A Market Protocol for Decentralized Task Allocation*

William E. Walsh Michael P. Wellman

Artificial Intelligence Laboratory
University of Michigan
1101 Beal Avenue, Ann Arbor, MI 48109-2110 USA
{*wew, wellman*}@*umich.edu*

Abstract

We present a decentralized market protocol for allocating tasks among agents that contend for scarce resources. Agents trade tasks and resources at prices determined by an auction protocol. We specify a simple set of bidding policies that, along with the auction mechanism, exhibits desirable convergence properties. The system always reaches quiescence. If the system reaches quiescence below the consumer's reserve price for the high level task, it will be in a solution state. If the system finds a solution it will reach quiescence in a solution state. Experimental evidence supports our conjecture that the system will converge to a solution when one exists and the consumer bids sufficiently high. We describe the system's application to and implementation in an agent-based digital library.

1. Introduction

In a multiagent system (MAS), we must often address the problem of allocating resources and effort in such a way that the resulting collection of agents can accomplish a complex task. This problem is complicated if the agents contend for limited resources, which may preclude the use of simple greedy allocation strategies. Furthermore, because the agents are autonomous, we must generally assume that they have specialized knowledge about their own capabilities but limited knowledge about other individuals and the large-scale structure of the problem. Additionally, we may have cause to compute the allocation of each resource individually. Given the decentralized nature of the problem as an exogenous constraint, we seek to design principled, effective, resource allocation protocols, noting that we can do no better than if we were free to take a centralized approach.

We present a decentralized market protocol for allocating tasks among agents under conditions of resource scarcity. The protocol builds *supply chains* in a bottom-up fashion using strictly local knowledge and communication. In the market approach, agents' decisions are coordinated by the price system, and complex multilateral behaviors are implemented via relatively simple bilateral exchanges. Moreover, solution methods and analytical techniques from economics can provide useful concepts for designing and understanding market systems.

Our experience with the *market-oriented programming* approach has verified that it works predictably and effectively in several convex domains [13, 28], characterized by infinite divisibility of resources and nonincreasing returns to scale. However, many important resource allocation problems—such as task allocation—are inherently discrete, violating the standard general conditions for market effectiveness.[1] Within economics, protocols for allocation of discrete goods have been studied under the heading of auction theory. Although much of the auction literature focuses on the allocation of single items [6, 9, 24], several studies address the more challenging problem of allocating multiple items [4, 7, 8, 21], and recent experience with the United States FCC radio spectrum auctions [10, 11] has prompted further economic interest in these problems [12].

We describe the task allocation problem in Section 2. In Section 3 we present a market system for task allocation. We analyze the relationships between system quiescence and solution convergence in Sections 4. In Section 5 we describe the system's application to and implementation in an agent-based digital library. We discuss related work in Section 6 and suggest extensions and future work in Section 7.

*Extended version of a paper in *Proceedings of the Third International Conference on Multi Agent Systems (ICMAS-98)*. Paris, France. July, 1998

[1] It should not be surprising that discrete goods complicate matters, considering the relative difficulties of solving integer programming as compared to linear programming problems. Bikhchandani and Mamer use the correspondence between these constrained optimization problems to characterize the conditions under which a given discrete problem will be problematic for a market [2].

2. The Task Allocation Problem

In the *task allocation problem*, we are interested in the achievement of some task or tasks, and tasks may be performed by various agents. In order to perform a particular task, an agent may need to achieve some subtasks, which may in turn be delegated to other agents, forming a supply chain through a hierarchy of task achievement. Constraints on the task assignment arise from *resource contention*, where agents would need a common resource (e.g., a subtask achievement, or something tangible like a piece of equipment) to accomplish their own tasks.

Tasks are performed on behalf of particular agents; if two agents need a subtask then it would have to be performed twice to satisfy them both. In this way, tasks are the same as any other discrete resource. Hence, we make no distinction in our model, and use the economic term "good" to refer to any task or resource provided or needed by agents.

2.1. Problem Specification

We provide a formal description of the problem in terms of bipartite graphs. The two types of nodes represent goods and agents, respectively. A *task dependency network* is a directed, acyclic graph, (V, E), with vertices $V = G \cup A$:

$$\begin{aligned} G &= \text{the set of goods,} \\ A &= \Pi \cup S \cup \{c\}, \text{the set of agents,} \\ \Pi &= \text{the set of producers,} \\ S &= \text{the set of suppliers,} \\ c &= \text{the end consumer,} \end{aligned}$$

and a set of edges E connecting agents with goods they can make use of or can produce. There exists an edge $\langle g, a \rangle$ from $g \in G$ to $a \in A$ when agent a can make use of one unit of g, or an edge $\langle a, g \rangle$ when a can produce unit of g.

The edges can be further characterized by the type of agent involved. A *supplier* can supply a *primitive good* without requiring any input goods:

For all $s \in S$ there is one $g \in G$ such that $\langle s, g \rangle \in E$.

The *consumer* wishes to acquire some high-level good:

There is a unique $g_c \in G$ such that $\langle g_c, c \rangle \in E$.

We consider a single consumer to simplify analysis; it is straightforward to extend the analysis to multiple consumers (described in Section 7).

A *producer* can produce some *output* good conditional on acquiring some *input* goods:

For all $\pi \in \Pi$ there exists a nonempty subset G' of G and a single $g \in G - G'$ such that $\langle \pi, g \rangle \in E$ and $\langle g', \pi \rangle \in E$ for all $g' \in G'$.

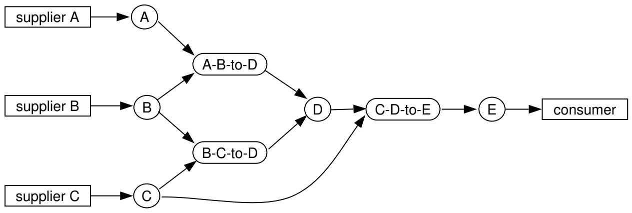

Figure 1. A task dependency network.

It is possible that a producer may require multiple units of a particular good as input. In this case we treat each unit as a separate edge. For instance, if π requires two units of t' as input, then its input edges are $\langle t', \pi \rangle_1$ and $\langle t', \pi \rangle_2$.

A producer's input requirements are *complementary* in that it must acquire each of its inputs; it cannot accomplish anything with only a partial set. The existence of different producers with the same output corresponds to different ways that a good can be produced.

Figure 1 shows an example network. Here the goods are indicated by circles, the consumer and suppliers are labeled as such in the boxes, and the producers are indicated by ovals. An arrow from an agent to a good indicates that the agent can supply that good, and an arrow from a good to an agent indicates that the agent wishes to acquire the good. For instance, producer labeled C-D-to-E requires one unit each of inputs C and D to provide one unit of E.

A *solution* is a partial ordering of production defined by a subgraph $(V', E') \subseteq (V, E)$. For $a \in A \cap V'$ and $g \in G \cap V'$, an edge $\langle a, g \rangle \in E'$ means that agent a provides g, and if $\langle g, a \rangle \in E'$, then a acquires g. To qualify as a solution, a subgraph must satisfy the following constraints.

1. The consumer acquires the good it desires:

   If $\langle g, c \rangle \in E$ then $\langle g, c \rangle \in E'$.

2. An agent is part of the solution iff it acquires or supplies a good:

   $a \in V' \cap A$ iff there exists a $t \in T$ such that $\langle t, a \rangle \in E'$ or $\langle a, t \rangle \in E'$.

3. All producers are *feasible*:

   For all $\pi \in \Pi$ such that $\langle \pi, g \rangle \in E'$, if $\langle g', \pi \rangle \in E$ then $\langle g', \pi \rangle \in E'$.

   Note that the feasibility constraint does not exclude the possibility that a producer acquires some inputs without providing its output.

4. Every good in the solution is both acquired and provided:

   If $g \in V' \cap G$ then there exist b and $s \in A \cap V'$ such that $\langle g, b \rangle \in E'$ and $\langle s, g \rangle \in E'$.

5. There is a one-to-one mapping between acquiring edges $\langle g, b \rangle \in E'$ and providing edges $\langle s, g \rangle \in E'$ for $g \in G \cup E'$.

It is often natural to associate a cost, *cost*(g), with primitive goods, that is, those provided by suppliers. In such cases, we define the cost of a solution as the cost of its primitive goods:

$$\textit{cost}((V', E')) = \sum_{\{\langle s,g \rangle \in E' | s \in S \cap V'\}} \textit{cost}(g).$$

2.2. Solving Task Allocation Problems

Our task dependency networks can be viewed as AND/OR graphs, with the consumer at the root. Agents correspond to AND nodes, and goods to ORs. One could thus solve task allocation problems in a centralized manner via AND/OR search techniques, with some extra bookkeeping to account for the fact that agents may participate in the solution in at most one way, and that the number of edges leading into a good must be the same as the number leading out in a solution (in other words, to treat these properly as graphs rather than trees). But we assume that we are constrained to solve the problem in a decentralized fashion.

Task allocation problems can be addressed in a somewhat more decentralized manner by the CONTRACT NET protocol [3]. However, because the contracting process proceeds top-down, an agent must commit to supplying a good before it is certain that it can actually acquire its input goods. That is, the protocol allocates goods in a greedy fashion, backward chaining from the consumer to the suppliers. Without lookahead, CONTRACT NET might allocate the production of good D to producer B-C-to-D in Figure 1 and thus fail to find a solution due to infeasibility.

In the remainder of this paper, we develop a decentralized protocol for solving task allocation problems, based on a market for good production.

3. The Market System

We have implemented our market-based protocol in WALRAS, a research platform for computational economies [28], and also the University of Michigan Digital Library (UMDL) experimental system. In each case, the system finds solutions in a decentralized multiagent environment where agents have only local knowledge about their own preferences or production technologies and the goods and related auctions that directly interest them.

The agents negotiate for the goods through auction mediators, one for each good. An auction in turn determines the price and allocation of its respective good as described in Section 3.1.

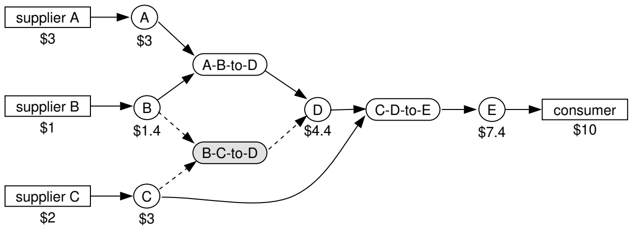

Figure 2. A valid solution to the example.

The agents are economically self-interested. The consumer is willing to buy its good of interest for no more than some fixed reserve price R_c. A supplier, s, is willing to sell its good for no less than some fixed reserve price, R_s (a natural choice might be the cost of the supplied good). Producers desire to make a profit while maintaining feasibility.

We say that a solution is *valid* if the consumer pays no more than its reserve price:

$$\text{price}(g_c) \leq R_c,$$

each supplier in the solution sells its good for at least its reserve price:

$$\text{price}(g_s) \geq R_s$$

for all $s \in S \cap V'$ such that $\langle s, g_s \rangle \in E'$,

and none of the active producers lose money:

$$\text{price}(g_\pi) \geq \sum_{g'_\pi \in I_\pi} \text{price}(g'_\pi)$$

for all $\pi \in \Pi \cap V'$ where $\langle \pi, g_\pi \rangle \in E'$ and $I_\pi = \bigcup g'_\pi$ such that $\langle g'_\pi, \pi \rangle \in E'$.

We make several further assumptions about the economy. The goods can be traded only in integer quantities, and producers sell only a single unit of their output. The latter is not a limitation to production because we can always replicate producers. We consider only economies where there are no potential cycles in production, that is no agent supplies goods that could be used to assemble its inputs. If negotiation resulted in such a production cycle, there would be no way to execute the transaction sequence.

Figure 2 shows one possible valid solution for the problem shown in Figure 1, for a specified set of reserve prices. Reserve prices are shown under their respective agents and good prices are shown under their respective goods. Dashed arrows indicate input/output relations not part of the solution. Observe that producers B-C-to-D and C-D-to-E both require good C as an input. (Perhaps they are competing for control of the same machine.) As a result, producer B-C-to-D cannot be the agent that produces good D. If it were, then B-C-to-D would have to be allocated good C, in which case C-D-to-E could not produce good E for the consumer.

The agents negotiate the terms of the trades by exchanging asynchronous *messages* with auctions. The *auction mechanism* defines the rules for determining prices and allocations as a function of the agents' messages. The agents' *bidding policies* govern their interaction with the auctions. The key distinction is that the mechanism is under the control of the system designers, while the bidding policies are determined by individual agents. Together, these elements constitute a *market protocol*.

3.1. Auction Mechanism

The task allocation market includes a simultaneous auction for each good of potential value. Each agent regularly sends new *bid* messages for some of the goods that it wishes to buy or sell (if an agent does not wish to change its bid, then it leaves its previous bid standing in the auction). A bid specifies the price below/above which the agent is willing to buy/sell. When an auction receives a new bid, it sends each of its bidders a *price quote* message specifying the price that would result if the auction ended in the current bid state. Because multiple agents may have bid the same price, the price quote also reports to each bidder the quantity it would buy or sell in the current state. Agents may then choose to revise their bids in response to the notifications.

When an auction sends a price quote to an agent, it includes the ID of the most recent bid received from that agent. An agent only responds to a price quote that reflects the most recent bid it sent. If an agent does not follow this policy, it may have difficulty establishing feasibility in an asynchronous system.[2]

Bidding continues until *quiescence*, a state where all messages have been received, no agent chooses to revise its bids, and no auction changes its allocation. At this point, the auctions *clear*, and each bidder is notified of the final prices and how many units it transacted in each good. Note that a quiescent economy may not be in a solution state, valid or invalid.

Each auction runs according to $(M+1)$st-price rules [19, 20, 31]. The $(M+1)$st price auction is the uniform price generalization of the second price Vickrey Auction [24] that allows for the sale and purchase of multiple units of a good. Given a set of bids including M units offered for sale, the $(M+1)$st-price auction sets a price equal to the $(M+1)$st highest of *all* of the bids. The price can be said to separate the winners from the losers, in that the winners include all sell bids strictly below the price and all buy bids strictly above the price. In order to maximize trade, some agents that bid at the $(M+1)$st price also win according to a tie breaking rule. Winning buy and sell bids are matched one-to-one.

Because producers' technologies are complementary, ensuring feasibility is a challenging problem. Inspired by the FCC actions, we designed our auctions to run simultaneously and reject an agent's bid if it does not increase over its previous bid (see [12] for a preliminary analysis of simultaneous ascending auctions). This design helps give producers an accurate indication of the relative prices for inputs and outputs. As we show in Section 4, the bid restriction also serves a key role in establishing the relationships between system quiescence and solution convergence of the economy.

3.2. Bidding Policies

Although MAS designers do not generally have control over the agents' behaviors, any conclusions about the outcome of a protocol must be based on some assumptions about these behaviors. Our analysis assumes that the agents follow a simple bidding policy, described in this section. Other variations may be reasonable, or perhaps better in some respects than the policies we describe. Rather than explore the range of possibilities, we chose in this work to investigate a particular set of policies in depth. Our chosen policies respect the ascending bid restriction enforced by the auction as well as agent autonomy, in that no agent utilizes private information of other agents in the system.

The suppliers' and consumer's optimization problems are simply to maximize or minimize, respectively, the difference between their reserve price and prices at which they transact. We assume that these agents do not behave strategically, and instead simply place fixed bids at their respective reserve prices. The auction rules ensure that this policy will result in a nonnegative surplus value.

A producer's optimization problem is much more complex, namely to maximize the difference between the price it receives for its output and the total price it pays for its inputs, while remaining feasible. A producer places a new bid for its output at a price equal to the sum of its *expected* input prices, if this sum exceeds its previous bid. For a given input good, we define the expected price for producer π to be the last price quote if that price quote indicated it was winning, or some small finite amount λ_π above the last price quote if that price quote indicated that it was losing.

A producer initially bids zero for each of its input goods and gradually increases its bids to ensure feasibility. A producer π will raise its bid for an input good by a small finite amount δ_π if and only if the price quotes indicate that it is losing that good but winning its output.

Note that, throughout the negotiation, the producers place bids for their output goods before they have received

[2] If the producer relies on a price quote for its output that does not reflect its most recent bid, it may incorrectly think it is winning its output. Based on the policy described in Section 3.2 it may then increase its bids on its inputs. This can result in situations where producers continue to incorrectly raise their bids and never establish feasibility. This problem does not occur with the ID reporting.

commitments on their input goods. Producers counteract potential risk by continually updating their bids based on price changes and their feasibility status.

4. System Quiescence and Solution Convergence

In this analysis, we assume that all messages are delivered in a finite period of time, that is, messages are never lost.

We define the *level* of a producer with output ω as follows: one if no producer has ω as input, and $k+1$ if the maximum level of any producer with input ω is k.

A given run of the market protocol has the following parameters:

- ϕ = the maximum level of any producer in the graph,
- Υ = the maximum number of input goods for any producer,
- $\Delta = \max_{\pi \in \Pi} \delta_\pi$,
- $\Lambda = \max_{\pi \in \Pi} \lambda_\pi$.

Lemma 1 *No agent places a buy bid above $R_c + 2\phi\Delta$.*

Proof. Clearly this holds for the consumer. We prove by induction on the producer level that no producer at level k places a buy bid above $R_c + 2k\Delta$. Because $k \leq \phi$ for all producers, the lemma follows immediately.

Only the consumer may wish to acquire the output of a producer π at level one. Thus π can only win its output bid if the expected price of its inputs is no greater than R_c. Assume that π will raise its bid for input ξ from β to β', where $R_c + \Delta < \beta' \leq R_c + 2\Delta$. Because $\delta_\pi \leq \Delta$, π must bid β' before bidding above $R_c + 2\Delta$. Similarly, it must be that $\beta > R_c$. It must also be that π is losing its bid for ξ, otherwise it would not raise the bid. But then the current price quote of ξ is greater than R_c. Thus, π will bid greater than R_c for its output. Because bids are nondecreasing, it will never again win its output bid, and hence will never again raise an input bid. Thus a level one producer will never place a buy bid above $R_c + 2\Delta$.

Now assume that no producer at any level i, where $i < k$, places a buy bid above $R_c + 2i\Delta$, to prove that no producer at level k places a buy bid above $R_c + 2k\Delta$. By the inductive assumption, no producer at level k can win its output bid for more than $R_c + 2(k-1)\Delta$. It is straightforward to apply the reasoning for level one to prove the inductive case. □

Lemma 2 *The price of no good exceeds $R_c + 2\phi\Delta$.*

Proof. Assume, contrary to which we wish to prove, that there is a good with price $p > R_c + 2\phi\Delta$. According to the auction protocol, there must be $M+1$ bids at or above p. By Lemma 1 these must all be sell bids. But, by definition, there are only M sell bids, which is a contradiction. □

Lemma 3 *No producer places a sell bid above $\Upsilon(R_c + 2\phi\Delta + \lambda)$.*

Proof. By Lemma 2, the expected price for any input to any producer does not exceed $R_c + 2\phi\Delta + \lambda$. No producer has more than Υ inputs, and thus by the producer bidding strategy, no producer places a sell bid above $\Upsilon(R_c + 2\phi\Delta + \lambda)$. □

Theorem 4 *The market protocol reaches quiescence within a finite period of time.*

Proof. According to Lemmas 1 and 3, there is a bound on each producer's bids. If the system is not in quiescence, at least one producer π raises at least one bid by at least $\min(\lambda_\pi, \delta_\pi)$. This can happen at most a finite number of times before a producer exceeds the bound on its bids, therefore the system must reach quiescence. □

If all sell bids for g_c, the good desired by the consumer, rise above the consumer's reserve price, then the economy will necessarily reach quiescence in a non-solution state. If, however, quiescence is reached before the price reaches the consumer's reserve price, we have a valid solution.

Theorem 5 *If the market reaches quiescence with price$(g_c) < R_c$, then the system's state represents a valid solution.*

Proof. Each producer must be feasible, otherwise it would change some its input bids and the economy would not have reached quiescence. For any active producer, the price of its output good must be no less than the total price of its input goods, otherwise it would increase its output bid and the economy would not have reached quiescence.

Because the price of g_c is less than the consumer's reserve price it must have won its bid for g_c. The auction guarantees that suppliers receive at least their reserve price if they win. Finally, the auction guarantees that there is a one-to-one mapping between successful buy bids and successful sell bids for any good.

Each of the constraints for a valid solution are satisfied in the given quiescence conditions. □

Lemma 6 *If the economy is in a valid solution state, then the subsequent behavior of the agents obeys the following properties:*

1. *No agent changes any buy bids.*

2. *No agent reduces any sell bids.*

3. *Each agent that won a sell bid does not bid above the price quote for that good.*

Proof. Recall that the consumer always bids its reserve price. Because we have a valid solution, each producer is feasible and thus will not raise any of its buy bids for inputs. Hence property 1 is satisfied.

Property 2 is satisfied because agents never decrease their bids.

A producer that won a sell bid must not have bid above the current quote for that good. By the definition of a valid solution, if a producer is active then the current price of its output good is no less than the total current price of its inputs. Thus an active producer will bid no higher than the current price of its output good. This, combined with the fact that suppliers do not change their bids, satisfies property 3. □

We say that tie breaking is *consistent* if, in any consecutive clearings with the same set of bids, the auction breaks ties in the same way.

Theorem 7 *If the economy is in a valid solution state, all auctions break ties consistently, and no sell bids are currently lost due to tie breaking, then after the subsequent price quote from each auction, the economy will be in a quiescent state with a valid solution.*

Proof. We refer to the three properties enumerated in Lemma 6.

Recall that the current price (along with tie breaking) separates the winning bids from the losing bids submitted prior. From property 1 we know that the current price and the consistent tie breaking will also separate the same winning buyers from the same losing buyers in the next price quote. Properties 2 and 3, and the fact that no sell bids are currently lost to tie breaking, ensure that the current price will also separate the same winning sellers from the same losing sellers in the next price quote. It follows that the allocations will not change after the next price quote. Also, because the current price will separate the winners from losers in the next price quote, the price will not change in the next price quote.

Since the prices and allocations do not change in the next set of price quotes, no agent will further change its bids, and the economy will be quiescent. Furthermore, because the economy is in a valid solution state based on the current price quotes, it must be in a valid solution state based on the next price quotes. □

We note here that consistent tie breaking itself is not sufficient to ensure that the allocation to sellers does not change. The properties established by Lemma 6 do not exclude a producer from increasing its next sell bid up to the

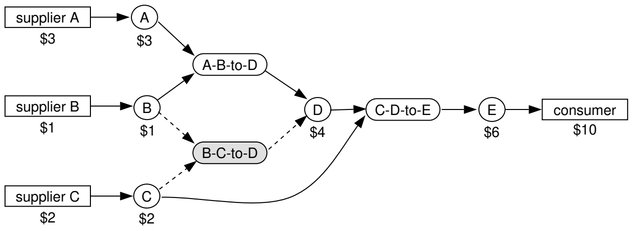

Figure 3. The minimum-price valid solution.

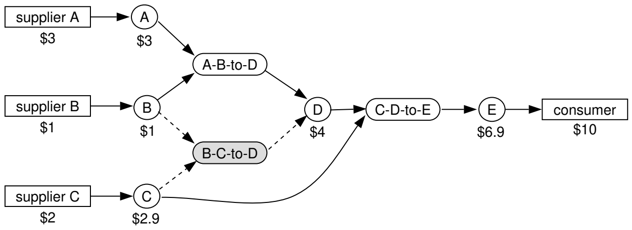

Figure 4. The minimum-price quiescent valid solution for $\lambda = \delta = 0.1$.

price quote. If this occurs, then, regardless of consistency, tie breaking of sell bids at the $(M + 1)$st price may be different in the next price quote.

Let Ψ to denote the minimum price quiescent valid solution in the economy, with associated price *price*(Ψ) for g_c. To reach a valid solution configuration, the consumer's reserve price must be no less than *price*(Ψ). Note that *price*(Ψ) will generally be greater than the minimum price of any valid solution. Consider the example shown in Figure 2. The minimum price for good E is \$6, as shown in Figure 3. However, if producer B-C-to-D has a λ less than \$0.5, then it will be willing to sell good D for less than \$4, and the prices and allocations shown cannot constitute a quiescent solution (it follows that our protocol would never generate the allocation shown in Figure 3). Because C-D-to-E is the only producer that sells E, the good C must be allocated to C-D-to-E rather than to B-C-to-D. Hence, in order for a valid solution to be quiescent, the price of C must be high enough such that B-C-to-D cannot win a bid to sell D. If the λ and δ for all producers is .1 then Ψ (with the constraint of no tie breaking) is as shown in Figure 4.

Our experiments suggest that if the consumer's reserve price is sufficiently higher than *price*(Ψ), then the economy will reach a quiescent valid solution. How much higher the reserve price must be depends on the λ and δ values as well as the structure of the economy. For instance, in our running example the system actually computes the prices shown in Figure 2, rather than those in Ψ. Good B is bid up higher than \$1 because at some times either A-B-to-D or B-C-to-D won its output bid yet did not win its bid for B.

Conjecture 8 *For any economy for which a valid solution exists, there exists a minimum reserve price such that if the consumer bids above this price then the economy will reach a quiescent valid solution with the specified auctions and bidding policies.*

Note that our conjecture does not claim that solutions found are those with minimum cost. Indeed, valid but suboptimal solutions appear in our experiments. Of course, the cost of any solution found is bounded by the consumer's reserve price.

We do not yet have a proof of this conjecture. However our intuition of its correctness is supported by a battery of randomly generated experiments. Each economy had a number of goods, suppliers, and two-input producers, each selected independently from the uniform distribution $[3, 53]$. Each supplier's price was uniformly distributed on $[0, 10]$, $\lambda = \delta = 0.1$ for all producers, and the consumer's reserve price was set sufficiently high. Over 3000 economies converged to a quiescent valid solution. Only economies with no solution failed.

That identifying convergent protocols is nontrivial is supported by our experience with several variants that fail to reliably terminate in a solution state. For example, the ascending price restriction appears pivotal in achieving the desired behavior. As a further test, we ran a set of task allocation exercises involving 25 human agents playing the role of producers, with the same price quote information available to our software agents. The humans failed to reach solutions in any of the ten problems (five different, run twice each), despite the fact that there were a multiplicity of solutions, found easily by agents pursuing the simple policy presented here.[3]

5. Task Allocation in an Agent-Based Digital Library

In an agent-based digital library, teams of specialized agents work together to provide users with information content and services. In the University of Michigan Digital Library (UMDL) [5, 29], mediators help user interface agents (UIAs) find and process information provided by collection interface agents (CIAs). An auction is a particular type of mediator that provides mediated negotiation service.

In our model of task allocation in the UMDL, the salient tasks are searching for information and revising queries to improve the search. A task planning agent (TPA) provides search coordination service to a UIA, and delegates the subtasks of revising queries to a terminology broadening/narrowing agent and a thesaurus agent. In our scenario, each agent must reserve high priority computation on one of two servers to avoid degradation of service; this good is the scarce resource that must be correctly allocated.

In the UMDL, agents and auctions run as separate processes with no central controller. Because the system is asynchronous and fully decentralized, it is a non-trivial task to detect system quiescence.

Each UMDL auction clears after a specified interval of inactivity. We prefer short inactivity periods, to minimize overall latency. But we have observed that bidding activity may not be uniform across all auctions. Thus if the inactivity period is too short, there is a danger that one auction may clear before quiescence is reached. In situations where a greater degree of centralization is allowed, we could run groups of auctions in a single process or allow them to communicate directly to determine when a global quiescent state is reached. This would be appropriate in situations where human designers can identify groups of auctions that should reach quiescence in concert. It is an interesting question whether one can identify appropriate clusters automatically based on local analysis of the task dependency network. For situations where this approach is not viable, we are also developing a more decentralized protocol — involving only communication between agents and their auctions — to detect when quiescence is reached.

6. Related Work

Rosenschein and Zlotkin [16] define a class of task allocation problems called *task-oriented domains* (TODs). In a TOD, any agent can potentially perform any subset of tasks, at designated costs. They analyze properties of protocols by which agents repartition the tasks to decrease their own costs. Sandholm [18] generalizes the model to include agent-dependent costs, and describes the implications of various restrictions on exchanges for achieving optimal allocations.

As noted in Section 2.2, the CONTRACT NET protocol attempts to allocate tasks by top-down hierarchical refinement. Sandholm [17] describes a variant of CONTRACT NET, in which tasks can be clustered to allow individual agents to bid for complementary inputs as a bundle. Bundling of interdependent resources is an important feature, the subject of much recent research in auction design [1, 15, 23, 30].

Auctions have been applied to various other discrete resource allocation problems in the distributed computing literature. The SPAWN system uses auctions to dynamically allocate underutilized processors [25]. Our analysis of market-based decentralized scheduling addresses equilibrium properties of and candidate market protocols for

[3]The exercise is described at `http://ai.eecs.umich.edu/people/wellman/courses/eecs498/w97/PS3.html`. We do not draw any serious conclusions from such an uncontrolled experiment, but report the outcome as suggestive of the subtlety of the underlying problem.

such problems [26].

Finally, some other recent work addresses the issue of convergence in market-based negotiation protocols. For example, Sierra et al. [22] specify a variety of bilateral negotiation policies, and present theoretical and empirical evidence bearing on their convergence and performance.

7. Extensions and Future Work

We have presented a decentralized market-based protocol for allocating tasks and scarce resources among agents. Analytical and experimental studies have indicated convergent behavior. As described in Section 5, we are exploring methods to detect quiescence in a decentralized fashion. We must also address the problem of executing the transactions. A transaction protocol would potentially include mechanisms for breaking local contracts when a global solution does not form.

Although we focus our attention on a single consumer, the economy also works if multiple consumers, perhaps with different reservation prices, wish to purchase the same good. If we substitute the value of the highest consumer reserve price for R_c in the definition of a valid solution and in each lemma and theorem, then all results continue to hold.

In a practical multiagent system, auctions may not exist for all goods of interest, and agents may not know *a priori* how to contact those auctions that do exist. Work is being performed by others in the UMDL to design mechanisms and policies for starting and maintaining auctions [14], and to create a *goods description language* and agents to interpret it to help bidders find the auctions they need [27].

We are using the current protocol as a basis for studying more general resource allocation problems. We are exploring the possibility of combining our task allocation protocol with our market-based scheduling model [26] to solve the problem of constructing supply chains with time dependencies.

Acknowledgments

We are grateful to Edmund Durfee for his detailed comments on an early draft of this paper, and also for his enlightening discussions on the topic. The bidding policy is based on a design by Peter Wurman, who also helped us clarify our thoughts on a number of occasions. Thanks to Terence Kelly, Jeffrey MacKie-Mason, and the anonymous reviewers for their insights, comments, and suggestions. We also thank Jonathan Mayer, on whose idea this work was initially pursued.

This work was supported by an NSF/DARPA/NASA Digital Library Initiative grant.

References

[1] J. S. Banks, J. O. Ledyard, and D. P. Porter. Allocating uncertain and unresponsive resources: an experimental approach. *RAND Journal of Economics*, 20(1):1–25, 1989.

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

[3] R. Davis and R. G. Smith. Negotiation as a metaphor for distributed problem solving. *Artificial Intelligence*, 20:63–109, 1983.

[4] G. Demange, D. Gale, and M. Sotomayor. Multi-item auctions. *Journal of Political Economy*, 94(4):863–872, 1986.

[5] E. H. Durfee, D. L. Kiskis, and W. P. Birmingham. The agent architecture of the University of Michigan Digital Library. *IEEE Proceedings – Software Engineering*, 144(1):61–71, 1997.

[6] D. Friedman and J. Rust, editors. *The Double Auction Market: Institutions, Theories, and Evidence*. Addison-Wesley, 1993.

[7] D. Grether. *The Allocation of Scarce Resources. Experimental Economics and the Problem of Allocating Airport Slots*. Westview Press, 1989.

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

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

[10] R. P. McAfee and J. McMillan. Analyzing the airwaves auction. *Journal of Economic Perspectives*, 10(1):159–175, Winter 1996.

[11] J. McMillan. Selling spectrum rights. *Journal of Economic Perspectives*, 8(3):145–162, Summer 1994.

[12] P. Milgrom. Auction theory in practice: The simultaneous ascending auction. Technical Report TR 98-0002, Stanford University, Dec. 1997. Department of Economics.

[13] T. Mullen and M. P. Wellman. Market-based negotiation for digital library services. In *Second USENIX Workshop on Electronic Commerce*, pages 259–269, Oakland, CA, 1996.

[14] T. Mullen and M. P. Wellman. The auction manager: Market middleware for large-scale electronic commerce. In *Third USENIX Workshop on Electronic Commerce*, Boston, 1998.

[15] S. J. Rassenti, V. L. Smith, and R. L. Bulfin. A combinatorial auction mechanism for airport time slot allocation. *Bell Journal of Economics*, 13:402–417, 1982.

[16] J. S. Rosenschein and G. Zlotkin. *Rules of Encounter*. MIT Press, 1994.

[17] T. W. Sandholm. An implementation of the CONTRACT NET protocol based on marginal cost calculations. In *Procedings from the Eleventh National Conference on Artificial Intelligence*, pages 256–262, 1993.

[18] T. W. Sandholm. Contract types for satisficing task allocation I: Theoretical results. Technical report, Washington University, 1997. Department of Computer Science.

[19] M. A. Satterthwaite and S. R. Williams. Bilateral trade with the sealed bid k-double auction: Existence and efficiency. *Journal of Economic Theory*, 48:107–133, 1989.

[20] M. A. Satterthwaite and S. R. Williams. *The Bayesian theory of the k-double auction*, chapter 4, pages 99–123. In Friedman and Rust [6], 1993.

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

[22] C. Sierra, P. Faratin, and N. R. Jennings. A service-oriented negotiation model between autonomous agents. *Robotics and Autonomous Systems*, to appear.

[23] H. R. Varian and J. K. MacKie-Mason. Generalized Vickrey auctions. Technical report, Dept. of Economics, Univ. of Michigan, June 1994.

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

[25] C. A. Waldspurger, T. Hogg, B. A. Huberman, J. O. Kephart, and S. Stornetta. Spawn: A distributed computational economy. *IEEE Transactions on Software Engineering*, Feb. 1992.

[26] W. E. Walsh, M. P. Wellman, P. R. Wurman, and J. K. MacKie-Mason. Some economics of market-based distributed scheduling. In *Eighteenth International Conference on Distributed Computing Systems*, 1998.

[27] P. Weinstein and W. P. Birmingham. Service classification in a proto-organic society of agents. In *Proceedings of the IJCAI-97 Workshop on Artificial Intelligence in Digital Libraries*, 1997.

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

[29] M. P. Wellman, W. P. Birmingham, and E. H. Durfee. The digital library as a community of information agents. *IEEE Expert*, 11(3):10–11, 1996.

[30] P. R. Wurman. Multidimensional auction design for computational economies, Nov. 1997. Dissertation proposal, department of Electrical Engineering and Computer Science, University of Michigan.

[31] P. R. Wurman, W. E. Walsh, and M. P. Wellman. Flexible double auctions for online commerce: Theory and implementation. *Decision Support Systems*, to appear.