European J. of Industrial Engineering

ISSN online: 1751-5262 - ISSN print: 1751-5254
https://www.inderscience.com/ejie

An auction mechanism for capacity allocation in identical parallel machines with time window constraints

Qianqian Zhu, Xiuli Wang

DOI: 10.1504/EJIE.2023.10050357

Article History:

Received:	20 April 2021
Accepted:	06 May 2022
Published online:	06 November 2023

An auction mechanism for capacity allocation in identical parallel machines with time window constraints

Qianqian Zhu and Xiuli Wang*

School of Economics and Management,
Nanjing University of Science and Technology,
Nanjing 210094, China
Email: zhuqq0819@163.com
Email: wangdu0816@163.com
*Corresponding author

Abstract: We study the scarce production capacity allocation problem in a decentralised decision-making environment. We focus on the design of an auction mechanism for effective allocation of scarce capacity, without private information. In our problem setting, the firm's machine environment is identical parallel machines, and each customer order must be processed within a time window. Here, resource scarcity depends not only on capacity, but also on the customer orders' time window constraints. Hence, we propose an ascending auction with a discriminatory pricing scheme for customers, to identify the real processing requirements of the customer orders and resolve resource conflicts. In our auction, the winner determination problem is NP-complete, we develop a heuristic to solve this problem using the Lagrangian relaxation technique. The computational study shows that our auction mechanism achieves over 93% of the global optimal value. [Submitted 20 April 2021; Accepted 6 May 2022]

Keywords: capacity allocation; auction mechanism; price discrimination; Lagrangian relaxation.

Reference to this paper should be made as follows: Zhu, Q. and Wang, X. (2023) 'An auction mechanism for capacity allocation in identical parallel machines with time window constraints', *European J. Industrial Engineering*, Vol. 17, No. 6, pp.833–855.

Biographical notes: Qianqian Zhu is a PhD candidate in Management Science and Engineering at the Nanjing University of Science and Technology. Her research interests are production scheduling and mechanism design.

Xiuli Wang is a Professor for Management Science and Engineering at the Nanjing University of Science and Technology. His research interests are production scheduling, supply chain management and service operation management.

1 Introduction

In many industries, firms adopt the make-to-order (MTO) strategy to fulfil orders with unique processing requirements. It is not possible to allocate the scarce capacity to all potential orders. Finding a solution for the allocation of scarce capacity to the right orders requires consideration of all private information of the orders. However, in a decentralised decision-making environment, the firm and customers share only limited private information in order to serve their own interests. As a result, local considerations will deteriorate global interests, and customers will have conflicting interests (Kutanoglu and Wu, 1999).

In this study, the firm possesses identical parallel machines. At the beginning of the planning horizon, the firm receives a set of customer orders, each with a release time and a deadline. The firm must either process the order within the time window constraint that be comprised of the release time and deadline, or reject the order; on the other hand, when the price given by the firm for processing the order is too high, the customer will decline processing. Operating in such an environment, we aim to design a mechanism that effectively and efficiently allocates the scarce capacity without considering all private information. Specifically, the mechanism should enable self-interested participants to make decisions that advance collective goals without knowing other private information, and the mechanism should be able to allocate the capacity within a reasonable time.

To the best of our knowledge, the auction mechanism is a suitable method to allocate scarce capacity in a decentralised decision-making environment (Wellman et al., 2001; McAfee and McMillan, 1987). Hall and Liu (2013) suggest that an ascending auction is more efficient and profitable than a sealed bid auction in production scheduling problems. Henceforth, we design a multi-round ascending auction mechanism to allocate scarce capacity to a number of competing customers. The main research work is as follows.

First, we design an auction mechanism in which the firm sets the price with discrimination for customers. In our discriminatory pricing scheme, we use

a the length and location of the order's time window

b the flexibility of order processing relative to its time window to describe the scarcity of the capacity.

The discriminate prices can assist the firm in distinguishing customers with different processing requirements, and allocating the capacity effectively.

Second, the winner determination (WD) problem of the auction mechanism needs to make simultaneous order selection and production scheduling decisions, which can be modelled as an order acceptance and scheduling (OAS) problem in identical parallel machines with time window constraints. A special case of the OAS problem on a single machine where all orders have the same time window constraints is equivalent to the Knapsack Problem. It is well known that the Knapsack Problem is an NP-complete problem (Pinedo, 1995). So the WD problem is an NP-complete problem. We develop a heuristic using the Lagrangian relaxation technique to solve the WD problem.

We structure the remainder of the paper as follows. Section 2 provides a literature review of related research. Section 3 outlines a detailed description of our problem setting. In Section 4, we present an auction mechanism to allocate the production capacity, and Section 5 outlines a computational study of the auction mechanism. Finally, Section 6 presents concluding remarks and future research directions.

2 Literature review

2.1 Auction for resource allocation problems

With the application of auction theory in radio spectrum rights (Jackson, 1976), auction theory, as a subfield of economics, has been attracting increasing research attention on resource allocation problems over the past few decades. Spectrum allocation is to regulate the use of the spectrum and divide it among various competing interests and organisations. The spectrum allocation problems are similar to the problem we consider, because of the scarcity and complementarity of resource. McMillan (1994, 1995), Cramton (2002), Kasbekar and Sarkar (2010) and Cramton and Ockenfels (2017) describe auctions for spectrum allocation problem. Auction theory also has been applied to many other fields, such as logistics services, electrical power and computational grids. See for examples, Ledyard et al. (2002), Yang et al. (2019) and Zhang et al. (2019) propose auction-based approaches for allocation of logistics services. Nicolaisen et al. (2001) and Voss and Madlener (2017) study auctions for electrical power. Das and Grosu (2005), Izakian et al. (2010) and Kaushik and Vidyarthi (2015, 2018) introduce the auction methods for grid resource management. However, using an auction mechanism to allocate production capacity differs from these studies in that it requires consideration of the heterogeneity of customer orders and the time dimension of the resource.

A few papers put forward the auction mechanisms for production capacity allocation problems. Wellman et al. (2001) consider the single machine capacity allocation problem with several customers each with a single order. They design an auction mechanism that uses time slots as market goods. Dewan and Joshi (2002) present an auction mechanism for the distributed scheduling problem in a job shop environment. The objective of their problem is to minimise earliness-tardiness penalties. They also introduce iterative price adjustments to reduce resource conflict. Hall and Liu (2013) propose an ascending auction mechanism for the allocation of the scarce single machine capacity among competing customers, each with a single order. They use time blocks as market goods, and apply flexibility to market goods design. Karabatı and Yalçın (2014) consider the integrated pricing and single machine capacity allocation problem in a decentralised decision-making environment. They propose an auction mechanism that takes the finished products as market goods. Our work is an extension of the research of Hall and Liu (2013), and it is different from the above studies in the following aspects. First, the firm's machine environment is identical parallel machines, which is common in the industry, the above studies present the auction mechanism in one single machine and job shop environment. Second, the customer orders in our study have time window constraints, which also affect the resource scarcity. These characteristics promote us to propose an auction mechanism with a new pricing scheme and an effective WD algorithm.

2.2 OAS problems

In our study, the firm's WD problem is to make a profit maximisation decision on the OAS problem in identical parallel machines with time window constraints. The OAS problem is initiated by Slotnick and Morton (1996) and Ghosh (1997). To the best of our knowledge, most studies consider the OAS problem on a single machine. For examples, Slotnick and Morton (2001) present an optimal branch-and-bound procedure for the OAS problem to minimise the weighted tardiness, which uses an integer relaxation for bounding. Rom and Slotnick (2009) use a genetic algorithm to solve the OAS problem with tardiness penalties. Oğuz et al. (2010) formulate a mixed integer linear program (MILP) for the OAS problem with tardiness and deadline, they also develop three heuristic algorithms to solve large-sized instances. Cesaret et al. (2012) present a tabu search algorithm that solves the OAS problem with release times and sequence-dependent setup times. Zhong et al. (2014) consider the OAS problem with machine availability constraints and study the approximability of the model. Chaurasia and Singh (2017) present two hybrid metaheuristic approaches for the OAS problem with release times and sequence-dependent setup times.

Recently, the OAS problems in more complicated machine environments have also been studied. Wang et al. (2013) study the OAS problem in a two-machine flow shop, they formulate the problem as MILP models and develop a heuristic and a branch-and-bound algorithm. Lei and Guo (2015) study the OAS problem in a flow shop environment, they formulate the problem as a MILP model and develop a parallel neighbourhood search algorithm. Esmaeilbeigi et al. (2016) consider the OAS problem in a two-machine flow shop, they present two new MILP formulations and develop several techniques to improve the formulations. Jiang et al. (2017) develop two approximation algorithms for an OAS problem with batch delivery on parallel machines. Naderi and Roshanaei (2020) study an OAS problem in identical parallel machines, they formulate the problem as a new MILP model and propose a novel branch-relax-and-check exact method for solving the model. In our problem setting, each customer order has a time window constraint. It is therefore different from that of Jiang et al. (2017) and Naderi and Roshanaei (2020), and their methods are not applicable.

3 Problem statement

We present the description of the problem setting as follows. The firm's machine environment is m identical parallel machines, and the planning horizon spans a time period $t = 1, 2, ..., T$. The firm sets a reserve value v for each time slot $[t-1, t]$. At the beginning of the planning horizon, the firm receives a set of customer orders $N = \{1, 2, ..., n\}$, where the release time, processing time, deadline and revenue of order $i(i \in N)$ are r_i, p_i, d_i, and u_i, respectively. We assume that the release times, processing times, and the deadlines of all orders are integers.

The firm maximises his profit by selling the capacity and holding any unallocated capacity at its reserve value. Each customer maximises profit by purchasing capacity to produce order, so as to get its revenue. Let α_i be the unit price for purchasing capacity to process order i. If order i can be processed after the release time r_i and finished before the deadline d_i on any machine, the customer will obtain a profit $u_i - \alpha_i p_i$, and the firm will earn a profit $\alpha_i p_i$ from order i. The firm's profit is the profit of the scheduled time slots plus the reserve value of the unscheduled time slots.

In decentralised decision-making environment, we assume that the processing requirements and revenues of customer orders are private information. Hence, we cannot make centralised decision of the capacity allocation problem. We design an auction mechanism that solves the private information case in Section 4.

4 Auction mechanism

In this section, we design an auction mechanism by which the firm can effectively allocate his capacity when the customers do not share all private information. The challenges for the design of the auction mechanism include: How to pass the messages between the firm and customers? How to guide the customers to transfer the messages in a way that is beneficial to effective resource allocation? How to determine the final schedule efficiently? For the first question, we propose auction protocols to define the responsibilities of all participants. For the second problem, we propose a pricing scheme as a dominant strategy for the customers to reveal their truthful processing requirements in their bids. For the third problem, we propose a heuristic to solve the WD problem using the Lagrangian relaxation technique.

4.1 Auction protocols

We propose a multi-round auction with increasing prices. The auctioneer is the firm, the market good is defined as a combination of continuous time slots, and the bidders are the customers who have orders to be processed. The auction protocols are described as follows: in each round,

a the firm updates the price for each customer to increase its profit over previous rounds

b all the customers are allowed to bid simultaneously to process orders and, the processing requirements of the customer orders cannot be changed, once they are confirmed at first round

c the firm determines which bids to admit, so as to maximise his profit from the submitted bids

d if none of the customers submit a new bid, the auction is terminated.

The auction protocols a to d are explained as follows: protocol a implies that the ask prices are ascending. In an ascending auction, keeping bidding until the ask price reaches its real revenue is each customer's dominant strategy. Compared with the second-price sealed bid auction, the ascending auction can generate higher profit for the firm. In protocol b, the processing requirements of each customer order include the processing time and processing time window, which are exogenous variables of the auction. Protocols c and d refer to Hall and Liu (2013). Protocol c is adopted because the objective of the firm is to maximise his own profit. It is obvious that the ascending auction will be terminated within a limited number of rounds. Compared with the fixed number of rounds, the firm can obtain higher profit by adopting the termination way in protocol d.

In each round of the auction, the firm needs to solve the pricing problem and the WD problem. Let k be the round index, $\alpha^k = (\alpha_1^k, \alpha_2^k, ..., \alpha_n^k)^T$ be the firm's unit time slot price for customers in round k. Let $B_i^k = (p_i, r_i, d_i, \alpha_i^k), i \in N$ be the bids submitted by the customers in round k, where α_i^k is the unit time slot price for customer i. Let $W^k(i), i \in N$ be an indicator function that denotes whether or not customer i 's bid is a winning bid in round k. We now formally describe the auction mechanism.

Procedure auction

Step 0 Initialisation: Set $w = 0$, $B_i^0 = 0$, $W^0(i) = 0$, $i \in N$.

Step 1 Set the round counter $k = k + 1$. The firm solves the pricing problem to update the price $\alpha^k = (\alpha_1^k, \alpha_2^k, ..., \alpha_n^k)$ for customers (see Subsection 4.2).

Step 2 For customers $i \in N$, if $W^{k-1}(i) = 1$, let $B_i^k = B_i^{k-1}$; if $W^{k-1}(i) = 0$, customer i submits a new bid $B_i^k = (p_i, r_i, d_i, \alpha_i^k)$.

Step 3 The firm gathers all bids and solves the WD problem to generate a temporary schedule $W^k(i), i \in N$ in round k of the auction (see Subsection 4.3).

Step 4 If none of the customers submit a new bid, where $B_i^k = B_i^{k-1}, i \in N$. The auction is terminated with the allocation $W^k(i), i \in N$, else go to step 1.

4.2 Pricing problem

We characterise the pricing problem by two elements:

a there is price discrimination among customers

b in each round, the price is adaptive and depends on the rounds and progress of the auction.

The goal of price updating is to resolve resource conflicts among customers. When the demand exceeds supply, conflict arises. We denote the price for each customer $i (i \in N)$ in round k by α_i^k. It is designed by two parts: current price and price increment.

We first define the current price for customer $i (i \in N)$ in round k, denoted by $\bar{\alpha}_i^k$, based on the temporary schedule in round $k - 1$:

1 If there is no time slots within $[r_i, d_i]$ allocated in round $k - 1$, then $\bar{\alpha}_i^k = v$.

2 If there are some or all of the time slots within $[r_i, d_i]$ allocated in round $k - 1$, then calculate $\bar{\alpha}_i^k$ as the weighted average of the values of all time slots within $[r_i, d_i]$, including the total bid prices of the allocated time slots and the total reserve value of the unallocated time slots.

Assuming bids $1, 2, ..., n_i$ are winning the time slots within $[r_i, d_i]$ at prices α_1^{k-1}, α_2^{k-1}, ..., $\alpha_{n_i}^{k-1}$. The first and last bids may be partially processed within $[r_i, d_i]$. Let $\bar{p}_j^{k-1} (j = n'_{i1}, ..., n'_{i2})$ be the length of the time slots within $[r_i, d_i]$ allocated to order j in round $k - 1$. Let I_i^{k-1} be the idle times within $[r_i, d_i]$ in round $k - 1$.

The expression of $\bar{p}_j^{k-1}$ is described as follows:

$$\bar{p}_j^{k-1} = \begin{cases} \min\{p_j, C_j^{k-1} - r_i\} & \text{if } j = n'_{i1} \\ p_j & \text{if } j = 2, ..., n'_{i2} - 1 \\ p_j - \max\{0, C_j^{k-1} - d_i\} & \text{if } j = n'_{i2} \end{cases}$$

where n'_{i1}, n'_{i2} are the first and last bids in one of the machines, $C_j^{k-1}(j = n'_{i1}, ..., n'_{i2})$ is bid j's completion time. The same calculation applies to the length of the time slots within $[r_i, d_i]$ allocated to bids on the other machines.

Hence, the current price $\bar{\alpha}_i^k$ is defined as follows:

$$\bar{\alpha}_i^k = \frac{\sum_{h=1}^{n_i} \alpha_h^{k-1} \bar{p}_h^{k-1} + I_i^{k-1} v}{m(d_i - r_i)}.$$

Second, we define the price increment for customer $i(i \in N)$ in round k, denoted by ϵ_i^k, based on the length and location of the bid's time window:

1 If there is no time slots within $[r_i, d_i]$ are allocated in round $k - 1$, then

$$\epsilon_i^k = \frac{\rho_1 p_i}{m(d_i - r_i)},$$

2 If there are some or all of the time slots within $[r_i, d_i]$ allocated in round $k - 1$, then

$$\epsilon_i^k = \frac{\rho_1 p_i}{m(d_i - r_i)} + \frac{\rho_2 \sum_{t=r_i+1}^{d_i} D_t^{k-1}}{m(d_i - r_i)N},$$

where ρ_1 and ρ_2 are price adjustment factors, they are pre-determined constants. ρ_1 adjusts the price according to the impact of order processing flexibility on prices, and ρ_2 adjusts the price according to the impact of resource conflicts on prices. If they are set too large or too small, the auction may miss the optimal solution or be low efficient. However, these two price adjustment factors should be determined experimentally. D_t^{k-1} is the number of bids received in round $k - 1$ that containing machine time slot t.

In our price increment function,

a We use the length and location of the order's time window, and the flexibility of order processing to define the impact on resource scarcity caused by potential orders. The less flexibility of order processing, the greater the impact on resource scarcity and the larger the price increment.

b We use the number of bids got in the last round to define the demand of these time slots within the order's time window. The more the demand, the greater the conflicts over resources among customers and the larger the price increment.

As a result, the discriminatory-linear price in round k for customer $i(i \in N)$ is described as follows:

$$\alpha_i^k = \max\{\alpha_i^{k-1} + \xi, \bar{\alpha}_i^k + \epsilon_i^k\}. \tag{1}$$

where ξ is a minimal positive number to ensure that $\alpha_i^k > \alpha_i^{k-1}$.

In our pricing scheme, we see that the prices are increasing during the auction, if the revenues of the customer orders are bounded, the auction is guaranteed to terminate, and the final winners' prices are closer to their true revenues. We also see that the price at the first round for customer $i(i \in N)$ is equal to $v + \frac{\rho_1 p_i}{m(d_i - r_i)}$, which discriminates based on his processing requirement. Hence, each customer's dominant strategy is to bid by his truthful processing requirement. The protocols of the auction describe that the customers are not allowed to change their processing requirements in the subsequent rounds. Thus, the customers will always bid by their truthful processing requirements, which will significantly improve the effectiveness of the firm's capacity allocation problem.

4.3 WD problem

The WD problem at each round involves maximisation of the firm's profit by selecting the customer bids and scheduling the accepted bids to the identical parallel machines for processing. The WD problem is NP-complete, we developed a fast heuristic that applies the Lagrangian relaxation technique to solve large-sized instances of the problem approximately. We first determine the subset of the accepted bids by the Lagrangian relaxation method, and then generate a feasible allocation from these accepted bids.

4.3.1 Integer linear programming formulation

Without loss of generality, we assume that the bids are numbered in non-decreasing order of their release times, with ties broken by the earliest deadline first. The objective of the WD problem is to maximise the firm's total profit which includes the profit of the sold capacity and the total reserve value of the unsold capacity. Let binary decision variables $x_{it} = 1$ denote that bid i is accepted and completed at time t, otherwise, $x_{it} = 0$, for $i \in N$ and $t = 1, 2, ..., T$. We formulate the WD problem as an integer linear programming model:

$$(ILP) \quad \max \sum_{i=1}^{n} \sum_{t=1}^{T} (\alpha_i - v) p_i x_{it} + mvT \tag{2}$$

$$\text{subject to } \sum_{i=1}^{n} \sum_{s=\max\{r_i+p_i,t\}}^{\min\{t+p_i-1,T\}} x_{is} \le m, \qquad \forall 1 \le t \le T \tag{3}$$

$$\sum_{t=1}^{T} x_{it} \le 1, \qquad \forall i \in N \tag{4}$$

$$\sum_{t=1}^{r_i+p_i-1} x_{it} = 0, \qquad \forall i \in N \tag{5}$$

$$\sum_{t=d_i+1}^{T} x_{it} = 0, \qquad \forall i \in N \tag{6}$$

$$x_{it} \in \{0, 1\}, \qquad \forall i \in N, 1 \le t \le T \tag{7}$$

In ILP, function (2) denotes the profit of the sold capacity $\sum_{i=1}^{n}\sum_{t=1}^{T}\alpha_i p_i x_{it}$ plus the total reserve value of the unsold capacity $mvT - \sum_{i=1}^{n}\sum_{t=1}^{T} vp_i x_{it}$. Constraint (3) states that in each time slot, at most m bids can be processed simultaneously, because of the limited capacity of the m machines. Constraint (4) states that the accepted bids should be processed exactly once. Constraints (5) and (6) make sure that the accepted bids are processed after their release times and completed before their deadlines, respectively.

We identify a property of an optimal solution which will be used to develop heuristic for the WD problem.

Lemma 4.1: For the WD problem, there is an optimal solution in which if at least one bid of A_i is accepted and processed between its release time and deadline, then bid i is also accepted for processing, where $A_i = \{j | r_j \geq r_i,\ d_j \leq d_i,\ p_j \geq p_i$ and $u_j < u_i\}$, $i \in N$.

Proof: Let π^* be an optimal solution for the WD problem. Suppose that bid i is not in π^*, but bid j in set A_i is accepted and began at time t in π^*, where $A_i = \{j | r_j \geq r_i$, $d_j \leq d_i,\ p_j \geq p_i$ and $u_j < u_i\}$, $i \in N$, $t \geq r_j$ and $t + p_j \leq d_j$. That means bid j is processed between its release time and deadline. We replace bid j by bid i. Denote the new schedule as π'.

In π', the beginning time of bid i is t, we see that $t \geq r_j$ and $r_j \geq r_i$, so $t \geq r_i$, we also see that $t + p_j \leq d_j$, $d_j \leq d_i$ and $p_j \geq p_i$, so $t + p_i \leq d_i$. Hence, bid i can also be produced between its release time and deadline in π'. As known $u_j < u_i$, so the profit of π' is greater than that of π^*. A contradiction arises. Hence, we reach the conclusion.

□

Lemma 4.1 implies that if an optimal solution includes bid j, it must include bid i. However, if an optimal solution includes bid i, it may not include bid j.

4.3.2 Lagrangian relaxation

Let $\lambda = (\lambda_1, \lambda_2, ..., \lambda_T)^T$ be a vector of the corresponding non-positive multipliers. The Lagrangian relaxation problem of the WD problem is described as follows:

$$(LR) \quad L(\lambda) = \max \left(\sum_{i=1}^{n}\sum_{t=1}^{T}(\alpha_i - v)p_i x_{it} + mvT + \sum_{t=1}^{T}\lambda_t \left(\sum_{i=1}^{n} \sum_{s=\max\{r_i+p_i,t\}}^{\min\{t+p_i-1,T\}} x_{is} - m \right) \right)$$

subject to equations (4), (5), (6) and (7)

In (LR), we add constraint (3) to the objective function with weights $\lambda = (\lambda_1, \lambda_2, ..., \lambda_T)^T$, and drop constraint (3). For any $\lambda \leq 0$, $L(\lambda)$ is no less than the objective value of the WD problem. That means the value of $L(\lambda)$, which depends on λ, is an upper bound on the optimal value of the WD problem.

The Lagrangian relaxation problem (LR) can be rewritten equivalently as:

$$(LR)\quad L(\lambda) = \max\left(\sum_{i=1}^{n}\sum_{t=1}^{T}\left(\alpha_i p_i - vp_i + \sum_{s=\max\{t-p_i+1,r_i\}}^{t}\lambda_s\right)x_{it}\right.$$
$$\left. + m\left(vT - \sum_{t=1}^{T}\lambda_t\right)\right)$$

subject to equations (4), (5), (6) and (7)

For each bid i, let

$$L_i(\lambda) = \sum_{t=1}^{T}(\alpha_i p_i - vp_i + \sum_{s=\max\{t-p_i+1,r_i\}}^{t}\lambda_s)x_{it}.$$

Then

$$L(\lambda) = \max\left(\sum_{i=1}^{n} L_i(\lambda) + m\left(vT - \sum_{t=1}^{T}\lambda_t\right)\right).$$

Let $\bar{\alpha}_i^t = (\alpha_i - v)$ and $\bar{\lambda}_i^t = \sum_{s=\max\{t-p_i+1,r_i\}}^{t}\lambda_s$, then

$$L(\lambda) = \sum_{i=1}^{n}\max_{r_i+p_i\le t\le d_i}\left(L_i(\bar{\alpha}_i^t p_i + \bar{\lambda}_i^t)\right) + m\left(vT - \sum_{t=1}^{T}\lambda_t\right).$$

Thus, we consider the Lagrangian dual of (LR) as follows:

$$(D)\quad L_D = \min_{\lambda\le 0} L(\lambda).$$

We use the subgradient algorithm to determine the Lagrange multipliers $\lambda = (\lambda_1, \lambda_2, ..., \lambda_T)^T$ to find near-optimal or optimal solutions of L_D:

Subgradient algorithm SA

Step 0 Initialisation: Set $\lambda^0 \le 0$ and $g = 0$. Set the values of G, μ^0 and γ.

Step 1 Calculate x^g and $L^g(\lambda^g)$ for the Lagrangian problem.

Step 2 Calculate $\lambda^{g+1} = \lambda^g + \beta_g G(x^g)/\| G(x^g) \|$, where β_g is the step size and $G(x^g)$ is the subgradient at point x^g, where

$$G(x^g) = (G_1(x^g), G_2(x^g), ..., G_T(x^g))^T,$$

$$G_t(x^g) = \sum_{i=1}^{n}\sum_{s=\max\{r_i+p_i,t\}}^{\min\{t+p_i-1,T\}} x_{is} - m, \text{ for } t = 1, ..., T,$$

and

$$\| G(x^g) \| = \sqrt{G_1(x^g)^2 + G_2(x^g)^2 + ... + G_T(x^g)^2}.$$

Step 3 Set $g \leftarrow g + 1$. If $g < G$, then perform step 2; otherwise, stop.

In algorithm SA, the difficulty lies in how to choose the step size β_g, we use the following method to define it:

$$\beta_g = \frac{\mu_g(\bar{Z} - L^g(\lambda^g))}{\parallel G(x^g) \parallel},$$

where $\mu_g = \gamma\mu_{g-1}$, $\bar{Z}$ is a feasible solution value for the WD problem and $\bar{Z} \leq L^g(\lambda^g)$.

We use algorithm SA to get an upper bound for the WD problem, and the vector x^G shows whether the bids are accepted or not, in the Lagrangian dual problem.

4.3.3 Heuristic for the WD problem

We need to develop a heuristic based on the Lagrangian relaxation method, since vector x^G may not be the optimal solution or even a feasible solution of the WD problem. First, we use x^G as a preliminary solution of the winning bids, and then generate a feasible schedule. We describe the heuristic as follows:

Heuristic algorithm HA

Step 0 Set Φ be the subset of accepted bids by algorithm SA and $\Phi' = N \setminus \Phi$. Index the bids in Φ in a non-decreasing order of the release times and break ties by the earliest deadline first.

Step 1 For the bid $i = 1, 2, ..., |\Phi|$, assign successively bid i to m identical parallel machines by steps 1-1 or 1-2.

- Step 1-1 If bid i can be started after r_i and completed before d_i on one of the machines, then schedule it to the machine on which bid i is started as close to its release time as possible. Next i.
- Step 1-2 If bid i cannot be started after r_i or completed before d_i on any machines, then discard it and let $\Phi' = \Phi' \cup \{i\}$. Next i.

Step 2 Confirm whether there are idle times between adjacent bids on all the machines. Successively select one bid s with $\alpha_s = \max\{\alpha_i | i \in \Phi'\}$. Insert bid s in all the idle times. If bid s can be started after r_s and completed before d_s in some idle times, then assign it such that it is started as close to its release time as possible and let $\Phi' = \bar{\bar{N}} \setminus \{s\}$.

Step 3 Check each bid $l \in A'$, calculate $A_l = \{j | r_j \geq r_l,\ d_j \leq d_l,\ p_j \geq p_l$ or $u_j < u_l,\ j \in \bar{N}\}$. If $A_l = \oslash$, then let $\Phi' = \Phi' \setminus \{l\}$.

Step 4 Successively select one bid w with $\alpha_w = \max\{\alpha_i | i \in \Phi'\}$. Insert the bid w in time slot $[r_w, d_w]$, and successively confirm one bid k after the order w. If it is not in its processing window, then discard it and process the succeeding bids in their processing windows as early as possible.

Step 5 Keep a record of the maximum objective value of the generated schedule in step 4. Let $\Phi' = \Phi' \setminus \{w\} \cup \{k\}$. If the maximum objective value is greater than the objective value of the current schedule, this schedule is taken as the current schedule.

Step 6 Confirm whether Φ' is an empty set. If $\Phi' \neq \oslash$, then go to step 4; otherwise stop.

In step 1 of algorithm HA, we generate a feasible schedule from the accepted bids. In step 2, we add the unscheduled bids to the idle times of the identical parallel machines. In step 3, we use Lemma 4.1 to assess whether the unscheduled bids can be greedily added to the machines. In steps 4 and 5, we greedily add the unscheduled bids to increase the firm's profit. The computing cost of algorithm HA is dominated by algorithm SA. Thus, the computational time of algorithm HA is $O(nT)$.

4.4 Special case

The global optimisation problem's objective is maximising the system profit. The integer linear program of the global optimal schedule can be readily obtained by changing the objective function of the WD problem [see equation (2)] as max $\sum_{i=1}^{n}\sum_{t=1}^{T}(u_i - vp_i)x_{it} + mvT$. For general cases, the solution generated by the auction mechanism may differ from the global optimal solution. Here, we consider a special case where all customer orders have the same processing time p. It reflects the practical situation in which assembling similar components according to different customer requirements. The special case is denoted as *the identical processing time problem*. We consider a simple example as follows:

Example 4.1: Consider an instance in which m = 2, T = 4. There are five customer orders (see Table 1 for the order information). We set v = 1, ρ_1 = ρ_2 = 1.

Table 1 The data of the customer orders in Example 4.1

Order i	p_i	r_i	d_i	u_i
1	2	0	2	10
2	2	0	3	7
3	2	0	4	4
4	2	1	3	8
5	2	2	4	6

The auction process is as follows:

In the first round, the firm sets the prices for each customer: α_1^1 = 1.5, α_2^1 = 1.33, α_3^1 = 1.25, α_4^1 = 1.5 and α_5^1 = 1.5. The customers submit bids B_1^1 = (2, 0, 2, 1.5), B_2^1 = (2, 0, 3, 1.33), B_3^1 = (2, 0, 4, 1.25), B_4^1 = (2, 1, 3, 1.5) and B_5^1 = (2, 2, 4, 1.5). The firm solves the WD problem, and customers 1, 2, 3, 5 win bids.

In the second round, the firm updates the price for customer 4: α_4^2 = 2.295. Customer 4 submits a new bid B_4^2 = (2, 1, 3, 2.295). The firm solves the WD problem and customers 1, 4, 5 win bids.

In the third round, the firm updates the price for customers 2 and 3: α_2^3 = 2.382, α_3^3 = 2.149. The price is too high for customer 3, only customer 2 submit new bids B_2^3 = (2, 0, 3, 2.382). The firm solves the WD problem and customers 2, 4, 5 win bids.

The details of rounds 4 to 11 are omitted.

In the 12th round, the firm updates the price for customer 2: α_2^{12} = 3.443. Customer 4 submits a new bid B_2^{12} = (2, 0, 3, 3.443). The firm solves the WD problem and customers 1, 2, 5 win bids.

In the 13th round, the firm updates the price for customer 4: α_4^{13} = 3.08. Customer 4 submits a new bid B_4^{13} = (2, 1, 3, 3.08). The firm solves the WD problem and customer 1, 2, 5 win bids.

In the last round, no one submits a new bid. The auction is terminated, and customers 1, 2, 5 win bids.

The system profit of the auction is 25. It is easy to show that the global optimal schedule is customers 1, 2, 3, 5, and the global optimal value is 27.

Example 4.1 illustrates that the auction mechanism cannot guarantee a global optimal solution for the special case. Next, we present a positive result under stronger conditions.

Theorem 4.2: For the identical processing time problem, if all the customer orders have the same time window constraint $[r, d]$, then the system value of the auction mechanism is less than the global optimal value by at most $\min\{m\lfloor\frac{d-r}{p}\rfloor, n-m\lfloor\frac{d-r}{p}\rfloor\}\frac{(\rho_1+\rho_2)p}{m}$.

Proof: According to the auction mechanism in Section 4, if all customer orders have the same processing time p and the same time window $[r, d]$, then in each round, price increment ϵ_i and price α_i are the same for the unaccepted customers.

There are a pool of customer orders $N = \{J_1, J_2, ..., J_n\}$, where $u_1 > u_2 > ... > u_n$. Assuming there is a feasible solution, that $A = \{J_1, J_2, ..., J_{m\lfloor\frac{d-r}{p}\rfloor}\}$ is a set of the accepted customer orders, $A' = \{J_{m\lfloor\frac{d-r}{p}\rfloor+1}, J_{m\lfloor\frac{d-r}{p}\rfloor+2}, ..., J_n\}$ is the set of the unaccepted customer orders, where $\min\{u_j|J_j \in A\} > \max\{u_i'|J_i' \in A'\}$, $|A| = m\lfloor\frac{d-r}{p}\rfloor$ and $|A'| = |N| - |A|$. For the global problem, it is clear that there exists a global optimal solution in which all orders in A are accepted for processing. Let $F^* = \sum_{J_j\in A} u_j + mv(T - \lfloor\frac{d-r}{p}\rfloor p)$ denote the global optimal value.

For the auction mechanism, we consider the worst solution in two possible situations:

1 When $|A| \le |A'|$, that means $m\lfloor\frac{d-r}{p}\rfloor \le n - m\lfloor\frac{d-r}{p}\rfloor$.

In the K round, assuming that$\alpha^K \le \min\{u_j/p|J_j \in A'\} = u_n/p$ and $J_{n-m\lfloor\frac{d-r}{p}\rfloor+1}, ..., J_n$ are accepted for processing. It means all winning orders are in A'. Then the firm updates the price $\alpha^{K+1} > \max\{u_j/p|J_j \in A\} = u_1/p$ for the unaccepted orders, where $\alpha^{K+1} = \bar{\alpha}^{K+1} + \epsilon^{K+1}$, and $\bar{\alpha}^{K+1} \ge \alpha^K$. Thus, none of the unaccepted orders can submit a new bid and the auction is terminated. So part or all of orders in A' are accepted for processing and the system value of the auction mechanism is $F = \sum_{J_i=n-m\lfloor\frac{d-r}{p}\rfloor+1}^{n} u_i + mv(T - \lfloor\frac{d-r}{p}\rfloor p)$, then

$$F^* - F = \sum_{J_j\in A} u_j + mv\left(T - \left\lfloor\frac{d-r}{p}\right\rfloor p\right) - \left(\sum_{J_i=n-m\lfloor\frac{d-r}{p}\rfloor+1}^{n} u_i + mv\left(T - \left\lfloor\frac{d-r}{p}\right\rfloor p\right)\right)$$
$$= u_1 + u_2 + ... + u_{m\lfloor\frac{d-r}{p}\rfloor} - (u_{n-m\lfloor\frac{d-r}{p}\rfloor+1} + u_{n-m\lfloor\frac{d-r}{p}\rfloor+2} + ... + u_n)$$

$$= (u_1 - u_n) + (u_2 - u_{n-1}) + \ldots + (u_{m\lfloor \frac{d-r}{p} \rfloor} - u_{n-m\lfloor \frac{d-r}{p} \rfloor+1})$$
$$< m\left\lfloor \frac{d-r}{p} \right\rfloor (\alpha^{K+1} - \alpha^K)p \le m\left\lfloor \frac{d-r}{p} \right\rfloor \epsilon^{K+1} p,$$

from Subsection 4.2, we see that

$$\epsilon^{K+1} = \frac{\rho_1 p}{m(d-r)} + \frac{\rho_2 \sum_{t=r+1}^{d} D_t^K}{m(d-r)N}$$
$$\le \frac{\rho_1}{m} + \frac{\rho_2}{m} = \frac{\rho_1 + \rho_2}{m},$$

thus, we have

$$F^* - F < \left\lfloor \frac{d-r}{p} \right\rfloor (\rho_1 + \rho_2)p.$$

2 When $|A| > |A'|$, that means $m\lfloor \frac{d-r}{p} \rfloor > n - m\lfloor \frac{d-r}{p} \rfloor$.

In the K round, assuming that $J_{n-m\lfloor \frac{d-r}{p} \rfloor+1}, \ldots, J_n$ are accepted for processing. It means all orders in A' and part of orders in A win. Then the firm updates the price $\alpha^{K+1} = \bar{\alpha}^{K+1} + \epsilon^{K+1} > \max\{u_j/p | J_j \in A\}$ for the unaccepted orders. Thus, none of the unaccepted orders can submit a new bid and the auction is terminated. So all orders in A' and part of orders in A are accepted for processing, similar to 1, we have

$$F^* - F = \sum_{J_j \in A} u_j + mv\left(T - \left\lfloor \frac{d-r}{p} \right\rfloor p\right) - \left(\sum_{J_i = n-m\lfloor \frac{d-r}{p} \rfloor+1}^{n} u_i \right.$$
$$\left. + mv\left(T - \left\lfloor \frac{d-r}{p} \right\rfloor p\right)\right)$$
$$= u_1 + u_2 + \ldots + u_{m\lfloor \frac{d-r}{p} \rfloor} - (u_{n-m\lfloor \frac{d-r}{p} \rfloor+1} + u_{n-m\lfloor \frac{d-r}{p} \rfloor+2} + \ldots + u_n)$$
$$= (u_1 - u_n) + (u_2 - u_{n-1}) + \ldots + (u_{n-m\lfloor \frac{d-r}{p} \rfloor} - u_{m\lfloor \frac{d-r}{p} \rfloor+1})$$
$$< \left(n - m\left\lfloor \frac{d-r}{p} \right\rfloor\right)(\alpha^{K+1} - \alpha^K)p \le \left(n - m\left\lfloor \frac{d-r}{p} \right\rfloor\right)\epsilon^{K+1} p$$
$$\le \left(n - m\left\lfloor \frac{d-r}{p} \right\rfloor\right)\frac{(\rho_1 + \rho_2)p}{m}.$$

Summarising the analysis of situations 1 and 2, we conclude that the system value of the auction mechanism is less than the global optimal value by at most $\min\{m\lfloor \frac{d-r}{p} \rfloor, n - m\lfloor \frac{d-r}{p} \rfloor\}\frac{(\rho_1+\rho_2)p}{m}$. □

5 Computational analysis

In this section, we present a computational analysis of the effectiveness and efficiency for the auction mechanism. All instances are solved on a PC computer with 2.9 GHz octa-core processor and 16 GB RAM.

5.1 Data generation

We first select $m \in \{2, 5, 10\}$, $n \in \{25, 50, 100\}$. For order $i(i \in N)$, the processing times, release time, deadline, and revenue are integer numbers. We generate p_i randomly from a uniform distribution in the interval [1, 100], u_i randomly from a uniform distribution in the interval [$p_i v$, 1,000]. We define the relative range factor parameters of release time R and deadline D. The values for R are 0.2, 0.4, 0.6 and the values for D are 0.6, 0.8, 1.0. This reflects a wide range of changes in the length and location of the time window for each customer order. For fixed values of R and D, we generate the release time r_i randomly from a uniform distribution in the interval $[0, \min\{T - p_i, RT\}]$, and the deadline d_i randomly from a uniform distribution in the interval $[r_i + p_i, \max\{r_i + p_i, DT\}]$. Let $P = \sum_{i \in N} p_i$. For each instance with total processing time P, the firm's capacity T is generated using $T = \tau P/2$, where $\tau \in \{0.5, 0.7, 0.9\}$. We generate the reserve value v randomly from a uniform distribution in the interval [1, 10]. For each of the 3 × 3 × 3 × 3 × 3 = 243 situations, we randomly generate 10 problem instances. Thus, we generate 2,430 problem instances.

5.2 Analysis of the performance of algorithm HA

In this subsection, we choose m = 2, $n \in \{25, 50, 100\}$ to analyse the influence of the changes of the parameters on the performance of the algorithm HA. Before the analysis of the performance of algorithm HA, we use a simple instance to illustrate that algorithm HA is well implemented.

Example 5.1: Consider an instance in which m = 2, T = 16. There are eight bids B_i = (p_i, r_i, d_i, α_i), $(i = 1, ..., 8)$ (see Table 2 for the information of the bids). We set v = 1.

We solve the ILP model of the WD problem by CPLEX, and get an optimal solution, B_1, B_2, B_4, B_5, B_6 are allocated time slots [4, 10], [1, 3], [9, 14], [1, 8] and [11, 12], respectively. The firm achieves a total profit of 334.

According to algorithm HA, we first use algorithm SA to get a preliminary result of the decision on bids selection. Set $\lambda^0 = (-1, ..., -1)$, $\mu^0 = 1$, $\gamma = 1$. After 16 iterations, we get a bids selection solution B_1, B_2, B_4, B_5, B_6. The solutions of the iterations are shown in Table 3. Then, we generate a feasible schedule for the accepted bids B_1, B_2, B_4, B_5, B_6. Finally, we get a solution where B_1, B_2, B_4, B_5, B_6 are allocated time slots [4, 10], [1, 3], [9, 14], [1, 8] and [11, 12], respectively. The firm achieves a total profit of 334. The solution is the same as the optimal solution.

Table 2 The information of the bids in Example 5.1

B_i	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
p_i	7	3	10	6	8	2	6	5
r_i	2	0	2	3	0	1	1	2
d_i	11	10	15	14	9	13	12	11
$\alpha_i p_i$	59	81	46	83	53	52	22	18

Since using the integer linear program of the WD problem can not be solved in a reasonable time, we establish its upper bound by using the Lagrangian relaxation method that is presented in Subsection 4.3.2. To evaluate the performance of algorithm HA, we compare its solution value with the optimal solution value or upper bound.

Table 3 The solutions for iterations of SA in Example 5.1

g	*Accepted bids*	$L^g(\lambda^g)$
1	$B_1, B_2, B_3, B_4, B_5, B_6, B_7, B_8$	384.0
2	$B_1, B_2, B_3, B_4, B_5, B_6, B_7, B_8$	381.1
3	$B_1, B_2, B_3, B_4, B_5, B_6, B_7, B_8$	361.4
4	$B_1, B_2, B_3, B_4, B_5, B_6$	349.9
5	B_1, B_2, B_4, B_5, B_6	341.3
6	$B_1, B_2, B_3, B_4, B_5, B_6$	331.4
7	B_1, B_2, B_4, B_5, B_6	339.8
8	B_1, B_2, B_4, B_5, B_6	336.5
9	B_1, B_2, B_4, B_5, B_6	336.6
10	B_1, B_2, B_4, B_5, B_6	337.4
11	B_1, B_2, B_4, B_5, B_6	336.1
12	$B_1, B_2, B_3, B_4, B_5, B_6$	334.1
13	B_1, B_2, B_4, B_5, B_6	337.4
14	B_1, B_2, B_4, B_5, B_6	334.9
15	B_1, B_2, B_4, B_5, B_6	334.2
16	B_1, B_2, B_4, B_5, B_6	334.0

Table 4 Performance of algorithm HA with $m = 2$, $n = 25$

τ	R	D	*POH(%) (n = 25)*			*POU(%) (n = 25)*		
			Max	*Min*	*Average*	*Max*	*Min*	*Average*
0.5	0.2	0.6	99.42	87.62	93.88	108.13	100.85	102.74
		0.8	96.21	89.56	92.95	104.74	100.55	102.19
		1.0	96.17	88.49	92.58	104.48	101.04	102.21
	0.4	0.6	100.00	88.86	94.94	106.93	100.17	102.20
		0.8	95.63	89.71	93.95	106.91	101.19	102.80
		1.0	100.00	88.77	93.56	105.29	101.15	102.72
	0.6	0.6	100.00	88.52	94.52	107.52	101.08	103.19
		0.8	99.52	90.31	94.34	105.31	100.47	102.41
		1.0	99.43	90.20	93.98	104.06	101.28	102.15
0.7	0.2	0.6	96.94	90.84	94.39	107.07	100.53	102.17
		0.8	96.26	88.30	93.69	103.28	100.63	101.78
		1.0	96.49	90.85	95.06	105.23	101.46	102.77
	0.4	0.6	99.35	91.38	95.92	104.52	100.89	102.22
		0.8	96.11	88.11	94.29	104.36	100.83	102.13
		1.0	96.38	90.81	94.02	102.61	100.85	101.78
	0.6	0.6	100.00	91.04	96.46	107.92	100.58	103.47
		0.8	96.89	91.97	94.73	104.20	100.48	102.41
		1.0	99.37	91.34	95.97	102.77	101.49	102.11
0.9	0.2	0.6	97.52	88.86	96.83	104.23	100.88	102.26
		0.8	97.38	88.51	94.58	103.92	101.08	102.00
		1.0	98.89	92.63	95.53	102.84	100.64	101.50
	0.4	0.6	99.33	93.36	96.86	104.89	100.84	102.16
		0.8	100.00	91.04	95.64	102.94	101.37	101.92
		1.0	99.12	94.00	97.91	102.90	100.63	101.64
	0.6	0.6	99.89	92.57	97.24	106.33	101.13	102.46
		0.8	98.00	90.23	96.76	116.85	101.29	102.88
		1.0	97.60	91.14	95.97	102.88	100.43	101.90
Average					95.06			102.30

Let H_{f-val} denote the objective value obtained by algorithm HA. Let OPT_f and $UB_{f-relax}$ denote the optimal value by IPL model and the objective value obtained by the Lagrangian relaxation method, respectively. For the case where n = 25, the performance of algorithm HA is defined as $POH = H_{f-val}/OPT_f$, and the performance of the upper bound is defined as $POU = UB_{f-relax}/OPT_f$. For the cases where n = 50 and 100, the performance of algorithm HA is defined as $POH = H_{f-val}/UB_{f-relax}$ since the optimal solution can not be obtained within reasonable computational time. The experimental results are displayed in Tables 4 and 5.

Table 5 Performance of algorithm HA with m = 2, $n \in \{50, 100\}$

τ	R	D	*POH(%) (n = 50)*			*POH(%) (n = 100)*		
			Max	*Min*	*Average*	*Max*	*Min*	*Average*
0.5	0.2	0.6	92.94	85.25	90.25	92.44	86.91	89.87
		0.8	91.91	83.21	89.94	90.40	85.10	88.16
		1.0	90.55	84.30	88.88	89.96	82.96	87.70
	0.4	0.6	95.77	88.46	92.52	96.00	88.31	92.78
		0.8	93.74	87.81	90.32	93.21	87.22	90.61
		1.0	93.26	85.08	89.73	90.79	84.65	89.34
	0.6	0.6	94.74	89.35	95.50	96.29	93.02	94.44
		0.8	93.69	88.13	93.49	92.84	90.04	93.36
		1.0	92.49	87.37	92.25	92.27	88.34	91.37
0.7	0.2	0.6	93.02	87.41	90.64	91.50	86.63	90.62
		0.8	94.20	87.11	90.58	92.28	84.27	90.05
		1.0	93.53	86.76	92.79	91.94	85.76	89.83
	0.4	0.6	93.56	89.87	92.91	94.81	88.32	92.85
		0.8	92.97	86.42	91.89	93.31	87.60	91.52
		1.0	93.75	88.39	91.32	93.35	85.03	90.59
	0.6	0.6	94.92	91.68	95.91	95.67	92.58	94.34
		0.8	94.24	90.86	93.73	96.19	90.14	93.35
		1.0	93.89	88.12	92.89	95.78	89.78	93.91
0.9	0.2	0.6	94.84	89.67	93.90	91.39	85.94	92.63
		0.8	92.26	88.62	92.79	93.08	84.77	92.13
		1.0	96.39	92.37	93.98	93.91	85.04	91.42
	0.4	0.6	96.29	89.52	94.94	93.74	90.62	93.42
		0.8	95.13	89.95	94.99	94.26	88.17	92.93
		1.0	97.16	94.28	95.45	93.86	89.42	92.95
	0.6	0.6	98.42	91.87	95.98	95.93	92.42	94.18
		0.8	96.66	92.10	95.69	96.14	90.91	93.29
		1.0	97.56	93.51	95.67	95.51	92.62	93.91
Average					92.98			91.91

For the instances with n = 25 shown in Table 4, the overall mean value of POU is 102.3%. This suggests that the upper bounds found by the Lagrangian relaxation method are typically close to optimal values.

From Tables 4 and 5, we also see that the overall mean values of POH for the case where n = 25, 50 and 100 are 95.06%, 92.98%, and 91.91%, respectively. The heuristic algorithm for the WD problem performs well across different combinations of parameters. However, we see that the values of POH for the large problem sizes, where n = 50 and 100, are significantly smaller than the value of the small-sized instances, where n = 25. This follows as POH may be underestimated due to the gaps between the upper bounds and the optimal objective values. We also see that algorithm HA performs slightly better for larger values of the factor for release times, and for smaller values of the factor for deadlines. Through the above analysis, we conclude that algorithm HA is effective for rapidly finding a good solution of the WD problem.

5.3 Analysis of the performance of the auction mechanism

In this subsection, we investigate the effectiveness and efficiency of the auction mechanism. First, we use the global optimal solution as a benchmark, and analysed the effectiveness of the auction mechanism by comparing the system profit of the auction mechanism to the benchmark solution. Since the global optimal solution can not be solved in a reasonable time by CPLEX for the large-sized instance. We establish an upper bound on the global optimal value by using the Lagrangian relaxation method presented in Subsection 4.3.2.

Let H_{auc} denote the objective value of the auction mechanism. Let OPT_s and $UB_{s-relax}$ denote the global optimal value and the upper bound obtained by Lagrangian relaxation method, respectively. For the case where m = 2, 5 and 10, n = 25, the effectiveness of the auction mechanism is defined as $POA = H_{auc}/OPT_s$. For the other cases where m = 2, 5 and 10, n = 50 and 100, the effectiveness of the auction mechanism is defined as $POA = H_{auc}/UB_{s-relax}$, as the global optimal value can not be obtained within reasonable computational times. The computational results with m = 2, 5 and 10, n = 25, 50 and 100 are summarised in Table 6.

Table 6 Performance of the auction mechanism with $m \in \{2, 5, 10\}$, $n \in \{25, 50, 100\}$

m	*n*	*POA(%)*		
		Max	*Min*	*Average*
2	25	100.00	86.59	94.00
	50	97.25	85.02	91.93
	100	96.47	83.09	91.03
5	25	100.00	88.42	95.36
	50	98.75	83.90	92.17
	100	95.49	82.58	91.52
10	25	100.00	89.92	96.28
	50	97.58	84.36	92.37
	100	97.01	82.92	92.54
Average			93.02	

From Table 6, we see that the overall mean value of POA is 93.02%. When n = 25, m = 2, 5, and 10, the auction mechanism can sometimes achieve the global optimal value. The minimum value of POA can reach more than 80%. The auction mechanism performs well with different numbers of machines.

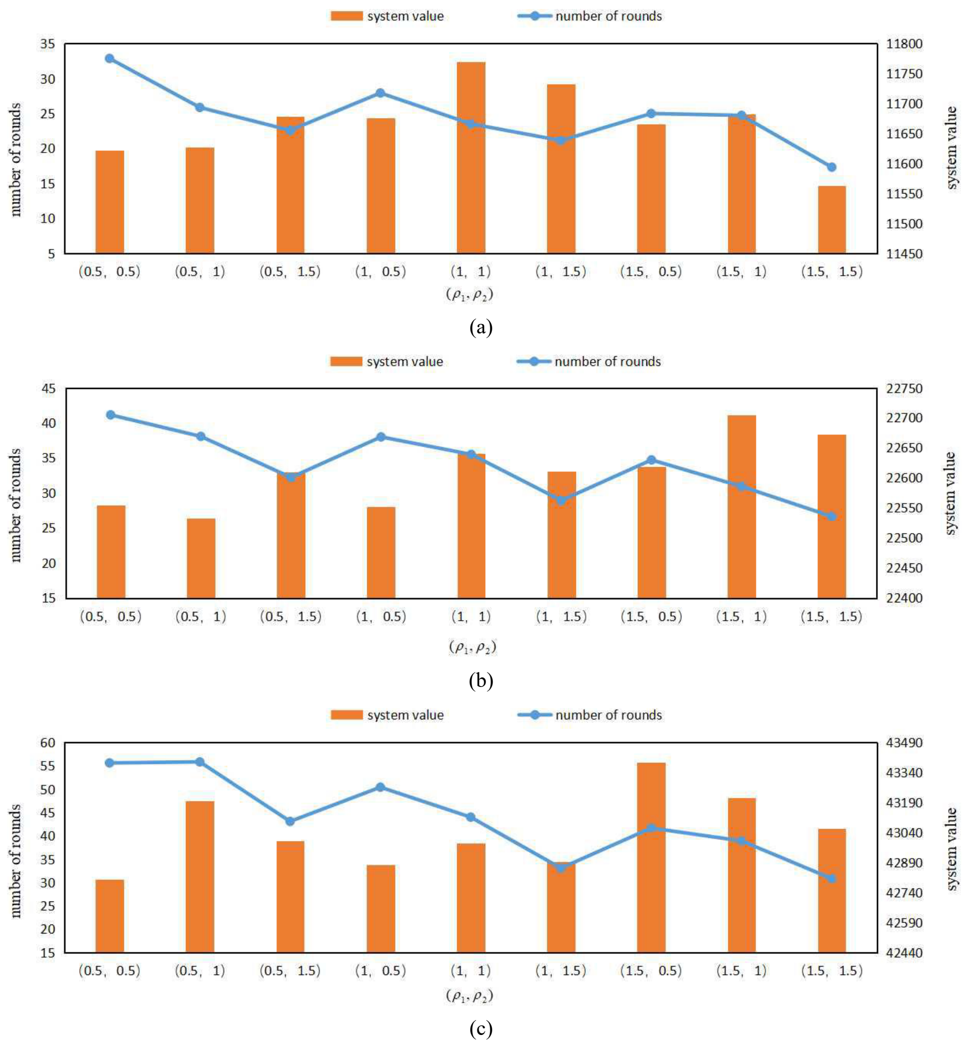
In Table 7, we choose $m = 2$, $n \in \{25, 50, 100\}$ to analyse the influence of the changes of the parameters on the performance of the auction mechanism. The detailed results are summarised in Table 7.

Table 7 Performance of the auction mechanism with $m = 2$, $n \in \{25, 50, 100\}$

τ	R	D	*POA(%) (n = 25)*			*POA(%) (n = 50)*			*POA(%) (n = 100)*		
			Max	*Min*	*Average*	*Max*	*Min*	*Average*	*Max*	*Min*	*Average*
0.5	0.2	0.6	97.43	89.79	93.93	93.26	86.80	90.66	90.88	86.58	89.88
		0.8	96.90	89.16	93.12	92.45	86.27	90.28	89.36	83.99	88.45
		1.0	96.25	90.02	92.36	92.55	85.02	89.84	90.06	83.09	87.56
	0.4	0.6	100.00	91.03	93.74	94.75	88.39	92.58	95.24	86.97	90.61
		0.8	95.37	88.16	92.94	92.65	87.50	90.61	91.88	88.04	89.96
		1.0	98.17	87.33	92.27	93.51	87.63	90.39	93.15	84.55	87.74
	0.6	0.6	98.77	90.27	95.05	97.25	89.79	92.72	94.96	89.88	92.33
		0.8	98.28	92.12	94.53	93.30	89.98	91.70	93.77	89.56	91.89
		1.0	97.10	86.59	93.66	94.92	85.86	91.57	92.52	88.54	90.29
0.7	0.2	0.6	97.86	90.80	93.97	91.96	85.35	90.20	91.63	86.21	89.96
		0.8	94.83	88.34	92.82	93.72	86.96	90.97	92.80	83.66	88.69
		1.0	98.24	87.43	92.49	94.56	86.35	90.57	95.12	83.45	90.84
	0.4	0.6	98.55	91.21	95.15	95.45	89.83	92.47	94.79	87.53	90.69
		0.8	97.84	87.72	93.82	94.34	87.41	91.68	92.92	88.17	90.55
		1.0	96.35	87.35	92.59	93.80	86.88	91.26	94.66	86.94	90.19
	0.6	0.6	97.58	88.55	94.45	95.33	90.06	92.67	94.94	88.47	92.44
		0.8	98.80	88.19	94.86	95.88	88.44	91.82	94.72	88.37	91.96
		1.0	98.38	89.97	93.84	94.96	90.30	92.68	94.38	87.20	91.43
0.9	0.2	0.6	96.27	90.51	95.04	94.91	88.06	91.59	92.84	85.23	91.56
		0.8	97.31	88.52	94.42	93.26	85.22	91.82	93.89	85.74	90.99
		1.0	97.66	89.60	93.62	95.02	87.42	90.98	94.82	84.72	91.59
	0.4	0.6	99.09	92.10	95.27	93.66	90.57	93.28	93.51	89.64	92.76
		0.8	97.19	90.79	94.32	95.64	88.61	93.56	95.10	86.48	91.86
		1.0	97.96	88.23	93.34	95.11	90.86	93.14	96.47	86.77	92.17
	0.6	0.6	100.00	92.61	95.97	96.40	92.38	94.97	95.39	91.05	93.93
		0.8	97.15	90.42	95.06	94.50	89.83	92.98	95.18	90.09	92.62
		1.0	97.70	90.07	95.44	96.94	91.66	93.91	96.47	90.97	93.91
Average					94.00			91.93			91.03

From Table 7, we see that the values of POA vary for situations associated with different combinations of parameters. The detailed analysis is as follows. First, when n = 25, 50 and 100, the overall mean values of POA are 94.00%, 91.93%, and 91.03%, respectively. The auction mechanism performs worse as the number of customers increases. Second, as the firm's capacity parameter τ increases from 0.5 to 0.9, the auction mechanism performs slightly better. This indicates that the smaller the scarcity of capacity, the better the performance of the auction mechanism. Third, as the relative range factor of release time R increases from 0.2 to 0.6, the auction mechanism performs slightly better. As the relative range factor of deadline D increases from 0.6 to 1.0, the auction mechanism performs slightly worse. This indicates that the length and location of an order's time window and the flexibility of order processing relative to its time window affect the scarcity of the production resources, which in turn affects the performance of the auction mechanism.

Figure 1 Number of rounds and system value of auction with $m = 2$, $n \in \{25, 50, 100\}$, (a) $m = 2$, $n = 25$ (b) $m = 2$, $n = 50$ (c) $m = 2$, $n = 100$ (see online version for colours)



We also take $m = 2$, $n \in \{25, 50, 100\}$ as examples, and use the mean number of rounds in which the auction reaches closure to test the computational efficiency of the auction mechanism. The number of rounds is mainly affected by the price adjustment factors. We construct the situations with nine different pairs of price adjustment factors (ρ_1, ρ_2). For each situation, we generate 20 problem instances. We calculate the number of rounds and the system value H_{auc} with different pairs of price adjustment factors (ρ_1, ρ_2). The results are presented in Figure 1.

From Figure 1, the mean numbers of rounds of the three problem sizes where n = 25, 50, and 100 are 24.5, 34.0, and 43.7, respectively. The mean numbers of rounds to reach closure for these nine different price adjustment factors are 43.2, 39.9, 32.6, 38.8, 34.3, 27.7, 33.8, 31.5, and 24.9, respectively. We observe that the price adjustment factors (ρ_1, ρ_2) significantly impact the system value and the number of rounds to reach closure. For fixed values of $\rho_1(\rho_2)$, when $\rho_2(\rho_1)$ increases from 0.5 to 1.5, the auction requires fewer rounds to reach closure. From Figure 1, the price adjustment factors (ρ_1, ρ_2) for which the system value reaches the highest of these three problem sizes where n = 25, 50, and 100 is (1, 1), (1.5, 1), and (1.5, 0.5), respectively. However, there is no clear relationship between the system value and the number of rounds. It may result from the combination optimisation characteristics of the allocation of scarce production capacity. It indicates that the suitable price adjustment factors (ρ_1, ρ_2) are important, which directly affect the system value of the auction mechanism. Hence, in order to obtain a better solution of the auction mechanism, it is necessary to select the suitable adjustment factors through a large number of experiments.

6 Conclusions

We consider the scarce production capacity allocation problem in a decentralised decision-making environment through an ascending auction mechanism. In this study, we consider the firm to possess identical parallel machines, which is different from the literature where the machine environment is a single machine or job shop. On the other hand, we assume the customer orders have time window constraints, because, in many situations, the customer orders can not be processed at the beginning of the production horizon. To the best of our knowledge, our study is the first to use an auction mechanism to allocate the production capacity in identical parallel machines with time window constraints.

We propose a discriminatory pricing scheme to resolve the resource conflicts and allocate the capacity effectively. A heuristic based on the Lagrangian relaxation technique is introduced to solve the WD problem. Our computational study shows that the heuristic for the WD problem is effective to find good solutions rapidly. The auction mechanism performs well, on average, it provides more than 93% of the global optimal value.

Future work can be directed to using the ascending auction mechanisms to allocate production capacity in more complex machine environments. Another major research direction is to extend the models in this study to the case where the capacity comes from several competing firms. In that case, a double auction mechanism would be designed to allocate the capacity from several firms to several customer orders.

Acknowledgements

The authors are grateful to the Editor, the Associate Editor, and the anonymous referees for their helpful comments, which substantially improved the quality of this work. This research was supported by the National Natural Science Foundation of China Grant No. 71871118.

References

Cesaret, B., Oğuz, C. and Salman, F.S. (2012) 'A tabu search algorithm for order acceptance and scheduling', *Computers & Operations Research*, Vol. 39, No. 6, pp.1197–1205.

Chaurasia, S.N. and Singh, A. (2017) 'Hybrid evolutionary approaches for the single machine order acceptance and scheduling problem', *Applied Soft Computing*, Vol. 52, No. 3, pp.725–747.

Cramton, P. (2002) 'Spectrum auction', in Cave, M., Majumdar, S. and Vogelsang, I. (Eds.): *Handbook of Telecommunications Economics*, Elsevier Science BV, Amsterdam, The Netherlands.

Cramton, P. and Ockenfels, A. (2017) 'The German 4G spectrum auction: design and behaviour', *The Economic Journal*, Vol. 127, No. 605, pp.305–324.

Das, A. and Grosu, D. (2005) 'Combinatorial auction-based protocols for resource allocation in grids', in *Proc. IEEE International Parallel and Distributed Processing Symposium*.

Dewan, P. and Joshi, S. (2002) 'Auction-based distributed scheduling in a dynamic job shop environment', *International Journal of Production Research*, Vol. 40, No. 5, pp.1173–1191.

Esmaeilbeigi, R., Charkhgard, P. and Charkhgard, H. (2016) 'Order acceptance and scheduling problems in two-machine flow shops: new mixed integer programming formulations', *European Journal of Operational Research*, Vol. 251, No. 2, pp.419–431.

Ghosh, J.B. (1997) 'Job selection in a heavily loaded shop', *Computers & Operations Research*, Vol. 24, No. 2, pp.141–145.

Hall, G.N. and Liu, Z.X. (2013) 'Market good flexibility in capacity auction', *Production and Operations Management*, Vol. 22, No. 2, pp.459–472.

Izakian, H., Abraham, A. and Ladani, B.T. (2010) 'An auction method for resource allocation in computational grids', *Future Generation Computer Systems*, Vol. 26, No. 2, pp.228–235.

Jackson, C. (1976) *Technology for Spectrum Market*, PhD thesis, Department of Electrical Engineering, Massachusetts Institute of Technology, Cambridge, MA.

Jiang, D., Tan, J. and Li, B. (2017) 'Order acceptance and scheduling with batch delivery', *Computers & Industrial Engineering*, Vol. 107, No. 5, pp.100–104.

Karabatı, S. and Yalçın, Z.B. (2014) 'An auction mechanism for pricing and capacity allocation with multiple products', *Production and Operations Management*, Vol. 23, No. 1, pp.81–94.

Kasbekar, G.S. and Sarkar, S. (2010) 'Spectrum auction framework for access allocation in cognitive radio networks', *IEEE/ACM Transactions on Networking*, Vol. 18, No. 6, pp.1841–1854.

Kaushik, A. and Vidyarthi, D.P. (2015) 'A model for resource management in computational grid for real-time jobs using game theory', *International Journal of Grid and Utility Computing*, Vol. 6, Nos. 3–4, pp.232–248.

Kaushik, A. and Vidyarthi, D.P. (2018) 'A model for resource management in computational grid using sequential auction and bargaining procurement', *Cluster Computing*, Vol. 21, No. 3, pp.1457–1477.

Kutanoglu, E. and Wu, S.D. (1999) 'On combinatorial auction and Lagrangean relaxation for distributed resource scheduling', *IIE Transactions*, Vol. 31, No. 9, pp.813–826.

Ledyard, J.O., Olson, M., Porter, D., Swanson, J.A. and Torma, D.P. (2002) 'The first use of a combined-value auction for transportation services', *Interfaces*, Vol. 32, No. 5, pp.4–12.

Lei, D. and Guo, X. (2015) 'A parallel neighborhood search for order acceptance and scheduling in flow shop environment', *International Journal of Production Economics*, Vol. 165, No. 7, pp.12–18.

McAfee, R.P. and McMillan, J. (1987) 'Auctions and bidding', *Journal of Economic Literature*, Vol. 25, No. 2, pp.699–738.

McMillan, J. (1994) 'Selling spectrum rights', *The Journal of Economic Perspectives*, Vol. 8, No. 3, pp.145–162.

McMillan, J. (1995) 'Why auction the spectrum', *Telecommunications Policy*, Vol. 19, No. 3, pp.191–199.

Naderi, B. and Roshanaei, V. (2020) 'Branch-relax-and-check: a tractable decomposition method for order acceptance and identical parallel machine scheduling', *European Journal of Operational Research*, Vol. 286, No. 3, pp.811–827.

Nicolaisen, J., Petrov, V. and Tesfatsion, L. (2001) 'Market power and efficiency in a computational electricity market with discriminatory double-auction pricing', *IEEE Transactions on Evolutionary Computation*, Vol. 5, No. 5, pp.504–523.

Oğuz, C., Salman, F.S. and Yalçın, Z.B. (2010) 'Order acceptance and scheduling decisions in make-to-order systems', *International Journal of Production Economics*, Vol. 125, No. 1, pp.200–211.

Pinedo, M. (1995) *Scheduling, Theory, Algorithms, and Systems*, 2nd ed., Pearson Education Asia Limited and Tsinghua University Press.

Rom, W.O. and Slotnick, S.A. (2009) 'Order acceptance using genetic algorithms', *Computers & Operations Research*, Vol. 36, No. 6, pp.1758–1767.

Slotnick, S.A. and Morton, T.E. (1996) 'Selecting jobs for a heavily loaded shop with lateness penalties', *Computers & Operations Research*, Vol. 23, No. 2, pp.131–140.

Slotnick, S.A. and Morton, T.E. (2007) 'Order acceptance with weighted tardiness', *Computers & Operations Research*, Vol. 34, No. 10, pp.3029–3042.

Voss, A. and Madlener, R. (2017) 'Auction schemes, bidding strategies and the cost-optimal level of promoting renewable electricity in Germany', *The Energy Journal*, KAPSARC Special Issue, Vol. 38, pp.229–264.

Wang, X., Xie, X. and Cheng, T.C.E. (2013) 'Order acceptance and scheduling in a two-machine flowshop', *International Journal of Production Economics*, Vol. 141, No. 1, pp.366–376.

Wellman, M.P., Walsh, W.E., Wurman, P.R. and MacKie-Mason, J.K. (2001) 'Auction protocols for decentralized scheduling', *Games and Economic Behavior*, Vol. 35, Nos. 1–2, pp.271–303.

Yang, K., Mireia, R-R. and Menéndez, M. (2019) 'An auction-based approach for prebooked urban logistics facilities', *Omega, The International Journal of Management Science*, Vol. 89, No. 8, pp.193–211.

Zhang, J., Xiang, J., Cheng, T.C.E., Hua, G. and Chen, C. (2019) 'An optimal efficient multi-attribute auction for transportation procurement with carriers having multi-unit supplies', *Omega, The International Journal of Management Science*, Vol. 83, No. 2, pp.249–260.

Zhong, X., Ou, J. and Wang, G. (2014) 'Order acceptance and scheduling with machine availability constraints', *European Journal of Operational Research*, Vol. 232, No. 3, pp.435–441.