PRODUCTION AND OPERATIONS MANAGEMENT

PRODUCTION AND OPERATIONS MANAGEMENT

Vol. 22, No. 2, March–April 2013, pp. 459–472
ISSN 1059-1478 | EISSN 1937-5956 | 13 | 2202 | 0459

DOI 10.1111/j.1937-5956.2012.01355.x

Market Good Flexibility in Capacity Auctions

Nicholas G. Hall
Fisher College of Business, The Ohio State University, Columbus, Ohio 43210-1144, USA, hall_33@fisher.osu.edu

Zhixin Liu
College of Business, University of Michigan-Dearborn, Dearborn, Michigan 48126-2638, USA, zhixin@umd.umich.edu

We consider the allocation of limited production capacity among several competing agents through auctions. Our focus is on the contribution of flexibility in market good design to effective capacity allocation. The application studied is a capacity allocation problem involving several agents, each with a job, and a facility owner. Each agent generates revenue by purchasing capacity and scheduling its job at the facility. Ascending auctions with various market good designs are compared. We introduce a new market good that provides greater flexibility than those previously considered in the literature. We allow ask prices to depend both on agents' utility functions and on the number of bids at the previous round of the auction, in order to model and resolve resource conflicts. We develop both optimal and heuristic solution procedures for the winner determination problem. Our computational study shows that flexibility in market good design typically increases system value within auctions. A further increase is achieved if each agent is allowed to bid for multiple market goods at each round. On average, the multiple flexible market goods auction provides over 95% of the system value found by centralized planning.

Key words: noncooperative game; auction; market good flexibility; capacity allocation
History: Received: January 2011; Accepted: January 2012 by Panos Kouvelis, after 2 revisions.

1. Introduction

We consider a capacity allocation problem involving several competing agents, each with a job, and a facility owner. Each agent generates revenue by purchasing production capacity and scheduling its job at the facility. However, the facility can process only one job at a time. Each agent maximizes its profit, that is, its revenue less its cost of purchasing the capacity and scheduling the job. The facility owner maximizes its revenue from selling its capacity and holding any unsold capacity at its reserve value. The facility owner allocates its limited capacity among the agents through the use of auctions. Our focus is on the design of an auction that allocates capacity effectively in this problem.

An important component in auction design is the choice of the market good. We discuss the importance of *flexibility in market good design* within auctions for resource allocation. By introducing a new market good that provides more flexibility than traditional market goods, we evaluate the extent to which increased market good flexibility enables auctions to gather information from bidders more completely and more efficiently. We show that the bidding process typically converges to a solution that is closer in value to that of an optimal centralized planning solution, and also does so faster, compared to a similar process with traditional market goods. Moreover, market good flexibility is a very general concept that can be applied to a wide variety of auctions for resource allocation.

The use of flexibility is familiar in revenue management (Talluri and van Ryzin 2004), where a business class airline seat can substitute for a coach class seat. However, our work is apparently among the first to apply it to auctions for resource allocation. Market good flexibility can improve the performance of auctions in various scheduling applications, such as the allocation of network television commercial time among advertisers (Reddy et al. 1998), and the allocation of airport gates for departure and arrival among airlines (Ball et al. 2006, Rassenti 1982). Here, flexibility is defined with respect to *time*. However, in an auction for telecommunications rights (Arbib et al. 2004), flexibility can be defined with respect to *bandwidth*. Also, in an auction for siting noxious facilities (Kunreuther and Kleindorfer 1986) or for distribution rights (Financialexpress 2007), flexibility can be defined with respect to *physical location*. Further, in an auction for investment opportunities, flexibility can be defined with respect to *the number and price of shares* (Derrien and Womack 2003).

A second contribution of our study is an evaluation of the effectiveness of *adaptive ask pricing* within auctions for resource allocation. Here, the ask price depends on the number of bids received at the previous round. Consequently, if the ask price of a market good is defined too high, fewer bids are received and its ask price declines. Adaptive ask pricing is designed to resolve resource conflicts among the agents. Kutanoglu and Wu (1999) define the ask price of each time slot in proportion to the excess number of bids above one at an auction round. However, such ask prices are not ascending and are difficult to implement for auctions with competing agents. Our work is apparently the first to apply adaptive ask pricing to ascending auctions for resource allocation. Potential applications of adaptive ask pricing include auctions for distributed Web services (Huang et al. 2005) and for downlink resources in a deep space network (NASA 2008).

Brucker (1998) and Pinedo (2012) discuss many centralized scheduling problems and procedures for solving them. Curiel et al. (1989, 1994, 2002) study cooperative decentralized sequencing games, where the cost saving of an agent relative to a given schedule is used to compensate other agents through side payments. Agnetis et al. (2004, 2007), Cheng et al. (2006), and Kubzin and Strusevich (2006) study cooperative multiple agent scheduling problems. They develop algorithms to find Pareto optimal schedules. However, where agents have competing interests, they may not communicate their information fully (Clearwater 1996), which makes centralized planning unreliable. An alternative approach to auctions is a *direct revelation mechanism* (DRM), which solicits all relevant information from the agents (Varian and MacKie-Mason 1994, Wellman et al. 2001). However, auctions provide several advantages over DRMs (Krishna 2002). First, they are *universal*, that is, they can be used to allocate any resource without knowing any details about how the resource is used by the agents. Second, they are *anonymous*, whereas payments defined by DRMs may depend on the identities of the bidders. Also, for scheduling problems, DRMs may not be *budget balanced* (Wellman et al. 2001). Finally, many DRMs require the solution of an intractable combinatorial problem. These reasons motivate our use of auctions.

In general, auction-based methods provide several advantages over simple capacity allocation rules such as first-come-first-serve (Wellman et al. 2001). Auctions are naturally decentralized and require communication only about bids and prices. Also, auctions can be designed to achieve Pareto or global optimality in various situations. When used as a decentralized optimization mechanism, an auction can sometimes find better solutions than centralized heuristics (Pinedo 2012). Bidders behave in an automated way and the auction is essentially a heuristic to solve a centralized problem (Dewan and Joshi 2002, Geng et al. 2006, Kutanoglu and Wu 1999).

Reviews of the multi-agent scheduling literature are provided by Shen et al. (2006) and Agnetis et al. (2008). Market-based mechanisms for decentralized scheduling are applied to time-shared computer systems (Sutherland 1968), airport gate allocation (Ball et al. 2006, Rassenti 1982), railroad track allocation (Brewer and Plott 1996), and multimedia services integration (Arbib et al. 2004). Shaw (1987, 1988) proposes a distributed scheduling method for intelligent manufacturing. Kutanoglu and Wu (1999), Dewan and Joshi (2002), and Geng et al. (2006) study job scheduling problems, using combinatorial auctions of time slots. Wellman et al. (2001) and Reeves et al. (2005) consider market-based scheduling mechanisms that allocate capacity over time, based on the bids of agents. Davidsson et al. (2005) provide an analysis of agent-based approaches to transport logistics. Karabati (2010) designs an auction mechanism for the pricing and capacity allocation decisions of a supplier that supplies multiple buyers with bundles of products.

Because of their characteristics of multiple market goods and complementarities between them, the applications that are most similar to the one we consider are the auctions of telecommunications spectrum held in the United States and Canada (Federal Communications Commission 1998, Industry Canada 2007). These are ascending auctions with multiple rounds. A major advantage of multiple round auctions is the information that they provide to bidding agents about the value other agents place on licenses. For example, the FCC's 700-MHz auction, Auction 73, reached closure in March 2008 after 261 rounds (Computech 2011). An alternative type of auction that can be used for scheduling applications is a sealed bid auction. A sealed bid auction can be useful for some applications, for example if a quick allocation is required, as in some auctions for content-based Internet advertising space (Google Display Network 2011). However, there are several reasons why a sealed bid auction would not be effective in the competitive resource allocation environment we are considering. First, they suffer from a lack of transparency. That is, unsuccessful bidding agents cannot observe the process under which they lost. Second, sealed bid auctions typically generate less revenue than ascending auctions (Cramton 1998), hence the facility owner who has a free choice of auction type will not prefer a sealed bid auction. Third, sealed bid auctions require bidding agents to forecast other agents' likely bids, which would be difficult in scheduling applications (McAfee and Lewis 2009). The difficulty arises from the fact that an agent typically does not know the processing time, value or scheduling cost information of another agent's job. Fourth, efficiency is important. A solution that is not efficient may fail to clear the market, even where a sufficient number of agents is bidding. However, sealed bid auctions are typically less efficient than ascending auctions (Cramton 1998). The issues of lower revenue and efficiency are especially problematic for scheduling applications, where a large number of market goods with complementary value need to be allocated. For all these reasons, both the general literature of auctions and the specific characteristics of the scheduling application we consider favor the use of an ascending auction over a sealed bid auction. Similarly, sealed bid auctions are not widely used for spectrum auctions. Henceforth, we consider ascending auctions.

We design an ascending auction mechanism that allocates capacity to the competing agents based on their bids and the facility owner's reserve values. We introduce a new market good, a *flexible time block*. We demonstrate that this market good provides both theoretical and computational advantages over the previously studied market good, a *fixed time block*. We compare the *effectiveness* and *computational efficiency* of auctions using the two market goods. Effectiveness is measured by the total value of the solution to the agents and the facility owner. Computational efficiency is measured by the number of rounds that the auction requires to reach closure. We develop both optimal and heuristic solution procedures for the facility owner's winner determination problem. Also, we extend the flexible time block auction to allow bidding for multiple time blocks at each round.

The remainder of the article is organized as follows. Section 2 provides definitions and preliminary results. In section 3, we design and analyze auction mechanisms with our two alternative market goods, and also allow each agent to bid for multiple flexible time blocks. In section 4, a computational study compares the performance of auctions with the two market goods. Section 5 discusses the properties of equilibrium solutions within our auction mechanism. Finally, section 6 provides concluding remarks and managerial insights. All proofs appear in Appendix S1.

2. Preliminaries

In section 2.1, we define the problem studied. In section 2.2, we provide a formal statement of our ascending auction. In section 2.3, we provide a computationally efficient algorithm for an important special case of the problem, and discuss its performance.

2.1. Definitions

We consider n competing agents, $\mathcal{A} = \{1, \ldots, n\}$. Each agent i has a single job i with processing time p_i. This job can be processed nonpreemptively on a common facility. The facility can process at most one job at a time. Let $P = \sum_{i=1}^{n} p_i$. We assume that all the jobs are available at the start of the planning horizon. Assuming that each agent has a single job is not restrictive, since the facility owner treats each bid as if it is from a different agent. If an agent has multiple jobs, then it needs to select a bid from an exponential number of possible combinations of them, which is a computationally challenging problem. However, our model still applies under two simplifications. First, each agent can combine its jobs into a single job and bid for resources for it. Second, each agent can bid for resources for each job individually. Although these two simplified bidding policies are not in general optimal for the agents, they are reasonable heuristics in the absence of information about the other agents (Kutanoglu and Wu 1999).

Let σ indicate a feasible schedule of all the n jobs or a subset of them. The completion time of scheduled job i in σ is denoted by $C_i(\sigma)$, or C_i if the schedule is clear from context. If a job i is not scheduled, then C_i is not defined.

Each job i generates a revenue v_i for agent i if processed, and incurs a scheduling cost $f_i(C_i)$ if completed at time C_i. Thus agent i receives an attained value $v_i - f_i(C_i)$. Wellman et al. (2001) consider a cost structure $f_i(C_i) = 0$ for $C_i \leq d_i$ and $f_i(C_i) = \infty$ for $C_i > d_i$, where d_i is a predefined due date for job i. We consider a more general cost function $f_i(C_i)$ which is a nondecreasing function of C_i, for $i = 1,\ldots,n$. Except for Algorithm GS below and our computational study in section 4, all of our results hold for this function. The price for purchasing capacity to process job i is denoted b_i. Job i generates a profit $v_i - f_i(C_i) - b_i$ for agent i if it completes processing at time C_i, and generates a cost b_i if it is not processed. Each agent maximizes its own profit. Let e represent the facility owner, which sells capacity. The facility owner has a set of consecutive time slots, $\mathcal{T} = \{1, \ldots, T\}$ to allocate to agents. Each time slot t that defines a time interval $[t-1,t]$ is a resource that corresponds to a unit of processing time in a schedule. The facility owner sets a reserve value q_t for time slot t, and maximizes its revenue from selling capacity, that is, the total price of sold capacity plus the total reserve value of unsold capacity.

A *solution* is a mapping $F : \mathcal{T} \rightarrow \mathcal{A} \cup e$, indicating which agent, if any, receives each time slot. Let $F_i = \{t | F(t) = i\}$ denote the set of time slots allocated to agent i, and $F_e = \{t | F(t) = e\}$ the set of unallocated time slots in F, where $F_e \cup (\cup_{i \in \mathcal{A}} F_i) = \mathcal{T}$.

Since without loss of generality each agent schedules its job as early as possible within its allocated capacity, a solution uniquely defines a schedule. The *system value V(F)* of a solution F is the total profit of the n agents plus the revenue of the facility owner. A *globally optimal solution* maximizes the system value, assuming that all the agents' information is known. In a *preemptive* setting, the processing of a job can be stopped and resumed later.

We consider two alternative market goods for production capacity.

1. A fixed time block, $(p, \bar{u})$, $p \leq \bar{u}$, is a set of p consecutive time slots from $\bar{u} - p + 1$ to $\bar{u}$.
2. A flexible time block, (p,u), $p \leq u$, is a set of any p consecutive time slots, where the last one is no later than u.

Fixed time blocks are used as market goods by Shaw (1987, 1988). A fixed time block $(p, \bar{u})$ is feasibly scheduled if it is allocated the p consecutive time slots $\bar{u} - p + 1, \ldots, \bar{u}$. A flexible time block (p,u) is feasibly scheduled if it is allocated p consecutive time slots, where the last one is u or earlier. Let $\mathcal{G}_1$ and $\mathcal{G}_2$ denote the sets of fixed and flexible time blocks, respectively. For preemptive scheduling, Wellman et al. (2001) define a market good $(p, \hat{u})$ to be any set of p time slots with the last one being slot $\hat{u}$. Comparing this definition with that of our flexible time block, neither definition contains the other.

2.2. Auction Mechanism

An *auction mechanism* (McAfee and McMillan 1987) is a set of rules for finding a solution, based on received bids. We consider an *ascending auction*, where the facility owner receives successively higher revenue. At each *round*, all agents are allowed to submit bids $\mathcal{B} = \{B_1, \ldots, B_n\}$ simultaneously. A bid is a tuple $B_i = \langle g_i, b_i \rangle$, where g_i is a market good, and b_i is a bid price. The facility owner determines which bids to admit. If at any round no agent submits a bid, then the auction reaches *closure* and stops. Let $\varepsilon(\cdot)$ denote a bid increment function, which defines a bid's minimum increment over the current price for the bid to be admissible.

Our ascending auction mechanism is defined as follows. At each round, the *bid price* for market good j, denoted by β_j, is the price of the admitted bid for that market good, and is undefined if no bid has been admitted. Also, every market good j has an *ask price* α_j, which is the minimum bid price required for a new bid to be admissible. There are various ways to determine ask prices for market goods for scheduling problems (Kutanoglu and Wu 1999). For example, price discrimination can be used. We consider ask prices without price discrimination. Our pricing mechanism uses the reserve values and positions of unallocated time slots, the bid prices of admitted bids, and the number of bids received by market goods containing each individual time slot in the earlier rounds. This mechanism guarantees that the bidding process terminates, under the condition that the job values are bounded.

Given a vector of ask prices for market goods at each round, every agent solves its *bid determination problem*. An agent can submit multiple bids, and for any market good, an agent can replace its bid with a new bid at later rounds, but it may not withdraw admitted bids. We consider the bid determination problem under a *straightforward*, or *myopic* bidding policy, where each agent bids the ask prices for a set of market goods that *maximize the agent's profit if it wins all of its bids at those prices*. This bidding policy is simple, and a natural choice for agents who have no information about other agents in iterative combinatorial auctions. This straightforward bidding policy is applied to scheduling problems by Kutanoglu and Wu (1999), Wellman et al. (2001), Dewan and Joshi (2002), Geng et al. (2006), and Karabati (2010). Moreover, Reeves et al. (2005) find that optimal bidding strategies are rarely identifiable even for simple scheduling problems. Rothkopf (2007) compares the usefulness of decision theory and game theory in defining effective bidding strategies.

Given a set of admissible bids, including those that are already admitted, the facility owner solves its *winner determination problem*. This problem requires labeling each admissible bid as winning or losing so as to maximize the facility owner's revenue, including its reserve value for any unsold capacity. The winner determination problem is solved at every round. We now formally describe our ascending auction mechanism.

Procedure Ascending Auction

0. Given a reserve value q_t, for time slot t, $t = 1, \ldots, T$. Given a processing time p_i, revenue v_i and scheduling cost function $f_i(C_i)$, for job i, $i = 1, \ldots, n$.
1. For each market good j, construct an ask price α_j, from the reserve values of unallocated time slots and the previously received bids if any.
2. For each market good j, each agent may submit a bid price $b_j \geq \alpha_j$. For each market good j, the facility owner either determines a winner and sets β_j to be the bid price of the winner, or concludes that the market good has not been won.
3. If at least one agent submitted a bid at the most recent application of Step 2, then go to Step 1. Otherwise, allocate each market good to its winner, construct and evaluate the resulting schedule, and terminate.

In Procedure Ascending Auction, each application of Step 2 is a different round of the auction. At the first round ask prices depend only on reserve values, whereas at subsequent rounds they also depend on information about received bids.

2.3. Special Case

When discussing globally optimal schedules and in our computational study, we let $f_i(C_i) = w_iC_i$ denote the *weighted completion time*, where $w_i \geq 0$ is the weight of job i. This objective is a widely used measure of work in process inventory cost (Pinedo 2012). We propose the following dynamic programming algorithm to find a solution to this problem.

Algorithm Global Schedule (GS)

Input
p_i, v_i, w_i, for $i = 1,\ldots,n$; T; q_t, for $t = 1,\ldots,T$.
Initialization
Reindex the n jobs such that $w_1/p_1 \geq \ldots \geq w_n/p_n$.
Value Function
$H(i,t) =$ the maximum system value of agents $1,\ldots,i$ and the facility owner using time slots $1,\ldots,t$.
Boundary Condition
$H(0,t) = \sum_{j=1}^{t} q_j$, where $t \geq 0$.
Optimal Solution Value
$H(n,T)$.
Recurrence Relation

$$H(i,t) = \max\begin{cases} H(i-1,t-p_i) + v_i - w_i t \\ H(i-1,t) \\ H(i,t-1) + q_t. \end{cases}$$

The value function assumes that jobs are processed in index sequence. In the recurrence relation, the first equation processes job i in time slots $t - p_i + 1,\ldots,t$, a revenue v_i is earned and a scheduling cost w_it is incurred, and the state variables $i - 1$ and $t - p_i$ are updated to i and t, respectively. The second equation does not process job i, and the state variable $i - 1$ is updated to i. Finally, in the last equation, time slot t is reserved, a reserve value q_t is earned and the state variable $t - 1$ is updated to t.

Remark 1. Where the scheduling cost is the weighted completion time and the reserve values are the same for all the time slots, Algorithm GS finds a schedule that maximizes the system value in $O(nT)$ time.

Since the input size is $O(n)$ job revenues, weights, and processing times, Algorithm GS runs in pseudo-polynomial time. The following result shows that we cannot expect to develop a polynomial time algorithm that finds an optimal system schedule.

Remark 2. For a special case of the problem without scheduling cost and with $v_i = p_i$ for $i \in A$, and $q_t = 0$ for $t = 1,\ldots,T$, the problem of finding a schedule that maximizes the system value is binary NP-hard.

Moreover, when each time slot has an arbitrary reserve value, the problem of finding a globally optimal schedule is *unary NP-hard* (Agnetis et al. 2008). For this more general problem, Algorithm GS runs as a heuristic with the following characteristics.

Remark 3. Where the scheduling cost is the weighted completion time:

a. Algorithm GS finds a feasible schedule with total system value at least 1/2 that of an optimal schedule;
b. If there exists an optimal schedule without unused time slots, then Algorithm GS finds an optimal schedule.

Part b of Remark 3 shows that Algorithm GS optimally solves many naturally occurring problem instances where capacity is tight and the reserve values are not too high.

3. Ascending Auctions

In sections 3.1 and 3.2, respectively, we consider ascending auctions with fixed and flexible time blocks as market goods. In section 3.3, a new bidding format is introduced, where each agent can bid for multiple flexible time blocks in each round.

3.1. Fixed Time Blocks

The definition of a fixed time block creates a combinatorial auction where each agent bids for a bundle of time slots, and wins either all of them or nothing. In either case, the resulting schedule is nonpreemptive by construction.

We consider an ascending auction mechanism with fixed time blocks as market goods. A bid by agent i is a tuple $B_i = \langle (p_i, \bar{u}_i), b_i \rangle$, where $(p_i, \bar{u}_i)$ is a fixed time block containing time slots $\bar{u}_i - p_i + 1,\ldots,\bar{u}_i$, and b_i is a bid price. The winner determination problem at each round requires that each time slot is allocated to at most one bid and the block of each winning bid is feasibly scheduled. We propose the following algorithm to solve this problem.

Algorithm Winner Determination (WD)

Input
$\mathcal{B} = \{B_1,\ldots,B_n\}$, where $B_i = \langle (p_i, \bar{u}_i), b_i' \rangle$, for $i = 1,\ldots,n$, where $b_i' = b_i$ if bid i is newly submitted,

$b'_i = b_i + \delta$ if bid i is already admitted, with $\delta > 0$ suitably small; T; q_t, for $t = 1,\ldots,T$.

Initialization

Index all the bids by nondecreasing order of $\bar{u}_i$ values.

Value Function

The largest integer no larger than $H(i,t)$, where $H(i,t)$ = the maximum revenue for the facility owner obtained from the bids $B_1, \ldots, B_i$, where the blocks of admitted bids among $B_1, \ldots, B_i$ complete processing no later than t.

Boundary Condition

$$H(i,t) = \begin{cases} \sum_{j=1}^{t} q_j, & \text{if } i = 0 \text{ and } t > 0 \\ 0, & \text{if } t = 0 \\ -\infty, & \text{if } t < 0 \end{cases}.$$

Optimal Solution Value

$$\lfloor H(n, \min\{\bar{u}_n, T\}) \rfloor + \sum_{j=\bar{u}_n+1}^{T} q_j.$$

Recurrence Relation

$$H(i,t) = \max \begin{cases} H(i-1, t-p_i) + b'_i, & \text{if } \bar{u}_i = t \\ H(i, t-1) + q_t \\ H(i-1, t). \end{cases}$$

Output

Schedule the fixed time blocks of the winning bids.

In the recurrence relation, when $\bar{u}_i = t$, the first equation admits bid B_i, a revenue b'_i is earned, and the state variables $i - 1$ and $t - p_i$ are updated to i and t, respectively. The second equation reserves time slot t, a value q_t is earned, and the state variable $t-1$ is updated to t. Finally, the last equation rejects bid B_i, and the state variable $i - 1$ is updated to i. The definition of b'_i gives priority to previously admitted bids when ties occur. Also, if a tie occurs among the three equations in the recurrence relation, then Algorithm WD prioritizes the possible decisions in the order shown.

THEOREM 1. *For the winner determination problem with fixed time blocks as market goods, Algorithm WD finds a schedule with maximum total revenue for the facility owner in* $O(\max\{n \log n, T, n\bar{u}_n\})$ *time.*

Since the input size is $O(\max\{n,T\})$ and $\bar{u}_n \le T$, Algorithm WD runs in polynomial time. However, if the reserve value of all time slots is a constant, then the input size becomes $O(n)$, and a more efficient algorithm exists. Then, the winner determination problem can be modeled as a scheduling problem with fixed start and end times, with the objective of maximizing the total profit of the scheduled jobs. Arkin and Silverberg (1987) describe an $O(n^2)$ time optimal algorithm for this problem.

For each block $(p, \bar{u})$, its *current price*, denoted by $\gamma_{(p,\bar{u})}$, is defined as follows.

1. If no time slots within block $(p, \bar{u})$ are allocated, then $\gamma_{(p,\bar{u})} = \sum_{t=\bar{u}-p+1}^{\bar{u}} q_t$;
2. If block $(p, \bar{u})$ is allocated with bid price $\beta_{(p,\bar{u})}$, then $\gamma_{(p,\bar{u})} = \beta_{(p,\bar{u})}$;
3. Otherwise, $\gamma_{(p,\bar{u})}$ is equal to the total bid price of all the allocated fixed time blocks that contain at least one time slot within block $(p, \bar{u})$, plus the total reserve value of the unallocated time slots within block $(p, \bar{u})$.

Note that the current price of a time block may decrease when the time slots it contains are reallocated as the auction progresses. The ask price $\alpha_{(p,\bar{u})}$ is equal to the current price $\gamma_{(p,\bar{u})}$ if no time slots within block $(p, \bar{u})$ are allocated; otherwise, it is equal to $\gamma_{(p,\bar{u})}$, plus a positive bid increment $\varepsilon(\cdot)$. We propose a bid increment function that (a) is based on the facility owner's total revenue in the current solution and the number of bids received by time blocks that contain each individual time slot and (b) simulates the shape of the agents' utility function where the scheduling cost of each job is a linearly increasing function of its completion time. Let Z denote the facility owner's total revenue in the current solution. Let $\eta(t)$ denote the number of bids received by all the fixed time blocks containing time slot t in all the earlier rounds, and let $M = \sum_{t=1}^{T} \eta(t)$. Then the bid increment for block $(p, \bar{u})$ is defined as:

$$\varepsilon(p, \bar{u}) = \max\left\{ \varepsilon, \tau \left(\frac{\sum_{t=\bar{u}-p+1}^{\bar{u}} \eta(t)}{M} Z + \frac{\sum_{t=\bar{u}-p+1}^{\bar{u}} t}{T} \right) \right\} \tag{1}$$

where τ is a predetermined constant, and ε is a minimum bid increment.

Using the number of bids received by time slots within each market good to define its ask price reduces conflicts over resources among the agents. This ask price definition is *adaptive*, in that if the ask price of a time block is defined too high, then that block receives fewer bids relative to other blocks and consequently its ask price decreases. However, with a bid increment, the ask price is always higher than the current price of a time block, and thus the facility owner's revenue increases whenever at least one time block with an ask price containing a bid increment is sold during a round of the auction. Therefore, if job values are bounded, termination of the bidding process is guaranteed. Another useful property of this ask price definition is *subadditivity*. That is, for any two consecutive time blocks $(p_1, \bar{u}_1)$ and $(p_2, \bar{u}_2)$ satisfying $\bar{u}_2 = \bar{u}_1 + p_2$, their total ask price is no less than the ask price of block $(p_1 + p_2, \bar{u}_2)$, that is, $\alpha_{(p_1,\bar{u}_1)} + \alpha_{(p_2,\bar{u}_2)} \geq \alpha_{(p_1+p_2,\bar{u}_2)}$. Hence, without loss of profitability, each agent bids for a single fixed time block instead of two consecutive fixed time blocks.

Remark 4. For an agent's bid determination problem, due to subadditivity of the ask prices, there exists an optimal set of bids containing at most one bid. For agent i, the candidate fixed time blocks are $(p_i, p_i),\ (p_i, p_i + 1), \ldots, (p_i, T)$. Therefore, a straightforward enumeration procedure finds a fixed time block for which to bid in O(T) time.

Next, we define an equilibrium solution for the allocation of fixed time blocks. Let $v_i^*(p, \bar{u})$ be the value of agent i for holding fixed time block $(p, \bar{u})$, that is, $v_i - f_i(\bar{u} - p + p_i)$ if $p \geq p_i$, and 0 if $p < p_i$. Let $\rho = \{\rho_{(p,\bar{u})} | (p, \bar{u}) \in \mathcal{G}_1\}$ denote the prices of the fixed time blocks. Recall that $F_i = \{t | F(t) = i\}$. Let F_i^x denote a minimal set of mutually disjoint time blocks that cover F_i. Let $H_i(\rho)$ denote the maximum profit of agent i, given ρ, that is:

$$H_i(\rho) \equiv \max_{(p,\bar{u}) \in \mathcal{G}_1} \{v_i^*(p, \bar{u}) - \rho_{(p,\bar{u})}\}.$$

A solution F is in equilibrium at prices ρ if and only if:

1. For each agent i, $v_i^*(F_i) - \sum_{(p,\bar{u}) \in F_i^x} \rho_{(p,\bar{u})} = H_i(\rho)$;
2. For a block $(p, \bar{u})$ containing no allocated time slots, $\rho_{(p,\bar{u})} = \sum_{t=\bar{u}-p+1}^{\bar{u}} q_t$;
3. For an allocated block $(p, \bar{u})$, $\rho_{(p,\bar{u})} \geq \sum_{t=\bar{u}-p+1}^{\bar{u}} q_t$;
4. For a block $(p, \bar{u})$ not satisfying either of the two previous conditions, $\rho_{(p,\bar{u})}$ equals the total price of the allocated blocks containing at least one time slot within block $(p, \bar{u})$, plus the total reserve value of the unallocated time slots within block $(p, \bar{u})$.

Theorem 2. *In an auction with fixed time blocks as market goods, the scheduling cost (respectively, system value) of an equilibrium solution can be arbitrarily large (resp., small) compared with that of an optimal solution, even when the following three conditions hold:*

1. *All the reserve values are* 0;
2. *The facility owner has sufficient time slots that all jobs can be processed;*
3. *The scheduling cost of each job is the weighted completion time.*

Theorem 2 shows that an equilibrium solution with fixed time blocks as market goods may not be close to optimal. This is because the prices of fixed time blocks are not additive. This is due to *complementarity* and *substitutability* of the market goods for the agents, as a result of which a time block can be more or less valuable to an agent depending upon whether it also possesses another time block.

3.2. Flexible Time Blocks

The fixed time block auction does not provide a way for agents to express their value of scheduling flexibility. This motivates us to design a new market good, a flexible time block, where the facility owner can process a job at or earlier than its required completion time.

With flexible time blocks as market goods, a bid is a tuple $B_i = \langle (p_i, u_i), b_i \rangle$, where (p_i, u_i) is a flexible time block, and b_i is a bid price. We consider the winner determination problem for flexible time blocks. An agent bidding for time block (p_i, u_i) can be allocated any p_i time slots $C_i - p_i + 1, \ldots, C_i$, where $C_i - p_i + 1 \geq 1$ and $C_i \leq u_i$. Consider a variant of Algorithm WD, which we term Algorithm WD′. In Algorithm WD′, u_i replaces $\bar{u}_i$ throughout. Also, the condition in the first equation of the recurrence relation is changed from $\bar{u}_i = t$ to $u_i \geq t$. However, Algorithm WD′ does not in general find optimal solutions for the facility owner.

Example 1. Consider a facility owner with $T = 7$ time slots, where slot 3 has reserve value $q_3 = a$ and all other slots have reserve value 0. There are two submitted bids: $B_1 = \langle (2, 7), a \rangle$, and $B_2 = \langle (3, 6), a + b \rangle$, where $a > b$. In an optimal solution, B_1 is allocated time slots 1 and 2, and B_2 is allocated time slots 4, 5, and 6, and thus the facility owner receives a total revenue of $3a + b$. However, Algorithm WD′ allocates time slots 1, 2, and 3 to bid B_2, and then time slots 4 and 5 to bid B_1, and thus the facility owner receives $2a + b$.

The following theorem summarizes our results about the winner determination problem.

Theorem 3. *For the winner determination problem with flexible time blocks as market goods:*

a. *If the time slots have nonincreasing reserve values, then Algorithm WD′ finds an optimal schedule in* $O(\max\{n \log n, T, nu_n\})$ *time;*
b. *If there exists an optimal schedule without unallocated time slots, then Algorithm WD′ finds an optimal schedule;*

c. *Algorithm WD′ finds a schedule with total revenue at least as large as the revenue found by Algorithm WD, and at least* $1/2$ *that of an optimal schedule;*
d. *If the time slots have the same (respectively, arbitrary) reserve value, then it is binary (resp., unary) NP-hard to find an optimal schedule.*

Since scheduling costs are nondecreasing with job completion times, agents have no incentive to pay higher prices for flexible time blocks with larger u_i values. Therefore, it is natural for the facility owner to define the *nonincreasing* reserve values for time slots considered in part a of Theorem 3. Part c of Theorem 3 shows that the facility owner can expect at least as much revenue from flexible time blocks as from fixed time blocks.

Next, in order to evaluate the performance of Algorithm WD′, we establish an upper bound on the maximum revenue for the facility owner.

Algorithm Upper Bound (UB)
Input
$\mathcal{B} = \{B_1, \ldots, B_n\}$, where $B_i = \langle (p_i, u_i), b_i \rangle$, for $i = 1,\ldots,n$; T; q_t, for $t = 1,\ldots,T$.
Initialization
Index all the time slots in nondecreasing order of q_t values; index all the bids in nondecreasing order of u_i values.
Output
$H(n, \min\{u_n, T\}) + \sum_{j=u_n+1}^{T} q_j$, as found by Algorithm WD′.

REMARK 5. Algorithm UB finds an upper bound on the maximum revenue for the winner determination problem with flexible time blocks as market goods.

For fixed time blocks, each time slot either belongs to a block or not, and the winner determination problem can be formulated as a multidimensional knapsack problem (see e.g., Sandholm et al. 2005). For flexible time blocks, this formulation does not work, since a time block does not contain a fixed set of time slots. Our approach is to regard each flexible time block (p,u) as $u - p + 1$ fixed time blocks (p,p), $(p, p+1)\ldots$, (p,u), and require that at most one of those fixed time blocks can be admitted by the facility owner. Our computational study shows that this integer program can be solved efficiently.

We compare the heuristic solution value and the upper bound from Algorithm UB with an optimal solution. Our data set consists of 360 problem instances, which are randomly generated as follows. The number of bids is $n \in \{5,10,20,30,40,50\}$, the length of blocks is $p_i \sim UI[1, \ldots, 10]$, and the bid price is $b_i \sim UI[1, 10]p_i$. Given $P = \sum_{i=1}^{n} p_i$, the capacity T is generated as $T \in \{[.5P],[.6P],\ldots,P\}$, where $[x]$ denotes the value of x rounded to the closest integer. The reserve value of each time slot is generated as $q_t \sim UI[1, \ldots, 10]$. The maximum completion time of each block is generated as $u_i \sim UI[p_i, T]$. For each of the 6 × 6 = 36 possible combinations of parameters, we randomly generate 10 problem instances. The model is coded using Visual C++ Express 2010 with 64-bit CPLEX Concert Technology, on a computer with a 2.67 GHz processor and 4.00 GB of RAM.

Our computational results are summarized in Table 1, where R and R' denote the percentage of revenue obtained by Algorithm WD′ and the upper bound found by Algorithm UB, relative to the optimal value found by the integer program, respectively. Also, TOPT denotes the CPU time in seconds required by the optimal algorithm. The overall mean values of R and R' are 99.29% and 106.95%, respectively. Algorithm WD′ performs consistently well across different values of n and T/P. The slightly better performance for smaller T/P values occurs because in many such data sets there exists an optimal schedule without unallocated time slots, in which case Algorithm WD′ is optimal, from Theorem 3. The results suggest that Algorithm WD′ is a very effective heuristic for the winner determination problem with flexible time blocks, up to $n = 50$. The upper bound found by Algorithm WD is also typically close to an optimal value. The optimal algorithm using the integer program is solved efficiently by CPLEX.

We now define the ask price $\alpha_{(p,u)}$ of a flexible time block (p,u). Letting $\gamma_{(p,u)}$ denote the current price of block (p,u), we have

1. If none of time slots $u - p + 1, u - p + 2, \ldots, u$ is allocated, then $\gamma_{(p,u)} = \sum_{t=u-p+1}^{u} q_t$;
2. If time slots $u - p + 1, u - p + 2, \ldots, u$ are allocated to a bid with a bid price β, then $\gamma_{(p,u)} = \beta$;
3. Otherwise, γ is equal to the total bid price of the bids which are allocated at least one time slot within $u - p + 1, u - p + 2, \ldots, u$, plus the

Table 1 Average Performance of Algorithm WD′ and the Upper Bound

Parameter	R%	R'%	TOPT
$n = 5$	99.51	106.18	0.01
$n = 10$	99.47	106.89	0.01
$n = 20$	99.21	106.64	0.04
$n = 30$	99.21	107.64	0.07
$n = 40$	99.19	107.26	0.14
$n = 50$	99.14	107.09	0.26
$T = [.5P]$	99.91	108.69	0.06
$T = [.6P]$	99.58	108.33	0.07
$T = [.7P]$	99.26	106.88	0.08
$T = [.8P]$	99.16	106.51	0.09
$T = [.9P]$	98.97	105.88	0.11
$T = P$	98.85	105.41	0.11

total reserve value of the unallocated time slots within $u - p + 1, u - p + 2, \ldots, u$.

4. Then, $\alpha_{(p,u)} = \gamma_{(p,u)} + \varepsilon(p,u)$, where $\varepsilon(p,u)$ is defined as in Equation (1).

REMARK 6. The bid determination problem in the auction with flexible time blocks can be solved via direct enumeration in $O(T)$ time.

An equilibrium solution of the scheduling problem with flexible time blocks as market goods is defined similarly to that with fixed time blocks. An additional requirement is that the facility owner's revenue is maximized for the admitted flexible time blocks.

COROLLARY 1. *In an auction with flexible time blocks as market goods, the scheduling cost (respectively, system value) of an equilibrium solution can be arbitrarily large (resp., small) compared with that of an optimal solution, even when the following three conditions hold:*
1. All the reserve values are 0;
2. The facility owner has sufficient time slots that all jobs can be processed;
3. The scheduling cost of each job is the weighted completion time.

More generally, however, a solution that is in equilibrium with fixed time blocks as market goods may not be in equilibrium with flexible time blocks, since the facility owner may be able to increase its revenue by rescheduling the admitted time blocks.

3.3. Multiple Flexible Time Blocks

To give the agents an opportunity to express their preferences in more detail, we develop an alternative bidding format. At each round, each agent i proposes a bidding function that requests p_i consecutive time slots, and offers a bid price $\beta_{(p_i,t)}$ if the last time slot it wins is no greater than t, for every $t = p_i, \ldots, T$. This is equivalent to allowing each agent to bid for multiple flexible time blocks, but win at most one. The ask price, $\alpha_{(p,u)}$, is as defined in section 3.2. We name this auction a *multiple flexible time block auction*.

Suppose that at a round of the multiple flexible time block auction, the maximum achievable profit for an agent i is r_i^*, assuming that no other agents bid. If an agent employs the straightforward bidding policy defined in section 2.2, then it only bids for flexible time blocks that result in profit r_i^*. As a result, each agent bids for very few flexible time blocks; hence, the multiple flexible time block auction performs similarly to the individual flexible time block auction. In practice, however, an agent may give up some profit in order to win market goods at earlier rounds of an auction. For example, in a *weakly straightforward* bidding policy, an agent i targets a profit no less than $k_i r_i^*$, $0 < k_i \leq 1$, and thus bids for any flexible time block of length p_i that results in a profit no less than $k_i r_i^*$, given a bid price equal to the block's ask price. Here, k_i can be understood as an *optimism parameter* of agent i. The choice of k_i value and the performance of this bidding policy are investigated in our computational study in section 4. The bid determination problem under this bidding policy can be solved in $O(T)$ time.

For the winner determination problem, the integer program with each flexible time block treated as multiple fixed time blocks with the same bid price is still computationally solvable. Next, we propose an easy-to-implement *local search* procedure to find a heuristic solution. This procedure is tested in section 4.

Algorithm Local Search (LS)
Input
$\mathcal{B}_i = \{B_{i1}, \ldots, B_{im_i}\}$, for $i = 1, \ldots, n$; T; q_t, for $t = 1, \ldots, T$; a preselected integer $l \geq 1$.
Selection
Randomly select a single bid from each $\mathcal{B}_i$ with $m_i \geq 1$.
Solution
For the bids chosen at the Selection step, run Algorithm WD′ to find a solution.
Termination
Run the Selection and Solution steps l times and find the best solution.

4. Computational Study

In this section, we investigate the effectiveness and computational efficiency of ascending auctions that allow bidding for individual fixed time blocks, individual flexible time blocks, and multiple flexible time blocks. Using the guidelines of Hall and Posner (2001), (a) we generate a wide range of parameter specifications, (b) all the parameters can be rescaled together without significantly affecting performance, and (c) the experimental design varies only the parameters that may affect the analysis. We first select $n \in \{5, 10, 20, 30, 40, 50\}$ and generate $p_i \sim UI[1, \ldots, 10]$. The scheduling cost is the total weighted completion time, where the weight of each job i is generated in three ways: as a constant using $w_i = 6$, randomized using $w_i \sim UI[1, \ldots, 10]$, and proportional with processing time using $w_i = p_i$. For each instance with total processing time P, the facility owner's capacity T is generated using $T \in \{[.5P], [.7P], [.9P]\}$. Since job revenue is naturally positively correlated with processing time and weight, we let each job j generate a revenue $v_i = (p_i + w_i)T$, if it is processed. The reserve value of each time slot is generated in three ways: as a constant using $q_t = [(1 + T)/2]$, randomized using $q_t \sim UI[1, \ldots, T]$, and linearly decreasing using $q_t = T - t + 1$. Finally, we define the bid increment function using Equation (1), where $\varepsilon = 1$ and $\tau \in \{5,10,15\}$. Here, changes in the bid increment function also reflect changes in bidding strategy. For example, a large bid increment simulates a jump in a bid price. For each of the $6 \times 3 \times 3 \times 3 \times 3 = 486$ possible combinations of parameters, we randomly generate 10 problem instances, for a total of 4,860.

For auctions with an individual fixed or flexible time block as the market good, the agents use the straightforward bidding policy. For the multiple flexible time block auction, the agents use the weakly straightforward bidding policy. If the optimism parameter k_i is small, then the bids reveal little information about the agents' true preferences. Our preliminary computational study shows that the system value provided by the auction is insensitive to the choice of k_i values, if $k_i \leq 0.6$. Therefore, for each agent i, at each round, we randomly generate a k_i value from 0.6 to 1, as a multiple of 0.01. The winner determination problem for the auction with fixed time blocks is optimally solved by Algorithm WD. For the auction with individual flexible time blocks, Algorithm WD′ provides only heuristic solutions unless the reserve values of the time slots are nondecreasing. We solve the winner determination using both the optimal integer program and Algorithm WD′. For the multiple flexible time blocks auction, the winner determination problem is solved using both the optimal integer program and Algorithm LS with $l = 100$.

Regarding the computation time needed to reach closure, we test instances with reserve values that are uniformly distributed. Using heuristic winner determination, the CPU times for instances with $n \in \{5,20,50\}$ have means < 0.01, 0.02, and 0.33 seconds, and maximum values < 0.01, 0.09, and 1.32 seconds, respectively. Using optimal winner determination, the CPU times for instances with $n \in \{5,20,50\}$ have means 0.03, 0.98, and 23.06 seconds, and maximum values 0.10, 5.12, and 1520.68 seconds, respectively. Thus, optimal winner determination requires extra computation time that increases rapidly with instance size. Therefore, for instances above $n = 50$, we recommend heuristic winner determination.

To evaluate the effectiveness of the three auctions, we use the system value of a schedule from Algorithm GS as a benchmark, and compare their system values. Each job i generates a value $v_i - w_iC_i = (p_i + w_i)T - w_iC_i \geq p_iT$ for completion time $C_i \leq T$, which is at least the total reserve value of any p_i consecutive time slots. Therefore, if $T < P$, there typically exists an optimal schedule without unused time slots; hence, from Remark 3, Algorithm GS finds an optimal schedule. Let R^*_{fx}, R^*_{fl}, and R^*_{mf} denote the percentages of system value provided by the auctions using individual fixed and flexible time blocks, and multiple flexible time blocks with optimal winner determination, respectively, relative to that from Algorithm GS. Further, let R_{fl} and R_{mf} denote the percentages of system value provided by the auctions using individual and multiple flexible time blocks with Algorithms WD′ and LS, respectively, relative to that from Algorithm GS. These percentages are first calculated for each individual instance, and then averaged across all the instances tested.

The effects of different parameters on the values of R^*_{fx}, R^*_{fl}, R^*_{mf}, R_{fl}, and R_{mf} are summarized in Table 2. For the auctions with flexible time blocks as market goods, the difference that results from optimal *vs.* heuristic winner determination is negligible. This is consistent with the near optimal average performance of the heuristic winner determination algorithm shown in Table 2. Mean values over all 4,860 instances are $R^*_{fx} = 92.68\%$, $R^*_{fl} = 95.01\%$, and $R^*_{mf} = 95.55\%$. Thus, flexible time block auctions generate more system value on average than fixed time block auctions, and multiple flexible time blocks provide a small further improvement. These two statements are verified by t-tests, for which the two p-values are both < 0.01.

The results in Table 2 provide several observations. First, as the number of agents increases, all three auctions perform better. Second, all three auctions perform best when the weight of a job is proportional to its processing time. This is because the processing sequence of the jobs does not affect the scheduling cost. Third, as the available scheduling resources increase, all three auctions generate system value that is closer to optimal. Fourth, the reserve values of time slots do not have a significant impact on the system values generated by the auctions with flexible time

Table 2 System Value for Various Parameter Settings

Parameter	$R^*_{fx}\%$	$R^*_{fl}\%$	$R^*_{mf}\%$	$R_{fl}\%$	$R_{mf}\%$
$n = 5$	90.89	91.04	93.17	91.20	92.27
$n = 10$	91.97	93.08	92.90	93.01	93.04
$n = 20$	92.69	95.12	95.24	95.09	94.81
$n = 30$	93.28	96.62	96.76	96.58	96.58
$n = 40$	93.33	96.89	97.37	96.85	97.32
$n = 50$	93.91	97.33	97.83	97.20	97.74
$w_i = 6$	87.38	92.21	93.21	92.07	92.79
$w_i \sim UI[1,\ldots,10]$	92.16	94.30	94.81	94.32	94.55
$w_i = p_i$	98.49	98.52	98.61	98.58	98.54
$T = [.5P]$	90.17	93.03	93.70	93.09	93.59
$T = [.7P]$	92.72	95.00	95.67	94.88	95.24
$T = [.9P]$	95.14	97.01	97.27	97.00	97.06
$q_t = [(T+1)/2]$	91.86	94.94	95.63	94.94	95.48
$q_t \sim UI[1,\ldots,T]$	92.15	94.91	95.29	94.84	95.17
$q_t = T - t + 1$	94.02	95.19	95.71	95.19	95.24
$\tau = 5$	92.41	95.09	95.57	95.11	95.56
$\tau = 10$	92.80	95.07	95.65	95.03	95.32
$\tau = 15$	92.82	94.88	95.42	94.83	95.00

blocks, but the auction with fixed blocks performs better with decreasing reserve values. Finally, as the bid increment parameter τ increases from 5 to 15, the fixed time block auction performs slightly better, whereas the auctions using flexible time blocks perform slightly worse. This indicates that flexible time block auctions provide higher system value if ask prices are more sensitive to the agents' preferences as revealed through their bids. Also, auctions with flexible time blocks as market goods perform well across different bidding strategies by the agents.

Next, we study the benefits of flexible time block auctions to the agents and the facility owner. In Table 3, columns RA^*_{fx} and RA^*_{fl} (respectively, RF^*_{mf} and RF^*_{fl}) show the percentages of the agents' value (resp., facility owner's value) given by the individual and multiple flexible time block auctions with optimal winner determination, relative to those given by the fixed time block auction, respectively. Flexible time block auctions favor the agents more when the number of agents is large, the jobs have equal weights, the available scheduling resource is scarce, the reserve values of time slots are the same, or the bid increment is small. Note that when jobs have weight proportional to processing time, flexible time blocks generate less value to the agents than the fixed time block auction. The benefit of flexible time block auctions to the facility owner is higher when the jobs have equal weights or weight proportional to processing time, the amount of scheduling resource is large, or the reserve values of time slots are random. Overall, the individual and multiple flexible time block auctions generate 18.99% and 21.89% more profit on average for the agents than the fixed time block auction, respectively. Also, the individual and multiple flexible time block auctions generate 3.48% and 3.57% more revenue on average for the facility owner than the fixed time block auction, respectively. This motivates the facility owner to choose flexible time blocks as the market good, and to allow bidding for multiple time blocks.

We also consider the computational efficiency of the three auctions. The mean number of rounds to reach closure in the individual fixed and flexible time block and multiple flexible time block auctions with optimal winner determination is 57.7, 55.5, and 25.1, respectively. Whereas, the mean number of rounds to reach closure in the individual flexible time block and multiple flexible time block auctions with heuristic winner determination is 55.4 and 31.0, respectively. Thus, the multiple flexible time block auction requires fewer rounds to reach closure using optimal winner determination. Detailed results appear in Table 4. All three auctions require more rounds to reach closure when the number of agents is larger, more scheduling resources are available, or the bid increment function is smaller. Also, all three auctions require more rounds when the weights of all jobs are the same, since more jobs are competitive with each other. Finally, linearly decreasing reserve values lead to more rounds in all three auctions. This is because larger reserve values early in the time horizon result in competitive bidding for both early and late capacity.

We summarize our computational results for the relative performance of the three auctions. Using fixed time blocks as market goods is not recommended, since this provides low system value, low profit for the agents, and low revenue for the facility owner, and is also inefficient at reaching closure. The use of flexible time blocks as market goods is

Table 3 Agents and Facility Owner's Values for Various Parameter Settings

Parameter	RA^*_{fl}%	RA^*_{ml}%	RF^*_{fl}%	RF^*_{ml}%
$n = 5$	108.19	126.13	104.01	102.13
$n = 10$	113.05	109.22	103.09	103.56
$n = 20$	115.46	111.64	103.62	104.54
$n = 30$	119.95	120.62	103.48	103.68
$n = 40$	126.63	134.82	103.27	103.53
$n = 50$	124.68	128.91	103.41	103.96
$w_i = 6$	144.47	152.94	104.68	104.65
$w_i \sim UI[1, \ldots, 10]$	114.85	118.43	101.88	102.18
$w_i = p_i$	94.66	94.30	103.88	103.87
$T = [.5P]$	128.14	133.99	103.10	103.11
$T = [.7P]$	120.82	124.92	102.70	102.93
$T = [.9P]$	105.02	106.76	104.64	104.66
$q_t = [(T+1)/2]$	125.41	130.93	103.69	103.55
$q_t \sim UI[1, \ldots, T]$	116.33	117.48	104.48	105.28
$q_t = T - t + 1$	112.24	117.26	102.27	101.87
$\tau = 5$	125.49	128.95	103.43	104.48
$\tau = 10$	117.95	122.23	103.38	103.36
$\tau = 15$	110.54	114.50	103.63	102.86

Table 4 Numbers of Rounds to Reach Closure

Parameter	Fix*	Flex*	MFlex*	Flex	MFlex
$n = 5$	4.4	4.8	3.2	4.7	3.4
$n = 10$	12.8	14.0	8.2	13.9	9.2
$n = 20$	35.0	36.1	19.4	36.2	23.4
$n = 30$	62.5	60.3	29.5	60.2	36.1
$n = 40$	93.7	88.9	39.3	88.1	49.4
$n = 50$	138.1	128.5	51.3	129.2	64.7
$w_i = 6$	65.9	59.8	31.4	59.7	37.3
$w_i \sim UI[1, \ldots, 10]$	47.6	45.7	22.4	45.5	27.3
$w_i = p_i$	59.8	60.9	21.6	60.9	28.5
$T = [.5P]$	29.6	27.8	13.5	27.5	16.1
$T = [.7P]$	48.8	45.7	21.1	45.7	25.9
$T = [.9P]$	94.9	92.8	40.8	92.9	51.1
$q_t = [(T+1)/2]$	56.1	55.8	24.2	55.8	29.6
$q_t \sim UI[1, \ldots, T]$	50.2	44.7	20.8	44.5	29.7
$q_t = T - t + 1$	67.0	65.9	30.3	65.9	33.9
$\tau = 5$	83.9	81.6	36.6	81.1	43.8
$\tau = 10$	50.6	47.8	22.0	48.2	27.7
$\tau = 15$	38.7	36.9	16.8	36.8	21.6

recommended, but a careful design of the associated bidding format is important. The multiple flexible time block auction consistently outperforms the auction with the individual flexible time block as the bidding format, both in effectiveness and computational efficiency.

5. Equilibrium Solutions

We discuss the properties of equilibrium solutions with time blocks as market goods. It is well known that, using time slots as market goods for the scheduling problem with or without preemption, an equilibrium solution may not exist (Wellman et al. 2001). However, with time blocks as market goods, an equilibrium solution always exists.

Theorem 4. *There exists an equilibrium solution for any instance of the resource allocation problem with fixed or flexible time blocks as market goods. However, the equilibrium solution is not unique.*

Even though an equilibrium solution is guaranteed to exist with fixed or flexible time blocks as market goods, due to the positive bid increment function that guarantees the auction to reach closure in a finite number of rounds, the ascending auction may not find an equilibrium solution.

Remark 7. The ascending auction may not find an equilibrium solution for an auction with fixed or flexible time blocks as market goods.

In Theorem 2 and Corollary 1, we show that the system value and scheduling cost of an equilibrium solution can be arbitrarily far from globally optimal for the fixed and flexible time block auctions, respectively. In addition, even for an instance with non-increasing reserve values for time slots, there may not exist an equilibrium solution that is globally optimal (Hall and Liu 2010). We now present a positive result under stronger conditions.

Theorem 5. *For auctions with fixed or flexible time blocks as market goods, if $p_1 = \ldots = p_n = p$, and $f_i(p_i) = f_i(p_i + 1) = \ldots = f_i(T) = f_i$ for $i = 1,\ldots,n$, then every globally optimal schedule is supported by an equilibrium solution where every allocated time block has the same price.*

The conditions of Theorem 5 specify that all jobs have the same processing time and each job has a scheduling cost that is independent of its completion time. However, the revenues of the jobs and the reserve values of the time slots are arbitrary. The constructed equilibrium solution is fair, in that all agents pay the same price for their winning time blocks. For the capacity allocation problem defined in Theorem 5, we describe an efficient algorithm to find a globally optimal schedule.

```
Algorithm Global Optimal Schedule (GOS)
Input $f_i(p) = \ldots = f_i(T) = f_i$ and $v_i$, for $i = 1,\ldots,n$; $p_1 = \ldots = p_n = p$; $T$; $q_t$, for $t = 1,\ldots,T$.
Recurrence Relation
```

$$H(i,t) = \max \begin{cases} H(i-1,t-p) + v_i - f_i \\ H(i-1,t) \\ H(i,t-1) + q_t. \end{cases}$$

The value function, boundary condition and optimal solution value of Algorithm GOS are the same as in Algorithm GS. We have the following result.

Remark 8. When all jobs have the same processing time and each job has a scheduling cost that is independent of its completion time, Algorithm GOS finds a schedule that maximizes the system value in $O(nT)$ time.

6. Conclusions

We evaluate the usefulness of market good design for the allocation of production capacity on a single resource among competing agents, through an ascending auction mechanism. We consider a previously studied market good, a fixed time block. We also introduce a new market good, a flexible time block, which is implemented using bidding for each individual time block and bidding for multiple time blocks. Flexible time block auctions allow the agents to explain their preferences better, since they can adjust their bid prices for their value of scheduling flexibility. We study the effectiveness and computational efficiency of the associated auctions, and develop optimal or near optimal algorithms for the bid determination and winner determination problems. Our computational study validates the practical effectiveness and computational efficiency of the flexible time block auction. On average, the multiple flexible time block auction delivers over 95% of the system value achieved by optimal centralized planning, and this performance improves as the number of agents increases. Moreover, both the agents as a whole and the facility owner benefit.

We discuss possible extensions of our work on the capacity allocation problem. First, the fixed and flexible time block goods can be used to allocate resources for natural variations of the problem. Examples occur where a partially processed job earns value for its agent, and/or preemption is allowed. Second, if each agent has multiple jobs, the facility owner faces the same problem as considered here, and our solution approach for the bid determination problem can be used. Third, ascending auctions can be used to allocate scheduling resources in more complex planning environments such as flow shops and job shops. Fourth, allowing reserve values to have density defined as a continuous function of time is a valuable generalization. Another useful generalization occurs where some capacity allocation is performed by centralized planning, but the remaining capacity is allocated by auction. For example, capacity may be partially allocated in advance based on complete information from agents, but then further allocated in a real time auction using updated information. The problem of designing an ascending auction with bounded performance ratios for scheduling cost and system value remains open. The possibility of collusion among the agents, and how it can be prevented, should also be studied. More generally, our work suggests the increased use of flexibility in market good design within auctions for resource allocation.

Our work provides several managerial insights about the allocation of production capacity among multiple competitive agents. First, efficient capacity allocation is particularly important for time-sensitive products. Second, auctions can efficiently allocate capacity among the agents, to achieve a high system value and reach closure quickly. Third, the choice of market good and bidding format makes a significant difference to the value received by the facility owner and the bidding agents. All parties typically benefit from an efficient choice. Fourth, it is valuable for the facility owner to maintain some control over the production schedule during the bidding process, as implemented through the use of flexible time blocks in our work. Fifth, when an efficient auction mechanism is used, the values received by the facility owner and the system are insensitive to the capacity reserve values specified by the facility owner, which is a helpful feature when reserve values are hard to estimate. Finally, it is important to use an auction design where the bidding agents can express their evaluation of production capacity in detail, such as by allowing bidding for multiple flexible time blocks.

Acknowledgments

The authors are grateful to the Senior Editor and two anonymous referees for their constructive comments, which substantially improved the quality of this work.

References

Agnetis, A., P. B. Mirchandani, D. Pacciarelli, A. Pacifici. 2004. Scheduling problems with two competing agents. *Oper. Res.* **52**(2): 229–242.

Agnetis, A., D. Pacciarelli, A. Pacifici. 2007. Multi-agent single machine scheduling. *Annals Oper. Res.* **150**(1): 3–15.

Agnetis, A., D. Pacciarelli, A. Pacifici. 2008. Combinatorial models for multi-agent scheduling problems. E. Levner, ed. *Multiprocessor Scheduling: Theory and Applications*. I-Tech Education and Publishing, Vienna, Austria, 21–46.

Arbib, C., S. Smriglio, M. Servilio. 2004. A competitive scheduling problem and its relevance to UMTS channel assignment. *Networks* **44**(2): 132–141.

Arkin, E. M., E. B. Silverberg 1987. Scheduling jobs with fixed start and end times. *Discrete Appl. Math* **18**(1): 1–8.

Ball, M., G. L. Donohue, K. Hoffman. 2006. Auctions for the safe, efficient and equitable allocation of airspace system resources. P. Cramton, Y. Shoham, R. Steinberg, eds. *Combinatorial Auctions*. MIT Press, Cambridge, MA, 507–538.

Brewer, P. J., C. R. Plott. 1996. A binary conflict ascending price (BICAP) mechanism for the decentralized allocation of the right to use railroad tracks. *Int. J. Ind. Organ.* **14**(6): 857–886.

Brucker, P. 1998. *Scheduling Algorithms*, 2nd edn. Springer, Berlin, Germany.

Cheng, T. C. E., C. T. Ng, J. J. Yuan. 2006. Multi-agent scheduling on a single machine to minimize total weighted number of tardy jobs. *Theor. Comput. Sci.* **362** (1–3): 273–281.

Clearwater, S. H., ed. 1996. *Market-Based Control: A Paradigm for Distributed Resource Allocation*. World Scientific, Hackensack, NJ.

Computech. 2011. Spectrum auctions at the FCC. Available at http://www.computechinc.com/experience/federal/spectrum-auctions (accessed date October 20, 2011).

Cramton, P. 1998. Ascending auctions. *Eur. Econ. Rev.* **42** (3–5): 745–756.

Curiel, I., H. Hamers, F. Klijn. 2002. Sequencing games: A survey. P. Borm, H. Peters, eds. *Chapters in Game Theory: In Honor of Stef Tijs*. Kluwer Academic Publishers, Boston, MA, 27–50.

Curiel, I., G. Pederzoli, S. Tijs 1989. Sequencing games. *Eur. J. Oper. Res.* **40**(3): 344–351.

Curiel, I., J. Potters, R. Prasad, S. Tijs, B. Veltman. 1994. Sequencing and cooperation. *Oper. Res.* **42**(3): 566–568.

Davidsson, P., L. Henesey, L. Ramstedt, J. Tornquist, F. Wernstedt. 2005. An analysis of agent-based approaches to transport logistics. *Transport. Res., Part C* **13**(4): 255–271.

Derrien, F., K. L. Womack. 2003. Auctions vs. bookbuilding and the control of underpricing in hot IPO markets. *Rev. Finan Stud.* **16**(1), 31–61.

Dewan, P., S. Joshi. 2002. Auction-based distributed scheduling in a dynamic job shop environment. *Int. J. Prod. Res.* **40**(5): 1173–1191.

Federal Communications Commission. 1998. *Federal communications commission: All about auctions*. Report, Federal Communications Commission Wireless Telecommunications Bureau, Washington, DC.

Financialexpress. 2007. Panel seeks views on e-auction, coal distribution by Jan 29. Available at http://www.financialexpress.com/news/panel-seeks-views-on-eauction-coal-distribution-by-jan-29/191771/ (accessed date October 20, 2011).

Geng, J., R. H. Kwon, J. C. Beck. 2006. A combinatorial auction-based heuristic for minimizing total weighted tardiness in job shop scheduling. Working paper, University of Toronto, Ontario, Canada.

Google Display Network. 2011. Creativity at the right place and right time delivers measurable results. Available at http://www.google.com/intl/en/ads/displaynetwork/ (accessed date October 20, 2011).

Hall N. G., Z. Liu. 2010. On auction protocols for decentralized scheduling. *Games and Econ Behav.* **72**(2): 583–585.

Hall, N. G., M. E. Posner. 2001. Generating experimental data for computational testing with machine scheduling applications. *Oper. Res.* **49**(6): 854–865.

Huang, S., H. Chen, L.-J. Zhang. 2005. Progressive auction based resource allocation in service-oriented architecture. Proceedings of the 2005 IEEE International Conference on Services Computing (SCC'05), **2**, 85–92.

Industry Canada. 2007. Licensing framework for the auction for spectrum licenses for advanced wireless services and other spectrum in the 2 GHz range. White paper, Industry Canada, Ottawa, Ontario, Canada.

Karabati, S. 2010. Pricing and scheduling with multiple products and non-cooperative agents. Working paper, College of Administrative Sciences and Economics, Koç University, Istanbul, Turkey.

Krishna, V. 2002. *Auction Theory*. Academic Press, St. Louis, MO.

Kubzin, M. A., V. A. Strusevich. 2006. Planning machine maintenance in two-machine shop scheduling. *Oper. Res.* **54**(4): 789–800.

Kunreuther, H., P. R. Kleindorfer. 1986. A sealed-bid auction mechanism for siting noxious facilities. *The Am. Econ. Rev.* **76** (2): 295–299.

Kutanoglu, E., S. D. Wu. 1999. On combinatorial auction and Lagrangean relaxation for distributed resource scheduling. *IIE Trans.* **31**(9), 813–826.

McAfee, R. P., T. R. Lewis 2009. *Introduction to Economic Analysis*. Flat World Knowledge, Irvington, NY.

McAfee, R. P., J. McMillan. 1987. Auctions and bidding. *J. Econ. Lit.* **25**(2): 699–738.

NASA. 2008. Game theoretic scheduling of the deep space network. Available at http://ti.arc.nasa.gov/tech/project/details.php?pid= 236&gid= 8&ta= 2 (accessed date January 3, 2011).

Pinedo, M. 2012. *Scheduling, Theory, Algorithms and Systems*, 4th edn. Springer, New York.

Rassenti, S. J. 1982. A combinatorial auction mechanism for airport time slot allocation. *Bell J. Econ.* **13**(2): 402–417.

Reddy, S. K., J. E. Aronson, A. Stam. 1998. SPOT: Scheduling programs optimally for television. *Manage. Sci.* **44**(1): 83–102.

Reeves, D. M., M. P. Wellman, J. K. MacKie-Mason, A. Osepayshvili. 2005. Exploring bidding strategies for market-based scheduling. *Decision Support Systems* **39**(1): 67–85.

Rothkopf, M. 2007. Decision analysis: The right tool for auctions. *Decision Analysis* **4**(3): 167–172.

Sandholm, T., S. Suri, A. Gilpin. 2005. CABOB: A fast optimal algorithm for winner determination in combinatorial auctions. *Manage. Sci.* **51**(3): 374–390.

Shaw, M. J. 1987. A distributed scheduling method for computer integrated manufacturing: The use of local area networks in cellular-systems. *Int. J. Prod. Res.* **25**(9): 1285–1303.

Shaw, M. J. 1988. Dynamic scheduling in cellular manufacturing systems: A framework for networked decision-making. *J. Manuf. Systems* **7**(2): 83–94.

Shen, W., Q. Hao, H.-J. Yoon, D. H. Norrie. 2006. Applications of agent-based systems in intelligent manufacturing: An updated review. *Adv. Eng. Inform.* **20**(4): 415–431.

Sutherland, I. E. 1968. A futures market in computer time. *Commun. ACM* **11**(6): 449–451.

Talluri, K. T., G. J. van Ryzin. 2004. *The Theory and Practice of Revenue Management*. Kluwer Academic Publishers, Norwell, MA.

Varian, H. R., J. K. MacKie-Mason. 1994. *Generalized Vickrey auctions*. Report, Department of Economics, University of Michigan, Ann Arbor, MI.

Wellman, M. P., W. E. Walsh, P. R. Wurman, J. K. MacKie-Mason. 2001. Auction protocols for decentralized scheduling. *Games Econ. Behav.* **35**(1–2): 271–303.

Supporting Information

Additional Supporting Information may be found in the online version of this article:

Appendix S1. Technical proofs.