristic based ε-constraint method (BPH-ECM) is developed to effectively and fast solve the problem. Finally, computational results on randomly generated instances are presented to evaluate the effectiveness of the proposed method.

The remainder of this chapter is organized as follows. Section 4.2 describes and mathematically formulates the problem, proves its complexity, and proposes an improved MILP based on analysed properties. In Section 4.3, a bin-packing heuristic based ε-constraint method is developed to fast solve the problem. Computational results are reported in Section 4.4. Finally, Section 4.5 summarizes this work and indicates future research directions.

4.2 Problem formulation

This study considers a bi-criteria single machine batch scheduling problem with machine on/off switching under TOU pricing. With the three-field notation proposed by Graham [59], the studied problem can be denoted by $TOU, 1|on/off, B|E, C_{max}$, which is described in detail as follows.

A batch processing machine with C jobs' capacity consumes P^{proc} (resp. P^{idle}) units of electricity per unit time when processing (resp. staying idle), and $P^{proc} > P^{idle}$. Turning on the machine is in a relatively short time and is assumed to require P^{on} units electricity. Note that turning off the machine does not consume energy. A given set $J = \{1, 2, \ldots, |J|\}$ independent nonpreemptive jobs have to be processed on the single batch processing machine within a scheduling horizon with $|I|$ periods, where we use I to denote the set of all periods. Each job j is available at time 0 and has a processing time p_j. Without loss of generality, we assume that all the jobs are numbered in nonincreasing order of the processing times, i.e., $p_1 \geq p_2 \geq \cdots \geq p_{|J|}$. All jobs can be regrouped into $|B|$ (to be determined) batches, where $\lceil |J|/C \rceil \leq |B| \leq |J|$ and B is the set of batches. The processing time of a batch is determined by the largest job processing time in the batch.

In this paper, a period $i \in I$ is considered as a work shift with duration S_i. Let s_i denote the starting time of period i, respectively. The length of the scheduling horizon is $s_{|I|+1}$. Any p_j is less than the period duration $S_i, \forall i \in I$. The unit electricity price of period i, denoted by Pr_i, is calculated as the average price of period i based on the tariffs information as $TOU, 1|B|E, C_{max}$ in Chapter 3. Thus,

the unit electricity cost for processing jobs in period i, denoted by e_i^p, is calculated as $e_i^p = P^{proc} \times Pr_i$. Similarly, we can compute the unit electricity cost of machine staying idle as $e_i^s = P^{idle} \times Pr_i$ and turn-on $e_i^o = P^{on} \times Pr_i$, respectively. To determine the NPE cost, we need to analyze the turn-on/off strategies between any two adjacent periods. All cases are analyzed as follows.

Case 1: there exists processing in period $i \in I \backslash \{|I|\}$ while period $i+1$ has not, then the machine will be turned off when finishing processing in period i and thus the machine idling cost in period i as well as the NPE cost in period $i+1$ are 0;

Case 2: no processing exists in period $i \in I \backslash \{|I|\}$ while period $i+1$ has, then the machine will be turned on in period $i+1$ as it is in a shut-down state in period i and thus the NPE cost in period i is 0 and there exists turn-on energy cost e_{i+1}^o in period $i+1$;

Case 3: there exists processing in both periods $i \in I \backslash \{|I|\}$ and $i+1$, then if $e_i^s d_i^{idle} > e_{i+1}^o$, where d_i^{idle} denotes the idling duration in period i, then the machine will be turned off in period i when finishing processing and thus the machine idling cost in period i is 0 and there exists turn-on energy cost e_{i+1}^o in period $i+1$; and otherwise the machine will be kept idling in period i and thus the machine idling cost in period i is $e_i^s d_i^{idle}$ and there is no turn-on energy cost in period $i+1$.

The objective of the problem is to find an optimal schedule that consists of batching the jobs, allocating the batches to periods, and deciding whether the machine to be turned off or kept running idle for an idle duration in order to optimize the total electricity cost (E) and the makespan (C_{max}) simultaneously.

4.2.1 Mathematical formulation

To formulate the model of the problem, we first denote indices, parameters and define decision variables as follows.

Indices:

j: index of jobs; $j \in J = \{1, 2, \ldots, |J|\}$
b: index of batches; $b \in B = \{1, 2, \ldots, |B|\}$
i: index of periods; $i \in I = \{1, 2, \ldots, |I|\}$

Paremeters:

C: capacity of the machine;
p_j: processing time of job $j, \forall j \in J$;
s_i: starting time of period $i, \forall i \in I$;
S_i: duration of period i;

e_i^p: electricity cost for processing jobs per unit time in period i , $\forall i \in I$;

e_i^s: electricity cost for machine idling per unit time in period i, $\forall i \in I$;

e_i^o: electricity cost produced by turning on the machine in period i, $\forall i \in I$;

Decision Variables:

$x_{j,b,i}$: 1 if job j is allocated to batch b and processed in period i; 0 otherwise, $\forall j \in J, \forall b \in B, \forall i \in I$;

$y_{b,i}$: 1 if batch b is allocated to period i; 0 otherwise, $\forall b \in B, \forall i \in I$;

z_i: 1 if at least one batch is allocated to period i; 0 otherwise, $\forall i \in I$;

$|B|$: number of the batches;

$P_{b,i}$: $\max\{p_j \mid j \in b\}$ if batch b is processed in period i; 0 otherwise, $\forall b \in B, \forall i \in I$;

v_i: 0 if the machine is turned off or no job is processed in period i, or no job is to be processed in period $i+1$; 1 otherwise, $\forall i \in I, v_0 = 0$;

u_i: > 0 if the machine is not turned off in period i; 0 otherwise, $\forall i \in I \backslash \{|I|\}$;

E: total electricity cost for completing all jobs;

C_{max}: completion time of the last job in J.

With the notations and variables defined above, the investigated problem can be formulated as the following model $\mathcal{P}'_f$.

$$\mathcal{P}'_f: \quad \min \ f_1 = E \tag{4.1}$$

$$\min \ f_2 = C_{max} \tag{4.2}$$

$$\text{s.t.} \quad \sum_{i \in I} \sum_{b \in B} x_{j,b,i} = 1, \forall j \in J \tag{4.3}$$

$$\sum_{i \in I} y_{b,i} = 1, \forall b \in B \tag{4.4}$$

$$\sum_{j \in J} x_{j,b,i} \leq y_{b,i} C, \forall b \in B, \forall i \in I \tag{4.5}$$

$$x_{j,b,i} p_j \leq P_{b,i}, \forall j \in J, \forall b \in B, \forall i \in I \tag{4.6}$$

$$z_i \leq \sum_{b \in B} y_{b,i}, \forall i \in I \tag{4.7}$$

$$\sum_{b \in B} P_{b,i} \leq S_i z_i, \forall i \in I \tag{4.8}$$

$$u_i \geq e_i^s (S_i v_i - \sum_{b \in B} P_{b,i}), \forall i \in I \tag{4.9}$$

$$v_i \leq z_{i+1}, \forall i \in I \backslash \{|I|\} \tag{4.10}$$

$$v_i \leq z_i, \forall i \in I \tag{4.11}$$

$$\sum_{i \in I} \sum_{b \in B} e_i^p P_{b,i} + \sum_{i \in I} e_i^o (z_i - v_{i-1}) + \sum_{i \in I/|I|} u_i \leq E \tag{4.12}$$

$$s_i z_i + \sum_{b \in B} P_{b,i} \leq C_{max}, \forall i \in I \tag{4.13}$$

$$x_{j,b,i}, y_{b,i}, z_i, v_i \in \{0,1\}, \forall j \in J, \forall b \in B, \forall i \in I \tag{4.14}$$

$$P_{b,i} \geq 0, u_i \geq 0, E \geq 0, C_{max} \geq 0, \forall b \in B, \forall i \in I \tag{4.15}$$

Objectives (4.1) and (4.2) are to minimize total electricity cost E and makespan C_{max}, respectively. Equation (4.3) ensures that any job $j \in J$ is allocated to one batch and one period. Equation (4.4) ensures that any batch $b \in B$ is processed in one period. Equation (4.5) restricts that the number of jobs in any batch does not exceed the capacity C and ensures that any job $j \in J$ is processed in period i only if its formed batch is processed in this period. Equation (4.6) states that the processing time of any batch is determined by the longest time of its involved jobs. Equation (4.7) states that variable z_i takes value 0 if no batches are processed in period $i \in I$. Equation (4.8) ensures that the total batch processing time in period $i \in I$ will not exceed the duration of this period. Equation (4.9) calculates the idling electricity cost in period $i \in I$. Equation (4.10) states that the machine is turned off in period $i \in I$ if there is no jobs to be processed in period $i+1$. Equation (4.11) ensures that the machine keeps being shut down in period $i \in I$ if there is no jobs to be processed in period i. Equation (4.12) calculates the total electricity cost. Equation (4.13) defines the makespan. Equations (4.14) and (4.15) are the restrictions on decision variables. Note that the number of batches $|B|$ is initially considered as its upper bound $|J|$ to derive a linear model.

4.2.2 Optimal batch rule analysis

In this section, we devote our attention to reducing the search space of the proposed model for optimal solutions by analyzing the properties of the problem. In what follows, we demonstrate that job batching and batch scheduling can be solved with two independent steps without loss of optimality for the considered problem.

A solution of the problem $TOU, 1|on/off, B|E, C_{max}$ can be uniquely defined by $(|B|, \{J_b, 1 \leq b \leq |B|\}, \{\tau_b, 1 \leq b \leq |B|\}, \{\nu_i, i \in I\})$, where $|B|, J_b$ and τ_b are the number of batches, the set of jobs allocated into batch b ($J_1 \cup J_2 ... \cup J_{|B|} = J$), and the period in which batch b is processed, respectively. ν_i denotes the machine status in the idle duration of period i, i.e., the machine is kept idling or turned off.

We particularly consider those solutions where the batches are formed with a so-called LPT-based method. With such method, jobs in the nonincreasing order

are put into batches in the following way: job j with $(b-1)C < j \leq bC$ and $1 \leq b \leq \lceil |J|/C \rceil - 1$ is allocated to batch b, and the remaining jobs to batch $\lceil |J|/C \rceil$.

Theorem 3 shows that we only need to consider such solutions to derive the Pareto front of the considered problem.

Theorem 3 *Any solution of $TOU, 1|on/off, B|E, C_{max}$ in which the batches differ from those formed with the LPT-based method is (at least weakly) dominated.*

Proof: Similarly to the proof for problem $TOU, 1|B|E, C_{max}$ studied in Chapter 3, we first define the following notations.

J_b: the set of jobs contained in batch b, $J_b \subseteq J$;
$n(J_b)$: the serial number of the least indexed job (thus with the largest processing time) in set J_b, i.e., $n(J_b) = \min\{j | j \in J_b\}$;
$P(J_b)$: the processing time of batch b (the processing time of the least indexed job in J_b), i.e., $P(J_b) = \max_{j \in J_b} p_j = p_{n(J_b)}$.

Let $|B^*|$ and J_b^* represent the number of batches formed with LPT-based method and the set of jobs involved in the batch b ($1 \leq b \leq |B^*|$), respectively. We have

$$\begin{aligned} |B^*| &= \lceil |J|/C \rceil, \\ J_b^* &= \{(b-1)C+1, (b-1)C+2, \ldots, bC\}, \quad b = 1, 2, \ldots, |B^*|-1, \\ J_{|B^*|}^* &= \{(|B^*|-1)C+1, (|B^*|-1)C+2, \ldots, n\}. \end{aligned}$$

With the above construction, we have

$$n(J_b^*) = (b-1)C+1, \tag{4.16}$$
$$P(J_b^*) = p_{(b-1)C+1}. \tag{4.17}$$

Suppose there is a feasible schedule $\hat{S}$ with solution $(|\hat{B}|, \{\hat{J}_b, 1 \leq b \leq |\hat{B}|\}, \{\hat{\tau}_b, 1 \leq b \leq |\hat{B}|\}, \{\hat{\nu}_i, i \in I\})$, in which the batches differ from those formed with LPT-based method. With similar proof to Theorem 1 in Chapter 3 (from formulas 3.14 to 3.19), we can conclude that the processing time of the batches are at least as long as those formed with the LPT-based method, i.e.,

$$P(\hat{J}_b) = p_{n(\hat{J}_b)} \geq p_{(b-1)C+1} = P(J_b^*), 1 \leq b \leq |B^*|. \tag{4.18}$$

Renew schedule $\hat{S}$ to S^* with the batches formed with LPT-based method, the new solution $(|B^*|, \{J_b^*, 1 \leq b \leq |B^*|\}, \{\hat{\tau}_b, 1 \leq b \leq |\hat{B}|\}, \{\hat{\nu}_i, i \in I\})$ can be achieved by removing batches $|B^*|+1, \ldots, |\hat{B}|$, if any, and replacing each batch $\hat{J}_b (1 \leq b \leq |B^*|)$ by the corresponding one formed with the LPT-based method (i.e., batch J_b^*) without changing the starting time. Relation (4.18) indicates that the new schedule S^* is also feasible. Because some batches are removed and the processing time of the rest batches are reduced, the makespan is not increased. Next, we prove that the total electricity cost is also not increased.

For any period $i \in I$ that involves job-processing, i.e., $z_i = 1$, according to (4.18), the total processing time in period i of schedule $\hat{S}$, calculated by $\sum_{b=1}^{|\hat{B}|} \hat{P}_{b,i}$, and that of schedule S^*, calculated by $\sum_{b=1}^{|B^*|} P^*_{b,i}$, must have the following relation.

$$\sum_{b=1}^{|\hat{B}|} \hat{P}_{b,i} \geq \sum_{b=1}^{|B^*|} P^*_{b,i} \tag{4.19}$$

Let $\hat{E}_i$ (resp. E^*_i) denote the total processing and idling cost of period i, and the turn-on cost of period $i+1$ of schedule $\hat{S}$ (resp. S^*). Since $z_i = 1$, the magnitude relationship of $\hat{E}_i$ and E^*_i can be analyzed through the following three cases.

Case 1: $z_{i+1} = 0$, then, $\hat{E}_i - E^*_i = e^p_i \sum_{b=1}^{|\hat{B}|} \hat{P}_{b,i} - e^p_i \sum_{b=1}^{|B^*|} P^*_{b,i} \geq 0$.

Case 2: $z_{i+1} = 1$ and $e^s_i(S_i - \sum_{b=1}^{|\hat{B}|} \hat{P}_{b,i}) > e^o_{i+1}$, then, we have

$$\hat{E}_i = e^p_i \sum_{b=1}^{|\hat{B}|} \hat{P}_{b,i} + e^o_{i+1}. \tag{4.20}$$

According to (4.19), we have

$$e^s_i(S_i - \sum_{b=1}^{|B^*|} P^*_{b,i}) \geq e^s_i(S_i - \sum_{b=1}^{|\hat{B}|} \hat{P}_{b,i}) > e^o_{i+1},$$

thus, for the solution of E^*_i, we have

$$E^*_i = e^p_i \sum_{b=1}^{|B^*|} P^*_{b,i} + e^o_{i+1}. \tag{4.21}$$

Comparing (4.20) with (4.21), it is obvious that $\hat{E}_i - E^*_i \geq 0$.

Case 3: $z_{i+1} = 1$ and $e^s_i(S_i - \sum_{b=1}^{|\hat{B}|} \hat{P}_{b,i}) \leq e^o_{i+1}$, then we have

$$\begin{aligned}
&\hat{E}_i - E^*_i \\
&= e^p_i \sum_{b=1}^{|\hat{B}|} \hat{P}_{b,i} + e^s_i(S_i - \sum_{b=1}^{|\hat{B}|} \hat{P}_{b,i}) - (e^p_i \sum_{b=1}^{|B^*|} P^*_{b,i} + \min\{e^s_i(S_i - \sum_{b=1}^{|B^*|} P^*_{b,i}), e^o_{i+1}\}) \\
&\geq e^p_i \sum_{b=1}^{|\hat{B}|} \hat{P}_{b,i} + e^s_i(S_i - \sum_{b=1}^{|\hat{B}|} \hat{P}_{b,i}) - (e^p_i \sum_{b=1}^{|B^*|} P^*_{b,i} + e^s_i(S_i - \sum_{b=1}^{|B^*|} P^*_{b,i})) \\
&= (e^p_i - e^s_i)(\sum_{b=1}^{|\hat{B}|} \hat{P}_{b,i} - \sum_{b=1}^{|B^*|} P^*_{b,i}).
\end{aligned}$$

Since $e^p_i > e^s_i$ and $\sum_{b=1}^{|\hat{B}|} \hat{P}_{b,i} \geq \sum_{b=1}^{|B^*|} P^*_{b,i}$, thus we have $\hat{E}_i - E^*_i \geq 0$.

The above results of the three cases indicate that the total electricity cost of the new schedule S^* is not greater than that of schedule $\hat{S}$. Consequently, neither

electricity cost nor makespan is increased in the schedule with batches formed with LPT-based method, which means that the initial schedule $\hat{S}$ is (at least weakly) dominated by the new one. □

From Theorem 3, we can conclude that the solution from the LPT-based method (at least weakly) dominates other solutions for the studied problem, where the jobs are nonresumable and of non-identical processing times. Moreover, Theorem 3 holds for any time-of-use pricing strategy. However, for problems involving other types of jobs, for example, the jobs with non-identical size or non-identical release time, the solution from the LPT-based method might not be able to dominate other solutions, and it needs more research efforts.

Besides, the following theorem also holds.

Theorem 4 *The batch scheduling problem* $TOU, 1|on/off, B|E, C_{max}$ *is strongly NP-hard.*

Proof: For the special case that $e_i^s = 0, \forall i \in I$ and $e_i^o = 0, \forall i \in I$, the problem $TOU, 1|B|E, C_{max}$ has been proved to be NP-hard in the strong sense [32], even when the single-objective is to minimize the makespan. Therefore, the problem $TOU, 1|on/off, B|E, C_{max}$ is also strongly NP-hard. □

4.2.3 An improved MILP model

By pre-processing the batches of $TOU, 1|on/off, B|E, C_{max}$ with the LPT-method according to Theorem 3, we have $P_b = p_{(b-1)C+1}$ and $|B^*| = \lceil |J|/C \rceil$, a new MILP model, denoted by $\mathcal{P}_f$, can be derived as follows.

$$
\begin{aligned}
\mathcal{P}_f: \quad & \min \ f_1 = E \\
& \min \ f_2 = C_{max} \\
\text{s.t.} \quad & \sum_{i \in I} y_{b,i} = 1, \forall b \in B^* && (4.22) \\
& \sum_{b \in B^*} P_b y_{b,i} \leq S_i z_i, \forall i \in I && (4.23) \\
& z_i \leq \sum_{b \in B^*} y_{b,i}, \forall i \in I && (4.24) \\
& u_i \geq e_i^s (S_i v_i - \sum_{b \in B^*} P_b y_{b,i}), \forall i \in I && (4.25) \\
& \sum_{i \in I} e_i^p \sum_{b \in B^*} P_b y_{b,i} + \sum_{i \in I} e_i^o (z_i - v_{i-1}) + \sum_{i \in I/|I|} u_i \leq E && (4.26) \\
& s_i z_i + \sum_{b \in B^*} P_b y_{b,i} \leq C_{max}, \forall i \in I && (4.27)
\end{aligned}
$$

$$y_{b,i}, z_i, v_i \in \{0,1\}, \forall b \in B^*, \forall i \in I \tag{4.28}$$

$$u_i \geq 0, C_{max} \geq 0, E \geq 0, \forall i \in I \tag{4.29}$$

and constraints (4.10) and (4.11)

where $B^* = \{1, 2, \ldots, |B^*|\}$ is the set of batches formed with the LPT-based method, $P_b = p_{(b-1)C+1}$. Equation (4.22) states that a formed batch $b, \forall b \in B^*$ should be entirely processed in one period. Equation (4.23) ensures that the total processing time in period $i \in I$ can not exceed its duration. Equation (4.24) ensures that variable z_i takes the value of 1 only if there are batches to be processed in period $i, \forall i \in I$. Equation (4.25) denotes the total electricity cost when the machine is left running idle. Equations (4.26) and (4.27) define the total electricity cost E and makespan C_{max}, respectively. Equations (4.28) and (4.29) enforce the restrictions on decision variables. Since part of variables and constraints are removed, the search space for Pareto optimal solutions of the initial problem is significantly reduced. To be more specific, model $\mathcal{P}_f$ reduces $|I| \cdot |J|^2 + (|J| - \lceil |J|/C \rceil) \cdot |I|$ binary variables, $|I| \cdot |J|$ real variables and $(2 + |I| \cdot |J| + |I|) \cdot |J| - (|J| \cdot |I| + |I| + 1) \cdot \lceil |J|/C \rceil$ constraints comparing to model $\mathcal{P}'_f$.

4.3 Resolution approach

In this section, we consider to implement the framework of ε-constraint method for the understudied problem since the method has good performance on the bi-objective problems. That is, $TOU, 1|on/off, B|E, C_{max}$ is solved by being transformed into a series of single objective problems (SOP) and then resolving each SOP. However, for a sequence of SOPs, although commercial optimization softwares, e.g., CPLEX, can solve them optimally, it would be very time-consuming and even impractical for solving medium- and large-size problems due to their strong NP-hard nature. As shown in our computational results in Section 4.4, CPLEX fails to find optimal solution of even one single objective problem for an instance with 40 batches (see set 61 in Table 4.6) within 18000s. Alternatively, to rapidly provide a set of Pareto optimal or near-optimal solutions $\mathcal{A}$, which approximates $\mathcal{F}$, may be desired by decision-makers. Although the KPH-ECM and MKPH-ECM proposed in Chapter 3 are able to generate approximation Pareto fronts, they become slow for larger sized instances since the embedded series of knapsack problems and multiple knapsack problems are also NP-hard, and the problem $TOU, 1|on/off, B|E, C_{max}$ is more complicated

than the $TOU, 1|B|E, C_{max}$ studied in Chapter 3. Consequently, to quickly provide good approximation Pareto front for the considered problem, a *Bin Packing Heuristic based ε-constraint method* (BPH-ECM) is developed in this section.

The BPH-ECM consists of transforming the studied bi-objective problem into a series of single objective problems (SOPs) with the framework of equidistant ε-constraint method, and then each SOP is solved with Bin packing heuristic that inspired by analysing the characteristics of $\mathcal{P}_f$. We can find that one of the key decisions for $TOU, 1|on/off, B|E, C_{max}$ is to determine the period in which each batch will be processed. Such decision can be considered as a bin packing process, where a period (resp. batch) is regarded as a bin (resp. item). With such observation, we propose a heuristic to obtain an optimal or near-optimal solution for each SOP in the BPH-ECM based on the inspiration of assignment rules for bin packing problems and the characteristics of the studied problem. In subsections 4.3.1 and 4.3.2, we detail the framework of ECM and the main steps of Bin packing heuristics, respectively.

4.3.1 Framework of ε-constraint method

The ε-constraint problem for $TOU, 1|on/off, B|E, C_{max}$ can be formulated as follows by selecting total electricity cost minimization as the preferred objective.

$$\begin{aligned} \mathcal{P}_f(\varepsilon^k): \quad & \min E \\ \text{s.t.} \quad & \text{Constraints (4.10), (4.11), (4.22)-(4.29), and } C_{max} \leq \varepsilon^k \end{aligned}$$

where the range of ε^k is determined by the ideal point (E^I, C^I_{max}) and nadir point (E^N, C^N_{max}). The two points are calculated as follows according to Definition 7.

$\mathcal{P}^1_f: E^I = \min E$ s.t. Constraints (4.10), (4.11) and(4.22)-(4.29)
$\mathcal{P}^2_f: C^I_{max} = \min C_{max}$ s.t. Constraints (4.10), (4.11) and(4.22)-(4.29)
$\mathcal{P}^3_f: E^N = \min E$ s.t. Constraints (4.10), (4.11), (4.22)-(4.29) and $C_{max} = C^I_{max}$
$\mathcal{P}^4_f: C^N_{max} = \min C_{max}$ s.t. Constraints (4.10), (4.11),(4.22)-(4.29), and $E = E^I$

From our preliminary results, we found that CPLEX failed to obtain even a feasible solution of any problem of $\mathcal{P}^1_f$ to $\mathcal{P}^4_f$ with 40 batches within 3600s due to its NP-hard nature. Therefore, the BPH-ECM is first to determine the range of ε by heuristically solving the four problems. Let $(E^{I\prime}, C^{I\prime}_{max})$ and $(E^{N\prime}, C^{N\prime}_{max})$ be the obtained objective value vectors.

To determine a set of ε-constraint problems, equidistant ε-constraint method is implemented to provide desired number of solutions for decision makers to avoid overmuch decision information. In this method, the iteration step is first calculated as follows:

$$\delta = (C_{max}^{N\prime} - C_{max}^{I\prime})/K \tag{4.30}$$

where K is a given number of iterations. Then, the value of ε^k, for the k-th ε-constraint problem $\mathcal{P}_f(\varepsilon^k)$ is defined as follows:

$$\varepsilon^k = \varepsilon^{k-1} - \delta,\ k \in \{1, ..., K\} \tag{4.31}$$

where ε^0 is initialized as $C_{max}^{N\prime}$. Thus, the K ε-constraint problems, i.e., $\mathcal{P}_f(\varepsilon^k), k \in \{1, 2, ..., K\}$, are determined.

By observing $\mathcal{P}_f(\varepsilon^k)$, it is not hard to find that all batches have to be completed within the allowable scheduling horizon $[0, \varepsilon^k]$, which is less than the scheduling horizon of the original problem $\mathcal{P}_f$. For each $\mathcal{P}_f(\varepsilon^k)$, a period i is said to be available if its start time is less than ε^k, i.e., $s_i < \varepsilon^k, \forall i \in I$, and the duration of the last period is $\varepsilon^k - s_i$ where i satisfies $s_i \leq \varepsilon^k \leq s_{i+1}$. Let (E^k, C_{max}^k) denote the solution obtained by heuristically solving problem $\mathcal{P}_f(\varepsilon^k)$.

4.3.2 Main steps of the BPH

The bin packing heuristic (BPH) is composed of five steps: 1) batch ordering, 2) period sequencing, 3) batch assignment, 4) batch adjustment, 5) machine turn-on/off determination, which are are detailed as follows, respectively.

1) In the batch ordering step, all the batches are arranged in nonincreasing order of their processing times in a list, i.e., $P_1 \geq P_2 \geq ... \geq P_{|B^*|}$.

2) All available periods are ordered by considering the characteristics of the SOP to be solved. More specifically, i) for the SOPs with the objective of minimizing makespan, all available periods are arranged in chronological order, such that earlier periods are firstly considered to process batches to reduce C_{max}; ii) for the SOPs with the objective of minimizing total electricity cost E, all available periods are arranged in nondecreasing order of their electricity prices, such that available periods with lower prices are preferentially considered to process batches to reduce the E.

3) In the batch assignment step, three fundamental assignment rules: *first fit* (FF), *best fit* (BF) and *best two fit* (B2F), initially used for solving bin packing problems, are respectively adapted in this study to assigning batches in the list to the ordered periods. They are presented as follows:

FF: each batch in the list is sequentially assigned to the first period in which it can be accommodated for processing.

BF: each batch in the list is sequentially assigned to the period with the smallest residual capacity among the periods having sufficient capacities.

B2F: i) consider the first empty period in the order, and allocate the batches in the unassigned list one by one to the current period until no batch can be accommodated; ii) (a) if all batches are assigned in the current period, end; otherwise, (b) check whether the assigned batch with the shortest processing time in the period can be replaced by two unassigned batches with longer total processing time. If yes, the shortest processing time batch is replaced by such two batches with the largest total processing time and it is put back to the unassigned batch list. Go back to i).

After the assignment, let B_i and d_i denote the set of batches processed in period i (batch set i in short) and its total processing time, respectively.

4) The batch adjustment is to further adjust the formed batch sets, i.e., B_i's, with longer total processing times, i.e., d_i's, to periods with higher priorities, since the assignment in 3) might not ensure that a period with higher priority involves a batch set with longer processing time.

5) Finally, we determine the machine turn-on/off strategy according to Case 1-3 in the Section 4.2.

Through the above five steps, the values of all decision variables are determined, thus a feasible solution is obtained. With different assignment rules, the solution may be different. Thus, in the computational results presented in Section 4.4, the solutions with different assignments: FF, BF and B2F, will be evaluated, and the corresponding versions of BPH-ECM are denoted by BPH-ECM$_1$, BPH-ECM$_2$ and BPH-ECM$_3$, respectively.

4.3.3 Overall algorithm

Based on the above statement, the BPH-ECM for $TOU, 1|on/off, B|E, C_{max}$ that described by model $\mathcal{P}_f$ can be outlined as Fig. 4.1.

Moreover, the BPH-ECM can be adapted to solving the original model $\mathcal{P}'_f$. According to Theorem 3, model $\mathcal{P}'_f$ can be equivalently transformed into $\mathcal{P}_f$ by preforming the batches with the LPT-based method. Thus, we can obtain the framework of BPH-ECM for $\mathcal{P}'_f$ after adding a batch formation step before the initialization of Algorithm 4.1.

The BPH-ECM for solving $\mathcal{P}_f$

1: initialize set $\mathcal{A} = \varnothing$ and the value of K;
2: compute points $(E^{I\prime}, C^{I\prime}_{max})$, $(E^{N\prime}, C^{N\prime}_{max})$ by heuristically solving problems $\mathcal{P}^1_f$ to $\mathcal{P}^4_f$;
3: define the iteration step size $\delta = (C^{N\prime}_{max} - C^{I\prime}_{max})/K$;
4: set $\mathcal{A} = \{(E^{I\prime}, C^{N\prime}_{max}), (E^{N\prime}, C^{I\prime}_{max})\}$ and let $\varepsilon^1 = C^{N\prime}_{max} - \delta$;
5: **for** $k = 1; k < K$ **do**
6: obtain ε^k and define problem $\mathcal{P}(\varepsilon^k)$;
7: obtain (E^k, C^k_{max}) by heuristically solving $\mathcal{P}(\varepsilon^k)$;
8: add (E^k, C^k_{max}) to $\mathcal{A}$ if the solution is feasible;
9: let $\varepsilon^{k+1} = \varepsilon^k - \delta$;
10: **end for**
11: remove dominated points if existing and derive final set $\mathcal{A}$.

Fig. 4.1: The BPH-ECM for solving $\mathcal{P}_f$

4.4 Computational results

In this section, computational results of 335 randomly generated instances (67 sets $\times$ 5 instances) are reported to evaluate the performance of the proposed methods. All algorithms were coded in C++. To evaluate their effectiveness, the set of points $\mathcal{A}$ obtained by the BPH-ECM is compared with a reference set, denoted as $\mathcal{R}$, obtained by CPLEX (Version 12.6). To be more specific, $\mathcal{R}$ are obtained by Algorithm 4.1, where each single-objective optimization problem is solved by CPLEX instead of heuristics. The solution quality of the BPH-ECMs are evaluated by *overall non-dominated solutions* (Q), *e-dominance indicator* (D) and *hypervolume ratio* (H), which respectively reflect the cardinality-, distance- and volume-based PIs. Besides, the efficiency of the BPH-ECM in terms of computational time (CPU seconds) is compared with that shown by CPLEX. For each instance, the total computational time and the computational time for each single-objective problem are limited to 18000s and 3600s, respectively. All experiments were conducted on a PC with 1.7 GHz processor and 3.12 GB RAM under windows 7 operating system.

All test instances are randomly generated in a similar way to those in Chapter 3. The processing times $p_j, \forall i \in J$, are randomly and uniformly generated from (100, 200] and (50, 100] respectively. Batch capacity C is set to be 10. Under the work shift, the period duration T is set to be 480 min, which reflects one of the three work shifts in a workday, i.e., 8×60 min. The number of periods $|I|$ is generated as

$|I| = \alpha_{|I|} max_{j \in J}\{p_j \times |B^*|/T\}$, where $\alpha_{|I|}$ is randomly and uniformly generated in the interval [0.6, 1.0]. P_{idle}, P_{proc}, and P_{on} are set to be 0.1 kWh/min, 0.3 kWh/min, 0.5 kWh [109]. The electricity price of each period is randomly generated from the uniform distribution on the interval of [1,7] ¢/kWh [55]. Both in the ε-constraint method implemented by CPLEX and BPH-ECM, K is set to be 50 in this study. The computational results are reported in Tables 4.1-4.6, in which each value is its average value of five instances. In the tables, $\mathcal{P}_f$ and $\mathcal{P}'_f$ denote the results of two models solved by CPLEX, respectively. Q, H and D (resp. CT) denote the performance metrics of solution quality (resp. computational time).

Table 4.1 reports the computational results of the small-size instances that solved with model $\mathcal{P}_f$. By comparing the computational results of BPH-ECMs with that of CPLEX, we can see that the number of Pareto solutions found are similar for the BPH-ECMs and CPLEX, and both the hypervolume ratio and average e-dominance indicators of the three BPH-ECMs are equal to or close to 1. These results indicate that the solution sets of BPH-ECMs are very close to $\mathcal{R}$. Especially, BPH-ECM$_3$ yields the same set of solutions with CPLEX. In terms of computational time, we can observe from Table 4.1 that CT of the BPH-ECMs is far less than that of CPLEX. This confirms the efficiency of the proposed methods. Moreover, BPH-ECM$_3$ yields the best performance among the BPH-ECMs in terms of solution quality. This may be because B2F assignment rule performs better than FF and BF for solving bin packing problems.

Table 4.2 presents the computational results of the instance sets 1-16 solved with model $\mathcal{P}'_f$. From the table, we can obtain similar conclusion as Table 4.1. Specifically, BPH-ECMs can generate good quality solutions that very close to $\mathcal{R}$, and they are much more efficient than CPLEX. From the comparison results of $\mathcal{P}_f$ and $\mathcal{P}'_f$ solved by CPLEX in the two tables, we can see that the obtained points by both models are the same. This implies both models can yield solutions of same quality over the small-size problem sets 1-16. However, we can find that CT of $\mathcal{P}_f$ is far less than that of $\mathcal{P}'_f$ on each problem set, which indicates that the improved model $\mathcal{P}_f$ is much more efficient than model $\mathcal{P}'_f$ in terms of computational time. This can be attributed to the derived properties that significantly reduce the search space. In the following, the computational results of $\mathcal{P}_f$ solved by CPLEX are taken as the reference set $\mathcal{R}$ to evaluate the performance of the BPH-ECMs.

Table 4.1: Comparison results with model $\mathcal{P}_f$ for small-size instances

Set	$\|J\|$	$\|I\|$	$\mathcal{P}_f$		BPH-ECM$_1$				BPH-ECM$_2$				BPH-ECM$_3$			
			Q	CT	Q	H	D	CT	Q	H	D	CT	Q	H	D	CT
1	12	2	4.0	1.195	4.0	1.000	1.000	0.234	4.0	1.000	1.000	0.231	4.0	1.000	1.000	0.223
2	14	2	4.0	1.179	4.0	1.000	1.000	0.194	4.0	1.000	1.000	0.229	4.0	1.000	1.000	0.235
3	16	2	4.0	1.217	4.0	1.000	1.000	0.208	4.0	1.000	1.000	0.230	4.0	1.000	1.000	0.236
4	18	2	4.0	1.285	4.2	1.000	1.000	0.219	4.0	1.000	1.000	0.228	4.0	1.000	1.000	0.207
5	20	2	4.0	1.226	4.0	1.000	1.000	0.227	4.0	1.000	1.000	0.234	4.0	1.000	1.000	0.272
6	22	2	8.0	2.103	7.4	1.000	1.008	0.231	7.0	0.922	1.016	0.245	8.0	1.000	1.000	0.261
7	24	2	8.0	2.028	7.0	1.000	1.007	0.275	6.8	0.921	1.015	0.248	8.0	1.000	1.000	0.239
8	26	2	7.2	2.056	6.2	1.000	1.006	0.303	6.2	0.910	1.011	0.257	7.2	1.000	1.000	0.250
Average			5.4	1.536	5.1	1.000	1.003	0.236	5.0	0.969	1.005	0.238	5.4	1.000	1.000	0.240
9	12	3	7.0	1.753	7.0	1.000	1.000	0.216	7.0	1.000	1.000	0.240	7.0	1.000	1.000	0.255
10	14	3	7.0	1.866	7.0	1.000	1.000	0.224	7.0	1.000	1.000	0.301	7.0	1.000	1.000	0.248
11	16	3	7.0	1.909	7.0	1.000	1.000	0.202	7.0	1.000	1.000	0.251	7.0	1.000	1.000	0.244
12	18	3	7.0	1.906	7.2	1.000	1.000	0.209	7.0	1.000	1.000	0.246	7.0	1.000	1.000	0.214
13	20	3	7.0	1.962	7.0	1.000	1.000	0.272	7.0	1.000	1.000	0.241	7.0	1.000	1.000	0.230
14	22	3	15.0	2.505	13.0	1.000	1.007	0.225	13.0	0.971	1.012	0.218	15.0	1.000	1.000	0.275
15	24	3	13.2	2.643	11.4	1.000	1.005	0.241	10.8	0.970	1.008	0.283	13.2	1.000	1.000	0.268
16	26	3	14.4	2.652	12.4	1.000	1.006	0.291	11.6	0.966	1.012	0.262	14.4	1.000	1.000	0.286
Average			9.7	2.150	9.0	1.000	1.002	0.235	8.8	0.988	1.004	0.255	9.7	1.000	1.000	0.253

Table 4.2: Comparison results with model $\mathcal{P}'_f$ for small-size instances

$\|J\|$	$\|I\|$	$\mathcal{P}'_f$		BPH-ECM$_1$				BPH-ECM$_2$				BPH-ECM$_3$			
		Q	CT	Q	H	D	CT	Q	H	D	CT	Q	H	D	CT
12	2	4.0	15.672	4.0	1.000	1.000	0.238	4.0	1.000	1.000	0.234	4.0	1.000	1.000	0.222
14	2	4.0	19.326	4.0	1.000	1.000	0.200	4.0	1.000	1.000	0.230	4.0	1.000	1.000	0.239
16	2	4.0	22.937	4.0	1.000	1.000	0.210	4.0	1.000	1.000	0.235	4.0	1.000	1.000	0.238
18	2	4.0	26.053	4.2	1.000	1.000	0.219	4.0	1.000	1.000	0.230	4.0	1.000	1.000	0.209
20	2	4.0	29.904	4.0	1.000	1.000	0.230	4.0	1.000	1.000	0.237	4.0	1.000	1.000	0.276
22	2	8.0	519.680	7.4	1.000	1.008	0.237	7.0	0.922	1.016	0.247	8.0	1.000	1.000	0.263
24	2	8.0	603.973	7.0	1.000	1.007	0.282	6.8	0.921	1.015	0.251	8.0	1.000	1.000	0.241
26	2	7.2	1596.231	6.2	1.000	1.006	0.309	6.2	0.910	1.011	0.260	7.2	1.000	1.000	0.253
		5.4	354.222	5.1	1.000	1.003	0.241	5.0	0.969	1.005	0.241	5.4	1.000	1.000	0.243
12	3	7.0	17.436	7.0	1.000	1.000	0.218	7.0	1.000	1.000	0.241	7.0	1.000	1.000	0.254
14	3	7.0	21.891	7.0	1.000	1.000	0.230	7.0	1.000	1.000	0.305	7.0	1.000	1.000	0.250
16	3	7.0	26.643	7.0	1.000	1.000	0.206	7.0	1.000	1.000	0.270	7.0	1.000	1.000	0.247
18	3	7.0	30.769	7.2	1.000	1.000	0.211	7.0	1.000	1.000	0.245	7.0	1.000	1.000	0.217
20	3	7.0	36.328	7.0	1.000	1.000	0.275	7.0	1.000	1.000	0.243	7.0	1.000	1.000	0.229
22	3	15.0	2796.540	13.0	1.000	1.007	0.226	13.0	0.971	1.012	0.220	15.0	1.000	1.000	0.278
24	3	13.2	3397.289	11.4	1.000	1.005	0.240	10.8	0.970	1.008	0.285	13.2	1.000	1.000	0.266
26	3	14.4	12053.474	12.4	1.000	1.006	0.295	11.6	0.966	1.012	0.267	14.4	1.000	1.000	0.288
		9.7	2297.546	9.0	1.000	1.002	0.238	8.8	0.988	1.004	0.260	9.7	1.000	1.000	0.254

Table 4.3: Comparison results for the instances with $p_j \in (50, 100]$ and $|B^*|$ =5-70

Set	$\|B^*\|$	$\|I\|$	$\mathcal{P}_f$		BPH-ECM$_1$				BPH-ECM$_2$				BPH-ECM$_3$			
			Q	CT	Q	H	D	CT	Q	H	D	CT	Q	H	D	CT
17	5	1.2	5.2	0.668	3.4	0.956	1.002	0.154	3.4	0.956	1.002	0.137	4.6	0.984	1.001	0.125
18	10	2.2	49.4	5.594	16.8	0.928	1.012	0.328	16.6	0.934	1.012	0.365	27.4	0.974	1.008	0.381
19	15	3.0	50.0	18.452	25.6	0.838	1.015	0.331	25.6	0.838	1.015	0.387	42.2	0.950	1.007	0.416
20	20	4.0	49.8	72.625	23.0	0.738	1.022	0.347	23.0	0.738	1.007	0.321	44.2	0.915	1.009	0.500
21	25	4.2	26.0	63.742	6.8	0.883	1.039	0.309	6.8	0.883	1.039	0.407	14.8	0.952	1.013	0.428
22	30	5.0	50.0	129.231	15.8	0.582	1.007	0.350	15.6	0.582	1.007	0.508	23.6	0.708	1.005	0.528
23	35	7.0	47.2	3949.350	27.8	0.848	1.015	0.436	27.6	0.849	1.014	0.557	45.8	0.964	1.003	0.672
24	40	7.4	49.2	1415.930	25.6	0.802	1.013	0.573	25.8	0.805	1.013	0.579	47.2	0.943	1.004	0.754
25	45	8.6	44.2	13457.900	27.0	0.897	1.011	0.634	27.0	0.897	1.011	0.703	48.2	1.001	1.004	0.829
26	50	9.4	37.0	14717.000	28.0	0.954	1.011	0.699	28.0	0.954	1.011	0.782	47.4	1.077	1.003	0.966
27	55	9.8	43.6	8830.760	30.2	0.833	1.012	0.719	30.0	0.833	1.012	0.811	45.4	0.969	1.003	0.947
28	60	11.4	36.0	13753.800	27.6	0.879	1.010	0.796	27.2	0.881	1.010	0.935	45.2	1.003	1.004	1.062
29	65	12.0	32.8	13515.200	30.8	0.962	1.012	0.818	30.8	0.962	1.012	1.076	46.0	1.125	1.003	1.158
30	70	13.2	18.6	18000.000	29.8	1.434	1.011	0.991	29.8	1.434	1.011	1.088	44.2	1.744	1.003	1.275
Average			38.5	6280.732	22.7	0.895	1.014	0.535	22.7	0.896	1.013	0.618	37.6	1.022	1.005	0.717

In addition, by comparing the results of $\mathcal{P}'_f$ and $\mathcal{P}_f$ solved by BPH-ECMs in the two tables, it can be found that Q, D, and H of $\mathcal{P}'_f$ are equal to that of $\mathcal{P}_f$. This indicates that the BPH-ECMs can obtain the same solutions for the two models. While computational time for $\mathcal{P}'_f$ is a little bit longer than that for $\mathcal{P}_f$. This is due to the computational time for sorting the jobs with LPT-based method and obtaining the values of batch processing times. Since the computational results of BPH-ECMs for solving model $\mathcal{P}_f$ and $\mathcal{P}'_f$ are very close, we only present the former in the following tables.

According to our preliminary experiments, we find that the computational complexity of the studied problem is affected by the instance scale as well as the length of job processing time. In what follows, we test larger sized instances with two generation schemes of job processing time times to examine the performance of the proposed methods. Tables 4.3 and 4.4 show the results for the instances with job processing time $p_j \in (50, 100]$, and Tables 4.5 and 4.6 illustrate the results for the instances with $p_j \in (100, 200]$. In addition, we use the number of batches $|B^*|$ instead of the number of jobs $|J|$ in the following tables, since the batches are pre-formed with LPT-based method.

We can see from Table 4.3 that in terms of cardinality, BPH-ECM$_1$ and BPH-ECM$_2$ performs a bit worse than CPLEX but BPH-ECM$_3$ obtains almost the same number of points with CPLEX on average. The hypervolume ratios are close to or exceed 90% and the average e-dominance indicators are slightly greater than 1, which implies the solution sets obtained by the proposed methods are very near to $\mathcal{R}$. It is worth pointing out that BPH-ECM$_1$ and BPH-ECM$_2$ derive more solutions and higher hypervolume ratio than CPLEX for problem set 30, and BPH-ECM$_3$ achieves better performance in terms of the two indicators than CPLEX over sets 25-30. These results show that the proposed BPH-ECMs can obtain good-quality solutions. In addition, we can also find that BPH-ECM$_3$ exhibits the best performance in the three BPH-ECMs. This may be because the procedure of replacing the assigned batch with two proper ones in B2F is effective to improve the solution quality. Moreover, Q, H and D of BPH-ECM$_1$ are respectively very close to that of BPH-ECM$_2$ in the same set, this indicates that BPH-ECM$_1$ and BPH-ECM$_2$ obtain roughly equivalent performance. On the other hand, we can observe that computational time of $\mathcal{P}_f$ solved by CPLEX varies from 0.668s to 18000s with its average value being 6280.732s, which increases 26946 times (18000/0.668) from set 17 to 30. However, CT of BPH-ECMs varies between 0.125s and 1.275s with its average value being less than 1s. This demonstrates that the proposed methods significantly outperform CPLEX in terms

of computational time. Moreover, Figure 4.2 provides an graphical comparison of the BPH-ECMs, we can find that computational time of BPH-ECM$_1$ is almost less than those of BPH-ECM$_2$ and BPH-ECM$_3$, which indicates that BPH-ECM$_1$ performs best in computational efficiency.

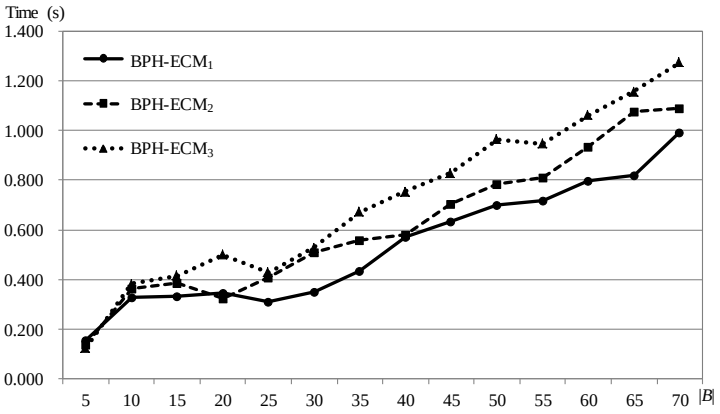


Fig. 4.2: Computational time of BPH-ECMs for the instances with $p_j \in (50, 100]$, $|B^*|$=5-70

Table 4.4 presents the results for the instances with $p_j \in (50, 100]$, $|B^*|$ and $|I|$ increasing from 80 to 5000 and 16.8 to 896.4, respectively. It can be observed from Table 4.4 that CPLEX almost loses power to solve the large-size problem instance sets due to the strong NP-hardness of the problem. It can only obtain very few Pareto solutions for the smallest-size sets 31 and 32 and fails to find even one Pareto solution for the rest larger-size instances, whereas the BPH-ECMs can obtain considerable Pareto solutions for all problem sets within 5 minutes. Moreover, CT of BPH-ECMs varies very slightly with the problem size, which suggest the proposed methods are relatively stable. This shows the effectiveness and efficiency of the proposed methods in solving large-size problems. Furthermore, it can be seen that numbers of non-dominated solutions Q found by BPH-ECM$_1$ and BPH-ECM$_2$ is slightly less than Q of BPH-ECM$_3$, which once again indicates that BPH-ECM$_3$ yields the best performance in terms of solution quality. While its computational time increases faster than those of BPH-ECM$_1$ and BPH-ECM$_2$ (see Figure 4.3), which shows that BPH-ECM$_1$ and BPH-ECM$_2$ perform better in terms of computational time.

In Table 4.5, we report the results for the instances with $p_j \in (100, 200]$ and $|B^*|$ varying from 5 to 20. From the table, we can find similar results to those reported in Table 4.3. BPH-ECM$_3$ performs better than BPH-ECM$_1$ and BPH-ECM$_2$, with 29.1 Pareto solutions (slightly less than that of CPLEX), H being greater than 0.951

Table 4.4: Comparison results for the instances with $p_j \in (50, 100]$ and $|B^*|$=80-5000

Set	$\|B^*\|$	$\|I\|$	$\mathcal{P}_f$		BPH-ECM$_1$		BPH-ECM$_2$		BPH-ECM$_3$	
			Q	CT	Q	CT	Q	CT	Q	CT
31	80	16.8	5.2	18000.0	33.8	1.147	34.2	1.158	34.8	1.394
32	100	19.8	5.2	8086.0	34.8	1.308	34.8	1.490	38.0	1.653
33	120	21.2	-	-	30.6	1.559	30.4	1.642	39.0	1.894
34	150	26.2	-	-	31.4	1.966	30.6	2.307	43.6	2.384
35	200	36.4	-	-	37.6	2.694	37.4	2.938	36.2	3.313
36	500	89.0	-	-	35.2	5.039	35.0	5.990	35.0	8.453
37	1000	174.4	-	-	38.8	11.384	38.6	14.032	33.0	18.619
38	2000	384.0	-	-	35.6	27.409	34.2	35.294	41.0	49.078
39	3000	537.4	-	-	35.4	38.506	35.0	46.984	38.6	120.412
40	4000	699.6	-	-	35.2	57.325	35.2	69.468	39.2	199.197
41	5000	896.4	-	-	34.8	70.438	34.6	98.092	38.8	289.675
Average			-	-	34.8	19.889	34.5	25.400	37.9	63.279

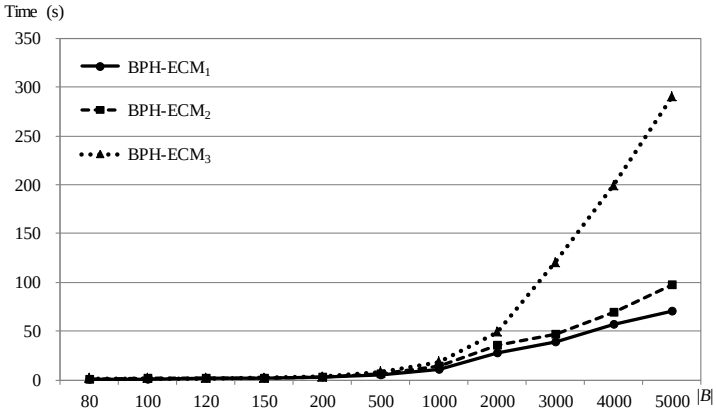


Fig. 4.3: Computational time of BPH-ECMs for instances with $p_j \in (50, 100]$, $|B^*|$=80-5000

and D being 1.006 on average. Although the cardinalities and hypervolume ratios of BPH-ECM$_1$ and BPH-ECM$_2$ are not as good as BPH-ECM$_3$, both values of D and H are slightly greater than 1, which implies the approximated Pareto fronts derived by BPH-ECM$_1$ and BPH-ECM$_2$ are also close to $\mathcal{R}$. In addition, it can be seen in Table 4.5 that CT of BPH-ECMs is far less than that of CPLEX over all sets 42-57. Moreover, CT of CPLEX exponentially increases with the problem size, whereas that of BPH-ECMs varies very slightly. This shows the remarkable advantage of BPH-ECMs in terms of computational time compared with CPLEX.

Table 4.5: Comparison results for the instances with $p_j \in (100, 200]$ and $|B^*|$=5-20

Set	$\|B^*\|$	$\|I\|$	$\mathcal{P}_f$		BPH-ECM$_1$				BPH-ECM$_2$				BPH-ECM$_3$			
			Q	CT	Q	H	D	CT	Q	H	D	CT	Q	H	D	CT
42	5	2.0	11.4	3.204	2.0	0.723	1.051	0.278	2.0	0.723	1.051	0.279	7.4	0.834	1.008	0.324
43	6	2.6	14.4	3.900	8.0	0.911	1.023	0.244	8.0	0.894	1.023	0.284	10.6	0.952	1.006	0.368
44	7	3.0	29.6	6.287	8.0	0.790	1.037	0.362	8.0	0.789	1.037	0.385	20.6	0.936	1.005	0.435
45	8	3.0	36.0	7.775	4.0	0.640	1.037	0.353	4.0	0.640	1.037	0.392	18.0	0.940	1.005	0.450
46	9	3.8	31.4	10.190	12.6	0.753	1.034	0.350	12.6	0.754	1.034	0.399	23.0	0.939	1.008	0.456
47	10	4.0	39.6	13.241	10.2	0.691	1.040	0.389	10.4	0.692	1.040	0.473	26.6	0.931	1.006	0.498
48	11	4.2	44.8	63.405	8.4	0.627	1.036	0.403	7.8	0.609	1.038	0.466	25.4	0.944	1.005	0.474
49	12	4.6	38.0	59.699	11.6	0.663	1.025	0.397	11.8	0.664	1.025	0.472	23.0	0.919	1.006	0.488
50	13	5.0	47.2	67.483	13.6	0.536	1.041	0.452	13.4	0.535	1.041	0.480	37.2	0.901	1.007	0.526
51	14	5.2	47.8	98.281	8.8	0.677	1.029	0.438	8.6	0.677	1.029	0.481	30.4	0.930	1.005	0.587
52	15	6.0	40.6	260.341	20.2	0.637	1.035	0.415	19.4	0.634	1.035	0.476	37.0	0.971	1.003	0.569
53	16	6.6	45.8	767.338	19.0	0.663	1.032	0.466	18.8	0.659	1.033	0.476	41.8	0.956	1.004	0.603
54	17	6.6	45.6	2840.280	17.0	0.657	1.030	0.465	16.6	0.657	1.030	0.650	40.2	0.954	1.006	0.683
55	18	7.0	40.8	9306.510	21.2	0.684	1.026	0.507	20.2	0.683	1.026	0.608	40.0	0.986	1.005	0.672
56	19	7.6	43.6	10362.500	19.0	0.631	1.036	0.562	18.8	0.631	1.036	0.611	43.2	0.988	1.006	0.694
57	20	7.8	26.8	15616.600	18.2	0.713	1.034	0.613	18.2	0.684	1.036	0.690	40.4	1.141	1.003	0.772
Average			36.5	2467.940	12.6	0.687	1.034	0.418	12.4	0.683	1.034	0.476	29.1	0.951	1.006	0.537

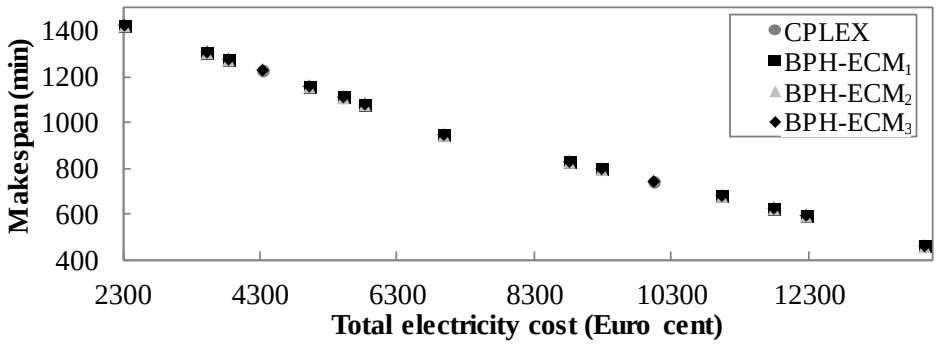
Table 4.6 reports the computational results for the larger sized instances with $p_j \in (100, 200]$ and $|B^*|$ (resp. $|I|$) varying from 21 to 2000 (resp. 8.0 to 799.6). Here, we kept the two generation schemes with approximately equivalent scheduling horizon and test the instances with up to 2000 batches. From Table 4.6, we can find that CPLEX almost loses power to solve these large-size instances within 18000s but the BPH-ECMs can obtain a set of Pareto solutions for all instances within 80 seconds. Furthermore, the BPH-ECM$_3$ still performs best in solution quality.

Table 4.6: Comparison results for the instances with $p_j \in (100, 200]$ and $|B^*|$=21-2000

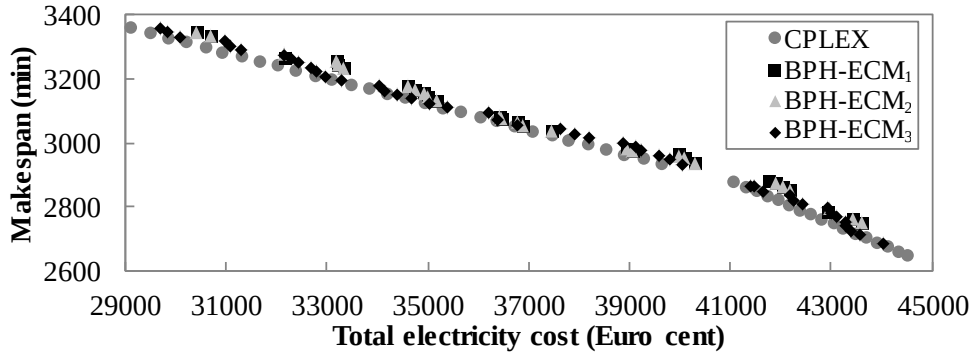
Set	$\lvert B^*\rvert$	$\lvert I\rvert$	$\mathcal{P}_f$		BPH-ECM$_1$		BPH-ECM$_2$		BPH-ECM$_3$	
			Q	CT	Q	CT	Q	CT	Q	CT
58	21	8.0	19.8	17160.7	16.2	0.563	16.6	0.601	36.6	0.828
59	25	9.2	5.8	18000	17.0	0.550	16.4	0.625	35.0	0.850
60	30	10.6	3.4	18000	18.8	0.607	19.0	0.636	40.0	0.963
61	40	16.2	-	-	23.2	0.988	23.6	1.056	35.2	1.325
62	50	19.4	-	-	24.2	1.046	24.6	1.159	38.4	1.544
63	100	39.6	-	-	29.8	2.358	29.8	2.623	36.0	2.991
64	200	75.0	-	-	34.6	4.794	34.6	5.768	35.8	5.900
65	500	178.6	-	-	28.6	9.697	28.6	11.573	35.2	15.372
66	1000	392.4	-	-	32.2	25.966	34.6	28.944	40.6	31.788
67	2000	799.6	-	-	28.4	44.116	28.2	45.489	45.8	79.144
Average			-	-	25.3	9.068	25.6	9.847	37.9	14.070

By comparing the results of the BPH-ECMs for the instances with different generation schemes, we can observe that BPH-ECMs perform better for the instances with less job processing time in terms of both solution quality and solution time. Taking the average values in Tables 4.3 and 4.5 for example, we can see from Q, H and D of the BPH-ECMs that the Pareto solutions found in the former table are more and closer to $\mathcal{R}$ than that in the latter one. For the same batch number instances, we can also observe that BPH-ECMs consume less time for the instances with less job processing time, such as sets 19 and 52, sets 24 and 61, sets 38 and 67.

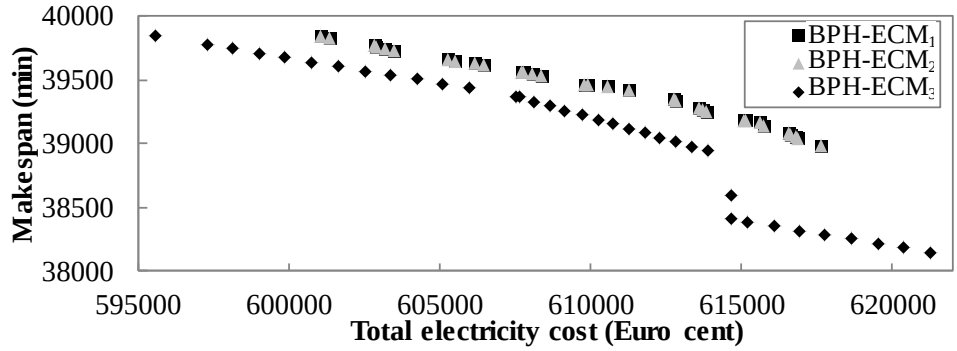
To facilitate the visualization of the solution quality comparison results, the fronts generated by CPLEX and BPH-ECMs for different scale instances are illustrated in Figure 4.4. Since as many as 67 different scales are involved in the study, we only present the graphical results of three instances from sets 14, 20 and 36, respectively. They are represented as small-, medium- and large-size instances. As shown in Figure 4.4(a) and (b), the BPH-ECMs can obtain good quality solutions that are very close to $\mathcal{R}$. Meanwhile, BPH-ECM$_3$ achieves the best performance among the BPH-ECMs in solution quality. From these figures, we can find a clear tradeoff between the total electricity cost and the makespan. Such tradeoff can provide some decision-making basis for manufactures to balance the electricity cost and production efficiency.



(a) small-size instance (22 jobs with $p_j \in (50,100]$)



(b) medium-size instance (200 jobs with $p_j \in (50,100]$)



(c) large-size instance (5000 jobs with $p_j \in (50,100]$)

Fig. 4.4: Dicovered Pareto solutions by different methods

Summing up, (1) model $\mathcal{P}_f$ is much more CPLEX-effective than model $\mathcal{P}'_f$; (2) the three BPH-ECMs, taking within 80s for any given instance, is much more efficient and stable than CPLEX in terms of running time overall instance sets; (3) the three BPH-ECMs outperform CPLEX in terms of solution quality on large-size instances; (4) the three BPH-ECMs perform better in the scenario with less job processing time and BPH-ECM$_3$ perfoms more stable than the other two methods in different scenarios; (5) BPH-ECM$_3$ has the best solution quality among the three methods; (6) BPH-ECM$_1$ and BPH-ECM$_2$ have less running time than BPH-ECM$_3$. Furthermore,

since the BPH-ECMs all are constructive, they are very easy to implement.

4.5 Conclusions

In this chapter, we have addressed a new bi-criteria single-machine batch scheduling problem with machine on/off switching under TOU electricity prices, which aims to minimize the TEC and makespan simultaneously. A bi-objective MILP model is presented for the problem. Then, an improved model is derived based on optimal batch rule analysis, with which the search space for Pareto optimal solutions is greatly reduced. The problem is demonstrated to be strongly NP-hard. To efficiently solve the problem, especially for medium- and large-size problem instances, a heuristic based ε-constraint method is devised. Computational results show that the improved model is much more efficient and that the proposed method can effectively and efficiently solve instances with up to 5000 batches and 896 periods within relatively short time. The corresponding work has been published in the following paper.

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. *Computers & Industrial Engineering*, 112:721–734, 2017.

Chapter 5

Parallel machine batch scheduling under TOU tariffs

5.1 Introduction

Parallel batch processing systems representing a typical production environment are extensively encountered in manufacturing industry [150]. As reviewed in Chapter 2, parallel batch machine scheduling problems, which are basis of studying more complicated shop scheduling problems, have been widely investigated over the past decades. In realistic parallel batch machine systems, old manual machines and advanced new machines are often simultaneously used by the manufacturers to reduce investment. This leads to the different processing speed and energy consumption [98]. Under time-of-use tariffs, designing effective scheduling schemes for parallel batch machine systems to save energy cost can be of huge theoretical and practical significance.

Moreover, the problems studied in Chapters 3 and 4 both assumed each job has to be completed in one work shift. However, continuously processing manufacturing systems are more widespread and practical in real production environments. In this Chapter, we consider a parallel batch scheduling problem where a nonpreemptive job is allowed to be processed in multiple periods. The problem is motivated by the scheduling challenges of ceramization operations in glass manufacturing, which also involves different sized jobs [138]. Consequently, jobs characteristic with non-identical sizes is incorporated to the considered problem.

In a parallel batch machine system, less number of enabled machines can help to cut down material and labour resources investment. However, more enabled machines may reduce total electricity cost under TOU tariffs, since more jobs can be processed during the off-peak periods on the different machines. So there is a trade-off between total electricity cost and number of enabled machines, which may be desired by the

decision makers. According to the literature, this type of problem has not been investigated in literature.

The work in this chapter investigates a bi-objective uniform parallel batch processing machine scheduling problem with non-identical job size under time-of-use tariffs (PBMS-TOU). The objectives are to simultaneously minimize total electricity cost and number of enabled machines. The problem consists of batching non-identical sized jobs and allocating formed batches to parallel machines on the variable electricity pricing horizon, such that the total electricity cost and number of enabled machines are minimized. For the problem, we firstly establish a mixed-integer linear programming model. To efficiently obtain the approximation Pareto front, we decompose the original problem into two subproblems: batch formation and batch allocation. Then a two-stage heuristic approach is designed, where the first (resp. second) stage focuses on batch formation (resp. allocation).

The remainder of the chapter is organized as follows. Section 5.2 describes and mathematically formulates the problem and demonstrates its complexity. In Section 5.3, a two-stage heuristic approach is proposed. Computational results on a case study and random generated instances are reported in Section 5.4. Section 5.5 concludes this work.

5.2 Problem formulation and complexity

The bi-objective uniform parallel batch scheduling problem with non-identical job sizes under time-of-use tariffs (PBMS-TOU) can be expressed as $TOU, Q|c_j, B|E, N$, where N denote the number of enabled machines. The scheduling horizon can be divided into $|I|$ time periods according to electricity prices and each period i is associated with a starting time s_i, a duration S_i and a unit electricity cost e_i, where $S_i = s_{i+1} - s_i$. The unit electricity costs of two adjacent periods are different.

A set of jobs J with job j characterized by processing time p_j and size c_j is to be processed on a set of available uniform batch processing machines M within the $|I|$ periods. The processing of each job cannot be interrupted. All the jobs can be grouped to $|B|$ batches (to be determined) and the capacity of each batch is C. The processing time of batch P_b is determined by the longest processing time job in the batch. Machine m with processing speed v_m consumes amount of q_m energy per unit time. The processing time of job j on machine m is $p_{j,m} = p_j/v_m$, and the processing time of batch b on machine m is $P_{b,m} = \max\{p_{j,m}|j \in b\} = \max\{p_j|j \in b\}/v_m$.

The objectives are to optimize the total electricity cost and number of the enabled machines, where the electricity cost is determined by the real-time power rate of the machine and the TOU prices. The number of the enabled machines is the used machines for completing all jobs.

The scheduling of the problem needs to determine job assignment to batches, batch allocation to machines and their orders on corresponding machines, such that both electricity cost and number of enabled machines are simultaneously optimized.

5.2.1 Mathematical Modelling

To formulate the problem, we first define the following notations.

Indices:

j: index of jobs;

b: index of batches;

m: index of machines;

i: index of periods.

Parameters:

J: set of all jobs, i.e., $J = \{1, 2, ..., |J|\}$;

B: set of all batches, i.e., $B = \{1, 2, ..., |B|\}$;

M: set of all machines, i.e., $M = \{1, 2, ..., |M|\}$;

I: set of all periods on the planning horizon, i.e., $I = \{1, 2, ..., |I|\}$;

p_j: processing time of job j, $\forall j \in J$;

c_j: size of job j, $\forall j \in J$;

C: capacity of each batch;

v_m: processing speed of machine m, $\forall m \in M$;

q_m: power rate of machine m when processing, $\forall m \in M$;

e_i: electricity cost of period i, $\forall i \in I$;

s_i: starting time of period i, $1 \leq i \leq |I| + 1$;

S_i: duration of period i, $\forall i \in I$;

L: a sufficiently large integer.

Decision variables:

$|B|$: number of formed batches;

$x_{j,b}$: equal to 1 if job j is processed in batch b (or to say, job j is assigned to batch b), 0 otherwise, $\forall j \in J, \forall b \in B$;

y_b: equal to 1 if batch b is formed (or to say, batch b is not empty), 0 otherwise;

$z_{b,m,i}$: equal to 1 if (any part of) batch b is processed on machine m in period i, 0 otherwise;

$r_{b,m}$: equal to 1 if batch b is processed on machine m, 0 otherwise;

$t_{b,m,i}$: processing time of batch b on machine m in period i;

$w_{b,m,i}$: equal to 1 if batch b is processed on machine m in both periods i and $i+1$, 0 otherwise;

u_m: equal to 1 if any batch is processed on machine m, 0 otherwise;

P_b: the processing time of batch b;

E: total energy cost for completing all jobs;

N: number of enabled machines.

Based on the above description and notations, problem PBMS-TOU can be formulated as the following mixed-integer linear programming(MILP) model $\mathcal{P}'_Q$.

$$\mathcal{P}'_Q : \min f_1 = E \tag{5.1}$$

$$\min f_2 = N \tag{5.2}$$

$$s.t. \sum_{b \in B} x_{j,b} = 1, \ \forall j \in J \tag{5.3}$$

$$P_b \geq p_j x_{j,b}, \ \forall j \in J, \ \forall b \in B \tag{5.4}$$

$$\sum_{j \in J} x_{j,b} c_j \leq C y_b, \ \forall b \in B \tag{5.5}$$

$$t_{b,m,i} \leq z_{b,m,i} L, \ \forall b \in B, \ \forall m \in M, \ \forall i \in I \tag{5.6}$$

$$r_{b,m} \geq z_{b,m,i}, \ \forall b \in B, \ \forall m \in M, \ \forall i \in I \tag{5.7}$$

$$\sum_{m \in M} r_{b,m} = 1, \ \forall b \in B \tag{5.8}$$

$$\sum_{i \in I} t_{b,m,i} \geq P_b / v_m - (1 - r_{b,m}) * L, \ \forall b \in B, \ \forall m \in M \tag{5.9}$$

$$\sum_{b \in B} t_{b,m,i} \leq S_i, \ \forall m \in M, \ \forall i \in I \tag{5.10}$$

$$w_{b,m,i} \geq z_{b,m,i} + z_{b,m,i+1} - 1, \ \forall b \in B, \ \forall m \in M, \ 1 \leq i \leq |I| - 1 \tag{5.11}$$

$$2 * w_{b,m,i} \leq z_{b,m,i} + z_{b,m,i+1}, \ \forall b \in B, \ \forall m \in M, \ 1 \leq i \leq |I| - 1 \tag{5.12}$$

$$\sum_{b \in B} w_{b,m,i} \leq 1, \ \forall m \in M, \ 1 \leq i \leq |I| - 1 \tag{5.13}$$

$$t_{b,m,i} + t_{b,m,i'} + s_{i'} z_{b,m,i'} - s_{i+1} z_{b,m,i} \leq P_b + (2 - z_{b,m,i} - z_{b,m,i'}) L,$$

$$\forall m \in M, \forall b \in B, 1 \leq i < i' \leq |I| \tag{5.14}$$

$$u_m \geq z_{b,m,i},\ \forall b \in B,\ \forall m \in M,\ \forall i \in I \tag{5.15}$$

$$E \geq \sum_{i \in I} \sum_{m \in M} \sum_{b \in B} q_m e_i t_{b,m,i} \tag{5.16}$$

$$N \geq \sum_{m \in M} u_m \tag{5.17}$$

$$x_{j,b}, y_b, z_{b,m,i}, w_{b,m,i}, u_m \in \{0, 1\} \tag{5.18}$$

$$P_b \in \mathbb{Z}^+,\ t_{b,m,i}, E, N \geq 0 \tag{5.19}$$

Expression (5.1) is to minimize the total electricity cost E and expression (5.2) aims to minimize the number of enabled machines N. The constraints include two aspects: batch formation (5.3)-(5.5) and allocating the batches to the machines on the horizon (5.6)-(5.19). The link between batch formation and allocation is the batch processing time P_b. The key of the batch allocation is to assign the batches on available machines to the periods with different unit electricity cost and guarantee processing contiguity and completion. In what follows, we detail the meaning of each constraint.

Equation (5.3) ensures that each job is assigned exactly to one batch. Equations (5.4) presents the batch processing time and (5.5) means the batch capacity must be respected. Equation (5.6) ensures that batch processing time equal to 0 on machine m in period i if it is not processed on the machine in this period. Equations (5.7) means that batch b cannot be processed on machine m in periods i if it is not processed on this machine. Equation (5.8) guarantee that a batch is only processed on one machine. Equation (5.9) ensures that the processing of a batch is completed within the horizon and is equal to P_b. Equation (5.10) states that total processing time in a period does not exceed the period's duration if one or several batches are processed in a period. Equations (5.11) - (5.13) are to ensure that at most one batch can be processed across any two adjacent periods. Constraint (5.14) restricts that a batch processed in multiple periods is not interrupted. Specifically, it states that if a batch b is simultaneously processed in period i and i' on machine m, i.e., $z_{b,m,i} = 1$ and $z_{b,m,i'} = 1$, then the sum of its processing times in the two periods (i.e., $t_{b,m,i}$ and $t_{b,m,i'}$) and the distance between i and i' (i.e., $s_{i'} - s_{i+1}$) should not be larger than the batch processing time P_b. Equation (5.15) expresses that machine m is used if at least one batch is processed on it. Equations (5.16) and (5.17) calculates the total electricity and the number of enabled machines, respectively. Equations (5.18) and (5.19) define the restrictions on the variables. Notably, the number of batches $|B|$ is initially set as its upper bound $|J|$ to derive a linear model.

5.2.2 Further improvement of model $\mathcal{P}'_Q$

Model $\mathcal{P}'_Q$ provided a correct mathematical formulation for the understudied problem. However, we can observe that equation (5.14), which restricts non-preemption of the batches that processed in multiple periods, has $0.5|M||B|(|I|^2 - |I|)$ constrains. The number of constraints grows fast with the number of periods, which leads to difficulty of solving large sized instances. This motivates us to better formulate this restriction. By analysing the characteristics of non-preemption of a given batch b processed on a given machine m, we find that its total number of processed periods must be less than its total number of crossed periods plus 1 (equation 5.20). Fig. 5.1 gives an general example, where a batch is processed in periods i to $i+3$. Then we have $\sum_{l=i}^{i+3} z_{b,m,l} = 4$ and $\sum_{l=i}^{i+2} w_{b,m,l} = 3$, where the difference is 1. Further guaranteeing that if a batch b is processed in periods $i-1$ and $i+1$ on a certain machine, period i on the machine must be entirely occupied by b (equation 5.21). Then the non-preemption requirement can be reformulated as follows.

$$\sum_{i \in I} z_{b,m,i} \leq \sum_{i \in I/|I|} w_{b,m,i} + 1, \ \forall b \in B, \ \forall m \in M \tag{5.20}$$

$$t_{b,m,i} \geq (z_{b,m,i-1} + z_{b,m,i+1} - 1)S_i, \forall b \in B, \forall m \in M, 2 \leq i \leq |I| - 1 \tag{5.21}$$

In equation (5.20), $I/|I|$ means set I excludes period $|I|$. In the new formulation, there are only $|B||M|(|I| - 1)$ constraints.

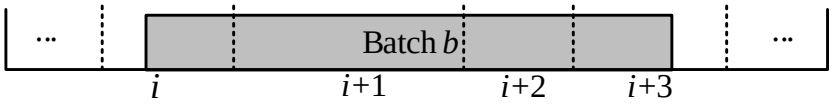


Fig. 5.1: An illustration of a batch processed in multiple periods

Now the improved model $\mathcal{P}_Q$ can be formulated as follows.

$$\begin{aligned} \mathcal{P}_Q: \quad & \min f_1 = E \\ & \min f_2 = N \\ \text{s.t.} \quad & \text{Constraints (5.3)-(5.13),(5.15)-(5.21).} \end{aligned}$$

5.2.3 Problem complexity

Theorem 5 *The problem PBMS-TOU is NP-hard in the strong sense.*

Proof: Consider a special case of problem PBMS-TOU that with identical machines and only a unique pricing period in the scheduling horizon. For such a special case,

to optimize the single objective of total electricity cost is to obtain a batch set with shortest total processing time. This batch formation problem corresponds to the problem of minimizing makespan on a single batch processing machine with non-identical job sizes, which has been proved to be NP-hard by Uzsoy [129]. Since the special case is strongly NP-hard, the theorem holds. □

5.3 Solution method

In model $\mathcal{P}_Q$, there are $|J|^2+(2|M||I|+1)|J|+|M|$ binary variables, $(|M||I|+1)|J|+2$ real variables and $|J|^2+|J||M|(6|I|-2)+4|J|+2|M||I|-|M|$ constraints. Take an instance with $|J|=10, |M|=10$ and $|I|=10$ for an example, there are 2120 binary variables, 1012 real variables and 6130 constraints. According to our preliminary experiments, it is computationally expensive to solve model $\mathcal{P}_Q$ due to its complexity. Moreover, it can be observed from the model that $t_{b,m,i}$ and P_b are two key variables for PBMS-TOU optimization. To determine $t_{b,m,i}$, we have to know the batch processing time P_b and its allocation to machine. This inspires us to decompose model problem into two sub-problems: batch formation and batch allocation. The former focuses on grouping job into batches; and the second stage allocates the formed batches to the parallel machines on the scheduling horizon. The structure of problem decomposition and the resulting two-stage heuristic approaches are illustrated in Fig. 5.2. Note that the two-stage heuristic approaches have two version, i.e., $\mathcal{P}_{Qf}$-$\mathcal{P}_{Qa}$ and SKM-$\mathcal{P}_{Qa}$, which are different with each other in the solution methods for batch formation.

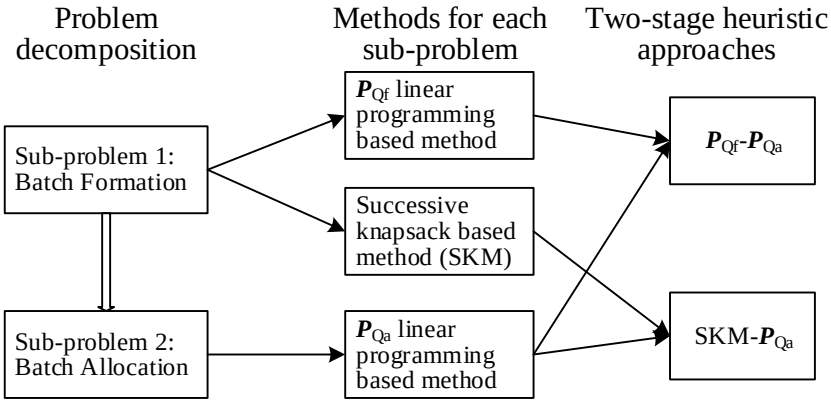


Fig. 5.2: Structure of problem decomposition and two-stage heuristic approaches

5.3.1 Methods for sub-problem 1: batch formation

Two batch formation methods are developed in this work. The first one is based on linear programming optimization model that solved by CPLEX; the second is

successive knapsack based method (SKM).

5.3.1.1 Linear programming model based batch formation

For batching the jobs, we can observed from model $\mathcal{P}_Q$ that batch processing time P_b is a crucial element, and lower total batch processing time may help for using less number of enabled machine N. Consequently, we set the objective for the first sub-problem as minimizing total batch processing time, denoted by T. Then the linear programming model for batch formation can be expressed as $\mathcal{P}_{Qf}$:

$$\begin{aligned} \mathcal{P}_{Qf}: \qquad & \min T = \sum_{b \in B} P_b \\ & \text{s.t.} \quad \text{Constraints (5.3)-(5.5).} \end{aligned} \tag{5.22}$$

by exactly solving model $\mathcal{P}_{Qf}$, we can obtain the minimum total batch processing time T and corresponding batch set B.

5.3.1.2 Successive knapsack based batch formation

Successive knapsack based method (SKM) is a heuristic method for batch formation with the objective of minimizing total batch processing time, i.e., heuristically solve problem $\mathcal{P}_{Qf}$. By observing $\mathcal{P}_{Qf}$, we find that it can be regarded as minimizing maximum completion time of batches in single machine environment where jobs are available at time 0, i.e., no machine idle time will exist between processing batches. Further considering constraints (5.3)-(5.5), batch formation problem in this work can be equivalent to optimizing makespan on a single batch processing machine with non-identical job sizes, i.e., $1|c_j, B|C_{max}$. This latter problem has been proved to be equivalent to minimizing total waste space of all batches, i.e., $\sum_{b \in B}(WS_b)$ [30],[143], where waste space of batch b (WS_b) is defined by formula 5.23 and illustrated with an example in Fig. 5.3.

$$WS_b = C \cdot P_b - \sum_{j \in b} c_j \cdot p_j \tag{5.23}$$

Inspired by the idea of minimizing $\sum_{b \in B}(WS_b)$, we propose to obtain the minimum total batch processing time by solving successive knapsack problems. It aims to maximize the occupied area of assigned jobs, i.e., $\sum_{j \in b} c_j \cdot p_j$, in batch $b, b \in B$. Consequently, the jobs can be regarded as items with weight c_j and value $c_j \cdot p_j$. The batches are considered as knapsacks. Each knapsack has to be firstly filled by the

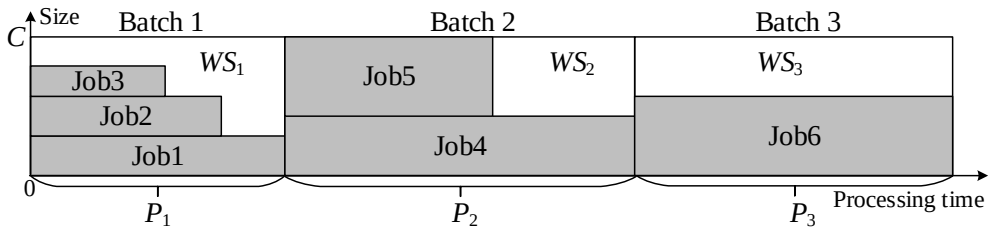


Fig. 5.3: An illustration of WS_b

unassigned longest processing time job f, then the residual capacity of the knapsack is $C - c_f$. The corresponding 0-1 knapsack problem can be expressed as follows:

$$\begin{aligned} \mathcal{P}_{KP_b}: \quad \max \quad & \sum_{j \in J \setminus f} (p_j c_j) x_j \\ s.t. \quad & \sum_{j \in J \setminus f} c_j x_j \leq C - c_f \\ & x_j \in \{0, 1\} \end{aligned}$$

where x_j is a binary variable which equals to 1 if job j is filled to batch b, 0 otherwise. The knapsacks will be sequentially used until all the jobs are assigned. Then the total number of used knapsacks, i.e., number of constructed batches $|B|$, can be obtained. The successive knapsack based method is presented in Algorithm 5.4.

Algorithm 5.4: successive knapsack based method

1: consider job $j, j \in J$ to be batched as an item with value $p_j \cdot c_j$ and weight c_j, sort them in non-decreasing order of p_j; and set batch index $b = 1$;
2: select the first job f from set J, solve the knapsack problem $\mathcal{P}_{KP_b}$ with residual capacity $C - c_f$;
3: form batch b that has processing time $P_b = p_f$ with selected jobs in $\mathcal{P}_{KP_b}$;
4: update set J by removing the selected jobs;
5: if $J \neq \emptyset$, let $b = b + 1$ and go to step 2; otherwise, output formed batches and their processing P_b.

Fig. 5.4: successive knapsack based method

Denote the batch set obtained by Algorithm 5.4 as B' and its total processing time as T', then we have $T' = \sum_{b \in B'} P_b$. Next, we focus on assign the formed batches to the scheduling horizon on the available machines.

5.3.2 Method for sub-problem 2: batch allocation

With the solution results of sub-problem 1, batch formation is determined. Each batch can be considered as a job and its processing time as a parameter. Then sub-problem 2 can be formulated as model $\mathcal{P}_{Qa}$.

$$\begin{aligned}
\mathcal{P}_{Qa}: \quad & \min f_1 = E \\
& \min f_2 = N \\
\text{s.t.} \quad & \text{Constraints (5.6)-(5.8),(5.10)-(5.13), (5.15)-(5.21) and} \\
& \sum_{i \in I} t_{b,m,i} \geq r_{b,m} P_b / v_m, \forall b \in B, \forall m \in M \qquad (5.24)
\end{aligned}$$

Constraint (5.24) is a modified version of constraint (5.9) since P_b is considered as a parameter. To solve the bi-objective model $\mathcal{P}_{Qa}$, an ε-constraint method is adapted in this work. By selecting total electricity cost E as the preferred objective, $\mathcal{P}_{Qa}$ can be transformed into a series of single objective problems $\mathcal{P}_{Qa}(\varepsilon)$'s.

$$\begin{aligned}
\mathcal{P}_{Qa}(\varepsilon): \quad & \min E \\
\text{s.t.} \quad & \text{Constraints (5.6)-(5.8),(5.10)-(5.13), (5.15)-(5.21), (5.24) and} \\
& N \leq \varepsilon \qquad (5.25)
\end{aligned}$$

where the range of ε is limited by the ideal and nadir values of N, i.e., $N^I \leq \varepsilon \leq N^N$. N^I and N^N can be calculated by solving the following three problems presented in Definition 7 of Chapter 2.

$\mathcal{P}^1_{Qa}: E^I = \min E$ s.t. Constraints (5.6)-(5.8), (5.10)-(5.13), (5.15)-(5.21) and (5.24)
$\mathcal{P}^2_{Qa}: N^I = \min N$ s.t. Constraints (5.6)-(5.8),(5.10)-(5.13), (5.15)-(5.21) and (5.24)
$\mathcal{P}^3_{Qa}: N^N = \min N$ s.t. Constraints (5.6)-(5.8),(5.10)-(5.13), (5.15)-(5.21), (5.24) and $E = E^I$

The value of ε is initialized as $N^N - 1$ in the first iteration. Then in $(k+1)$-th iteration, ε^{k+1} is set as $N^k - 1$, where N^k is obtained from the solution of $\mathcal{P}_{Qa}(\varepsilon)$ in k-th iteration.

Now the ε-constraint method for model $\mathcal{P}_{Qa}$ can be summarized as Algorithm 5.5.

With the above models and Algorithm 5.5, the completed two-stage heuristic approach can be concluded as Algorithm 5.6.

Algorithm 5.5: ε-constraint method for model $\mathcal{P}_{Qa}$

1: set $\mathcal{A}' = \emptyset$ and $k = 1$;
2: exactly solve the models P^1_{Qa}, P^2_{Qa} and P^3_{Qa} to obtain E^I, N^I and N^N;
3: set $\mathcal{A}' = (E^I, N^N)$, and $\varepsilon^k = N^N - 1$;
4: **while** $\varepsilon^k \geq N^I$ **do**
5: solve problem $\mathcal{P}_{Qa}(\varepsilon^k)$ exactly to obtain E^k, and calculate N^k with the solution of $\mathcal{P}_{Qa}(\varepsilon^k)$;
6: add (E^k, N^k) to set $\mathcal{A}'$, and let $\varepsilon^{k+1} = N^k - 1$, reset $k = k + 1$;
7: **end while**
8: remove dominated points from $\mathcal{A}'$ if any and obtain the Pareto optimal set $\mathcal{A}$.

Fig. 5.5: ε-constraint method for model $\mathcal{P}_{Qa}$

Algoirthm 5.6: Two-stage Heuristic approach for PBMS-TOU

1: form the batches with model $\mathcal{P}_{Qf}$ (resp. SKM) and output formed batch b and its processing time P_b, where $b \in B$ (resp. $b \in B'$);
2: construct model $\mathcal{P}_{Qa}$ with b and P_b, where $b \in B$ (resp. $b \in B'$); then call Algorithm 5.5 to obtain batch schedules and the corresponding approximation set $\mathcal{A}$ for problem PBMS-TOU.

Fig. 5.6: Two-stage Heuristic approach for PBMS-TOU

5.4 Computational results

In this section, a real-life case and random generated instances are tested to validate the effectiveness of the proposed models and two-stage heuristic approach. The formulated models are coded in C++ and solved by commercial optimization software CPLEX 12.6. The algorithms are coded in C++ and the embedded 0-1 knapsack problem is solved by calling the exact algorithm developed by David Pisinger [112]. All the experiments are implemented o a PC with 1.7 GHz Intel i5-3317U CPU and 3.12 GB RAM. The quality of the obtained solution sets are evaluated with the number of solutions Q, hypervolume ration H and the average e-dominance indicator D. CT denotes the CPU time consumed by the proposed methods. The computational time for each instances is limited in 3600 s.

5.4.1 A case study

The example is originated from a glass production company in Shanghai, China [138], where furnaces (batch machines) are a key resource of the manufacturing process. They have a fixed capacity that can contain different sized glasses for simultaneous processing. The furnaces that can complete the same operation have different speeds and energy consumption rates, which is corresponding to a batch processing parallel machines scheduling problem.

In the case, there are 30 glasses need to be ceramization in two furnaces within 300 minutes, where three pricing periods are involved. The detailed job information and period information are presented in Table 5.1. Fig. 5.7 provide a visualization of the jobs. The jobs are batched on a pallet then put into the furnace. The capacity of each pallet is 24. The power rate of the first furnace is 90 kW with speed rate equalling to 2, and the second one has speed rate equalling to 1 and consume 66 kW power each unit time.

Table 5.1: The detailed data of the case

Job type	p_j	c_j	Number of jobs		Period No.	S_i	e_i
1	20	3	14		1	120	0.7
2	40	6	10		2	100	1.2
3	80	12	3		3	80	0.3
4	100	18	3		-	-	-

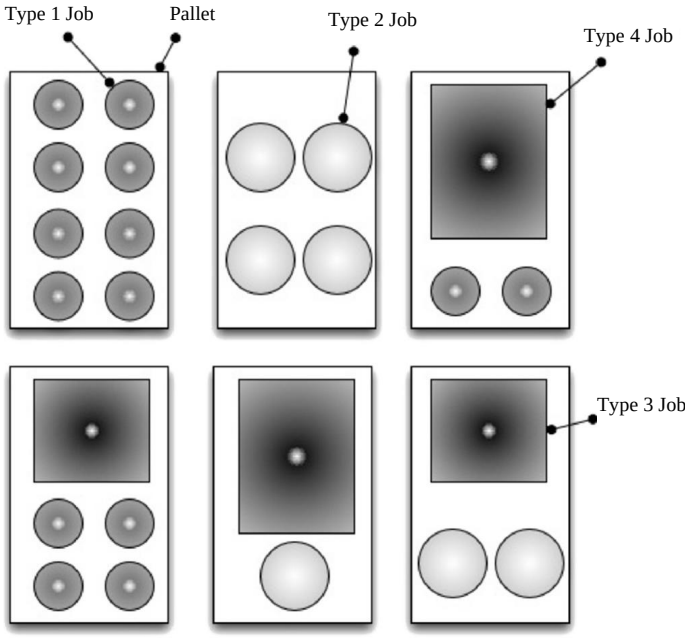


Fig. 5.7: The examples of batch combinations of the four job types [138]

The computational results are shown in Table 5.2, where (E, N) denotes the obtained objective vectors. Note that the bi-objective model $\mathcal{P}'_Q$ and $\mathcal{P}_Q$ are solved by adapting the framework of exact ε-constraint method. We can find that model $\mathcal{P}_Q$ is much efficient than $\mathcal{P}'_Q$, which owes to the effective reformulation that has much less constraints. By decomposing the problem into two stages, the computational time is further reduced since $\mathcal{P}_{Qf}$-$\mathcal{P}_{Qa}$ consumes 60 seconds less than $\mathcal{P}_Q$. Moreover, batching the jobs with successive knapsack based method (SKM) is much faster than directly solving the linear programming model $\mathcal{P}_{Qf}$. In terms of solution quality, the four methods obtain the same solutions. That is, the two-stage heuristic methods, i.e., $\mathcal{P}_{Qf}$-$\mathcal{P}_{Qa}$ and SKM-$\mathcal{P}_{Qa}$, have obtain the Pareto front for the studied case. Comparing the two solutions, we can find that enabling two machines can reduce 3360 RMB Yuan electricity cost than the circumstance where only one machine is involved for processing.

Table 5.2: The solutions and computation time of each method

	$\mathcal{P}'_Q$	$\mathcal{P}_Q$	$\mathcal{P}_{Qf}$-$\mathcal{P}_{Qa}$	SKM-$\mathcal{P}_{Qa}$
CT	65.81	65.36	5.29	0.30
(E, N)	(15000,2) (18360,1)	(15000,2) (18360,1)	(15000,2) (18360,1)	(15000,2) (18360,1)

The detailed solution information is illustrated in Table 5.3 and Fig. 5.8. Table 5.3 shows that there are 8 formed batches, whose total processing time equals to 560 min. All the batches can be completed processing either on one machine or two machines. The two schedules are graphically described by Fig. 5.8. Note that since the speed rate of the first machine is 2, the processing time of batch b, $b = 1, 2, ..., 8$, on the machine is $P_b/2$. For example, in solution 1, batch 8 consumes 20 minutes on machine 2; while in solution 2, its processing can be completed in 10 minutes on machine 1. In addition, when only one machine is enabled, the first furnace is selected, which has higher speed and lower cost for completing a unit processing time.

Table 5.3: The detailed batch information

Batch No.	Job type in the batch	Batch size	P_b
1	4,2	24	100
2	4,2	24	100
3	4,2	24	100
4	3,3	24	80
5	3,2,2	24	80
6	2,2,2,2	24	40
7	2,1,1,1,1,1,1	24	40
8	1,1,1,1,1,1,1,1	24	20

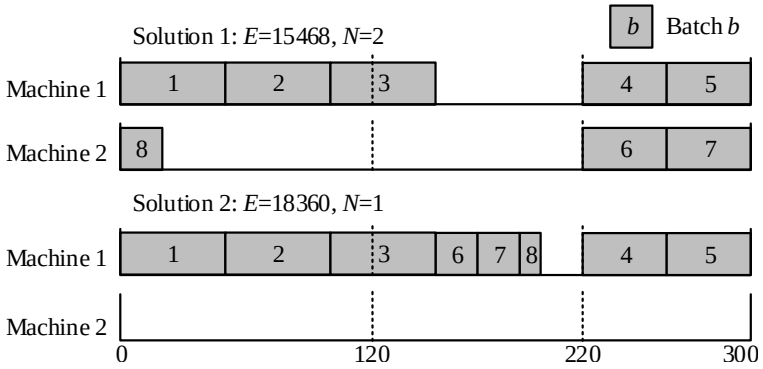


Fig. 5.8: The schedules for the formed batches

5.4.2 Random test instances

In this section, random test instances are used to examine the performance of the proposed methods, which are generated according to the characteristics of the studied case. The number of jobs increases from 10 to 40. The characteristics of the jobs, i.e., processing time and size, are generated as the four types of jobs in the studied case. The length of the scheduling horizon $s_{|I|+1}$ is set as 300 min. The electricity pricing scheme are the same as the studied case. The number of machines is generated as $|M| = \alpha \sum_{j \in J} p_j c_j / (C s_{|I|+1})$, where α is a positive factor that reflects the resource scarcity and randomly generated between 2 to 5. The processing speed rate v_m of machine m varies from 1 to 3. After generating the speeds for all machines, we sort all the machines in non-increasing order of their speed, i.e., $v_1 \geq v_2 \geq ... \geq v_{|M|}$. Since higher speed generally results in greater power consumption in real-life, the power consumption is generated as $q_{m-1} = q_m + rd_m$, where $q_{|M|} = 66$kw/h according to the real-life example and rd_m is a random parameter generated between 0 to 30. The computational results are reported in Tables 5.4 to 5.6. In the tables, the bi-objective models $\mathcal{P}'_Q$ and $\mathcal{P}_Q$ are transformed into single objective problems with exact ε-constraint method, and each single objective problem is solved by CPLEX 12.6.

The computational results for small sized instances are shown in Tables 5.4 and 5.5. Comparing the results obtained by the two models in Table 5.4, it can be observed that the improved model is more efficient than the basic one, and model $\mathcal{P}'_Q$ even fails to find a solution for the instances with up to 15 jobs. Q, H, D are the performance metrics for evaluating the solution equality, and the Pareto optimal set obtained by the improved model $\mathcal{P}_Q$ is set as the reference set. From the table, we can find that number of solutions find by different method is the same, and hypervolume

ratio H and average e-dominance indicator D both equal to 1. This indicates that all the methods have obtained the same solutions, i.e., found the Pareto front. In terms of solution efficiency, we can find that for each method, the computational time increases with the problem size. Both two-stage heuristic approaches consume far less time than the models, which demonstrates the effectiveness of the problem decomposition. Moreover, SKM-$\mathcal{P}_{Qa}$ can solve most of the problems faster than $\mathcal{P}_{Qf}$-$\mathcal{P}_{Qa}$, because successive knapsack based method is more efficient than the linear programming model.

Table 5.4: Comparison results for small sized instances

$\|J\|$	$\mathcal{P}'_Q$		$\mathcal{P}_Q$		$\mathcal{P}_{Qf}$-$\mathcal{P}_{Qa}$				SKM-$\mathcal{P}_{Qa}$			
	Q	CT	Q	CT	Q	H	D	CT	Q	H	D	CT
10	3	5.05	3	2.03	3	1.000	1.000	0.45	3	1.000	1.000	0.52
11	2	0.48	2	0.55	2	1.000	1.000	0.31	2	1.000	1.000	0.16
12	3	99.98	3	1.56	3	1.000	1.000	0.50	3	1.000	1.000	0.33
13	3	333.70	3	1.81	3	1.000	1.000	0.66	3	1.000	1.000	0.34
14	4	364.16	4	30.53	4	1.000	1.000	1.03	4	1.000	1.000	0.78
15	-	3600.00	3	211.86	3	1.000	1.000	2.53	3	1.000	1.000	1.55
Average		733.90		41.39		1.000	1.000	0.91		1.000	1.000	0.61

Table 5.5 reports the obtained solutions of the instances in Table 5.4. N and E denote the number of enabled machines and total electricity cost. It is obvious that there is a trade-off between the two objective values. Specifically, lower total electricity cost can be obtained when more machines are enabled.

Table 5.5: Solutions for small sized instances

$\|J\|$	N	E	$\|J\|$	N	E
10	1	15000	13	1	13200
	2	12959		2	10984
	3	12472		3	10868
11	1	7238	14	1	11880
	2	7051		2	8949
				3	8424
				4	8259
12	1	11484	15	2	15475
	2	9762		3	14768
	3	9393		4	14680

Table 5.6 presents the computational results on larger sized instances. Since there are limited number of solutions for each set of instances, the detail information of the obtained solution set is provided. In the table, UB is the upper-bound of model $\mathcal{P}_Q$ obtained by CPLEX within the limited computational time. CT(*) denote the

Table 5.6: Comparison results for larger sized instances

$\|J\|$	N	$\mathcal{P}_Q$		$\mathcal{P}_{Qf}$-$\mathcal{P}_{Qa}$				SKM-$\mathcal{P}_{Qa}$			
		UB	CT	E	CT($\mathcal{P}_{Qf}$)	CT($\mathcal{P}_{Qa}$)	CT	E	CT(SKM)	CT($\mathcal{P}_{Qa}$)	CT
16	1	-	3600	11232	0.23	0.72	0.95	11232	0.00	0.72	0.72
	2	-		7881				7881			
	3	-		6225				6225			
	4	6204		6204				6204			
20	2	-	3600	39390	1.09	5.59	6.69	39390	0.02	4.86	4.88
	3	-		34238				34238			
	4	-		32573				32573			
	5	-		31292				31292			
	6	30824		30824				30824			
24	2	-	3600	50328	1.89	12.28	14.17	50328	0.00	13.78	13.78
	3	-		40380				40380			
	4	-		36256				36256			
	5	-		34664				34664			
	6	-		33815				33815			
	7	33515		33515				33515			
28	2	-	3600	50544	3.17	10.31	13.48	50544	0.00	10.80	10.80
	3	-		39804				39804			
	4	-		37074				37074			
	5	-		35460				35460			
	6	-		34560				34560			
	7	34380		34320				34320			
32	2	-	3600	47040	1.86	21.38	23.23	47040	0.00	15.45	15.45
	3	-		37072				37072			
	4	-		35082				35082			
	5	-		33491				33491			
	6	-		32296				32296			
	7	31672		31672				31672			
36	2	-	3600	32790	21.95	22.75	44.70	32790	0.00	20.11	20.11
	3	-		28840				28840			
	4	-		26365				26365			
	5	-		24831				24831			
	6	24720		24720				24720			
40	3	-	3600	52988	231.68	40.92	272.61	52988	0.00	42.00	42.00
	4	-		47083				47083			
	5	-		44194				44194			
	6	-		42521				42521			
	7	-		41177				41177			
	8	-		40490				40490			
	9	40065		40065				40065			

computational time spent by method *. Due to the complexity of problem PBMS-TOU, the improved model is failed to obtain the Pareto front for any set of the instances. According to our observation on the experiments, it is because the lower bound converges very slow. However, the two-stage heuristic approaches are able to find the solutions within 280 s. The solution set obtained by both two-stage heuristic methods are the same, and SKM-$\mathcal{P}_{Qa}$ spent less time than $\mathcal{P}_{Qf}$-$\mathcal{P}_{Qa}$, because the SKM is more efficient than $\mathcal{P}_{Qf}$.

5.5 Conclusions

This chapter investigates a parallel batch processing machine scheduling problem under TOU tariffs with non-identical job sizes and processing contiguity requirement. The objectives are to minimize total electricity cost and number of enabled machines. We first formulated a mixed integer linear programming model for the problem. Then the model is further improved with processing contiguity reformulation. A two-stage heuristic approach is developed to efficiently find the optimal or near optimal solutions, where the first stage aims to form the batches and the second stage allocates the batches to the machines on the scheduling horizon. For the first stage, a linear programming based method ($\mathcal{P}_{Qf}$) and a successive knapsack based method (SKM) are developed to construct the batches; then the batches are scheduled to the machines by solving a bi-objective linear programming model $\mathcal{P}_{Qa}$ with ε-constraint method. Thus two versions of heuristic approach are resulted, i.e., $\mathcal{P}_{Qf}$-$\mathcal{P}_{Qa}$ and SKM-$\mathcal{P}_{Qa}$. The performance of the proposed methods is evaluated by a case study and random generated instances. The computational results show that the improved model is more efficient than the original one, and the two-stage heuristic approach can find the very good quality solutions. In addition, SKM-$\mathcal{P}_{Qa}$ is much efficient than $\mathcal{P}_{Qf}$-$\mathcal{P}_{Qa}$, especially for large sized instances. Part of this work has been published in the following papers.

J. Cheng, F. Chu, and M. Zhou. An improved model for parallel machine scheduling under time–of–use electricity price. *IEEE Transactions on Automation Science and Engineering*, DOI:10.1109/TASE.2016.2631491, 2017.

J. Cheng, F. Chu, M. Liu , and W. Xia. Single–machine batch scheduling under time–of–use tariffs: new mixed–integer programming approaches. In *2016 IEEE International conference on Systems, Man, and Cybernetics, Budapest, Hungary*, pages 3498–3503, 2016.

Chapter 6

Conclusions and perspectives

This thesis investigates several bi-objective batch scheduling problems under time-of-use (TOU) electricity tariffs, which aim to design production plans for batch processing machines under fluctuating electricity prices, with the objectives of simultaneously optimizing total electricity cost and production efficiency or resource. For each of the considered problems, appropriate mathematical models were formulated, the problem complexities were demonstrated, and problem characteristic based solution methods, including knapsack heuristic based ε-constraint method, multiple knapsack heuristic based ε-constraint method, bin-packing heuristic based ε-constraint method and two-stage heuristic based iterative search algorithm, were developed. The problem properties were also analysed to help resolve the problems. Computational results on randomly generated instances showed that the proposed methods outperform the state-of-the-art solver CPLEX.

We first considered scheduling a single batch processing machine under time-of-use tariffs with the objectives of minimizing total electricity cost and makespan. A mixed-integer linear programming (MILP) model was formulated for the problem. Then the model was tightened based on property analysis and the studied problem was proved to be NP-hard. Two efficient heuristic based ε-constraint methods (ECM), i.e., knapsack heuristic based ECM (KH-ECM) and multiple knapsack heuristic based ECM (MKH-ECM), were respectively developed to find approximation Pareto fronts for the problem. The computational results on random generated instances show the effectiveness and efficiency of the proposed methods compared with the commercial solver CPLEX. Specifically, the first algorithm spends less than two minutes for all instances and achieved better quality solutions than CPLEX for large-size problems. The second one can almost find exact Pareto front within one minute, while CPLEX consumed near five hours for a medium-size instance.

Subsequently, to further optimize non-processing energy consumption, the previous problem was extended to a single batch machine scheduling problem with machine on/off switching under TOU tariffs. The objectives are to optimize total electricity cost and makespan. The problem was firstly formulated as a bi-objective MILP model. Then an improved model was proposed based on the optimal batch rule demonstration, which significantly reduced solution search space. A bin-packing heuristic based ε-constraint method (BPH-ECM) was developed for the problem. The proposed method can find a set of efficient solutions within 5 minutes for the instance up to 5000 batches, while CPLEX failed to find any solution for instances with 120 batches.

Finally, we studied a bi-objective parallel batch processing machine scheduling problem with non-identical job sizes under TOU tariffs to minimize total electricity cost and number of enabled machines. A bi-objective MILP model was formulated for the problem and then a two-stage MILP model was further proposed to quickly generating optimal or near-optimal solutions. A two-stage heuristic based iterative search algorithm was developed to find the approximation Pareto front. The computational results on a case study and randomly generated instances demonstrated the well performance of the proposed methods.

Batch scheduling considering energy saving is receiving increasing attention due to the necessity of environmental protection and sustainability development. This dissertation is an attempt and exploration of energy cost savings under time-of-use electricity pricing, which can help enterprises reduce their energy bills. However, there is still enough space for conducting further research.

1) In the present works, the proposed solution methods find the approximation Pareto fronts by transforming a multi-objective problem into a series of single objective ones with the framework of scalarization methods, and then heuristically solving the single objective problems successively. In the future, we may combine the proposed problem characteristics with other sophisticated meta-heuristics (e.g. NSGA-II, SPEA2) to further improve the solution quality for the problem. Especially for the problem studied in Chapter 5, it is necessary to develop more efficient method for large sized problems. By analysing the problem property, an intuitive idea is to assign the batches to the periods with lower processing cost while ensuring the continuous processing and non-preemption.

2) For large sized batch scheduling problems under TOU tariffs, the solution quality evaluation is always a question since MILP models failed to obtain optimal solutions. Consequently, it is meaningful to continue developing exact multi-objective optimization methods that can efficiently find the exact Pareto fronts for large sized

problems. Alternatively, proposing good lower bound for the bi-objective problems also can be useful to evaluate how far the obtained solution set from the Pareto optimal solution set.

3) The technique of the mathematical formulation and heuristic approaches can be adapted to more complicated problems, such as the problems involving other machine environments (e.g., unrelated parallel machines, flow shop, job shop), and job characteristics (e.g., dynamic release times, non-identical due dates). More regular objective functions (e.g., total completion time, maximum lateness) and production features (e.g., serial batching, maintenance activity) also deserve investigation under TOU tariffs.

4) In many countries, other Demand Response strategies, such as Peak Load Pricing (PLP) and Real-Time Pricing (RTP), are also extensively encountered. Different from TOU tariffs that set the prices and pricing periods in advance, RTP provides volatile prices that may hourly or daily change. It is a challenge to explore scheduling approaches for production enterprises under such electricity pricing scheme, where stochastic optimization methods can be one possible option for solving the problems.

5) In the context of sustainable development, it would be significant to incorporate other environmental related criteria into the optimization framework. These criteria can be carbon emissions reduction, liquid consumption minimization, and material loss optimization, etc.

Bibliography

[1] Hamid Abedinnia, Christoph H. Glock, Michael Schneider, and Eric H. Grosse. Machine scheduling problems in production: A tertiary study. *Computers & Industrial Engineering*, 2017.

[2] Energy Information Administration. Annual energy outlook 2015 - with projections to 2040. Technical report, Energy Information Administration, 2015.

[3] Energy Information Administration. International energy outlook 2017. Technical report, EIA, 2017.

[4] Paolo Albertelli. Energy saving opportunities in direct drive machine tool spindles. *Journal of Cleaner Production*, 165:855–873, 2017.

[5] Eric Angel, Evripidis Bampis, and Vincent Chau. *Low Complexity Scheduling Algorithm Minimizing the Energy for Tasks with Agreeable Deadlines*. Springer Berlin Heidelberg, 2012.

[6] Eric Angel, Evripidis Bampis, Vincent Chau, and Dimitrios Letsios. *Throughput Maximization for Speed–Scaling with Agreeable Deadlines*. Springer Berlin Heidelberg, 2013.

[7] Eric Angel, Evripidis Bampis, Vincent Chau, and Nguyen Kim Thang. *Throughput Maximization in Multiprocessor Speed–Scaling*. Springer International Publishing, 2014.

[8] Eric Angel, Evripidis Bampis, Fadi Kacem, and Dimitrios Letsios. Speed scaling on parallel processors with migration. *Lecture Notes in Computer Science*, 7484(1):128–140, 2011.

[9] Asia Pacific Energy Research Centre APERC. Apec energy demand and supply outlook – 5th edition. Technical report, Asia–Pacific Economic Cooperation, 2013.

[10] José Elias C. Arroyo and Y. T. Leung. An effective iterated greedy algorithm for scheduling unrelated parallel batch machines with non–identical capacities and unequal ready times. *Computers & Industrial Engineering*, 105:84–100, 2017.

[11] José Elias C. Arroyo and Y. T. Leung. Scheduling unrelated parallel batch processing machines with non–identical job sizes and unequal ready times. *Computers & Operations Research*, 78:117–128, 2017.

[12] Meral Azizoglu and Scott Webster. Scheduling a batch processing machine with non–identical job sizes. *International Journal of Production Research*, 38(10):2173–2184, 2000.

[13] Evripidis Bampis, Dimitrios Letsios, Ioannis Milis, and Georgios Zois. *Speed Scaling for Maximum Lateness*. Springer Berlin Heidelberg, 2013.

[14] Philippe Baptiste and Antoine Jouglet. On minimizing total tardiness in a serial batching problem. *RAIRO-Operations Research*, 35(1):107–115, 2001.

[15] S Bechikh, L Ben Said, and K Ghedira. Estimating nadir point in multi–objective optimization using mobile reference points. In *Evolutionary Computation*, pages 1–9, 2010.

[16] Abderrahmane Bensmaine, Mohammed Dahane, and Lyès Benyoucef. A non-dominated sorting genetic algorithm based approach for optimal machines selection in reconfigurable manufacturing environment. *Computers & Industrial Engineering*, 66(3):519–524, 2013.

[17] Jorne Van Den Bergh, Jeroen Beliën, Philippe De Bruecker, Erik Demeulemeester, and Liesje De Boeck. Personnel scheduling: A literature review. *European Journal of Operational Research*, 226(3):367–385, 2013.

[18] Jean-François Bérubé, Michel Gendreau, and Jean-Yves Potvin. An exact ε–constraint method for bi–objective combinatorial optimization problems: Application to the traveling salesman problem with profits. *European Journal of Operational Research*, 194(1):39–50, 2009.

[19] AAG Bruzzone, D Anghinolfi, M Paolucci, and F Tonelli. Energy–aware scheduling for improving manufacturing process sustainability: A mathematical model for flexible flow shops. *CIRP Annals–Manufacturing Technology*, 61(1):459–462, 2012.

[20] Massimiliano Caramia and Paolo DellÂ´Olmo. *Multi–objective Management in Freight Logistics*. Springer London, 2008.

[21] Vijaya Chandru, C-Y Lee, and Reha Uzsoy. Minimizing total completion time on batch processing machines. *International Journal of Production Research*, 31(9):2097–2121, 1993.

[22] Vijaya Chandru, Chung-Yee Lee, and Reha Uzsoy. Minimizing total completion time on a batch processing machine with job families. *Operations Research Letters*, 13(2):61–65, 1993.

[23] P.-Y. Chang, P. Damodaran, and S. Melouk. Minimizing makespan on parallel batch processing machines. *International Journal of Production Research*, 42(19):4211–4220, 2004.

[24] A. Charnes and W. W. Cooper. Management models and industrial applications of linear programming. *John Wiley & Sons volumes I and II, New York*, 1961.

[25] A. Charnes, W. W. Cooper, and R. O. Ferguson. Optimal estimation of executive compensation by linear programming. *Management Science*, 1(2):138–151, 1955.

[26] Ada Che, Ke Lv, Eugene Levner, and Vladimir Kats. Energy consumption minimization for single machine scheduling with bounded maximum tardiness. In *IEEE International Conference on Networking, Sensing and Control*, pages 146–150, 2015.

[27] Ada Che, Xueqi Wu, Jing Peng, and Pengyu Yan. Energy–efficient bi–objective single–machine scheduling with power–down mechanism. *Computers & Operations Research*, 85:172–183, 2017.

[28] Ada Che, Yizeng Zeng, and Ke Lyu. An efficient greedy insertion heuristic for energy–conscious single machine scheduling problem under time–of–use electricity tariffs. *Journal of Cleaner Production*, 129:565–577, 2016.

[29] Huaping Chen, Bing Du, and George Q Huang. Metaheuristics to minimise makespan on parallel batch processing machines with dynamic job arrivals. *International Journal of Computer Integrated Manufacturing*, 23(10):942–956, 2010.

[30] Huaping Chen, Bing Du, and George Q Huang. Scheduling a batch processing machine with non–identical job sizes: a clustering perspective. *International journal of production research*, 49(19):5755–5778, 2011.

[31] Ba Yi Cheng, Y T. Leung, and Kai Li. Integrated scheduling on a batch machine to minimize production, inventory and distribution costs. *European Journal of Operational Research*, 258(1):104–112, 2016.

[32] Junheng Cheng, Feng Chu, Chengbin Chu, and Weili Xia. Bi–objective optimization of single–machine batch scheduling under time–of–use electricity prices. *RAIRO – Operations Research*, 50(4–5):715–732, 2016.

[33] S.H. Chung, Y.T. Tai, and W.L. Pearn. Minimising makespan on parallel batch processing machines with non–identical ready time and arbitrary job sizes. *International Journal of Production Research*, 47(18):5109–5128, 2009.

[34] C. A. C. Coello, G. T. Pulido, and M. S. Lechuga. Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 8(3):256–279, 2004.

[35] Yunfei Cui, Zhiqiang Geng, Qunxiong Zhu, and Yongming Han. Review: Multi–objective optimization methods and application in energy saving. *Energy*, 125:681–704, 2017.

[36] Piotr Czyz˙zak and Adrezej Jaszkiewicz. Pareto simulated annealing–a metaheuristic technique for multiple –objective combinatorial optimization. *Journal of Multi–criteria Decision Analysis*, 7, 1998.

[37] Min Dai, Dunbing Tang, Adriana Giret, Miguel A. Salido, and W. D. Li. Energy–efficient scheduling for a flexible flow shop using an improved genetic–simulated annealing algorithm. *Robotics & Computer Integrated Manufacturing*, 29(5):418–429, 2013.

[38] Purushothaman Damodaran and Ping Yu Chang. Heuristics to minimize makespan of parallel batch processing machines. *International Journal of Advanced Manufacturing Technology*, 37(9–10):1005–1013, 2008.

[39] Purushothaman Damodaran, Don Asanka Diyadawagamage, Omar Ghrayeb, and Mario C. Vélez-Gallego. A particle swarm optimization algorithm for minimizing makespan of nonidentical parallel batch processing machines. *International Journal of Advanced Manufacturing Technology*, 58(9–12):1131–1140, 2012.

[40] Purushothaman Damodaran, Neal S Hirani, and Mario C Velez-Gallego. Scheduling identical parallel batch processing machines to minimise makespan using genetic algorithms. *European J of Industrial Engineering*, 3(2):187–206, 2009.

[41] Purushothaman Damodaran, Praveen Kumar Manjeshwar, and Krishnaswami Srihari. Minimizing makespan on a batch–processing machine with non–identical job sizes using genetic algorithms. *International Journal of Production Economics*, 103(2):882–891, 2006.

[42] Indraneel Das and John E Dennis. A closer look at drawbacks of minimizing weighted sums of objectives for pareto set generation in multicriteria optimization problems. *Structural optimization*, 14(1):63–69, 1997.

[43] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[44] Kalyanmoy Deb. Multiobjective optimization using evolutionary algorithms. *Computational Optimization & Applications*, 39(1):75–96, 2001.

[45] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T. Meyarivan. *A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi–objective Optimization: NSGA–II*. Springer Berlin Heidelberg, 2000.

[46] Emrah Demir, Tolga Bektaş, and Gilbert Laporte. The bi–objective pollution–routing problem. *European Journal of Operational Research*, 232(3):464–478, 2014.

[47] Hongwei Ding, Lyès Benyoucef, and Xiaolan Xie. A simulation-based multi-objective genetic algorithm approach for networked enterprises optimization. *Engineering Applications of Artificial Intelligence*, 19(6):609–623, 2006.

[48] Jian Ya Ding, Shiji Song, and Cheng Wu. Carbon–efficient scheduling of flow shops by multi–objective optimization. *European Journal of Operational Research*, 248(3):758–771, 2016.

[49] Jian Ya Ding, Shiji Song, Rui Zhang, Raymond Chiong, and Cheng Wu. Parallel machine scheduling under time–of–use electricity prices: New models and optimization approaches. *IEEE Transactions on Automation Science and Engineering*, 13:1138 – 1154, 2016.

[50] Lionel Dupont and Clarisse Dhaenens-Flipo. Minimizing the makespan on a batch machine with non–identical job sizes: an exact procedure. *Computers & Operations Research*, 29(7):807–819, 2002.

[51] Lionel Dupont and Fariborz Jolai Ghazvini. Minimizing makespan on a single batch processing machine with non–identical job sizes. *Journal européen des systèmes automatisés*, 32(4):431–440, 1998.

[52] Masoud Esmaili, Heidar Ali Shayanfar, and Nima Amjady. Multi–objective congestion management incorporating voltage and transient stabilities. *Energy*, 34(9):1401–1412, 2009.

[53] Kan Fang, Nelson Uhan, Fu Zhao, and John W Sutherland. A new approach to scheduling in manufacturing for power consumption and carbon footprint reduction. *Journal of Manufacturing Systems*, 30(4):234–240, 2011.

[54] Kan Fang, Nelson A Uhan, Fu Zhao, and John W Sutherland. Flow shop scheduling with peak power consumption constraints. *Annals of Operations Research*, 206(1):115–145, 2013.

[55] Kan Fang, Nelson A Uhan, Fu Zhao, and John W Sutherland. Scheduling on a single machine under time–of–use electricity tariffs. *Annals of Operations Research*, 238(1–2):199–227, 2016.

[56] MP Fanti, B Maione, G Piscitelli, and B Turchiano. Heuristic scheduling of jobs on a multi–product batch processing machine. *International Journal of Production Research*, 34(8):2163–2186, 1996.

[57] Christian Gahm, Florian Denz, Martin Dirr, and Axel Tuma. Energy–efficient scheduling in manufacturing companies: A review and research framework. *European Journal of Operational Research*, 248(3):744–757, 2015.

[58] Carlos García-Martínez, Oscar Cordón, and Francisco Herrera. An empirical analysis of multiple objective ant colony optimization algorithms for the bi–criteria tsp. *European Journal of Operational Research*, 180(1):116–148, 2007.

[59] Ronald L Graham, Eugene L Lawler, Jan Karel Lenstra, and AHG Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics*, 5:287–326, 1979.

[60] Stephen C. Graves. A review of production scheduling. *Operations Research*, pages 646–675, 1981.

[61] Timothy Gutowski, Cynthia Murphy, David Allen, Diana Bauer, Bert Bras, Thomas Piwonka, Paul Sheng, John Sutherland, Deborah Thurston, and Egon Wolff. Environmentally benign manufacturing: Observations from japan, europe and the united states. *Journal of Cleaner Production*, 13(1):1–17, 2005.

[62] Isermann H. Proper efficiency and the linear vector maximum problem. *Operations Research*, 22:189–191, 1974.

[63] Hubert Hadera and Iiro Harjunkoski. Continuous–time batch scheduling approach for optimizing electricity consumption cost. In *23rd European Symposium on Computer Aided Process Engineering*, volume 32, page 403. Elsevier, 2013.

[64] Iskandar Halim and Rajagopalan Srinivasan. Sequential methodology for scheduling of heat–integrated batch plants. *Industrial & Engineering Chemistry Research*, 48(18):8551–8565, 2009.

[65] Michael Pilegaard Hansen and Andrzej Jaszkiewicz. Evaluating the quality of approximations to the non-dominated set. *Journal Für Die Reine Und Angewandte Mathematik*, 405(2):352–4, 1998.

[66] Cheng He, Y. T. Leung, Kangbok Lee, and Michael L. Pinedo. Scheduling a single machine with parallel batching to minimize makespan and total rejection cost. *Discrete Applied Mathematics*, 204(C):150–163, 2016.

[67] Yoshiro Ikura and Mark Gimple. Efficient scheduling algorithms for a single batch processing machine. *Operations Research Letters*, 5(2):61–65, 1986.

[68] Min Ji, Jen Ya Wang, and Wen Chiung Lee. Minimizing resource consumption on uniform parallel machines with a bound on makespan. *Computers & Operations Research*, 40(12):2970–2974, 2013.

[69] Zhao Hong Jia, Kai Li, and Y. T. Leung. Effective heuristic for makespan minimization in parallel batch machines with non–identical capacities. *International Journal of Production Economics*, 169:1–10, 2015.

[70] Nicolas Jozefowiez, Frédéric Semet, and El Ghazali Talbi. The bi–objective covering tour problem. *Computers & Operations Research*, 34(7):1929–1942, 2007.

[71] Imed Kacem, Slim Hammadi, and Pierre Borne. Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 32(1):1–13, 2002.

[72] A H Kashan and B. Karimi. Scheduling a single batch–processing machine with arbitrary job sizes and incompatible job families: An ant colony framework. *Journal of the Operational Research Society*, 59(9):1269–1280, 2008.

[73] A. H. Kashan, B. Karimi, and F. Jolai. Effective hybrid genetic algorithm for minimizing makespan on a single–batch–processing machine with non–identical job sizes. *International Journal of Production Research*, 44(12):2337–2360, 2006.

[74] Ali Husseinzadeh Kashan, Behrooz Karimi, and S. M. T. Fatemi Ghomi. A note on minimizing makespan on a single batch processing machine with nonidentical job sizes. *Theoretical Computer Science*, 410(27):2754–2758, 2009.

[75] Ali Husseinzadeh Kashan, Behrooz Karimi, and Masoud Jenabi. A hybrid genetic heuristic for scheduling parallel batch processing machines with arbitrary job sizes. *Computers & Operations Research*, 35(4):1084–1098, 2008.

[76] David Kathan. Assessment of demand response and advanced metering: Staff report 2012. *Washington: Federal Energy Regulatory Commission*, 2012.

[77] S G Koh, P H Koo, and J W Ha. Scheduling parallel batch processing machines with arbitrary job sizes and incompatible job families. *International Journal of Production Research*, 42(19):4091–4107, 2004.

[78] Shie Gheun Koh, Pyung Hoi Koo, Dong Chun Kim, and Won Suk Hur. Scheduling a single batch processing machine with arbitrary job sizes and incompatible job families. *International Journal of Production Economics*, 98(1):81–96, 2005.

[79] Liulin Kong, Hanbin Luo, and Heng Li. Optimal single–machine batch scheduling for the manufacture, transportation and jit assembly of precast construction with changeover costs within due dates. *Automation in Construction*, 81:34–43, 2017.

[80] C-Y Lee. Minimizing makespan on a single batch processing machine with dynamic job arrivals. *International Journal of Production Research*, 37(1):219–236, 1999.

[81] Chung-Yee Lee, Reha Uzsoy, and Louis A Martin-Vega. Efficient algorithms for scheduling semiconductor burn–in operations. *Operations Research*, 40(4):764–775, 1992.

[82] Wonkyun Lee, Seong Hyeon Kim, Jaesang Park, and Byung Kwon Min. Simulation–based machining condition optimization for machine tool energy consumption reduction. *Journal of Cleaner Production*, 150:352–360, 2017.

[83] Markus Leitner, Ivana Ljubić, and Markus Sinnl. Solving the bi–objective prize–collecting steiner tree problem with the ε–constraint method. *Electronic Notes in Discrete Mathematics*, 41:181–188, 2013.

[84] Pieter Leyman and Mario Vanhoucke. A new scheduling technique for the resource–constrained project scheduling problem with discounted cash flows. *International Journal of Production Research*, 53(9):2771–2786, 2015.

[85] Chung Lun Li and Chung Yee Lee. Scheduling with agreeable release times and due dates on a batch processing machine. *European Journal of Operational Research*, 96(3):564–569, 1997.

[86] Lin Li, Zeyi Sun, and Zhijun Tang. Real time electricity demand response for sustainable manufacturing systems: challenges and a case study. In *2012 IEEE International Conference on Automation Science and Engineering (CASE)*, pages 353–357. IEEE, 2012.

[87] Shuguang Li. Parallel batch scheduling with inclusive processing set restrictions and non–identical capacities to minimize makespan. *European Journal of Operational Research*, 260:12–20, 2016.

[88] Shuguang Li, Guojun Li, Xiaoli Wang, and Qiming Liu. Minimizing makespan on a single batching machine with release times and non–identical job sizes. *Operations Research Letters*, 33(2):157–164, 2005.

[89] Xiaolin Li. *Research on Scheduling Batch Processing Machines in Parallel*. CNKI, 2012.

[90] XiaoLin Li, HuaPing Chen, Bing Du, and Qi Tan. Heuristics to schedule uniform parallel batch processing machines with dynamic job arrivals. *International Journal of Computer Integrated Manufacturing*, 26(5):474–486, 2013.

[91] Xiaolin Li, YanLi Huang, Qi Tan, and HuaPing Chen. Scheduling unrelated parallel batch processing machines with non–identical job sizes. *Computers & Operations Research*, 40(12):2983–2990, 2013.

[92] Zhongya Li, Huaping Chen, Rui Xu, and Xueping Li. Earliness–tardiness minimization on scheduling a batch processing machine with non–identical job sizes. *Computers & Industrial Engineering*, 87:590–599, 2015.

[93] Cheng-Hsiang Liu and Ding-Hsiang Huang. Reduction of power consumption and carbon footprints by applying multi–objective optimisation via genetic algorithms. *International Journal of Production Research*, 52(2):337–352, 2014.

[94] Peiji Liu, Fei Liu, and Hang Qiu. A novel approach for acquiring the real–time energy efficiency of machine tools. *Energy*, 121:524–532, 2017.

[95] Lingfa Lu and Jinjiang Yuan. Unbounded parallel batch scheduling with job delivery to minimize makespan. *Operations Research Letters*, 36(4):477–480, 2008.

[96] Shenpeng Lu, Haodi Feng, and Xiuqian Li. Minimizing the makespan on a single parallel batching machine. *Theoretical Computer Science*, 411(7):1140–1145, 2010.

[97] Shyi-Min Lu, Ching Lu, Kuo-Tung Tseng, Falin Chen, and Chen-Liang Chen. Energy–saving potential of the industrial sector of taiwan. *Renewable and Sustainable Energy Reviews*, 21:674–683, 2013.

[98] Hao Luo, Bing Du, George Q Huang, Huaping Chen, and Xiaolin Li. Hybrid flow shop scheduling considering machine electricity consumption cost. *International Journal of Production Economics*, 146(2):423–439, 2013.

[99] Sujay Malve and Reha Uzsoy. A genetic algorithm for minimizing maximum lateness on parallel identical batch processing machines with dynamic job arrivals and incompatible job families. *Computers & Operations Research*, 34(10):3016–3028, 2007.

[100] M. Mathirajan, V. Chandru, and A. I. Sivakumar. Heuristic algorithms for scheduling heat–treatment furnaces of steel casting industries. *Sadhana*, 32(5):479–500, 2007.

[101] M Mathirajan and AI Sivakumar. A literature review, classification and simple meta–analysis on scheduling of batch processors in semiconductor. *The International Journal of Advanced Manufacturing Technology*, 29(9–10):990–1001, 2006.

[102] George Mavrotas. Effective implementation of the ε–constraint method in multi–objective mathematical programming problems. *Applied Mathematics and Computation*, 213(2):455–465, 2009.

[103] Sharif Melouk, Purushothaman Damodaran, and Ping Yu Chang. Minimizing makespan for single machine batch processing with non-identical job sizes using simulated annealing. *International Journal of Production Economics*, 87(2):141–147, 2004.

[104] Hadi Mokhtari and Aliakbar Hasani. An energy–efficient multi–objective optimization for flexible job–shop scheduling problem. *Computers & Chemical Engineering*, 104:339–352, 2017.

[105] Joon-Yung Moon, Kitae Shin, and Jinwoo Park. Optimization of production scheduling with time–dependent and machine–dependent electricity cost for industrial energy efficiency. *The International Journal of Advanced Manufacturing Technology*, 68(1–4):523–535, 2013.

[106] Joon-Yung Moon, Kitae Shin, and Jinwoo Park. Optimization of production scheduling with time–dependent and machine-dependent electricity cost for industrial energy efficiency. *The International Journal of Advanced Manufacturing Technology*, 68(1–4):523–535, 2013.

[107] Baruch Mor and Gur Mosheiov. Batch scheduling on uniform machines to minimize total flow–time. *Computers & Operations Research*, 39(3):571–575, 2012.

[108] Gilles Mouzon and Mehmet B Yildirim. A framework to minimise total energy consumption and total tardiness on a single machine. *International Journal of Sustainable Engineering*, 1(2):105–116, 2008.

[109] Gilles Mouzon, Mehmet B Yildirim, and Janet Twomey. Operational methods for minimization of energy consumption of manufacturing equipment. *International Journal of Production Research*, 45(18–19):4247–4271, 2007.

[110] Mehdi Nikzad, Babak Mozafari, Mahdi Bashirvand, Soodabeh Solaymani, and Ali Mohamad Ranjbar. Designing time–of–use program based on stochastic security constrained unit commitment considering reliability index. *Energy*, 41(1):541–548, 2012.

[111] Tatsuya Okabe, Yaochu Jin, and Bernhard Sendhoff. A critical survey of performance indices for multi–objective optimisation. In *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, volume 2, pages 878–885. IEEE, 2003.

[112] David Pisinger. A minimal algorithm for the 0–1 knapsack problem. *Operations Research*, 45(5):758–767, 1997.

[113] David Pisinger. An exact algorithm for large multiple knapsack problems. *European Journal of Operational Research*, 114(3):528–541, 1999.

[114] Chris N Potts and Mikhail Y Kovalyov. Scheduling with batching: a review. *European Journal of Operational Research*, 120(2):228–249, 2000.

[115] Marco Pranzo. Batch scheduling in a two–machine flow shop with limited buffer and sequence independent setup times and removal times. *European Journal of Operational Research*, 153(3):581–592, 2004.

[116] Markus Rager, Christian Gahm, and Florian Denz. Energy–oriented scheduling based on evolutionary algorithms. *Computers & Operations Research*, 54:218–231, 2015.

[117] Peter Reiter and Walter J Gutjahr. Exact hybrid algorithms for solving a bi–objective vehicle routing problem. *Central European Journal of Operations Research*, 20(1):19–43, 2012.

[118] Atle Riise, Carlo Mannino, and Edmund K. Burke. Modelling and solving generalised operational surgery scheduling problems. *Computers & Operations Research*, 66:1–11, 2016.

[119] Jason R Schott. Fault tolerant design using single and multicriteria genetic algorithm optimization. Technical report, Massachusetts Institute of Technology, 1995.

[120] B. Shahidi-Zadeh, R. Tavakkoli-Moghaddam, A. Taheri-Moghadam, and I. Rastgar. Solving a bi–objective unrelated parallel batch processing machines scheduling problem: A comparison study. *Computers & Operations Research*, 88:71–90, 2017.

[121] Fadi Shrouf, Joaquin Ordieres-Meré, Alvaro García-Sánchez, and Miguel Ortega-Mier. Optimizing the production scheduling of a single machine to minimize total energy consumption costs. *Journal of Cleaner Production*, 67:197–207, 2014.

[122] Nurul Suhaimi, Christine Nguyen, and Purushothaman Damodaran. Lagrangian approach to minimize makespan of non–identical parallel batch processing machines. *Computers & Industrial Engineering*, 101:295–302, 2016.

[123] Chang S Sung and YI Choung. Minimizing makespan on a single burn–in oven in semiconductor manufacturing. *European Journal of Operational Research*, 120(3):559–574, 2000.

[124] Vishnu Swaminathan and Krishnendu Chakrabarty. Energy–conscious, deterministic i/o device scheduling in hard real–time systems. *IEEE Transactions on Computer–Aided Design of Integrated Circuits and Systems*, 22(7):847–858, 2003.

[125] K. C. Tan, C. K. Goh, Y. J. Yang, and T. H. Lee. Evolving better population distribution and exploration in evolutionary multi–objective optimization. *European Journal of Operational Research*, 171(2):463–495, 2006.

[126] Lixin Tang, Xiaoli Zhao, Jiyin Liu, and Y. T. Leung. Competitive two–agent scheduling with deteriorating jobs on a single parallel–batching machine. *European Journal of Operational Research*, 2017.

[127] Vincent T'Kindt and Jean Charles Billaut. *Multicriteria Scheduling: Theory, Models and Algorithms*. Springer–Verlag, Berlin, Heidelberg, 2005.

[128] Fabien Tricoire, Alexandra Graf, and Walter J. Gutjahr. The bi–objective stochastic covering tour problem. *Computers & Operations Research*, 39(7):1582–1592, 2012.

[129] Reha Uzsoy. Scheduling a single batch processing machine with non–identical job sizes. *International Journal of Production Research*, 32(7):1615–1635, 1994.

[130] Reha Uzsoy and Yaoyu Yang. Minimizing total weighted completion time on a single batch processing machine. *Production & Operations Management*, 6(1):57–73, 1997.

[131] DJ Van De Rzee, A Van Harten, and PC Schuur. Dynamic job assignment heuristics for multi–server batch operations–a cost based approach. *International Journal of Production Research*, 35(11):3063–3094, 1997.

[132] David A Van Veldhuizen and Gary B Lamont. On measuring multiobjective evolutionary algorithm performance. In *Proceedings of the Congress on Evolutionary Computation*, volume 1, pages 204–211. IEEE, 2000.

[133] David Allen Van Veldhuizen. Multiobjective evolutionary algorithms: classifications, analyses, and new innovations. *Evolutionary Computation*, 8(2):125 – 147, 1999.

[134] David A. Van Veldhuizen and Gary B. Lamont. Multiobjective evolutionary algorithm test suites. In *ACM Symposium on Applied Computing*, pages 351–357, 1999.

[135] Cheng Shuo Wang and Reha Uzsoy. A genetic algorithm to minimize maximum lateness on a batch processing machine. *Computers & Operations Research*, 29(12):1621–1640, 2002.

[136] Shijin Wang and Ming Liu. A genetic algorithm for two–stage no–wait hybrid flow shop scheduling problem. *Computers & Operations Research*, 40(4):1064–1075, 2013.

[137] Shijin Wang and Ming Liu. A heuristic method for two–stage hybrid flow shop with dedicated machines. *Computers & Operations Research*, 40(1):438–450, 2013.

[138] Shijin Wang, Ming Liu, Feng Chu, and Chengbin Chu. Bi–objective optimization of a single machine batch scheduling problem with energy cost consideration. *Journal of Cleaner Production*, 137:1205–1215, 2016.

[139] Yong Wang and Lin Li. Time–of–use based electricity demand response for sustainable manufacturing systems. *Energy*, 63:233–244, 2013.

[140] Jin Wu and Shapour Azarm. Metrics for quality assessment of a multiobjective design optimization solution set. *Journal of Mechanical Design*, 123(1):18–25, 2001.

[141] Peng Wu. A study on lane reservation problems in transportation networks. *Ph.D thesis, University of Paris–Saclay, France*, 2016.

[142] Peng Wu, Ada Che, Feng Chu, and MengChu Zhou. An improved exact–constraint and cut–and–solve combined method for biobjective robust lane reservation. *IEEE Transactions on Intelligent Transportation Systems*, 16(3):1479–1492, 2015.

[143] Rui Xu, Huaping Chen, and Xueping Li. Makespan minimization on single batch–processing machine via ant colony optimization. *Computers & Operations Research*, 39(3):582–593, 2012.

[144] Shubin Xu and James C. Bean. A genetic algorithm for scheduling parallel non–identical batch processing machines. In *IEEE Symposium on Computational Intelligence in Scheduling*, pages 143–150, 2007.

[145] Mehmet Bayram Yildirim and Gilles Mouzon. Single–machine sustainable production planning to minimize total energy consumption and total completion time using a multiple objective genetic algorithm. *Engineering Management, IEEE Transactions on*, 59(4):585–597, 2012.

[146] Yunqiang Yin, Tai Chiu Edwin Cheng, Dujuan Wang, and Chin Chia Wu. Improved algorithms for single–machine serial–batch scheduling with rejection to minimize total completion time and total rejection cost. *IEEE Transactions on Systems Man & Cybernetics Systems*, pages 1–11, 2015.

[147] L. Zadeh. Optimality and non–scalar–valued performance criteria. *IEEE Transactions on Automatic Control*, 8(1):59–60, 1963.

[148] Guochuan Zhang, Xiaoqiang Cai, C-Y Lee, and Chak Kuen Wong. Minimizing makespan on a single batch processing machine with nonidentical job sizes. *Naval Research Logistics (NRL)*, 48(3):226–240, 2001.

[149] Hao Zhang, Fu Zhao, Kan Fang, and John W Sutherland. Energy–conscious flow shop scheduling under time–of–use electricity tariffs. *CIRP Annals–Manufacturing Technology*, 63(1):37–40, 2014.

[150] Rui Zhang, Pei-Chann Chang, Shiji Song, and Cheng Wu. A multi–objective artificial bee colony algorithm for parallel batch–processing machine scheduling in fabric dyeing processes. *Knowledge–Based Systems*, 116:114–129, 2017.

[151] Yuzhong Zhang, Zhigang Cao, and Qingguo Bai. A ptas for scheduling on agreeable unrelated parallel batch processing machines with dynamic job arrivals. In *International Conference on Algorithmic Applications in Management*, pages 162–171, 2005.

[152] Shengchao Zhou, Huaping Chen, and Xueping Li. Distance matrix based heuristics to minimize makespan of parallel batch processing machines with arbitrary job sizes and release times. *Applied Soft Computing*, 52:630–641, 2017.

[153] Shengchao Zhou, Huaping Chen, Rui Xu, and Xueping Li. Minimising makespan on a single batch processing machine with dynamic job arrivals and non–identical job sizes. *International Journal of Production Research*, 52(8):2258–2274, 2014.

[154] Shengchao Zhou, Ming Liu, Huaping Chen, and Xueping Li. An effective discrete differential evolution algorithm for scheduling uniform parallel batch processing machines with non–identical capacities and arbitrary job sizes. *International Journal of Production Economics*, 179:1–11, 2016.

[155] Zhen Zhou, Feng Chu, Ada Che, and MengChu Zhou. ε–constraint and fuzzy logic–based optimization of hazardous material transportation via lane reservation. *IEEE Transactions on Intelligent Transportation Systems*, 14(2):847–857, 2013.

[156] Eckart Zitzler. Evolutionary algorithms for multiobjective optimization: Methods and applications. *Ph.D. Thesis.*, 1999.

[157] Eckart Zitzler, Joshua Knowles, and Lothar Thiele. Quality assessment of pareto set approximations. In *Multiobjective Optimization*, pages 373–404, 2008.

[158] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. Spea2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. In *Evolutionary Methods for Design, Optimization and Control with Applications To Industrial Problems. Proceedings of the Eurogen'2001. Athens. Greece, September*, 2001.

[159] Eckart Zitzler and Lothar Thiele. *Multiobjective Optimization Using Evolutionary Algorithms – A Comparative Case Study.* Springer Berlin Heidelberg, 1998.

[160] Eckart Zitzler and Lothar Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 1999.

[161] Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M Fonseca, and Viviane Grunert Da Fonseca. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE transactions on evolutionary computation*, 7(2):117–132, 2003.

My publications

Journal papers

J. Cheng, F. Chu, and M. Zhou. An improved model for parallel machine scheduling under time–of–use electricity price. *IEEE Transactions on Automation Science and Engineering*, DOI:10.1109/TASE.2016.2631491, 2017.

J. Cheng, F. Chu, C. Chu, and W. Xia. Bi–objective optimization of single–machine batch scheduling under time–of–use electricity prices. *RAIRO – Operations Research*, 50(4–5):715–732, 2016.

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. *Computers & Industrial Engineering*, 112:721–734, 2017.

J. Cheng, F. Chu, M. Liu, and W. Xia. Effective heuristics for single machine batch scheduling under time-of-use tariffs. *Sustainability*, under review.

Conference papers

J. Cheng, F. Chu, M. Liu, and W. Xia. Electricity cost and makespan optimization on a single batch processing machine under time–of–use pricing policy. In *45th International Conference on Computers & Industrial Engineering, Metz, France*, pages 236–241, 2015.

J. Cheng, F. Chu, M. Liu , and W. Xia. Single–machine batch scheduling under time–of–use tariffs: new mixed–integer programming approaches. In *2016 IEEE International conference on Systems, Man, and Cybernetics, Budapest, Hungary*, pages 3498–3503, 2016.

J. Cheng, F. Chu, and W. Xia. Single machine batch scheduling under time–of–use policy. *16ème Conférence ROADEF Société Française de Recherche Opérationnelle et Aide à la Décision*, 2015.

J. Cheng, F. Chu, W. Xia, J. Ding, and X. Ling. Bi–objective optimization for single–machine batch scheduling considering energy cost. In *2014 IEEE International Conference on Control, Decision and Information Technologies*, 236–241, 2014.

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre : Ordonnancement multicritère par lots avec tarifs d'électricité différenciés

Mots clés : Optimisation bi-objectif, Ordonnancement par lots, Tarification d'électricité différenciée, Programmation mathématique, Méthodes de ε-contrainte.

Résumé : L'industrie est le plus grand consommateur d'énergie dans le monde et la majeure partie de sa consommation est électrique. Pour moduler la consommation et équilibrer les périodes creuses et de pic, les producteurs d'électricité dans de nombreux pays pratiquent une tarification différenciée, en anglais "time-of-use (TOU) policy", afin d'encourager les industriels et les particuliers à adapter leur consommation. Cette stratégie incite les gros consommateurs industriels, en particulier le secteur semi-conducteur où la fabrication se fait souvent par lots, à réduire leurs factures d'électricité en adaptant leur production. Dans ce travail, nous étudions plusieurs problèmes d'ordonnancement de production par lots avec tarification différenciée d'électricité. Nous nous intéressons d'abord à l'ordonnancement d'une machine par lots pour minimiser le coût total d'électricité et le makespan. Le deuxième problème étudié généralise le premier en considérant le coût d'électricité pendant les périodes inactives de la machine telles que les périodes de réglage ou d'attente. Enfin, nous traitons l'ordonnancement sur machines parallèles par lots avec des pièces non identiques. Pour chacun de ces problèmes, nous construisons des modèles mathématiques appropriés, et évaluons sa complexité. Pour la résolution, nous proposons plusieurs méthodes de ε-contrainte dans lesquelles des sous-problèmes sont transformés en problèmes de sac-à-doc, de sacs-à-doc multiples et ou de bin packing. Nous développons aussi une méthode itérative à deux étapes. Les performances des méthodes développées sont évaluées à l'aide d'un grand nombre d'instances représentatives générées au hasard. Les résultats numériques montrent l'efficacité de ces méthodes par rapport au logiciel commercial CPLEX.

Title : Multi-criteria Batch Scheduling under Time-of-Use Tariffs

Keywords : Batch scheduling, Multi-objective combinatorial optimization, Time-of-use electricity tariffs, Mixed-integer linear programming, Algorithms.

Abstract : The industrial sector is the largest consumer of the world's total energy and most of its consumption form is electricity. To strengthen the grid's peak load regulation ability, time-of-use (TOU) electricity pricing policy has been implemented in many countries to encourage electricity users to shift their consumption from on-peak periods to off-peak periods. This strategy provides a good opportunity for manufacturers to reduce their energy bills, especially for energy-intensive ones, where batch scheduling is often involved. In this thesis, several bi-objective batch scheduling problems under TOU tariffs are studied. We first investigate a single machine batch scheduling problem under TOU tariffs with the objectives of minimizing total electricity cost and makespan. This primary work is extended by considering machine on/off switching. Finally, a parallel batch machines scheduling problem under TOU tariffs with non-identical job sizes to minimize total electricity cost and number of enabled machines is studied. For each of the considered problems, appropriate mathematical models are established, their complexities are demonstrated. Different bi-objective resolution methods are developed, including knapsack problem heuristic based ε-constraint method, multiple knapsack problem heuristic based ε-constraint method, bin packing heuristic based ε-constraint method and two-stage heuristic based iterative search algorithm. The performance of proposed methods is evaluated by randomly generated instances. Extensive numerical results show that the proposed algorithms are more efficient and/or effective than the commercial software CPLEX.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France