Representing Market Design Choices in Market-Oriented Programming

Christopher J. Hazard
cjhazard@ncsu.edu
North Carolina State University
Raleigh, NC 27695

Peter R. Wurman
wurman@ncsu.edu
North Carolina State University
Raleigh, NC 27695

Abstract

Market-Oriented Programming (MOP) is a multi-agent systems paradigm in which resources are managed through price-based communications. Based on the subdivisibility and complexity of tasks, a given problem may have many possible MOP designs. The structure of MOP designs can be difficult to communicate. We present a method to represent and characterize designs of market systems and discuss design trade-offs with respect to combinatorial optimization. We demonstrate our MOP design representation using two systems, the first being a warehouse system involving item and vehicle routing with combinatorial orders, and the second being the Trading Agent Competition Supply Chain Management (TAC SCM) game.

Introduction

The Market-Oriented Programming (MOP) approach (Wellman 1996) provides a means to subdivide problems into separate auctions. In each auction, agents interact by communicating prices in order to negotiate the allocation and trade of resources. The MOP paradigm allows a system designer to modularly apply domain knowledge and heuristics to agents. Further, MOP scales effectively by distributing the work among the agents and markets.

Systems which contain cascading dependencies of specialized activities are often designed such that a particular good or service might only be exchanged in certain locations or between certain types of agents. Supply chain management (Walsh & Wellman 2003; Fan, Stallaert, & Whinston 2003) and robot coordination (Berhault *et al.* 2003; Lagoudakis *et al.* 2005) are two such applications of MOP. Decentralized scheduling is another application of MOP (Wellman *et al.* 2001), and a given scheduling problem may be well suited for more complex MOP designs if constraints or valuations strongly segment the market.

MOP design involves splitting a system into agents, auctions, and commodities, and deciding which commodities are traded in each auction and under what rules. Segmenting a system into multiple auctions can reduce the domain on which agents are able to express combinatorial valuations, which may reduce the range of possible global outcomes. Following the model presented by Bikhchandani and Ostroy (2002), we will use the term non-linear pricing to refer to an auction policy that allows combinatorial offers, and the term non-anonymous pricing to refer to a policy that allows discriminatory prices in which buyers (or sellers) may see different prices for the same combination of objects.

Tâtonnement, that is, iterated adjustment of supply and demand until the market converges using linear prices, can reach optimal allocations even with a less expressive, linear bidding scheme (Cheng & Wellman 1998). However, tâtonnement may not converge unless aggregate demand is nondecreasing for each good with respect to all other goods.

An alternative and supplement to tâtonnement is to allow non-linear pricing. While non-linear pricing can find globally optimal solutions in certain cases where tâtonnement cannot, winner determination in combinatorial exchanges can be less tractable.

In a supply chain, an end consumer might bid on items from distributors or producers of finished goods. However, the consumer will often not differentiate between different suppliers of raw materials when bidding, because the consumer deems those attributes of the goods substitutable. For example, the particular supplier of wheat from which bread is made will typically not affect a consumer's decision on which sandwich to buy (assuming the sandwiches in consideration have bread that lie within the customer's tolerance of wheat quality). By restricting the consumers to bidding only on finished products from suppliers, the computational complexity of the resource allocations is greatly reduced.

In addition to substitutability, the expected time before a buyer receives a good may affect the buyer's bids. In terms of global system throughput, minimizing the average delivery time of goods is paramount. Further, suppose that each consumer's value of a good can potentially change non-monotonically with respect to time. In this case, consumers could benefit from placing non-linear bids on all of the possible paths of goods through the supply chain in order to achieve the globally optimal solution.

A market designer must make trade-offs between expressivity and tractability based on domain knowledge or heuristics. In systems involving many different agent and item types, the MOP design space can become exceedingly large. In order to determine best MOP practices for a given domain, it is important for MOP practitioners to be able to communicate their designs.

To the best of our knowledge, no expressive means of diagramming market designs has been demonstrated, despite

the number of MOP implementations. Depictions of market systems often represent only the transfer of goods within the system. From the perspective of market details, Neumann and Holtmann (2004) present a blueprint scheme for a single market, but the UML-based blueprints are too detailed to meaningfully scale up to a large market system. Similarly, Fox et al. (2000) represent detailed action graphs, but their graphs do not denote the flow of goods nor details of bids. Babaioff and Walsh (2005) include bids along with the direction of exchange, but do not indicate any properties of the individual markets.

In this paper, we present both a diagramming method and a symbolic representation that express both linearity and anonymity of pricing in market systems, in addition to the flow of goods. We apply this representation to two example systems and discuss design implications.

This paper is organized as follows. First, we formally present the notation for our diagrams and symbolic representation. Next, we present an example of routing in a warehouse system that demonstrates many possibilities of market design. We introduce our diagrams and symbolic methods, and discuss how different market designs can impact performance and utility function design. Next, we present a model of the Trading Agent Competition Supply Chain Management (TAC SCM) game. Finally, we draw some conclusions.

Market Representation

Bid/Ask Offer Notation

A key feature of market systems is the manner in which goods and services are transferred. We use the term *item type* to refer to a type of good or service. Each item type has a set of attributes that allow it to play a particular role in the system. We use the term *item* to denote a particular instance of an item type. An item may have attributes in addition to those inherited from the item type. These additional attributes may be a property of the item itself, such as color, or a property of the agent that owns the item, such as location.

In MOP, agents interact by sending bid (buy) and ask (sell) offers to the market. Each offer represents an agent's willingness to pay for an item or item combination. Interfaces between agents are defined by the set of item types for which agents are permitted to make ask and bid offers. We denote a simple bid for an agent of type B to buy an item of type I from an agent of type A as

$$B : bid\{A, I\}. \tag{1}$$

The braces enclose a tuple of a seller type and the corresponding item type. Similarly, we denote a simple ask offer for an agent of type A to sell an item of type I to an agent of type B as

$$A : ask\{B, I\}. \tag{2}$$

All notation defined in this section may be used symmetrically for both ask and bid offers. For the sake of brevity, we omit the ask representations.

Depending on the system, items of the same type bought from different agents may or may not be substitutable. If the item type's substitutability is dependent on the offering agent, then agents making offers for that good will have non-anonymous pricing. A buyer's valuation may account for attributes of the item which are inherited from the seller, such as the distance between the buyer and seller. For example, the buyer may not be willing to pay as much for an item that costs extra time and energy to retrieve. If agent-item pairs are non-anonymous, then other agents must place separate bids for each seller for a given item type. We represent non-anonymous bids with a postfix plus. An agent of type B offering to buy item type I from agents of type A with non-anonymous pricing would express its bid as

$$B : bid\{A+, I\}. \tag{3}$$

Non-linear pricing is useful in MOP, as it allows agents to explicitly express complementary valuations when bidding on multiple items. An agent may value having a hammer and a box of nails together much higher than having only a hammer or a box of nails. An agent may obtain these multiple items in the same market from one type of agent or a set of different agent types. Different agent types may also be selling the same item type either in the same auction or in different auctions. We denote agent type B offering a non-linear bid on item types I and J from agents of type A and C respectively as

$$B : bid\{A, I\}, bid\{C, J\}. \tag{4}$$

An agent may also submit non-linear bids of a given item type to the same seller. For example, an agent may bid on sets of shapes with which to build scaffolding. An agent of type B may bid for a combination of items of type I from agents of type A as

$$B : bid\{A, I*\}. \tag{5}$$

We may further combine the notation for non-anonymous and non-linear bids. Suppose an agent of type B bids on an item of type I from agents of type A in and on a combination of items of type J from agents of type C in the same bid. Suppose further that I is substitutable across agents of type A, but J is not substitutable across agents of type C. This example may be written as

$$B : bid\{A, I\}, bid\{C+, J*\}. \tag{6}$$

Although the notation is symmetric for ask and bid offers, this does not mean that a particular pair of ask and bid offers is symmetric. For example, one buyer type may bid non-linearly and non-anonymously on a seller's items, while the seller may use linear and anonymous pricing. For these reasons, a system must be represented by all of the possible bid and ask offers for each agent type.

The notation we present is useful for several reasons. First, the notation defines the contents of the bids of all agents in the system. Second, using inference, bids and asks may be chained together to determine the paths of items within the system. Finally, this notation may be converted to a diagram of the market system.

Diagrammatic Representation

As markets are composed of interacting agents and tradeable items, a useful diagrammatic representation must represent

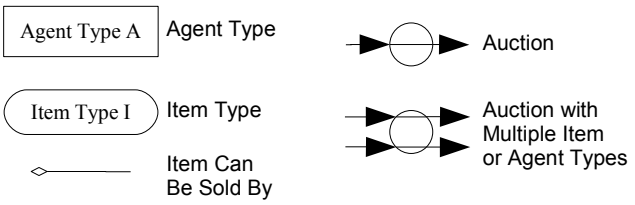

Figure 1: Diagrammatic representation of market features.

arrow	anonymous price?	linear price?	representation
	no	yes	{A+, I}
	no	no	{A+, I*}
	yes	yes	{A, I}
	yes	no	{A, I*}

Figure 2: Arrow types and corresponding representation.

their relationships. Figure 1 shows the diagrammatic representations that support the symbolic market description. Agent types are represented by boxes and item types are represented by rounded boxes. An agent type is denoted to have the ability to sell an item by a line between the agent type and the item type with a diamond on the side of the agent type. Items are transferred from one agent to another through auctions, with each auction represented by a circle. The lines entering an auction come from sellers' items and the lines exiting a circle go to buyers. If an agent can propose a combinatorial bid of item types or items from different agent types, then multiple lines go through the same auction circle.

Instead of enumerating each agent and item in a system, we represent agent and item types and do not explicitly address cardinality. Cardinality relationships may be easily added to the diagrams. However, denoting cardinality may be restrictive in systems with a dynamic number of agents of a given type, and so we omit it.

Every bid/ask line has an arrow on each side of the market; the arrow closest to the buyer represents the relationship between the buyer type and the item type while the arrow closest to the item type represents the relationship between the item type and the buyer type. Note that an item may inherit attributes from its seller.

Figure 2 shows the meanings of each different arrow type and the corresponding bid/ask offer representation. No hash behind the arrow indicates that the agents on the corresponding side of the market are anonymous, meaning that an agent only needs to place one offer for each item among all buyers or sellers. Alternatively, a hash behind an arrow means that each buyer would need to place a separate bid for each seller. A single arrow means that the corresponding buyer or seller bids on each item separately, whereas a double arrow indicates non-linear bids.

When multiple lines go through the market to a given buyer or item, the buyer or seller may place non-linear offers that include multiple agents and item types. To prevent an agent from placing non-linear bids on multiple items, the market may be broken into several auctions, each drawn as its own separate circle.

Finally, we note that the bid/ask offer representation and the diagrammatic representation have a one-to-one mapping. The diagrammatic representation can better represent a market system visually to readers while the bid/ask offer notation is more compact and easier to represent individual relationships. Additionally, the offer notation is amenable to programmatic inference.

Example: Warehouse Routing

In this section, we apply our representation methods to a warehouse routing problem. First we define the problem and then represent different market solutions. Finally, we will provide a brief discussion on how different market designs affect performance.

Problem Definition

ALPHABET SOUP is analogous to the real-world problem of order fulfillment in a warehouse environment (Hazard, Wurman, & D'Andrea 2006). The objective of the ALPHABET SOUP warehouse is to assemble specific words out of component letters. The inventory of the system are the *letter tiles*, which are stored in moveable buckets. The buckets can be picked up and carried around the warehouse by *bucketbots*, which move buckets to and from stations to accomplish the overall system objectives. A *letter station* puts letter tiles into buckets, while a *word station* takes letters out of buckets to compose words. In a warehouse environment, stations are typically located on the borders of the map.

A *letter tile* is a combination of a letter and a tile color, and a *word* is a sequence of letter tiles. The letters in a word do not need to have the same tile color. ALPHABET SOUP uses a dictionary of English words and a color profile to construct the set of words that make up the incoming orders. These words are distributed to word stations as jobs that need to be completed. Each word is a set of specific letter tiles that must be brought together to form the word. Each word station has a finite number of jobs it may be actively working on at any one time. When a word is completed, the station puts it into the completed-words list and can accept a new word.

In order to build words, an adequate inventory of letter tiles must be available. New letters are received at the letter stations in homogeneous bundles. To get the letter tiles into inventory, bucketbots must bring buckets to the letter station. Like the limit on the number of active words in a word station, each letter station has a limit on the maximum number of bundles which may be simultaneously staged.

The final component of the system is the bucketbot. Bucketbots can grab and release a bucket, and move. A bucketbot can pick up only one bucket at a time, and likewise a bucket may be attached only to one bucketbot at a time.

Market Representations

First we consider a multiple simultaneous single-item auction design of ALPHABET SOUP, as shown in Figure 3. Word stations sell completed words to the word queue to fulfill words ordered by the system. In order to prevent multiple word stations from completing the same words and duplicating orders, word stations sell a contract to complete the

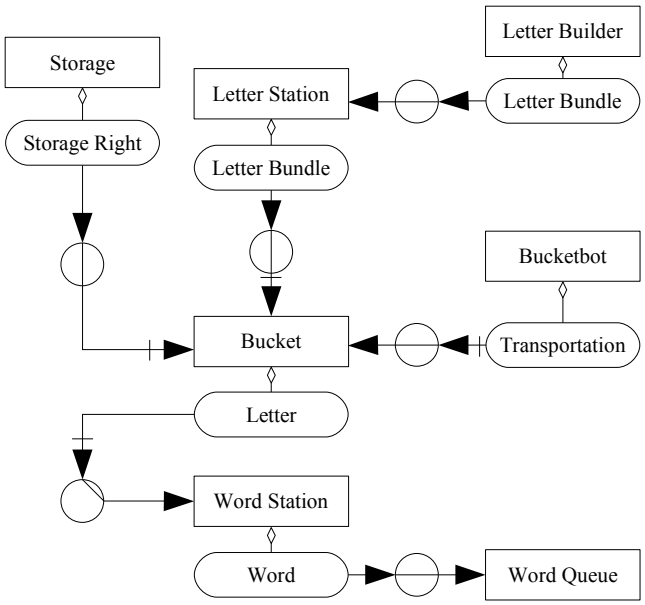

Figure 3: Multiple simultaneous single-item auction design of ALPHABET SOUP.

given word. The word queue bids the system's revenue for the order, and the word station's ask price is formed from the current market prices of the letters needed to complete the word.

Each bucket offers to sell its current letters to each word station. Letters are anonymously priced with respect to word stations. Buckets, however, must travel different distances to reach each word station, making a bucket's distance to a letter non-anonymously priced. Bucket travel is arranged by purchasing transportation from bucketbots. Transportation offered by bucketbots is anonymous, since bucketbots travel at the same speed. However, bucketbots must travel an initial distance in order to pick up a bucket, making the buckets non-anonymously priced to bucketbots. When a bucket no longer needs transportation, it must be stored at a storage location. Storage rights are offered by each storage location, and are sold to the highest bidder. Buckets are anonymously priced to storage locations, but storage locations are non-anonymously priced to buckets, as different storage locations will be different distances to the bucket and also word and letter stations. A bucket loses its storage rights when it is picked up by a bucketbot.

Letter auctions operate similarly to the word markets. Buckets bid on letter bundles at letter stations, their bid price partly dependent on the distance to the letter station, but buckets are anonymously priced with respect to letter stations. Letter stations buy letter bundles from the system letter builder based on demand from buckets.

The design of Figure 3 is represented symbolically as

$$\begin{aligned} LetterBuilder &: ask\{LetterStation, LetterBundle\} \\ LetterStation &: bid\{LetterBuilder, LetterBundle\} \\ LetterStation &: ask\{Bucket, LetterBundle\} \\ Bucket &: bid\{LetterStation+, LetterBundle\} \end{aligned}$$

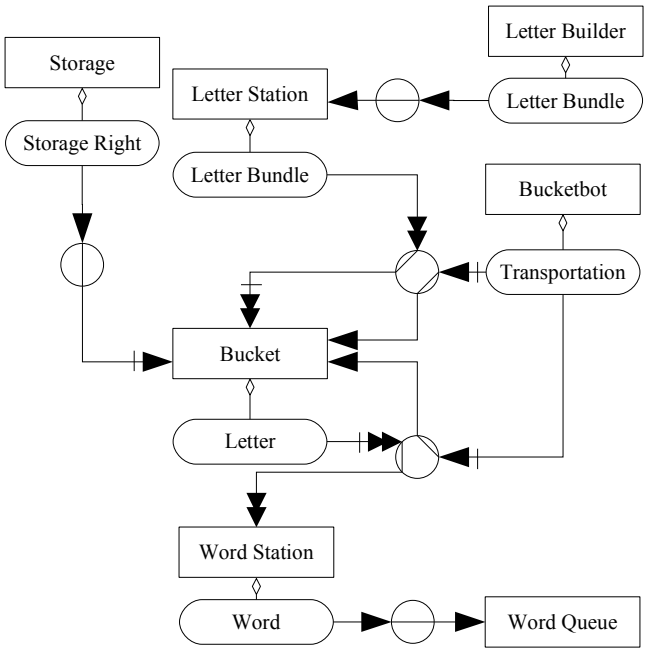

Figure 4: Partly non-linear price market model of ALPHABET SOUP.

$$\begin{aligned} Storage &: ask\{Bucket, StorageRight\} \\ Bucket &: bid\{Storage+, StorageRight\} \\ Bucketbot &: ask\{Bucket+, Transportation\} \\ Bucket &: bid\{Bucketbot, Transportation\} \\ Bucket &: ask\{WordStation+, Letter\} \\ WordStation &: bid\{Bucket, Letter\} \\ WordQueue &: bid\{WordStation, Word\} \\ WordStation &: ask\{WordQueue, Word\}. \end{aligned}$$

Figure 4 represents a second possible market system for ALPHABET SOUP, incorporating non-linear priced auctions while preserving price anonymity relationships from the previous model. The major change is that buckets bid on transportation in conjunction with a set of letters to buy or sell. Buckets' bids consist of a set of letters from across a set of letter stations in addition to transportation. The benefits of these non-linear bids are that buckets can express preferences to deliver multiple letters to word stations that are close to each other. Similarly, if a bucket has ten of letter q and two of letter u, the bucket may prefer to sell u in combination with q to lower its surplus of letter q. The non-linear offer to sell letters to word stations simultaneously allows a bucket to bid for transportation. An alternative design would be for the word station to buy the transportation for buckets in combination with letters. In providing transportation, the word station would also need to bid on storage for the buckets, unless the bucket retained the word station's transportation after the transaction. Further, to allow buckets to deliver to multiple word stations in the same trip, word stations would need to have a separate market to trade bucket and bucketbot combinations with each other.

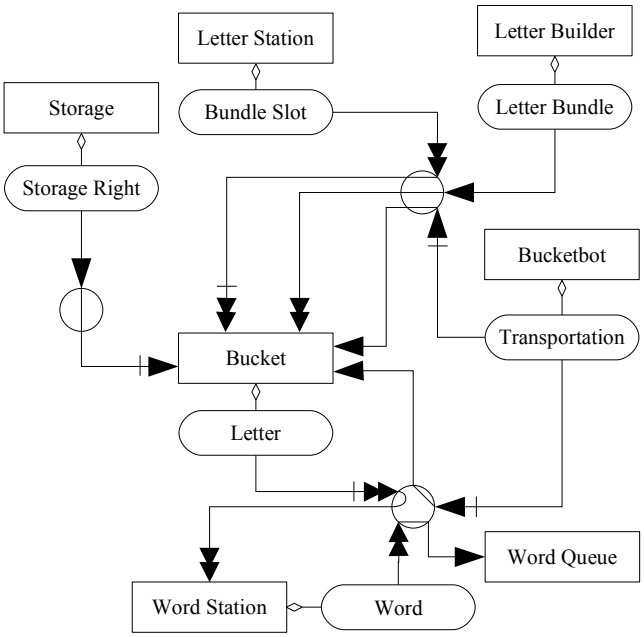

Figure 5: Mostly non-linear priced market model of ALPHABET SOUP.

The symbolic representation of Figure 4 is

$$
\begin{aligned}
LetterBuilder &: ask\{LetterStation, LetterBundle\}\\
LetterStation &: bid\{LetterBuilder, LetterBundle\}\\
LetterStation &: ask\{Bucket, LetterBundle*\}\\
Bucketbot &: ask\{Bucket+, Transportation\}\\
Bucket &: bid\{LetterStation+, LetterBundle*\},\\
&\quad bid\{Bucketbot, Transportation\}\\
Storage &: ask\{Bucket, StorageRight\}\\
Bucket &: bid\{Storage+, StorageRight\}\\
Bucket &: ask\{WordStation+, Letter*\},\\
&\quad bid\{Bucketbot, Transportation\}\\
WordStation &: bid\{Bucket, Letter*\}\\
WordQueue &: bid\{WordStation, Word\}\\
WordStation &: ask\{WordQueue, Word\}.
\end{aligned}
$$

We add further complexity to our ALPHABET SOUP market system in Figure 5. Here, all resources must be allocated at once to picking up a set of letters or delivering a word. A word station must bid on a set of words to complete, but the allocation will not occur until a set of buckets have also offered the letters to complete each word and bid on transportation to ensure the delivery. Similarly, to obtain a set of letters, a bucket must buy the letters directly from the letter builder and also buy a bundle slot from a letter station from which the bucket can pick the letter up. Figure 5 is represented symbolically as

$$
\begin{aligned}
LetterBuilder &: ask\{Bucket, LetterBundle\}\\
LetterStation &: ask\{Bucket, BundleSlot*\}\\
Bucketbot &: ask\{Bucket+, Transportation\}\\
Bucket &: bid\{LetterStation+, BundleSlot*\},\\
&\quad bid\{LetterBuilder, LetterBundle*\},\\
&\quad bid\{Bucketbot, Transportation\}\\
Storage &: ask\{Bucket, StorageRight\}\\
Bucket &: bid\{Storage+, StorageRight\}\\
Bucket &: ask\{WordStation+, Letter*\},\\
&\quad bid\{Bucketbot, Transportation\}\\
WordStation &: bid\{Bucket, Letter*\},\\
&\quad ask\{WordQueue, Word*\}\\
WordQueue &: bid\{WordStation, Word\}.
\end{aligned}
$$

Impact of Market Structures on Performance

One way that market design impacts ALPHABET SOUP is size and scope of the optimizations performed. Compared with the market system depicted in Figure 4, the market system in Figure 5 requires more coordination, but may benefit from the combinatorial auction's ability to express complementaries. Allocations within ALPHABET SOUP benefit greatly from combining similar tasks, such as a bucket delivering multiple letters per visit to a word station. We have found that the tâtonnement approach can perform quite well. Our initial experiments with simulation have indicated that linear pricing can out-perform greedy algorithms by 10-20% and can perform on-par with a simple non-linear priced implementation of the system shown in Figure 4.

Feedback is another major issue in market design, though feedback tends to reside more in the valuation models of the agents. One design choice would be for word stations to choose words that the closest buckets can best fulfill and buckets prefer to bring letters to the closest word stations. This positive feedback can specialize regions in terms of letters and letter combinations and affect the system's throughput. Using learning or other adaptive techniques in computing valuations can create more complex feedback.

Market design also directly affects agents' utility functions. Consider the Word Station in ALPHABET SOUP. In Figure 3, the word station needs to evaluate the expected cost to complete each word independently and bid on each letter from every bucket to fulfill words. Figure 4 allows the word station to offer non-linear bids on combinations of letters from buckets. The word station could value a set of letters that would complete several nearly-complete words more than a combination of letters that start to complete a new word. By completing several nearly complete words and starting new words in their place, the word station would have more choices for letters to buy and be less likely to remain idle while waiting on one or two specific letters. Finally, in the system in Figure 5, a word station would need to make sure that it had all the letters ready to buy before starting to fulfill a word.

Example: TAC SCM

In the Trading Agent Competition Supply Chain Management (TAC SCM) game (http://www.sics.se/tac/), agents compete against each other to earn the most profit as a personal computer (PC) manufacturer in a supply chain. Each

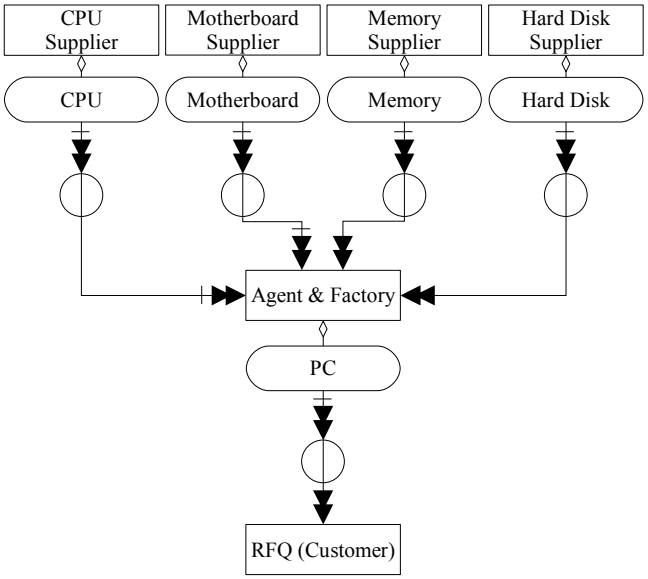

Figure 6: TAC SCM diagrammatic representation.

agent is responsible for procuring components from suppliers, managing assembly and shipping, and bidding on orders from customers. Each agent has identical factories, and competes with all other agents in the same market.

Figure 6 shows the diagrammatic representation of TAC SCM. An agent can submit several bids per day to each supplier asking for different quantities of different components, making all of the arrows between the agent and supplier double. All suppliers offer two grades of components except for the motherboard suppliers. The CPU and motherboard are non-anonymously priced, as CPU brands will only work with a corresponding brand of motherboards. This non-anonymous pricing is indicated by the hashes behind the arrows. Memory and hard drives are anonymously priced with respect to the agents. Each agent has a reputation with the suppliers based on its history, so agents are not anonymously priced with respect to the suppliers.

TAC SCM does not distinguish between customers, only between customer bids, known as requests for quotation (RFQ). To a customer, PC's are anonymously priced between agents. However, RFQ's are not anonymously priced between agents because RFQ's may call for PC's with different attributes. The customers' bids and the agents' asks are for multiple PC's, yielding the double arrows on both sides of that market.

Conclusions

We present a normative method to describe relationships in complex market systems. With the applicability of MOP, the ability to communicate market designs is very important in order to compare and evaluate designs. Our diagrammatic representation maps directly to our symbolic market notation so that market designs may be communicated effectively and concisely.

References

Babaioff, M., and Walsh, W. E. 2005. Incentive-compatible, budget-balanced, yet highly efficient auctions for supply chain formation. *Decision Support Systems* 39(1):123–149.

Berhault, M.; Huang, H.; Keskinocak, P.; Koenig, S.; Elmaghraby, W.; Griffin, P.; and Kleywegt, A. 2003. Robot exploration with combinatorial auctions. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, volume 2, 1957–1962.

Bikhchandani, S., and Ostroy, J. M. 2002. The package assignment model. *Journal of Economic Theory* 107(2):377–406.

Cheng, J. Q., and Wellman, M. P. 1998. The walras algorithm: A convergent distributed implementation of general equilibrium outcomes. *Computational Economics* 12(1):1–24.

Fan, M.; Stallaert, J.; and Whinston, A. B. 2003. Decentralized mechanism design for supply chain organizations using an auction market. *Information Systems Research* 14(1):1–22.

Fox, M. S.; Barbuceanu, M.; and Teigen, R. 2000. Agent-oriented supply-chain management. *International Journal of Flexible Manufacturing Systems* 12:165–188.

Hazard, C. J.; Wurman, P. R.; and D'Andrea, R. 2006. Alphabet soup: A testbed for studying resource allocation in multi-vehicle systems. In *Proceedings of the AAAI Workshop on Auction Mechanisms for Robot Coordination*, 23–30.

Lagoudakis, M. G.; Markakis, E.; Kempe, D.; Keskinocak, P.; Kleywegt, A.; Koenig, S.; Tovey, C.; Meyerson, A.; and Jain, S. 2005. Auction-based multi-robot routing. In *Proceedings of the International Conference on Robotics: Science and Systems (ROBOTICS)*, 343–350.

Neumann, D. G., and Holtmann, C. 2004. Embodiment design in market engineering. In *Proceedings of the Research Symposium on Emerging Electronic Markets (RSEEM)*, 85–96.

Walsh, W. E., and Wellman, M. P. 2003. Decentralized supply chain formation: A market protocol and competitive equilibrium analysis. *Journal of Artificial Intelligence Research* 19:513–567.

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

Wellman, M. P. 1996. Market-oriented programming: Some early lessons. In Clearwater, S., ed., *Market-Based Control: A Paradigm for Distributed Resource Allocation*. River Edge, New Jersey: World Scientific. chapter 4, 74 – 95.