Some Economics of Market-based Distributed Scheduling

William E. Walsh Michael P. Wellman Peter R. Wurman
Jeffrey K. MacKie-Mason

University of Michigan, Ann Arbor, USA
{ *wew, wellman, pwurman, jmm* }*@umich.edu*

October 1, 1997

Abstract

Market mechanisms solve distributed scheduling problems by allocating the scheduled resources according to market prices. We model distributed scheduling as a discrete resource allocation problem, and demonstrate the applicability of economic analysis to this framework. Drawing on results from the literature, we discuss the existence of equilibrium prices for some general classes of scheduling problems, and the quality of equilibrium solutions. We then present two protocols for implementing market solutions, and analyze their computational and economic properties.

1 Introduction

Solving scheduling problems with and for distributed computing systems presents particular challenges attributable to the distributed nature of the computation. System modules may represent independent entities with conflicting and competing scheduling requirements, and may possess localized information relevant to their tasks. To recognize this independence, we treat the modules as *agents*, ascribing each of them autonomy to decide how to deploy resources under their control in service of their interests. Within this model, a distributed scheduling method can be analyzed according to how well it exhibits the following properties:

- Decisions are made by self-interested agents with local (private) information.
- The method requires minimal communication overhead.
- Agents can make effective choices without knowing the private information and strategies of other agents.
- Solutions are *efficient* in that they do not waste resources. If there is some way to make some agent(s) better off without harming others, it should be done. A solution that cannot be improved in this way is called *Pareto optimal*.

In some settings, it might be appropriate to adopt some stronger optimality criteria, based on a judgment about social value of the various agents.

Straightforward scheduling policies—such as first-come first-served, shortest-job-first, priority-first, and combinations thereof—do not generally possess these properties. For example, queue-position schemes are insensitive to relative value based on the *substance* of the task being performed. On the other hand, priority-based schemes beg the question of how to set priorities so that desirable results follow. If self-interested agents are free to set their own priorities, then without some incentive to the contrary, they will specify maximum priority for whatever they are interested in.

Citing such limitations, several have proposed that distributed resource allocation problems be solved via market mechanisms [4], an approach we have called *market-oriented programming* (MOP) [20]. In MOP, we define agent activities in terms of resources required and produced, reducing an agent's decision problem to evaluating the tradeoffs of acquiring different resources. These tradeoffs are represented in terms of market prices, which define a common scale of value across the various resources. The problem for designers of computational markets is to specify the configuration of resources traded (formally designated *goods* in the market), and the mechanism by which agent interactions determine prices.

Markets can provide several advantages for distributed scheduling:

- Markets are naturally distributed. Agents make their own decisions about how to bid based on the prices and their own relative valuations of the goods.
- Communication is limited to the exchange of *bids* and *prices* between agents and the market mechanism. In particular settings, it can be shown that price systems minimize the dimensionality of messages required to determine efficient allocations [7].
- In some well-characterized situations we can achieve Pareto and system optima (or come within some tolerance of optimal).
- Since agents must back their representations with exchange offers, there is substantial disincentive to claiming that a task is more important than it is. As we see below, in some circumstances we can ensure that truthfully reporting private information is an optimal strategy.

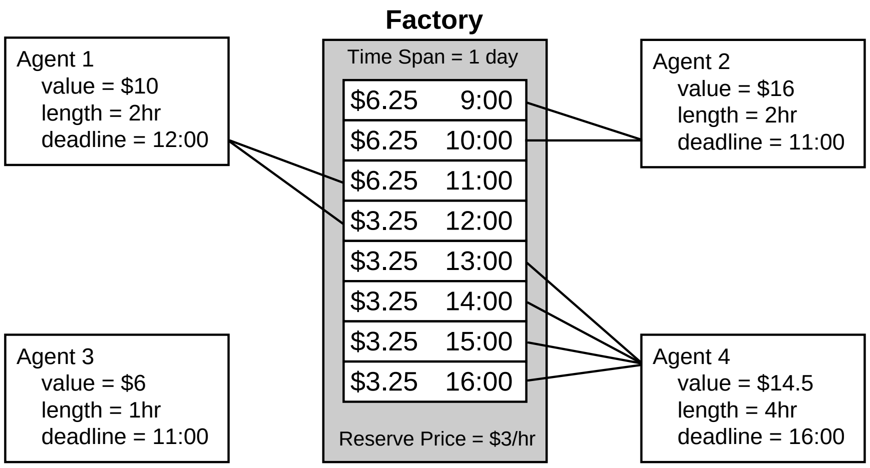
Of course, all of these benefits do not automatically accrue as a result of setting up a market-like environment. Prior work applying market-inspired mechanisms to scheduling [2, 18, 19] and other distributed computing problems [15] has produced promising empirical results. Understanding the scope of these methods, and developing a general design methodology for computational markets, however, requires an analytical characterization of their properties. In our own MOP work, we have adopted the framework of general equilibrium theory [9], and have found that our computational markets behave predictably when conditions of the theory are met [11, 20]. We have also applied the approach to discrete optimization problems—where the conditions guaranteeing desirable outcomes are not satisfied—and have found (not surprisingly) that the methods sometimes work, and other times break down [21].

Since scheduling problems very often involve discrete (indivisible) resource units, we have undertaken to analyze directly the behavior of computational market mechanisms for such problems. We start by defining a general class of discrete allocation problems, and characterizing some distinctions particularly meaningful in the scheduling domain. Fortunately, some recent results in economic theory bear directly on these problem classes. We report and discuss some of these results in the sequel, along with our own extensions and analysis.

In the next section, we motivate the work with a concrete example of a simple factory scheduling problem. In Section 3, we provide a formal economic model of a general version of the problem, and in Section 4 we relate some equilibrium and optimality properties associated with the problem. In Section 5, we describe and analyze two market protocols for solving the problem. Finally, we consider future work in Section 6.

2 A Factory Scheduling Economy

Consider a factory with an unscheduled day shift. There are eight one-hour time slots, labeled 9:00 to 16:00 according to their respective end times. Slots can be allocated for the production of



Factory
Time Span = 1 day
$6.25 9:00
$6.25 10:00
$6.25 11:00
$3.25 12:00
$3.25 13:00
$3.25 14:00
$3.25 15:00
$3.25 16:00
Reserve Price = $3/hr
Agent 1 value = $10 length = 2hr deadline = 12:00
Agent 2 value = $16 length = 2hr deadline = 11:00
Agent 3 value = $6 length = 1hr deadline = 11:00
Agent 4 value = $14.5 length = 4hr deadline = 16:00

Figure 1: A factory scheduling economy. Lines connecting the agents to time slots represent one feasible allocation.

customer orders. The factory has a *reserve price* for each time slot, representing the minimum price that the factory is willing to accept in exchange for that time slot.

Each customer agent has one job it wants completed. An agent's job is defined by its duration (length), its deadline, and the value (expressed in dollars) the agent places on the job. An agent is willing to spend up to this value to complete its job. To do so, the agent must acquire a number of slots no less than the length (not necessarily contiguous), no later than the deadline. The agent gets no value if its job cannot be completed before its deadline. The value of a solution is the sum of values of the agents holding the goods: the reserve price for each time slot that was not sold, plus the value associated with each customer agent that meets its job deadline.

Example 1 *The agents are shown in Figure 1.*[1] *Since the sum of lengths exceeds available factory time, it is not possible for all of the agents to produce their orders. The allocation depicted in Figure 1 represents a system optimum.*

Given an assignment of prices to goods, we can define an agent's optimal choice as a set of slots that complete the job at the minimum cost, or the empty set if the job costs more than its value. The reader can verify that at the prices shown in Figure 1, each agent makes a locally optimal choice in the globally optimal allocation.

3 Formal Model of the Scheduling Economy

A general discrete resource allocation problem consists of:

- G, a set of n discrete goods,
- A, a set of m agents, and $\perp$ representing the seller or null agent,
- prices $p = \langle p_1, \ldots, p_n \rangle$.

Agent j gets value $v_j(X)$ (measured in price units) if it acquires the set of goods X, $X \subseteq G$.

[1]An interactive online demonstration of this the ascending auction (Section 5.1) applied to this example can be found at `http://auction.eecs.umich.edu/FactoryDemoDocs/factory-demo.html`.

Let $H_j(p)$ denote the maximum surplus value achievable by agent j at prices p. That is,

$$H_j(p) \equiv \max_{X \subseteq G} \left[v_j(X) - \sum_{i \in X} p_i \right].$$

A *solution* is a mapping $f : G \rightarrow A \cup \{\perp\}$, indicating which agent, if any, gets each good. Let $F^j \equiv \{i | f(i) = j\}$ denote the set of goods allocated to agent j and $F^\perp \equiv \{i | f(i) = \perp\}$ denote the set of unallocated goods in f.

The *reserve value* of good i is q_i. Intuitively, the reserve value denotes the value to the owner, or the "system", of not allocating the good to one of the agents. Different time slots could potentially have different reserve values; for instance, it might be more expensive to produce during evening hours due to overtime pay.

The *value of a solution*, $v(f)$, is the sum of the agent values achieved and the reserve value of goods not used by agents,

$$v(f) \equiv \sum_{i \in F^\perp} q_i + \sum_{j=1}^{m} v_j(F^j).$$

We measure the value of a solution *ex post*, that is, conditional on knowing all agents' valuations.[2] A solution is *optimal* if no other solution has higher value.

In Section 5 we present market protocols for this very general resource allocation problem. However, the theoretical results and examples we present focus on particular classes of scheduling problems where each agent has one job to complete. For these problems, we associate each agent j with a job length λ_j and one or more deadlines $d_j^1, \ldots, d_j^{K_j}$. The value $v_j(X)$ of a set of goods X is determined by the earliest deadline d_j^k such that X includes at least λ_j time slots no later than d_j^i. For convenience we represent the time slots as integers and define the earliest time slot to be 1.

If $\lambda_j = 1$ for all j, we call the scheduling problem *single unit.* Problems violating this constraint are *multiple unit.* If each agent j has a single deadline ($K_j = 1$, we call the problem *fixed deadline.* If $K_j > 1$ for some j (i.e., j accrues greater value for finishing the job sooner), then we call the problem *variable deadline.*

4 Equilibrium and Optimality in the Scheduling Economy

Definition 1 *A solution f is in* equilibrium *at prices p iff*

1. *For all j such that $H_j(p) > 0$, $v_j(F^j) - \sum_{i \in F^j} p_i = H_j(p)$.*
2. *For all j such that $H_j(p) < 0$, $\sum_{i \in F^j} p_i = 0$.*
3. *For all j such that $H_j(p) = 0$, one of the first two consequents above holds.*
4. *For all $i \in F^\perp$, $p_i \leq q_i$.*
5. *For all $i \notin F^\perp$, $p_i \geq q_i$.*

Intuitively, this definition states that supply equals demand at equilibrium. Equilibria sometime exist, and are generally not unique. Consider Example 1. The solution shown, with only agent 3 receiving no goods, is in equilibrium at the set of prices suggested, with slots 9:00, 10:00, and 11:00 each having a price of \$6.25, and all other slots having a price of \$3.25. This equilibrium solution has value \$40.50, which is optimal. Indeed it had to be, as demonstrated by the following result.

[2]It is sometimes useful to measure the value of a solution *ex ante*, that is, with respect to the expectation of agent valuations.

Name	Job Length	Deadline	Value
Agent 1	2	2	\$3
Agent 2	1	2	\$2

Table 1: A Problem with no equilibrium.

Theorem 1 *For the general discrete resource allocation problem, if there exists a* p *such that* f *is in equilibrium at* p*, then* f *is an optimal solution.*

Proof. See [3]. □

This result confirms the usual consequence of competitive equilibrium: that no further gains from trade are possible and so the result is Pareto efficient. Since we assume that agent values are expressible in price units, Pareto optimality corresponds to global optimality.

Example 2 *There are two agents as described in Table 1, and the reserve price of each good is zero.*

The optimal solution, $f(1) = f(2) = 1$*, is not in equilibrium at any prices, and indeed no equilibrium exists in this case. If* p *were in equilibrium, then* $p_1 > \$2$ *and* $p_2 > \$2$*, otherwise agent 2 would demand one of the goods. But if these inequalities hold then agent 1 would not demand the two time slots it requires.*

In this example, the nonexistence of equilibrium prices is due to *complementarities* in agent preferences. Agent 1's preference for the two time slots are complementary because it values one iff if it has the other. Complementarities cannot arise in the single-unit scheduling problem.

Theorem 2 *Any optimal solution to the single-unit scheduling problem (fixed or variable deadline) is supported by a price equilibrium.*

Proof. If there is a set of prices p that supports an equilibrium then p supports an equilibrium for any optimal allocation [3, 6]. The single-unit scheduling problem always has a price equilibrium [14]. □

Together, Theorems 1 and 2 establish that a solution to the single-unit scheduling problem is optimal iff it is supported by a price equilibrium. Example 2 demonstrates that relaxing the single-unit restriction immediately leads to the possibility that an equilibrium will not exist.

In addition to the single-unit restriction of Theorem 2, we can identify a few other conditions that guarantee the existence of equilibrium. If all agents have separable preferences over goods then an equilibrium exists.[3] Separable preferences is a sufficient condition for *gross substitutability*—if the price for one good goes up, demand does not go down for any other good—which in turn guarantees the existence of equilibrium [8]. Bikhchandani and Mamer [3] present some other technical conditions for existence of equilibrium, which are not immediately expressible in scheduling terms.

5 Market Protocols

A *protocol* consists of a *mechanism*, along with agent *bidding policies*. The mechanisms we consider are generically called *auctions*. McAfee and McMillan provide the following definition [10]:

> An auction is a market institution with an explicit set of rules determining resource allocation and prices on the basis of bids from the market participants.

[3]Note that preferences are not separable in the multiple-unit scheduling problem. However, equilibrium would exist if agents had separable preferences for completing multiple single-unit jobs.

This definition includes the well known English open-outcry and first-price sealed bid auctions—commonly used to sell art and to award procurement contracts, respectively—as well as a broad range of other mechanisms, including fixed pricing, Dutch auction, Vickrey auction, commodities markets, and the ascending and Generalized Vickrey auctions described in Section 5.

In order to place greater structure on the space of mechanisms, and also to provide a common interface to agents, we define a somewhat restricted, but still very general auction protocol.

1. Agents send bids to the mechanism to indicate their willingness to exchange goods.

2. *Price quotes* may be posted by the auction to provide highly summarized information about the system-wide value of goods.

3. The auction determines an allocation and notifies the agents as to who purchases what from whom at what price.

The above sequence may be performed once or iterated a number of times.

Auctions can be differentiated by a number of parameterized values including, but not limited to: matching algorithm, price determination algorithm, intermediate price information revealed, whether bids are publicly revealed, and the timing of matching, information revelation, and other events. We have built the Michigan Internet AuctionBot [22, 1], an online auction server that operates as a research and development platform for creating and experimenting with auction-based economies. The AuctionBot provides interfaces for human and software agents to easily create and bid in auctions. Currently the AuctionBot supports a number of different types of auctions, including the ascending auction protocol described in Section 5.1.

In order to analyze an auction we must consider the agents' presumed bidding policies, which in turn are based on our model of their beliefs and preferences. With some auctions we are able to determine analytically that a particular bidding policy is the Bayes-Nash equilibrium or even the dominant strategy. With other auctions we rely on experimentation and rules of thumb based on economic principles to determine reasonable bidding policies.

The auction mechanisms we discuss are distributed in the sense that each agent calculates its own bid function, based on local information. The ascending auction can be further distributed in that allocation for each good can be computed separately.

In the following sections we present two auctions and associated bidding policies. These demonstrate the tradeoffs between solution quality under different problem restrictions, computational cost, and the degree that the mechanism can be distributed.

5.1 Ascending Auction

The ascending auction protocol is defined for the general discrete resource allocation problem. A separate auction is run for each good. Agents submit successively higher bids to an auction and the auction immediately reports a price quote to all interested agents. When the bidding stops, the auction allocates its respective good to the highest bidder at the price the agent bid, or the good is retained by the seller if there are no bids.

5.1.1 Bidding Rules for the Ascending Auction

The current *bid price* in the auction for good i, denoted β_i, is the current highest bid in the auction and is undefined if no bids have been submitted to the auction yet. If β_i is defined, the current *ask price*, denoted α_i, is $\beta_i + \epsilon$, for some fixed ϵ, otherwise it is q_i.

An agent can bid α_i for good i. Agents are not allowed to withdraw bids. An agent may replace its bid with another, but the new bid must be at the current ask price. These rules guarantee that prices do not decrease and that the bidding process will stop.

Name	Job Length	Deadline	Value
Agent 1	2	2	\$20
Agent 2	2	3	\$8
Agent 3	1	3	\$2

Table 2: A multiple-unit ascending auction example.

5.1.2 Bidding Policy for the Ascending Auction

When an agent j enters the market, it identifies the set of goods X that maximizes its utility, given the current ask prices. If it the cost is less than $v_j(X)$, the agent places a bid for each good at its ask price, otherwise it declines to participate in the market.

As other agents continue to bid, agent j may lose some of its bids. When this occurs, j bids the ask price on the set of goods that maximizes its utility, assuming that it must pay for the goods it is currently winning. Agent j continues to bid this way so long as it can maintain winning bids for a set of goods X' for a total cost of no more than $v_j(X')$, at which point it stops bidding.

This bidding strategy is reasonable and fairly simple. It is optimal for the single-unit problem because the ascending auction exhibits the *no regret* property with this restriction [3]. That is, bidders do not wish to change their bids after observing other agents bids during the auction. However, we do not claim that this strategy is optimal for the multiple-unit scheduling problem. But regardless of the bidding strategy, the no regret property does not hold for the ascending auction in multiple-unit case [3].

5.1.3 Analysis of the Ascending Auction

We define the price p_i of good i as the price payed for i. Thus it is β_i if it is defined, otherwise q_i.

It is possible that the ascending auction can miss an equilibrium of a multiple-unit scheduling economy by an arbitrarily large amount.

Example 3 *The bid increment is $\epsilon = \$1$ and the reserve prices are zero. The agents are described in Table 2.*

Although there are a number of equilibrium price sets (one of which is $p_1 = \$8$, $p_2 = \$8$, and $p_3 = \$1$), the ascending auction may not find an equilibrium. Agent 2 would bid up good 3 until $\alpha_3 > \$2$ while it and agent 1 both bid up the prices on 1 and 2. The reader can verify that any equilibrium must have agent 3 winning good 3 at a price no greater than \$2.

In the multiple-unit scheduling problem, the ascending auction can produce allocations that are arbitrarily far from optimal.

Example 4 *There are two agents as shown in Table 3. Reserve prices are $q_1 = \$1$ and $q_2 = \$9$, and the bid increment is $\epsilon = \$1$.*

If agent 2 places its bids first, it will bid \$1 for 1 and \$9 for 2. Agent 1 will then bid \$2 for 1. The bidding will stop with good 1 allocated to agent 1 and good 2 allocated to agent 2. This solution has a value of \$3 yet the optimal solution, with 2 unallocated, has a value of \$12. It is easy to see—by increasing q_2 and v_2 by the same value—that the ascending auction can produce a solution that is arbitrarily far from optimal.

If we restrict each agent's requirement to a single time slice, then by Theorem 2 an equilibrium exists. However, the ascending auction protocol is not guaranteed to reach an equilibrium even with this restriction. Consider the following economy.

Name	Job Length	Deadline	Value
Agent 1	1	1	\$3
Agent 2	2	2	\$11

Table 3: A suboptimal multiple-unit ascending auction example.

Name	Job Length	Deadline	Value
Agent 1	1	2	\$6
Agent 2	1	3	\$7

Table 4: A single-unit ascending auction example.

Example 5 *The bid increment is $\epsilon = \$1$. The reserve prices are $q_1 = \$4$, $q_2 = \$3$, and $q_3 = \$3$. The agents are described in Table 4.*

It is possible that agent 2 may bid first for 2. Then $\alpha_2 = \$4$. Agent 1 will then bid \$4 for either 1 or 2. If it bids for 1 then the bidding will stop and agent 1 will win 1 for \$4 and agent 2 will win 2 for \$3. But since $p_2 = \$3 < p_1$, agent 1 would maximize its surplus by demanding 2. However, the bidding rules prohibit any readjustment towards an equilibrium. The auction does not allow agent 1 to withdraw its bid for 1, and hence the final allocation violates condition 1 of the definition of equilibrium.

It is not hard to see that the potential failure to reach equilibrium can be demonstrated for any positive value of ϵ, no matter how small. Nevertheless, unlike the multiple-unit problem, we can bound the distance from the equilibrium price vector by ϵ.

Theorem 3 *For the variable-deadline, single-unit scheduling problem, the final price of any good determined by ascending auction protocol will differ from the unique minimum equilibrium price by at most $\kappa\epsilon$, where $\kappa = \min(n, m)$.*

Proof. See [5]. □

Consider again Example 5. The solution shown has a value of \$16. If agent 1 had received good 2 and agent 2 had received good 3 then the value of the solution would be \$17, which is optimal. Although the ascending auction protocol does not guarantee the optimal solution even when agents are restricted to requiring a single good, the error from optimum is bounded by ϵ.

Theorem 4 *The ascending auction protocol with a given ϵ produces a solution to the variable-deadline, single-unit scheduling problem that is suboptimal by at most $m\kappa\epsilon$, where $\kappa = \min(n, m)$.*

Proof. Let f be the allocation reached by the ascending auction and let f^* be an optimal allocation. p_i is the price found for i in the ascending auction, and p_i^*be the unique minimum equilibrium price for i. Let $e_i = p_i^* - p_i$. From Theorem 3 we know that $| e_i | \leq \kappa\epsilon$.

Let F^j and F^{*j} be the set of goods allocated to agent j in f and f^*, respectively. Let U and U^*be the set of goods unallocated in f and f^* , respectively. Similarly, B and B^*represent the goods allocated to agents in f and f^* . It follows that $B \setminus B^* \equiv U^* \setminus U$, and vice versa. To get the error, we can subtract the value of the final allocation from the optimal allocation.

$$\begin{aligned} v(f^*) - v(f) &= (\sum_{i\in U^*} q_i + \sum_{j=1}^{m} v_j(F^{*j})) - (\sum_{i\in U} q_i + \sum_{j=1}^{m} v_j(F^j)) \\ &= \sum_{i\in U^*\setminus U} q_i - \sum_{i\in U\setminus U^*} q_i + \sum_{j=1}^{m} v_j(F^{*j}) - \sum_{j=1}^{m} v_j(F^j). \qquad (1) \end{aligned}$$

The agents' strategy implies that when the ascending auction stops,

$$\sum_{j=1}^{m} v_j(F^j) - \sum_{i \in B} p_i \geq \sum_{j=1}^{m} v_j(F^{*j}) - \sum_{i \in B^*} \alpha_i \geq \sum_{j=1}^{m} v_j(F^{*j}) - \sum_{i \in B^*} p_i$$

$$\begin{aligned}
\sum_{j=1}^{m} v_j(F^{*j}) - \sum_{j=1}^{m} v_j(F^j) &\leq \sum_{i \in B^*} p_i - \sum_{i \in B} p_i \\
&= \sum_{i \in B^* \setminus B} p_i - \sum_{i \in B \setminus B^*} p_i \\
&= \sum_{i \in U \setminus U^*} p_i - \sum_{i \in U^* \setminus U} p_i. \qquad (2)
\end{aligned}$$

Goods that were bought in f^* must have minimum equilibrium prices greater than or equal to their reserve prices

$$\sum_{i \in U^* \setminus U} q_i \leq \sum_{i \in U^* \setminus U} p_i^* = \sum_{i \in U^* \setminus U} (p_i + e_i). \qquad (3)$$

Goods that were unallocated in f^* must have minimum equilibrium prices less than or equal to their reserve prices

$$\sum_{i \in U \setminus U^*} q_i \geq \sum_{i \in U \setminus U^*} p_i^* = \sum_{i \in U \setminus U^*} (p_i + e_i). \qquad (4)$$

Substituting 2, 3, and 4 into 1 gives

$$\begin{aligned}
v(f^*) - v(f) &\leq \sum_{i \in U^* \setminus U} (p_i + e_i) - \sum_{i \in U \setminus U^*} (p_i + e_i) + \sum_{i \in U \setminus U^*} p_i - \sum_{i \in U^* \setminus U} p_i \\
&= \sum_{i \in U^* \setminus U} (e_i) - \sum_{i \in U \setminus U^*} (e_i).
\end{aligned}$$

The total error is maximized when $(U^* \setminus U) \cup (U \setminus U^*)$ contains all of the goods, $e_i = ke$ for all $i \in U^* \setminus U$ and $e_i = -\kappa\epsilon$ for all $i \in U \setminus U^*$. This gives an upper bound on the total error, $v(f^*) - v(f) = m\kappa\epsilon$.

This makes intuitive sense. It means that each good that was allocated in one solution and unallocated in the other can subtract at most $\kappa\epsilon$ from the total value. $\square$

The computation of the clearing and price quotes is trivial in the ascending auction. The communications costs dominate the run time, which is inversely proportional with ϵ. Hence, in choosing the value for ϵ we decide how we wish to trade off optimality with communication efficiency.

5.2 Generalized Vickrey Auction

The ascending auction performs well for single-unit allocation problems. At the end of Section 3 we noted that a single-unit problem is only one sufficient condition for a price equilibrium allocation to exist. However, even when price equilibria exist for a multiple-unit problem, the ascending auction may not find an equilibrium; see Example 4. Further, as Example 2 demonstrates, any scheduling problems have an optimal, value-maximizing allocation that is not supportable by any price equilibrium, but the ascending auction allocates based on price.

The Generalized Vickrey Auction (GVA) [16] can implement optimal allocations in a broad class of scheduling problems. The GVA works for problems with multiple goods, multiple units, requirements contingencies, and externalities (i.e., values for one agent that depend on the allocations obtained by other agents). The GVA is not a price allocation, and thus can obtain optimality in problems for which a price equilibrium does not exist.

The GVA is a direct revelation mechanism. Agents declare their requirements and valuations in their bids. The auction mechanism then returns an allocation, and a vector of positive or negative payments to be made to the agents. The payments are designed so that truthful revelation of valuations is the dominant bidding policy.

5.2.1 Bidding Policy for the GVA

Recall that v_j is agent j's actual value function. Each agent announces $\hat{v}_j$, its alleged value function. The circumflexes are used to indicate that the agent is not constrained to be truthful, i.e., it may be that $\hat{v}_j \neq v_j$.

5.2.2 Optimality Analysis of the GVA

Recall that a solution is a mapping f, and the value of a solution is given by $v(f)$. The auction mechanism:

1. Computes a solution,
$$f^* = \arg\max_f \sum_{i \in F^\perp} q_i + \sum_{j=1}^{m} \hat{v}_j(F^j). \tag{5}$$
2. Computes payments to agents,
$$V_j \equiv W_{-j}(f^*) - P_j(\hat{v}_{-j}),$$
where
$$\begin{aligned} W_{-j}(f^*) &= \sum_{s \neq j} \hat{v}_s(F^{*j}) && (6) \\ P_j(\hat{v}_{-j}) = \tilde{P}_j(\hat{v}_{-j}) &= \max_f \sum_{i \in F^\perp} q_i + \sum_{s \neq j} \hat{v}_s(F^s). && (7) \end{aligned}$$
The W_{-j} function returns the total reported value for all agents other than j at the solution f^*. The GVA is incentive-compatible for any $P_j(\hat{v}_{-j})$ that is an arbitrary function of the other agents' reported value functions, independent of agent j's reported preferences. $\tilde{P}_j$ is a specific, convenient payment.

Theorem 5 *If there exists a solution f^* to problem (5), and if $v_j(F^{*j}) + V_j \geq 0$, then agents truthfully report their value functions ($\hat{v}_j = v_j \; \forall j$), and f^* is an optimal solution.*

Proof. See [16]. □

The intuition generalizes Vickrey's original [17] result. An agent receives $v_j(F^j) + V_j = v_j(F^j) + W_{-j} - P_j$, from the value of its allocation and the payment from the auction. The auction mechanism chooses the solution f^* to maximize $\hat{v}_j(F^j) + W_{-j}$. Therefore, if the agent bids truthfully ($\hat{v}_j = v_j$), it receives the auction mechanism's maximand less a function (P_j) that is unaffected by agent j's bid. Clearly the agent can do no better than to tell the truth and receive the full maximand of the auction mechanism.

The GVA solves problems with multiple units, and problems without a price equilibrium. Example 2 above had both features:

Example 6 (From Example 2.) *If the agents truthfully report their value functions, the auction mechanism finds the optimal solution f^*: $f^*(1) = f^*(2) = 1$. It then calculates $W_{-1} = 0$ and $W_{-2} = 3$. Agent 1 receives total value $3 + [0 - P_1]$, and agent 2 receives $0 + [3 - P_2]$. No untruthful bid can increase these payoffs, so the agents bid truthfully. The condition that $v_j(F^j) + V_j \geq 0$ requires that $P_j \leq 3$ for $j \in \{1, 2\}$; otherwise, rational agents would choose not to participate in the auction. If we use $P = \tilde{P}$ then agent 1 makes a payment of \$2, agent 2 makes a payment of \$0, and the mechanism has a revenue of \$2.*

5.2.3 Limitations on the GVA

Generally, we have three desired properties for a mechanism: incentive compatibility, participation, and optimality. In our scheduling problem we can obtain all three using $\tilde{P}_j$ from (7). The payment function $V_j(\tilde{P}_j)$ transfers to agent j the net value increment to all other agents that results from j's participation in the auction. Agent j's only effect on others is that it may get time slices that others desire, so its participation always makes other always weakly worse off. Thus, V_j is negative for all j, and the auction mechanism runs a surplus.

Theorem 6 *If the GVA uses the payment function $W_{-j} - \tilde{P}_j$ then the participation constraint will be satisfied and the net monetary payments to the auction mechanism will be nonnegative.*

However, the problem statement assumes that the auction mechanism knows the reservation values q_i. If instead the q_i are the private information of seller agents, then no mechanism can obtain more than two out of three of the desired properties. Myerson and Satterthwaite [12] have proven this impossibility theorem for bilateral exchange problems, some of which are scheduling problems with seller agents.

Example 7 (Bilateral exchange.) *Suppose there are two agents, S and B, and one unit of a good, which S owns. Let S's reserve value be s and B's be b. Suppose $b > s$. The GVA would induce truthful reporting of b and s, give the good to B, require B to pay s, and pay b to S. The no-revenue-deficit constraint would be violated for the auction mechanism.*

We can always use the GVA to obtain incentive compatibility and optimality, but with an auction deficit, by setting, e.g., $P_j = 0$. Alternatively, the GVA can obtain incentive compatibility and participation by setting a sufficiently high P_j, which, however, violates optimality because there is inefficiently low participation.

5.2.4 GVA Computation

As a baseline for computational efficiency we note that Neapolitan and Naimipour [13] show that a simple centralized greedy algorithm solves the single-unit, fixed-deadline scheduling problem optimally, in time $\Theta(m \lg m)$. For the GVA, the auction mechanism must solve multiple optimization problems to process the bids. For a single-unit, fixed-deadline problem we can use the centralized algorithm. For the P_j described above, the algorithm is run once to get the optimal allocation, then again for each agent with its bid removed to determine the price it pays. The total runtime is $\Theta(m^2 \lg m)$. Thus, inducing preference revelation (and thereby obtaining full optimality) raises the auction cost by at most a factor of m; this is the computational cost of distributing the problem via the GVA.

If we remove the single-unit restriction, then any centralized algorithm that can solve the scheduling problem optimally can solve the Integer Knapsack problem. Hence the multiple-unit scheduling problem is NP-Complete.[4] By the preceding argument, distributing the multiple-unit problem via the GVA adds at most a factor of m to the computation.

6 Discussion

We have presented two market mechanisms that can compute optimal or near-optimal solutions to the single-unit distributed scheduling problem in a computationally efficient manner. The multiple-unit problem is significantly more difficult and entails a sharper tradeoff among solution quality, computational efficiency, and the degree to which the mechanism is distributed. The computation performed by the ascending auction is trivial, and can be distributed by goods. However we cannot guarantee anything about the quality of solutions produced by this mechanism for the multiple-unit problem. The GVA, by solving multiple combinatorial problems, finds the optimal solution for this problem.

We view this work as the first important step in developing a broad framework for using markets to solve distributed scheduling problems. In order to move forward we must identify broader classes of scheduling problems and develop associated mechanisms such that we can effectively predict and analyze the behavior of the economy. We do not expect to find a single mechanism that reaches equilibrium in all situations where such equilibria exist. However we wish to develop a suite of mechanisms that collectively cover a broad range of problems. That is, we want to be able to choose a mechanism for a given problem and know that it will reach or come close to equilibrium when it exists or else perform acceptably in some other respect when equilibrium does not exist.

We are exploring the theoretical aspects of market mechanisms to support our experimental work in more complex, real-time network scheduling domains. These domains require more elaborate models, including multiple-stage scheduling which is necessary when, for instance, data must pass through several different network nodes. We are in the process of joining our top-down economic approach with a bottom-up analysis of network scheduling requirements.

Acknowledgments

This work was supported by a DARPA grant from the Information Survivability program.

References

[1] The Michigan Internet AuctionBot web site: `http://auction.eecs.umich.edu`.

[2] Albert D. Baker. Metaphor or reality: A case study where agents bid with actual costs to schedule a factory. In Clearwater [4].

[3] Sushil Bikhchandani and John W. Mamer. Competitive equilibrium in an exchange economy with indivisibilities. *Journal of Economic Theory*, 74:385–413, 1997.

[4]The problem can be viewed as pseudo-polynomial because it can be solved via dynamic programming in time polynomial in the sum of all agent values.

[4] Scott Clearwater, editor. *Market-Based Control: A Paradigm for Distributed Resource Allocation.* World Scientific, 1995.

[5] Gabrielle Demange, David Gale, and Marilda Sotomayor. Multi-item auctions. *Journal of Political Economy*, 94(4):863–872, 1986.

[6] Faruk Gul and Ennio Stacchetti. Walrasian equilibrium without complementarities. Technical report, Princeton University and University of Michigan, 1997.

[7] J. S. Jordan. The competitive allocation process is informationally efficient uniquely. *Journal of Economic Theory*, 28:1–18, 1982.

[8] A. S. Kelso and V. P. Crawford. Job matching, coalition formation, and gross substitutes. *Econometrica*, 50:1483–1504, 1982.

[9] Andreu Mas-Colell, Michael D. Whinston, and Jerry R. Green. *Microeconomic Theory.* Oxford University Press, New York, 1995.

[10] R. P. McAfee and John McMillan. Auctions and bidding. *Journal of Economic Literature*, 25:699–738, 1987.

[11] Tracy Mullen and Michael P. Wellman. A simple computational market for network information services. In *First International Conference on Multiagent Systems*, pages 283–289, San Francisco, CA, 1995.

[12] Roger B. Myerson and Mark A. Satterthwaite. Efficient maechanisms for bilateral trading. *Journal of Economic Theory*, 29:265–281, 1983.

[13] R. E. Neapolitan and K. Naimipour. *Foundations of Algorithms.* D. C. Heath and Company, Lexington, MA, 1996.

[14] L. S. Shapley and M. Shubik. The assignment game I: The core. *International Journal of Game Theory*, 1(2):111–130, 1972.

[15] Michael Stonebraker, Robert Devine, Marcel Kornacker, Witold Litwin, Avi Pfeffer, Adam Sah, and Carl Staelin. An economic paradigm for query processing and data migration in Mariposa. In *Third International Conference on Parallel and Distributed Information Systems*, Las Vegas, NV, 1994.

[16] Hal R. Varian and Jeffrey K. MacKie-Mason. Generalized Vickrey auctions. Technical report, Dept. of Economics, Univ. of Michigan, June 1994.

[17] William Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *Journal of Finance*, 16:8–37, 1961.

[18] Carl A. Waldspurger, Tad Hogg, Bernardo A. Huberman, Jeffrey O. Kephart, and Scott Stornetta. Spawn: A distributed computational economy. *IEEE Transactions on Software Engineering*, February 1992.

[19] Carl A. Waldspurger and William E. Weihl. Lottery scheduling: Flexible proportional-share resource management. In *Proceedings of the First Symposium on Operating System Design and Implementatin*, November 1994.

[20] Michael P. Wellman. A market-oriented programming environment and its application to distributed multicommodity flow problems. *Journal of Artificial Intelligence Research*, 1:1–23, 1993.

[21] Michael P. Wellman. A computational market model for distributed configuration design. *AI EDAM*, 9:125–133, 1995.

[22] Peter R. Wurman, Michael P. Wellman, and William E. Walsh. The Michigan Internet Auction-Bot: A configurable auction server for human and software agents. *Submitted for publication*, 1997.