Coordinating Multi-Attribute Reverse Auctions Subject to Finite Capacity Considerations

Jiong Sun and Norman M. Sadeh

CMU-ISRI-03-105

Institute for Software Research International
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213-3891

Appeared in the *Proceedings of the 37th Hawaii International Conference on System Sciences*, January 5-8, 2004, Big Island, Hawaii

The research reported in this paper has been conduced as part of the MASCHINE project, a joint research effort between Carnegie Mellon University and the University of Michigan funded by the National Science Foundation under ITR Grant 0205435.

Keywords: Supply Chain Formation, Reverse Auction, Finite Capacity Scheduling

Abstract

Reverse auctions offer the prospect of more efficiently matching suppliers and producers in the face of changing market conditions. Prior research has generally ignored the temporal and finite capacity constraints under which reverse auctioneers typically operate. In this paper, we consider the problem faced by a manufacturer (or service provider) that needs to fulfill a number of customer orders, each requiring a possibly different combination of components (or services). The manufacturer can procure these components or services from a number of possible suppliers through multi-attribute reverse auctions. Bids submitted by prospective suppliers include a price and a delivery date. The reverse auctioneer has to select a combination of supplier bids that will maximize its overall profit, taking into account its own finite capacity and the prices and delivery dates offered by different suppliers for the same components/services. The manufacturer's profit is determined by the revenue generated by the products it sells, the cost of the components/services it purchases, as well as late delivery penalties it incurs if it fails to deliver products/services in time to its own customers. We provide a formal model of this important class of problems, discuss its complexity and introduce rules that can be used to efficiently prune the resulting search space. We also introduce a branch-and-bound algorithm that takes advantage of these pruning rules along with two heuristic search procedures. Computational results are presented that evaluate the performance of our heuristic procedures under different conditions both in terms of computational requirements and distance from the optimum. Our experiments show that taking into account finite capacity considerations can significantly improve the manufacturer's bottom line, thereby confirming the importance of these constraints and the effectiveness of our search heuristics.

1. Introduction

Today's global economy is characterized by fast changing market demands, short product lifecycles and increasing pressures to offer high degrees of customization, while keeping costs and lead times to a minimum. In this context, the competitiveness of both manufacturing and service companies will increasingly be tied to their ability to identify promising supply chain partners in response to changing market conditions. With the emergence of e-business standards, such as ebXML, SOAP, UDDI and WSDL, the Internet will over time facilitate the development of more flexible supply chain management practices.

Today, however such practices are confined to relatively simple scenarios such as those found in the context of MRO (Maintenance, Repair and Operations) procurement. The slow adoption of dynamic supply chain practices and the failure of many early electronic marketplaces can in part be attributed to the one-dimensional nature of early solutions that forced suppliers to compete solely on the basis of price. Research in the area has also generally ignored key temporal and capacity constraints under which reverse auctioneers typically operate. For instance, a PC manufacturer can only assemble so many PCs at once and not all PCs are due at the same time. Such considerations can be used to help the PC manufacturer select among bids from competing suppliers.

In this paper, we present techniques aimed at exploiting such temporal and capacity constraints to help a reverse auctioneer select among competing multi-attribute procurement bids that differ in prices and delivery dates. We refer to this problem as the *Finite Capacity Multi-Attribute Procurement* (FCMAP) problem. It is representative of a broad range of practical reverse auctions, whether in the manufacturing or service industry. This article provides a formal definition of the FCMAP problem, discusses its complexity and introduces several rules that can be used to prune its search space. It also presents a branch-and-bound algorithm and two heuristic search procedures that all take advantage of these pruning rules. Computational results show that accounting for the reverse auctioneer's finite capacity can significantly improve its bottom line, confirming the important role played by finite capacity considerations in procurement problems. Results are also presented that compare the performance of our heuristics search procedures both in terms of solution quality and computational requirements under different bid profile assumptions. These results suggest that our procedures are generally capable of generating solutions that are just within a few percent of the optimum and that they scale nicely as problem size increases.

The balance of this paper is organized as follows. Section 2 provides a brief review of the literature. In section 3, we introduce a formal model of the FCMAP problem. Section 4 identifies three rules that can help the reverse auctioneer (or manufacturer) eliminate non-competitive bids or bid combinations. Section 5 introduces a branch-and-bound algorithm that takes advantage of our pruning rules. This is followed by the presentation of two heuristic search procedures that also take advantage of our pruning rules. In particular, Section 6 details a randomized early/tardy heuristic that exploits a property of the FCMAP problem introduced in Section 4. In Section 7, a second heuristic search procedure is presented that combines Simulated Annealing (SA) search with a cost estimator based on the well-known "Apparent Tardy Cost" rule first introduced by Vepsalainen and Morton (1987). An extensive set of computational results are presented and discussed in Section 8. Section 9 provides some concluding remarks and discusses future extensions of this research.

2. Literature Overview

Few researchers have studied supply chain formation problems in the context of capacity-constrained environments. A notable exception is the work of Gallien and Wein (2002) who have proposed a reverse auction mechanism that takes into account supplier capacity constraints. Babaioff and Nisan (2001) have designed information exchange protocols that enhance supply chain responsiveness in the face of surges or drops in demand and supply. Their work however assumes infinite production capacity, where an increase in the production volume of one product does not impact the ability of the manufacturer to possibly increase or maintain production levels for other products. Other relevant work includes that of Walsh and Wellman (1998), though here again capacity constraints are ignored. Sadeh et al. (2001) discuss MASCOT, an agent-based supply chain decision support tool that supports finite capacity models. Their work to date has focused on the empirical study of real-time "capable-to-promise" and "profitable-to-promise" functionality and on scheduling coordination across static supply chains (Kjenstad 1998). Another significant effort in this area is the work carried out by the team of Collins and Gini in the context of MAGNET (Collins et al. 1998).

3. The Finite Capacity Multi-Attribute Procurement Problem

The Finite Capacity Multi-Attribute Procurement (FCMAP) problem revolves around a reverse auctioneer – referred below as the "manufacturer", though it could also be a service provider. The manufacturer has to satisfy a set of customer commitments or orders $O_i, i \in M = \{1,...,m\}$ (see Figure 1). Each order i needs to be completed by a due date dd_i, and requires one or more components (or services), which the manufacturer can obtain from a number of possible suppliers. The manufacturer has to wait for all the components before it can start processing the order (e.g., waiting for all the components required to assemble a given PC). For the sake of simplicity, we assume that the processing required by the manufacturer to complete work on customer order O_i has a fixed duration du_i, and that the manufacturer can only process one order at a time ("capacity constraint").

Formally, for each order O_i and each component $comp_{ij}, j \in N_i = \{1,...n_i\}$, the manufacturer organizes a reverse auction for which it receives a set of multi-attribute bids $\beta_{ij} = \{B_1^{ij},...,B_{n_{ij}}^{ij}\}$ from prospective suppliers. Each bid B_k^{ij} includes a bid price bp_k^{ij} and a proposed delivery date dl_k^{ij}. Below we use the notation $B_k^{ij} = (dl_k^{ij}, bp_k^{ij})$.

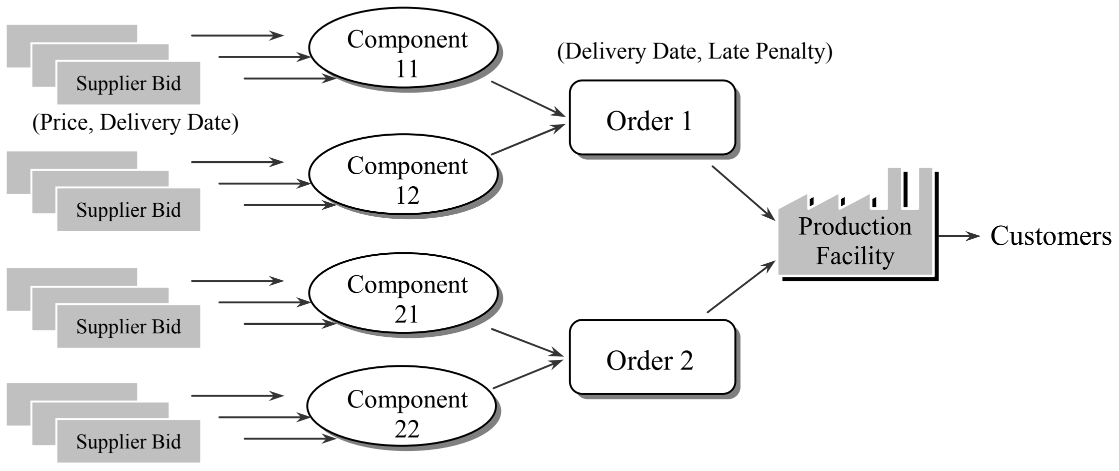

Figure 1. Finite capacity multi-attribute procurement problem

Failure by the manufacturer to meet an order O_i's due date results in a penalty $tard_i \times T_i$, where T_i is the time by which delivery of the product or service is late, and $tard_i$ is the marginal penalty for missing the delivery date. Such penalties, which are commonly used to model manufacturing scheduling problems, reflect actual contractual terms, loss of customer goodwill, interests on lost profits or a combination of the above (Pinedo 1995).

A solution to the FCMAP problem consists of:

- a selection of bids: $Bid\_Comb = \{Bid\_Comb_1,...,Bid\_Comb_m\}$, where Bid_Comb_i ($i \in M$) is a combination of n_i bids - one for each of the components required by order O_i, and
- a collection of start times: $ST = \{st_1,...,st_m\}$, where st_i is the time when the manufacturer is scheduled to start processing order O_i, and $st_i \geq dl^{ij}, \forall j = 1,..,n_i$, since orders cannot be processed before all the components they require have been delivered by suppliers.

Given a solution (Bid_Comb, ST), the profit of the manufacturer is the difference between the revenue generated by its customer orders (once they have been completed) and the sum of its procurement costs and tardiness penalties. This is denoted:

$$prof(Bid\_Comb, ST) = \sum_{i \in M} rev_i - \sum_{i \in M} \sum_{j \in N_i} bp^{ij} - \sum_i tard_i \times T_i \tag{1}$$

where,

- rev_i is the revenue generated by the completion of order O_i (i.e., the amount paid by the customer),
- bp^{ij} is the price of component $comp_{ij}$ in Bid_Comb, and
- $T_i = Max(0, st_i + du_i - dd_i)$ with st_i being the start time of order O_i in ST.

Note that because we assume a given set of orders, the term $\sum_{i \in M} rev_i$ is the same across all solutions. Accordingly, maximizing profit in Equation (1) is equivalent to minimizing the sum of procurement and tardiness costs: $cost(Bid\_Comb, ST) = \sum_{i \in M} \sum_{j \in N_i} bp^{ij} + \sum_i tard_i \times T_i$.

It is worth noting that the above model contrasts with earlier research in dynamic supply chain formation, which has generally assumed manufacturers with infinite capacity or fixed lead times and ignored delivery dates and tardiness penalties (Collins et al. 2001, Davis and Smith 1983, Faratin and Klein 2001, Sandholm 1993).

From a complexity standpoint, it can easily be seen that the FCMAP problem is strongly NP-hard, since the special situation where all components are free and available at time zero reduces to the single machine total weighted tardiness problem, itself a well known NP-hard problem (Du and Leung 1990).

An example of an exact procedure to solve FCMAP problems involves looking at all possible procurement bid combinations and, for each such combination, solving to optimality a single machine weighted tardiness problem with release dates (e.g., using a branch-and-bound algorithm). A release date is a date before which a given order is not allowed to be processed. Given a combination of procurement bids Bid_Comb_i, an order O_i has a release date:

$$r_i = \underset{j \in N_i}{Max}[dl^{ij}] \tag{2}$$

where dl^{ij} denotes the delivery date of component $comp_{ij}$ in Bid_Comb_i. In other words, the component that arrives the latest determines the order's release date.

Clearly, with the exception of fairly small problems, the requirements of the above procedure are computationally prohibitive. Below, we identify a number of rules that can be used to efficiently prune the search space associated with FCMAP problems.

4. Pruning the Search Space

Pruning Rule 1: Eliminating Expensive Bids with Late Delivery Dates

Consider an FCMAP problem P with an order O_i requiring a component $comp_{ij}$ for which the manufacturer has received a set of bids $\beta_{ij} = \{B_1^{ij},...,B_{n_{ij}}^{ij}\}$ from prospective suppliers. Let $B_k^{ij} = (dl_k^{ij}, bp_k^{ij})$ and $B_l^{ij} = (dl_l^{ij}, bp_l^{ij})$ be two bids in β_{ij} such that:

$$dl_l^{ij} \geq dl_k^{ij} \text{ and } bp_l^{ij} \geq bp_k^{ij}.$$

Then problem P' with $\beta_{ij}' = \beta_{ij} \setminus \{B_l^{ij}\}$ *admits the optimal solutions with the exact same profit as problem P.*

The correctness of this rule is obvious. Its application is illustrated in Figure 1, where an order requires two components: component 1 and component 2. The manufacturer has received bids for each component. Using Rule 1, it can be determined, for instance, that bid_{14} is not competitive given that it is more expensive than bid_{13} and arrives late. Similarly, bid_{22} and bid_{24} can also be pruned.

Pruning Rule 2: Eliminating Expensive Bids with Unnecessarily Early Delivery Dates

Consider an FCMAP problem P with an order O_i requiring a set of components $comp_{ij}, j \in N_i = \{1,...n_i\}$. Let $\beta_{ij} = \{B_1^{ij},...,B_{n_{ij}}^{ij}\}$ be the set of bids received by the manufacturer for each component $comp_{ij}$ with $B_k^{ij} = (dl_k^{ij}, bp_k^{ij})$. We define $r_i^{earliest}$ as the earliest possible release date for order O_i. It can be computed as:

$$r_i^{earliest} = \underset{1 \le j \le n_i}{Max}\ \underset{1 \le k \le n_{ij}}{Min}\ dl_k^{ij}$$

Let B_k^{ij} and B_l^{ij} be two bids for component $comp_{i\,j}$ such that:

$$bp_l^{ij} \ge bp_k^{ij} \text{ and } dl_l^{ij} \le dl_k^{ij} \le r_i^{earliest}.$$

Then problem P' with $\beta_{i\,j}{}' = \beta_{i\,j} \setminus \{B_l^{ij}\}$ *admits the exact same set of optimal solutions as problem P.*

An intuitive explanation should suffice to convince the reader. While bid B_l^{ij} has an earlier delivery date than bid B_k^{ij}, this earlier date is not worth paying more for: it does not add any scheduling flexibility to the manufacturer since the start of order O_i remains constrained by $r_i^{earliest} \ge dl_l^{ij}$. A formal proof can easily be built based on this observation.

Note that, in general, it is not possible to prune bid B_k^{ij}. This is because other bids for component $comp_{i\,j}$ may have delivery dates that are after $r_i^{earliest}$, which would reduce the number of available scheduling options possibly leading to lower quality solutions. Application of this rule is also illustrated in Figure 2, where it results in the pruning of bid_{11}. This is because both bid_{11} and bid_{12} arrive before the order's earliest release date, $r^{earliest}$, and bid_{11} is more expensive than bid_{12}.

Pruning Rule 3: Eliminating Expensive Bid Combinations with Unnecessarily Early Delivery Dates

Consider an FCMAP problem P whose search space has already been pruned using Rule 1. In other words, given two bids $B_k^{ij} = (dl_k^{ij}, bp_k^{ij})$ and $B_l^{ij} = (dl_l^{ij}, bp_l^{ij})$, $j \ne k$, for the same component $comp_{i\,j}$, if $dl_l^{ij} > dl_k^{ij}$, then $bp_l^{ij} < bp_k^{ij}$.

Let $Bid\_Comb_i^a = \{B_a^{i1}, \ldots, B_a^{in_i}\}$ be a combination of bids for the n_i components required by order O_i. Suppose also that there exist two bids $B_a^{ik} = (dl_a^{ik}, bp_a^{ik}) \in Bid\_Comb_i^a$ and $B_b^{ik} = (dl_b^{ik}, bp_b^{ik})$ such that $dl_a^{ik} < dl_b^{ik} \le dl_a^{il}$, then $Bid\_Comb_i^a$ is dominated by $Bid\_Comb_i^b$, where $Bid\_Comb_i^b$ = $(Bid\_Comb_i^a$

$\setminus \{B_a^{ik}\}) \cup \{B_b^{ik}\}$. *By "dominated" we mean that, for every solution to problem P involving* $Bid\_Sel_i^a$, *there is a better solution where* $Bid\_Comb_i^a$ *is replaced by* $Bid\_Comb_i^b$.

Again, intuitively, this is easy to understand. Given that $Bid\_Comb_i^a$ includes a bid for a second component $comp_{il}$ that gets delivered at time $dl_a^{il} \geq dl_b^{ik} > dl_a^{ik}$, replacing bid B_a^{ik} with bid B_b^{ik} will not delay the start of order O_i and can only help reduce the cost of its components since $bp_a^{ik} > bp_b^{ik}$ (as indicated earlier, we assume that Rule 1 has already been applied to prune bids). Once again, it is straightforward to build a formal proof based on the above observation. Note also that Rule 3 actually subsumes Rule 2 – Rule 2 is easier to visualize and also introduces the notion of earliest possible release date, which we use later in this article.

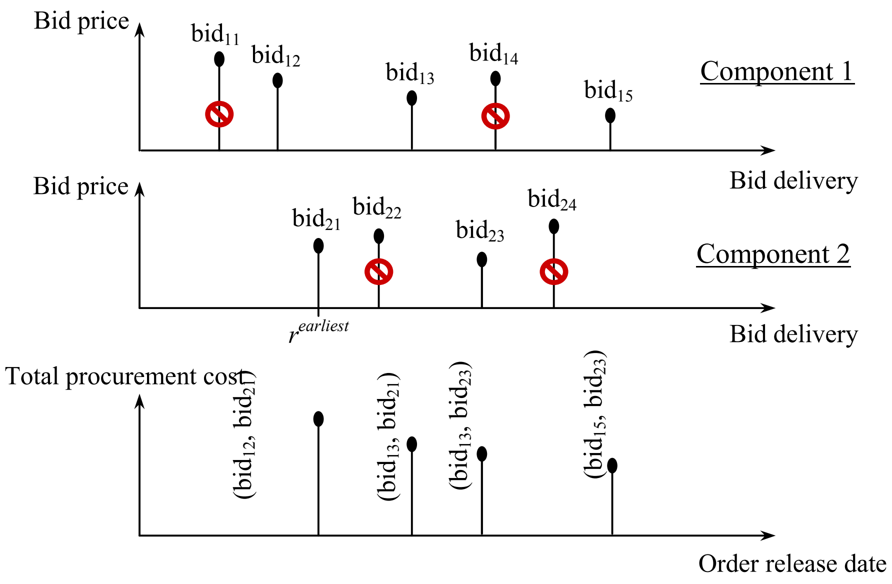

Figure 2. From 20 bid combinations to 4 non-dominated ones

The three pruning rules we just identified can be used to prune the set of bids to be considered. This is illustrated in Figure 2, where the combination of the three rules brings the number of bid combinations to be considered from 20 to just for 4 non-dominated combinations. In particular, the application of Rule 3 helps us prune bid combination (bid_{12}, bid_{23}). This is because this combination is dominated by (bid_{13}, bid_{23}), which results in the same release date but is cheaper. Another bid combination pruned using Rule 3 is (bid_{15}, bid_{21}).

It should be clear that, for each order, Rules 1 and 2 can be applied in $O(c \cdot b \cdot \log b)$ time, where b is an upper-bound on the number of bids received for a given component and c an upper-bound on the number of components required by a given order. It can also be shown that, for a given order O_i, Rule 3 can be applied in $O(tb \cdot \log tb)$ time, where tb is the total number of bids received for order O_i across all the components it requires. This is done as follows:

1. For each component $comp_{il}$, create a sorted list $\lambda_{ij} =< B_1^{ij},..,B_{n_{ij}}^{ij} >$ such that $dl_k^{ij} < dl_{k+1}^{ij}$. Create an overall list of delivery dates for all the bids received for O_i (i.e., for all the components required by the order) and sort the delivery dates in increasing order. Let Λ_i be this sorted list
2. For each date r_i in Λ_i, keep only those non-dominated combinations of bids that are compatible with having r_i as order O_i's release date. Note that such bid combinations are of the form $\{B_a^{i1},...,B_a^{in_i}\}$ where $dl_a^{ij} \le r_i, \forall j$, and there is no other bid B_b^{ij} such that $dl_a^{ij} < dl_b^{ij} \le r_i$. In other words, for each component $comp_{il}$, B_a^{ij} is the latest bid compatible with release date r_i (and hence also the cheapest such bid). Finding such bids requires very little time, given the sorted bid lists λ_{ij} created in step 1.

As a parenthesis, it is worth noting that the three pruning rules we just introduced apply to scenarios with more complex constraints, as they only take advantage of the release constraint that requires each order to have all its components before it can be processed. For instance, this includes problems where the manufacturer is modeled as a job shop, the capacity of some machines is greater than one and there are sequence dependent setup times. It can also be shown that the pruning rules can be extended to accommodate problems with inventory holding costs, as long as orders are not allowed to be shipped before their due dates – this assumption corresponds to having finished goods inventory and is representative of many supply chain situations.

Consider the non-dominated bid combinations resulting from the application of our three pruning rules to an FCMAP problem. Let the non-dominated bid combinations of order O_i be denoted:

$$Bid\_Comb_i^* = \{Bid\_Comb_i^1 = (r_{i1}, pc_{i1}), \ldots, Bid\_Comb_i^{m_i} = (r_{im_i}, pc_{im_i})\},$$

where r_{ik} is the release date of bid combination $Bid\_Comb_i^k$, as defined in Equation (2), and pc_{ik} is its total procurement cost, defined as the sum of its component bid prices. It follows that:

Property 1: *For each order O_i, $i \in M$, it must hold that, if $r_{ia} < r_{ib}$, then $pc_{ia} > pc_{ib}$, $\forall a,b \in \{1,\ldots,m_i\}, a \neq b$. In other words, the total procurement costs of non-dominated bid combinations strictly decrease as their release dates increase.*

<u>Proof</u>:

We have already shown that, following the application of Rule 1, the bids that remain for a given component have prices that strictly decrease as their delivery dates increase.

Let $Bid\_Comb_i^a$ be a non-dominated bid combination for order O_i – following the application of Rules 1 through 3. Let its release date r_{ia} be determined by the delivery date of component j, namely $r_{ia} = dl_a^{ij}$. Note that, by definition, the release date of a bid combination is always determined by one or more of its components. Given that Rule 3 has already been applied, the delivery date dl_a^{ik} of any component k must be the latest delivery date among those bids for component k that satisfy $dl_a^{ik} \leq dl_a^{ij}$.

Consider another non-dominated bid combination $Bid\_Comb_i^b$ for order O_i such that $r_{ib} > r_{ia}$. Let l be the index of one of the components determining the release date of bid combination $Bid\_Comb_i^b$, namely $r_{ib} = dl_b^{il} > r_{ia} = dl_a^{ij}$. Just as for bid combination $Bid\_Comb_i^a$, the fact that Rule 3 has been applied implies that the delivery date dl_b^{ik} of any component k in $Bid\_Comb_i^b$ must be the latest delivery date among those bids for component k that satisfy $dl_b^{ik} \le dl_b^{il}$. Given that $r_{ib} = dl_b^{il} > r_{ia} = dl_a^{ij}$, it also follows that, for any component k, we have $dl_b^{ik} \ge dl_a^{ik}$ with a strict inequality for at least one component, namely component *l*. Given that Rule 1 has been applied, it also follows that, for any component k, $bp_b^{ik} \le bp_a^{ik}$ with a strict inequality for at least one component (component *l*). Hence, $pc_{ia} = \sum_{1 \le k \le n_i} bp_a^{ik} > pc_{ib} = \sum_{1 \le k \le n_i} bp_b^{ik}$. □

Property 1 is illustrated in Figure 3, where we have two bid combinations $Bid\_Comb_i^a$ and $Bid\_Comb_i^b$ for an order O_i that requires three components. In this particular example, r_{ib} is determined by the delivery date of component 3, while r_{ia} is determined by that of component 2. The two bid combinations share the same delivery dates for two out of three of the components required by order O_i : components 1 and 2. The difference in procurement cost comes from the higher price associated with the later delivery of component 3 in bid combination $Bid\_Comb_i^b$ (namely, $dl_b^{i3} = r_{ib} > dl_a^{i3}$).

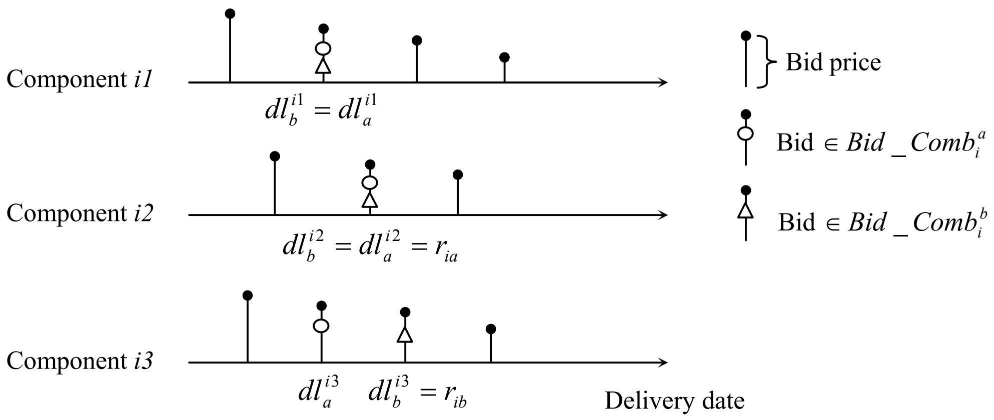

Figure 3. Illustration of Property 1

Note also that, if after application of the pruning rules there exist two non-dominated bid combinations, $Bid\_Comb_i^a$ and $Bid\_Comb_i^b$, such that $r_{ia} = r_{ib}$, it must hold that $pc_{ia} = pc_{ib}$.

In the following sections, we introduce a branch-and-bound algorithm to solve the FCMAP problem along with two (significantly faster) heuristic search procedures. All three procedures take advantage of the pruning rules we just introduced. One of the two heuristic procedures also takes advantage of Property 1.

5. A Branch-and-Bound Algorithm

Following the application of the pruning rules introduced in the previous section, optimal solutions to the FCMAP problem can be obtained using a branch-and-bound procedure. Branching is done over the sequence in which orders are processed by the manufacturer and over the release dates of non-dominated bid combinations of each order. Specifically, the algorithm first picks an order to be processed by the manufacturer then tries all the release dates (of non-dominated bid combinations) available for this order. Note that, as orders are sequenced in this fashion, some of their available release dates become dominated, given prior sequencing decisions. For instance, consider two orders O_1 and O_2, with O_2 having two release dates r_{21} and r_{22} with $r_{21} < r_{22}$ - following the application of pruning Rules 1 through 3. Suppose that, at the current node, O_1 is sequenced before O_2 and that O_1's earliest completion date is greater than r_{22}. It follows that release date r_{21} is strictly dominated by release date r_{22} at this particular node. Release dates that become dominated as a result of prior assignments can be pruned on the fly, thereby further speeding up the search procedure. Given a node *n* in the search tree, namely a partial sequence of orders and a selection of release dates for each of the orders already sequenced, it is possible to compute an upper-bound for the profit of all complete solutions (i.e. leaf nodes) compatible with this node:

$$UB_n = \sum_{i \in M} rev_i - \sum_{i \in OS_n} (pc_i + tard_i \times T_i) - \sum_{i \notin OS_n} [tard_i \times \max(0, cd_{OS_n} + du_i - dd_i) + mpc_i],$$

where:

- OS_n is the set of orders sequenced at node *n*;
- pc_i is the total procurement cost associated with the non-dominated release date (or bid combination) assigned to order $O_i \in OS_n$ and T_i is its tardiness. Note that each order is scheduled to start as early as possible, given prior sequencing decisions and the release date assigned to it: there are no benefits to starting later;
- cd_{OS_n} is the completion date of the last order in OS_n;
- mpc_i is the minimum possible procurement cost of order O_i - this cost is node-independent.

If the upper bound of a node *n* is lower than the best feasible solution found so far, the node *n* and all its descendants are pruned.

6. Early/Tardy Heuristic

Property 1 tells us that, following the application of the pruning rules, the procurement costs of non-dominated bid combinations strictly decrease as release dates increase. Figure 4 plots the total procurement cost and tardiness cost of an order for different possible start times. While tardiness costs increase linearly for start times that miss the order's due date, procurement costs vary according to a decreasing step function. Specifically, the circles in Figure 4 represent the order's non-dominated bid combinations. For instance, if the order starts at time *t*, its procurement cost is pc_i, namely the procurement cost of the latest non-dominated bid combination compatible with this start time (Bid_Comb^i). Its tardiness cost is equal to $tard \times \max(0, t + du - dd)$, where *tard* is its marginal tardiness penalty, *dd* its due date stand *du* its duration (or processing time). The end result is an early/tardy scheduling problem with non-linear earliness costs.

Ow and Morton have introduced an early/tardy dispatch rule for one-machine scheduling problems subject to linear earliness and tardiness costs (Ow and Morton 1989). Because our earliness costs are not linear, this heuristic can not readily be applied. Below, we briefly review some of its key elements and discuss how we have adapted it to produce a family of heuristic search procedures for the FCMAP problem.

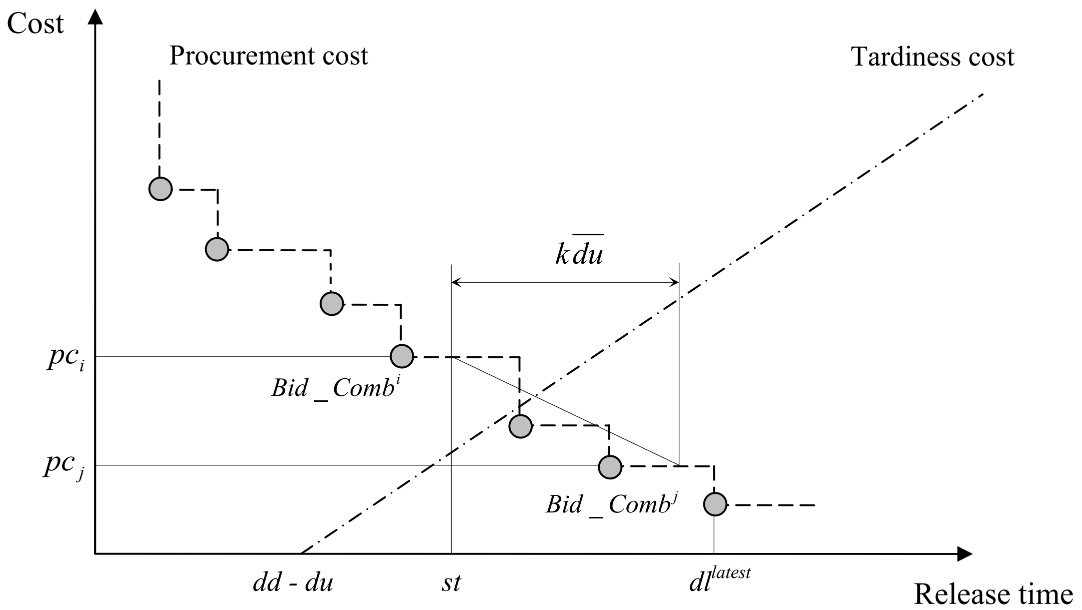

Figure 4. An order's tardiness and procurement costs

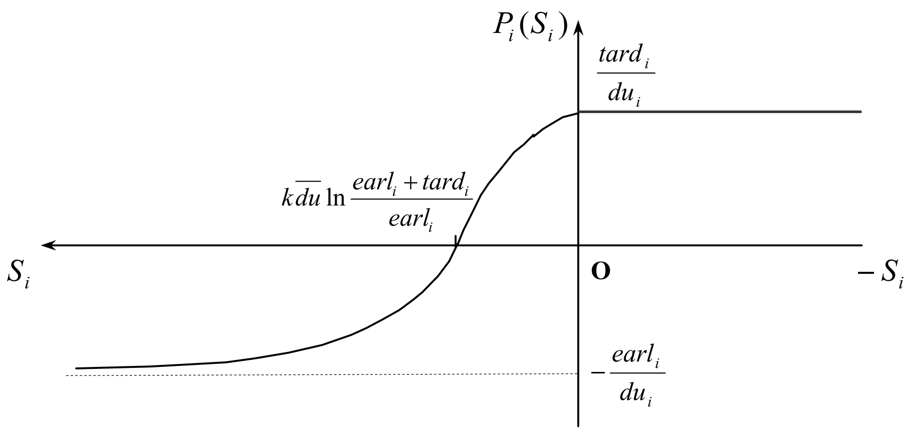

Figure 5. Early/Tardy dispatch rule

This Early/Tardy dispatch rule interpolates between two extreme cases (Figure 5). The first case is one where all orders are assumed to be late and where only tardiness needs to be minimized. The second situation is one where all orders are assumed to have plenty of time and where only earliness costs need to be minimized. In the former case, it can be shown that an optimal solution can be built by sequencing orders according to a *Weighted Shortest Processing Time* dispatch rule, where each order receives a priority:

$$P_i(S_i) = tard_i / du_i \, ,$$

where $P_i(S_i)$ is the priority of order O_i, S_i is its slack at time t, du_i its processing time and $tard_i$ its marginal tardiness penalty. Slack S_i at time t is defined as:

$$S_i = dd'_i - du_i - t \, ,$$

where dd'_i is an artificial due date of order O_i, which is defined as:

$$dd'_i = dd_i + (dl^{latest} - dd_i + du_i)^+ \, ,$$

where dl^{latest} is the delivery date of the cheapest non-dominated bid combination (see Figure 4) and $(X)^+$ denotes the positive part of X. By using this artificial due date, we avoid the situation where the procurement cost over-weights the tardiness cost in calculating the priority.

Conversely, in the latter case, when all jobs are assumed to be early, it can be shown that an optimal solution can be built by sequencing orders according to a *Weighted Longest Processing Time* dispatch rule of the form:

$$P_i(S_i) = -earl_i / du_i \, ,$$

where $earl_i$ is its marginal earliness cost – namely the penalty incurred for every unit of time the order finishes before its due date.

This Early/Tardy dispatch rule interpolates between these two cases by assigning to each order an early/tardy priority that varies with its slack:

$$P_i(S_i) = -\frac{earl_i}{du_i} + \frac{earl_i + tard_i}{du_i} \times \exp[-\frac{(S_i)^+}{k \cdot \overline{du}}] \qquad (3)$$

where $\overline{du}$ is the average processing time of an order and k is a look-ahead parameter. This parameter can intuitively be thought of as the average number of orders that will typically get processed ahead of an order queueing in front of the machine. The above formula can easily be seen to reduce to the Weighted Shortest Processing Time dispatch rule when slack $S_i \leq 0$ and to the Weighted Longest Processing Time dispatch rule when $S_i \to \infty$. The value of the look-ahead parameter *k* controls the transition between these two extremes, with higher values of *k* making the transition start earlier.

In the FCMAP problem, an order O_i cannot start before its earliest possible release date $r_i^{earliest}$ (see pruning Rule 2 – it should be clear that this release date is never pruned by Rule 3). In addition, earliness costs vary according to a step function. A marginal earliness cost can however be obtained through regression, whether locally or globally. Specifically, we distinguish between the following two approaches to computing marginal earliness costs for an order in the FCMAP problem:

1) *Local Earliness Weight*: At time *t*, the local marginal earliness cost associated with an order *O* (see Figure 4) can be approximated as the difference in procurement costs associated with the latest non-dominated bid combinations compatible with processing the order at respectively time *t* (namely Bid_Comb^i) and time $t + k \cdot \overline{du}$ (namely Bid_Comb^j):

$$earl^L = \frac{pc_i - pc_j}{k\overline{du}},$$

2) *Global Earliness Weight:* An alternative involves computing a single global marginal earliness cost for each order. This can be done using a Least Square Regression:

$$earl^G = \frac{\sum pc \cdot rd - n \cdot \overline{pc} \cdot \overline{rd}}{\sum rd^2 - n \cdot \overline{rd}^2},$$

where $\overline{pc}$ is the average procurement cost of non-dominated bid combinations for the order, and *rd* is their average release date.

The simplest possible release policy for the FCMAP problem involves releasing each order O_i at its earliest possible release date, namely $r_i^{earliest}$. We refer to this policy as an *Immediate Release Policy.* It might sometime result in releasing some orders too early and hence yield unnecessarily high procurement costs. Ow and Morton have suggested using what they refer to as an *Intrinsic Release Policy*, which amounts to releasing orders when their early/tardy priority $P_i(S_i)$ becomes positive. $P_i(S_i)$ can be viewed as the marginal cost incurred for delaying the start of order O_i at time *t*. As long as this cost is negative, there is no benefit to releasing the order. The tipping point, where $P_i(S_i) = 0$, is the order's intrinsic release date:

$$\hat{r}_i = dd_i' - du_i - k\overline{du}\ln\frac{earl_i + tard_i}{earl_i}.$$

Here again, one can use either the local or global earliness cost associated with an order. Intuitively, one would expect the global earliness cost to be more appropriate for the computation of an order's release date and its local earliness cost to be better suited for the computation of its priority at a particular point in time. This has generally been confirmed in our experiments. In Section 8, we only present results where priorities are computed using local earliness costs. We do however report results, where release dates are computed with both local and global earliness costs, as we have not found any significant differences between these two policies.

Rather than limiting ourselves to deterministic adaptations of Ow and Morton's dispatch rule, we have also experimented with randomized versions, where order release dates and priorities are modified by small stochastic perturbations. This enables our procedure to make up for the way in which it approximates procurement costs, sampling the search space in the vicinity of its deterministic solution. The resulting early/tardy search heuristic operates by looping through the following procedure for a pre-specified amount of time. As it iterates, the procedure alternates between the immediate and intrinsic release policies discussed earlier and successively tries a number of different values for the heuristic's look-ahead parameter *k*. The following outlines one iteration – i.e. with one particular release policy and one particular value of the look-ahead parameter.

1. For each order O_k, $k \in M = \{1,...,m\}$, compute the order's release date. When using the immediate release policy, this simply amounts to setting the order's release date $RD_k = r_k^{earliest}$. When using the intrinsic release policy, the order's

release date is computed as $RD_k = Max\{r_k^{earliest}, (1+\alpha)\times \hat{r}_k\}$, where α is randomly drawn from the uniform distribution [–dev_1, +dev_1] (dev_1 is a parameter that controls how widely the procedure samples the search space);

2. Dispatch the orders, namely let $t_0 = \underset{k\in M}{Min}\, RD_k$
   1) For all those orders O_k that have not yet been scheduled and whose release dates are before t_0, compute the order's priority at time t_0 as:
   $$PR_k(t_0) = (1+\beta)\cdot P_k(dd_k - du_k - t_0),$$
   where P_k is the early/tardy priority defined in (3) and β is randomly drawn from the uniform distribution [–dev_2, +dev_2] (dev_2 is a parameter that controls how widely the procedure samples the search space);
   2) Let order O_i be the order with the highest priority. Schedule O_i to start at time t_0;
   3) If all orders have been scheduled, then Stop. Else, let $t_1 = t_0 + du_i$ and t_2 be the earliest release date among those orders that have not yet been scheduled. Set $t_0 = Max\{t_1, t_2\}$ and repeat Steps 1-3.
   4) Compute the profit of the resulting solution. If it is higher than the best solution obtained so far, make this the new best solution.

A deterministic version of this procedure simply amounts to setting dev_1 and dev_2 to zero.

7. A Simulated Annealing (SA) Search Procedure

A second heuristic search procedure for the FCMAP problem involves using Simulated Annealing (SA) to explore different combinations of bids. Given a selection of non-dominated bid combinations $Bid\_Comb = \{Bid\_Comb_1, ..., Bid\_Comb_m\}$ - one combination per order, the procedure computes the release date r_i of each order O_i and sequences the orders, using the Apparent Tardiness Cost (ATC) dispatch rule first introduced in (Vepsalainen and Morton 1987). ATC is known to generally yield high quality schedules for the one-machine total weighted tardiness problem and has a $O\ (m \cdot \log m)$ complexity. As such it is an excellent estimator for the best solution compatible with a given selection of bid combinations. The following further details the SA procedure:

Step 1 – Initialization:

Set initial temperature $Temp = Temp_0$, and initial bid selection Bid_Comb^1 = $\{Bid\_Comb_1^1,...,Bid\_Comb_m^1\}$;

Use the ATC dispatch rule to build a schedule. Let $cost^1$ = $cost(Bid\_Comb^1, ST^1)$, where $ST^1 = \{st_1^1,...,st_m^1\}$ is the set of start times assigned by ATC to orders O_1 through O_m. Set Bid_Comb^{opt}=Bid_Comb^1 and $cost^{opt}$=$cost^1$.

Step 2 – Search:

Perform the following step N times:

Select Bid_Comb =$neighbor(Bid\_Comb^1)$ (randomly or through some heuristic), and compute $cost$ = $cost(Bid\_Comb, ST)$, where ST is the set of order start times assigned by the ATC dispatch rule;

If $cost^1 \geq cost \geq cost^{opt}$, set Bid_Comb^1=Bid_Comb;

Else if $cost>cost^1$ and $rand() \leq exp((cost^1\text{-}cost)/Temp)$, set Bid_Comb^1= Bid_Comb;

Else if $cost<cost^{opt}$, set Bid_Comb^{opt}=Bid_Comb^1=Bid_Comb.

If Bid_Comb^{opt} was not modified in the last N iterations, decrease the temperature $Temp = Temp \cdot \alpha$. Go to Step 3.

Step 3 – Termination Condition:

If Bid_Comb^{opt} has not been improved over the past K steps, then STOP and return (Bid_Comb^{opt}, ST^{opt}) as the best solution found by the procedure, otherwise go to Step 2.

In our experiments, parameter values were chosen as follows: $Temp_0$ =300, α =0.95, N=60 and K=40. The initial bid combination Bid_Comb^1 is randomly generated. Note also that the ATC dispatch rule is itself a parametric dispatch rule with a look-ahead parameter (Vepsalainen and Morton 1987). In our experiments, we systematically run ATC with values of the look-ahead parameter equal to 0.5, 1.0, 1.5, …, 6.0 and pick the best of the 12 solutions we have generated.

We have studied variations of this procedure that rely on different movesets. In particular, we have considered a one-bid moveset variation, where we modify the selection of a single bid (for a given component), and a two-bid moveset variation, where two bid selections (for two different components) are modified at once. For both types of movesets, we have also experimented with two ways of selecting moves:

- a random mechanism, where a move in the moveset is randomly selected, and
- an organized mechanism that replaces the bid(s) that reduce most the profit of the current solution (Bid_Comb^1, ST^1), namely, those bids for which $bp^{ij} + tard_i \cdot T_i$ is the greatest.

The experiments presented in Section 8 use the organized mechanism, as we found it to generally yield higher quality solutions than the random one. We have not found a great difference between variations of our procedure using one-bid movesets and two bid movesets.

8. Computational Evaluation

A number of experiments have been run to evaluate the impact of our pruning rules, the performance of our heuristic search procedures, and the benefits of our FCMAP model over traditional reverse auction models that ignore the manufacturer's finite capacity. These experiments are further detailed below.

Empirical Setup

Problems were randomly generated to cover a broad range of conditions by varying the distribution of bid prices and bid delivery dates as well as the overall load faced by the manufacturer. Results are reported for 2 groups of problems:

1. **Problems with 10 orders, 5 required components per order and 20 supplier bids per component:** These problems were kept small enough so

that they could be solved to optimality with our branch-and-bound algorithm. Key parameter values were drawn from the following uniform distributions:

- Order processing time: U[5,25]
- Order marginal tardiness cost: U[1,10]
- Order due dates: 2 distributions:
    1. Medium Load (***ml***) problems: U[100,300]
    2. Heavy Load (***hl***) problems: U[100,200]
- Component bid deliveries: 2 distributions:
    1. Narrow bid delivery distribution (***nd***): U[0,50]
    2. Wide bid delivery distribution (***wd***): U[0,100]
- Component bid prices: 2 distributions:
    1. Narrow bid price distribution (***np***): U[5,35]
    2. Wide bid price distribution (***wp***): U[5,65]

A total of 20 problems were generated in each category (ml/hl, nd/wd, np/wp), yielding a total of 160 problems.

2. **Problems with 50 orders, 5 required components per order and 20 supplier bids per component**: While these problems were too large to be solved with branch-and-bound (even with our pruning rules), they were used to validate results obtained on the smaller sets of problems. This includes, determining how our heuristic search procedures scale up and evaluating the benefits of our FCMAP model over traditional reverse auction models and policies that ignore the manufacturer's finite capacity – these latter being simply referred to below as "infinite capacity" policies. Key parameter values were drawn from the following uniform distributions:

- Order processing time: U[10,50]
- Order marginal tardiness cost: U[1,10]
- Order due dates:
    - Medium Load (***ml***) problems: U[500,1500]
    - Heavy Load (***hl***) problems: U[500,1000]
- Component bid deliveries: 2 distributions:
    1. Narrow bid delivery distribution (***nd***): U[0,300]

        ii. Wide bid delivery distribution (***wd***): U[0,500]

- Component bid prices: same 2 distributions as 10-order problems (***np/wp***)

A total of 20 problems were generated in each category for a total of 160 problems.

Note that order revenues are irrelevant, since the orders to be produced are fixed. In other words, all solutions admit the same overall revenue and overall profit is solely determined by the sum of tardiness and procurement costs associated with a given solution – see equation (1). Accordingly, we report overall costs rather than overall profits.

Distance From the Optimum

Tables 1 and 2 summarize results obtained on ten-order problems. The tables provide the average distance from the optimum of solutions obtained with different variations of our search heuristics across eight problem sets (four medium load problem sets in Table 1 and four heavy load problem sets in Table 2) with each problem set including a total of 20 problems. This distance was computed as: [*cost*(*solution*) – *cost*(*optimal_solution*)]/*cost*(*optimal_solution*). Standard deviations are provided between parentheses. Optimal solutions were obtained using the branch-and-bound procedure introduced in Section 5. Results are reported for the following techniques:

- **Infinite capacity**: This is a technique that reflects traditional reverse auction practices, where the manufacturer's capacity is ignored. Specifically, for each order and each component, the manufacturer selects the cheapest bid compatible with the order's due date. Orders are then scheduled according to the Apparent Tardiness Cost dispatch rule (Vepsalainen and Morton 1987), namely the same rule used in our Simulated Annealing procedure (See Section 7).
- **Finite capacity**: Results are reported for a number of variations of our search heuristics:
    - Simulated Annealing (SA) procedure: This is the procedure introduced in Section 7 with $Temp_0$=300, α=0.95, N=60 and K=40. The procedure

was run five times on each problem and we report both average performance and best performance over 5 runs.

- Early/Tardy (ET) Procedure: this is the early/tardy heuristic introduced in Section 6. Here again we report results for several variations of this heuristic:
  - *G-L* uses global earliness weights in its release policy and local earliness weights in its priority computations,
  - *L-L* uses local earliness weights for both release date and priority computations,
  - *Determ* is a deterministic variation of the Early/Tardy heuristic described in Section 7, namely $dev_1 = dev_2 = 0$,
  - *Random* is a stochastic variation of the same heuristic with $dev_1 = 0.3$ and $dev_2 = 0.3$. For comparison sake, the CPU time given to this heuristic was the same CPU time required by an average SA run in the same problem category,
  - *Hybrid* is a heuristic that runs SA once, ET/L-L/Random once and takes the best of the resulting two solutions.

Table 1. Percentage deviation from the optimum – 10-order, medium load problems (Standard deviations are provided between parentheses)

	Infinite Capacity	Finite Capacity						
		SA		ET				Hybrid
		Avg. of 5 Runs	Best of 5 Runs	L-L		G-L		
				Determ.	Random	Determ.	Random	
np/nd	48.56 (40.69)	29.41 (22.60)	17.91 (15.06)	6.99 (2.82)	<u>5.80</u> (3.11)	7.09 (2.80)	6.57 (2.57)	<u>4.52</u> (3.50)
wp/nd	41.78 (47.89)	33.04 (33.50)	21.47 (22.08)	12.89 (4.91)	<u>10.62</u> (4.42)	12.89 (4.91)	11.99 (3.80)	<u>7.51</u> (4.61)
np/wd	84.78 (82.08)	49.98 (35.36)	26.66 (15.00)	8.98 (3.42)	<u>6.94</u> (4.280	9.15 (3.23)	8.14 (2.92)	<u>6.42</u> (4.26)
wp/wd	62.10 (59.86)	45.11 (36.98)	31.44 (23.22)	15.05 (5.37)	<u>11.84</u> (5.13)	15.05 (5.37)	13.96 (5.45)	<u>9.11</u> (6.04)

Table 2. Percentage deviation from the optimum – 10-order, heavy load problems
(Standard deviations are provided between parentheses)

		Finite Capacity						
	Infinite Capacity	SA		ET				Hybrid
		Avg. of 5 Runs	Best of 5 Runs	L-L		G-L		
				Determ.	Random	Determ.	Random	
np/nd	56.94 (43.87)	35.84 (20.14)	26.54 (14.29)	8.58 (3.76)	7.10 (2.70)	8.51 (3.72)	7.21 (2.99)	6.80 (2.48)
wp/nd	30.05 (28.76)	25.47 (22.65	20.39 (17.10)	14.74 (7.73)	12.77 (5.30)	13.67 (6.34)	12.32 (5.50)	8.55 (6.69)
np/wd	147.58 (106.96)	65.36 (45.48)	37.54 (22.75)	9.52 (4.06)	8.52 (3.92)	9.23 (4.28)	8.10 (3.86)	7.52 (4.04)
wp/wd	129.72 (89.47)	70.05 (42.87)	39.80 (24.40)	16.20 (3.93)	13.62 (3.95)	15.41 (3.93)	13.73 (4.12)	12.23 (3.73)

Table 1 and 2 yield a number of observations:

- **Importance of the FCMAP Model:** All finite capacity heuristics yield solutions with significantly lower costs that the infinite capacity one, thereby confirming the importance of the FCMAP model. Taking into account finite capacity considerations when selecting supplier bids significantly improves the manufacturer’s bottom line. The results are most impressive in problem categories ml/np/wd and hl/np/wd, where our hybrid heuristics reduces total costs by as much as 92%.
- **Distance from the Optimum:** Our hybrid heuristics generally yields solutions that are within less than 10 percent across 7 out of the 8 problem categories
- **Effectiveness of Property 1:** Even deterministic versions of the early/tardy heuristic are within 16% of the optimum across all problem categories – and within less than 10 percent across several of them. This strongly suggests that Property 1 and the way in which our ET heuristic approximates earliness costs are rather effective. Note also that this deterministic version of our heuristic takes only a tiny fraction of a second on these problems.
- **ET heuristic versus SA heuristic:** Given the same amount of CPU time, the ET heuristic performs significantly better than the SA search procedure.
- **Global versus Local Earliness Weights:** The G-L and L-L variations of the ET heuristic seem to generally perform equally well, with L-L performing slightly

better on medium load problems and G-L performing slightly better on heavy load problems. Additional results not reported here show however that local earliness weights yield significantly better results than global earliness weights when it comes to priority computations. These results confirm our intuition that local earliness weight computations better capture the changing profiles of non-dominated bid combinations, while global earliness weights make more sense when it comes to dispatching decisions – especially in heavy load situations.

Computational Requirements

Our computational results suggest that the CPU time required by the SA procedure increases linearly with problem size, while that of branch-and-bound grows exponentially. By design, the CPU time allocated to the ET heuristic was set to be equal to that of the SA procedure. CPU times in Figure 6 were obtained using a 1GHz Pentium-III computer. CPU times on larger problems with 50 orders are reported in Figure 7. As expected, the infinite capacity heuristics is the fastest one, though SA and ET do not require more than 15 seconds on these larger problems – by design, their CPU times are identical and equal to half that of the hybrid heuristics.

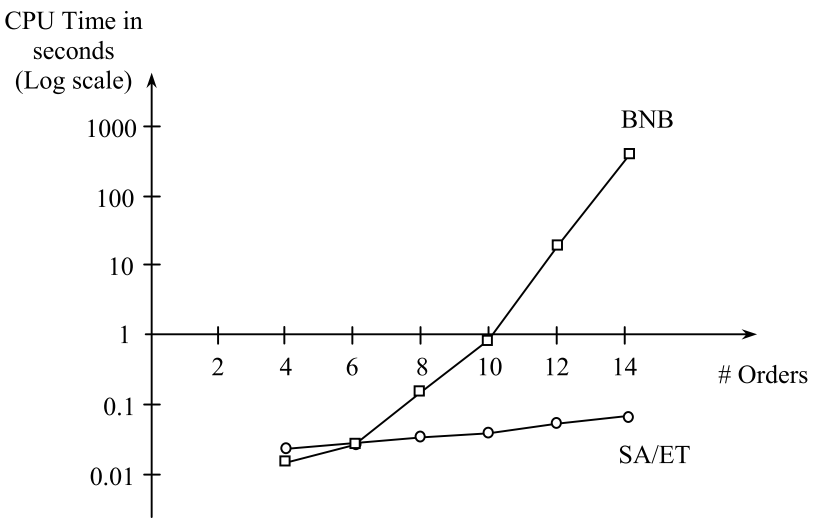

Figure 6. CPU time

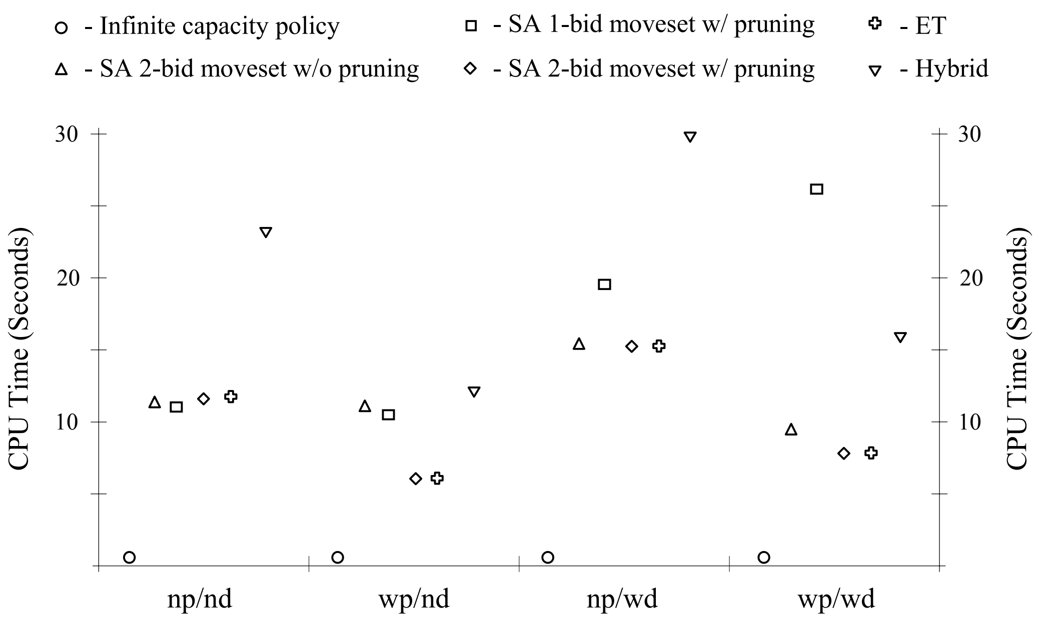

Figure 7. Overall CPU time (50 orders – medium load)

Impact of Ignoring the Manufacturer's Capacity on Larger Problems

Figure 8 and Table 3 summarize results evaluating the impact of ignoring the manufacturer's finite capacity on larger problems with 50 orders in medium load situations. Similar results for heavy load situations are provided in Figure 9 and Table 4. It can be seen that the ET and hybrid rules systematically yield the best results across all problem categories. A look at the cost breakdowns provided in Table 3 and 4 indicates that ET is capable of selectively sacrificing procurement costs to yield significant reductions in tardiness costs. On some problems, ET reduces overall costs by more than 88%. These results further validate the benefits of the FCMAP model advocated in this paper and the way in which our ET heuristic leverages Property 1.

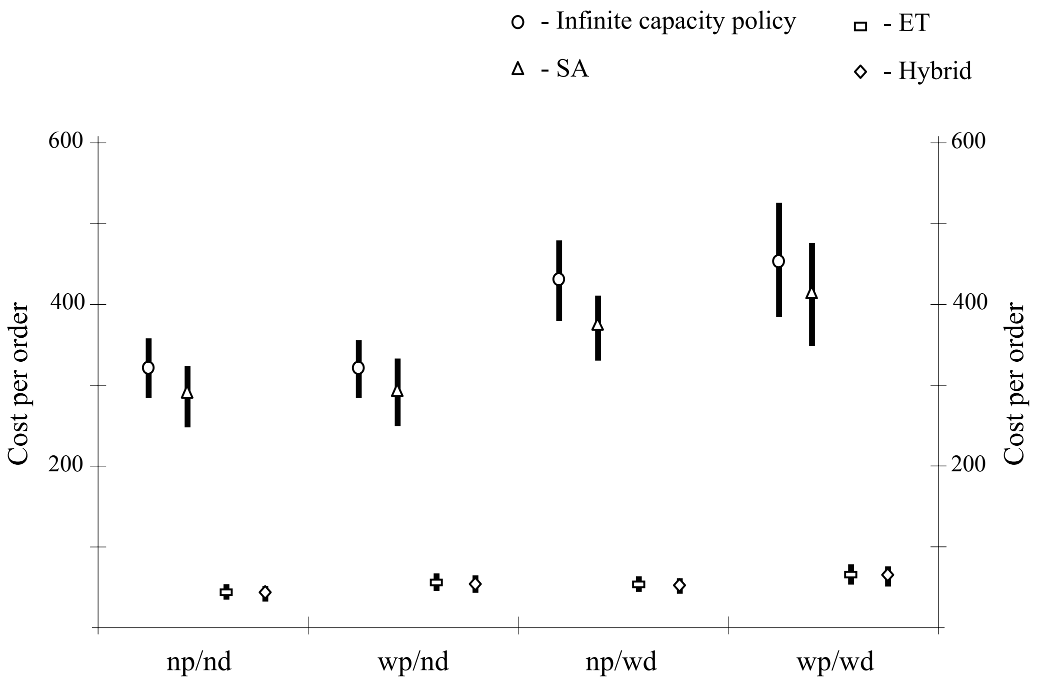

Figure 8. Infinite capacity policy vs. finite capacity policy – SA heuristic search, ET heuristic, Hybrid heuristic: 95% confidence intervals of overall cost per order (50 orders – medium load)

Table 3. Cost breakdown (50 orders – medium load)

		Procurement Cost	Tardy Cost	**Total Cost per Order**
np/nd	Inf. Cap.	30.11 (0.37)	297.85 (100.13)	**327.96** (100.06)
	ET	32.13 (1.05)	16.54 (18.86)	**48.67** (19.25)
wp/nd	Inf. Cap.	37.09 (0.75)	289.54 (97.88)	**326.63** (97.64)
	ET	40.71 (1.74)	20.71 (24.29)	**61.42** (24.86)
np/wd	Inf. Cap.	30.22 (0.28)	400.20 (127.02)	**430.42** (126.99)
	ET	35.01 (1.76)	17.79 (17.57)	**52.80** (18.63)
wp/wd	Inf. Cap.	36.97 (0.80)	421.17 (184.97)	**458.14** (185.02)
	ET	44.43 (3.46)	26.15 (32.25)	**70.58** (34.00)

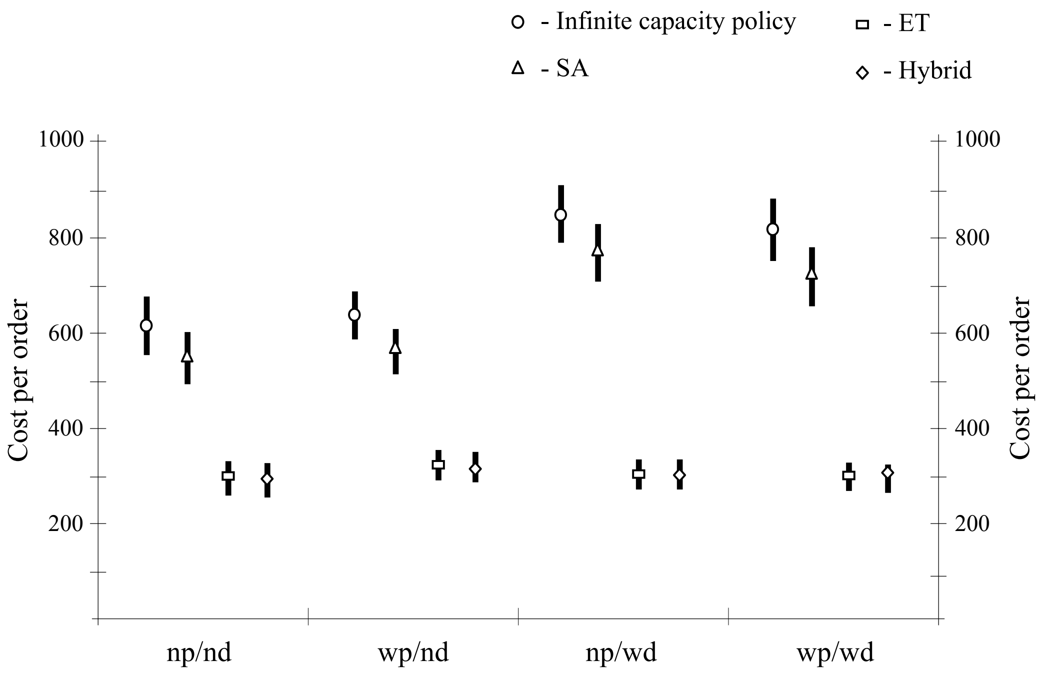

Figure 9. Infinite capacity policy vs. finite capacity policy – SA heuristic search, ET heuristic, Hybrid heuristic: 95% confidence intervals of overall cost per order (50 orders – heavy load)

Table 4. Cost breakdown (50 orders – heavy load)

		Procurement Cost	Tardy Cost	**Total Cost per Order**
np/nd	Inf. Cap.	30.11 (0.32)	594.79 (154.59)	**624.90** (154.61)
	ET	32.61 (0.97)	264.51 (92.42)	**297.12** (92.55)
wp/nd	Inf. Cap.	37.28 (0.91)	590.19 (112.63)	**627.47** (112.50)
	ET	42.28 (1.59)	278.53 (74.02)	**320.81** (74.01)
np/wd	Inf. Cap.	30.21 (0.58)	820.83 (150.77)	**851.04** (151.02)
	ET	35.46 (1.09)	268.97 (82.84)	**304.46** (82.42)
wp/wd	Inf. Cap.	37.44 (0.98)	783.39 (169.19)	**820.83** (169.42)
	ET	48.46 (2.79)	248.49 (70.85)	**296.95** (71.67)

9. Concluding Remarks

Prior work on dynamic supply chain formation has generally ignored capacity and delivery date considerations. In this paper, we have introduced a model for finite capacity multi-attribute procurement problems faced by manufacturers who have to select among supplier bids that differ in terms of prices and delivery dates. We have identified several dominance criteria that enable the manufacturer to quickly eliminate uncompetitive combinations of bids and have shown that the resulting problem can be modeled as an early/tardy problem with stepwise earliness costs. A branch-and-bound algorithm, a randomized early/tardy (ET) search heuristic and a Simulated Annealing (SA) procedure have been introduced to help the manufacturer select a combination of bids that maximizes its overall profit, taking into account its finite capacity as well as the prices and delivery dates associated with different supplier bids. We have shown that these procedures greatly improve over simpler infinite capacity bid selection models. Comparison with optimum solutions obtained using branch-and-bound, suggest that a hybrid heuristic that combines our ET and SA procedures generally yields solutions that are within 10% of the optimum.

It should also be noted that the model and techniques presented in this paper can easily be generalized to accommodate situations where the manufacturer can process multiple orders at the same time (non-unary capacity) or where the manufacturer incurs setup times for switching production between different product families. This is true for the pruning rules we introduced as well as the branch-and-bound procedure and two heuristic search procedures. At the same time, we have not attempted to evaluate our techniques on these problems and hence do not know, for instance, how far our heuristic search procedures would be from the optimum. It is also worth noting that our pruning rules also apply to situations where the manufacturer is modeled as a more complex job shop environment, where each order has to flow through a (possibly different) succession of machining (or service) centers. It can also be shown that our model and techniques can be extended to accommodate those problems with inventory holding costs, as long as orders are not allowed to be shipped before their due dates – this assumption corresponds to having finished goods inventory and is representative of many supply chain situations.

Future work will aim to refine our model in support of dynamic profitable-to-promise functionality, where the manufacturer needs to determine how to respond to requests for bids from prospective customers while also selecting among procurement bids from prospective suppliers (Arunachalam and Sadeh 2003, Sadeh et al. 2003).

Appendix 1. Additional Computational Results

Figures A1 and A2 compare the performance of 1- and 2-bid moveset variations of our SA search. As can be seen, the 2-bid moveset procedure consistently yields slightly better solutions across all 8 problem categories. In addition, it typically requires about half the CPU time of the 1-bid moveset procedure, suggesting that it is more effective at quickly converging towards better solutions (see Figure 7).

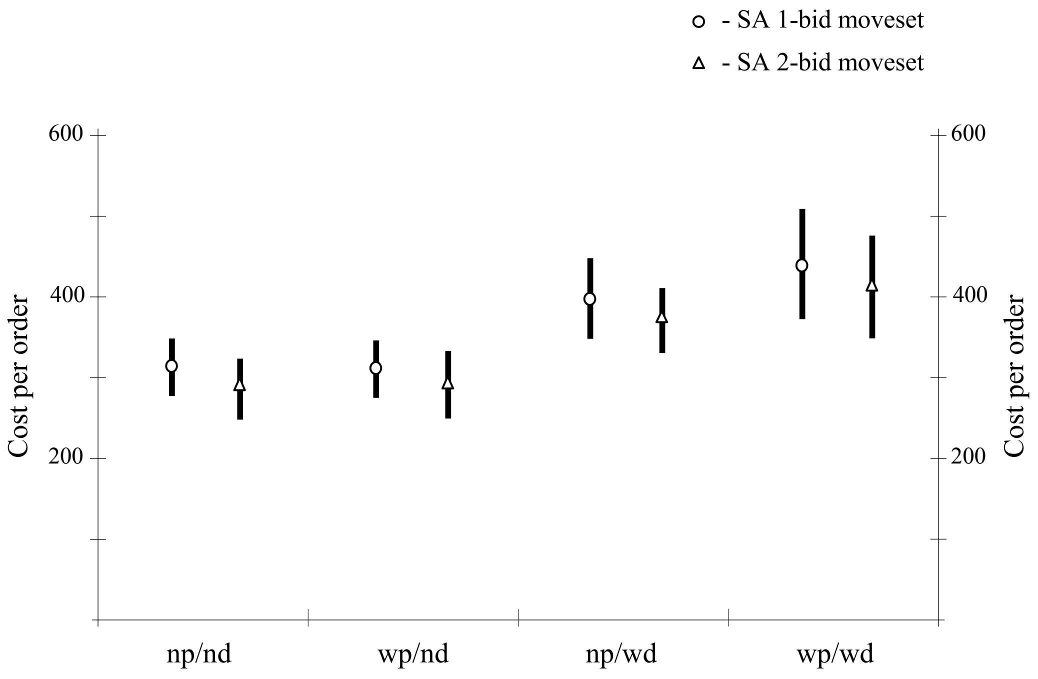

Figure A1. SA 1-bid moveset vs. 2-bid moveset (50 orders – medium load)

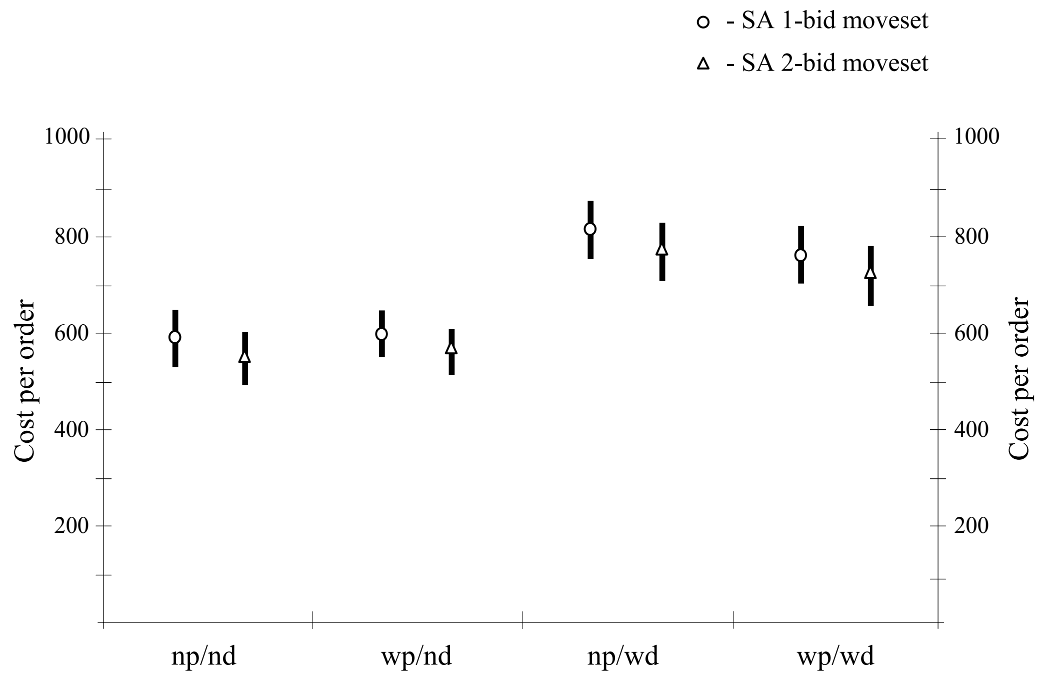

Figure A2. SA- 1-bid moveset vs. 2-bid moveset (50 orders – heavy load)

Appendix 2. ET – Parameter Sensitivity

Tables A1 and A2 show the sensitivity of our ET heuristic search procedure to different values of dev_1 and dev_2. The results are for small problems with 10 orders. The first line in each table entry reports distance of the best solution generated by the procedure from the optimum – the latter was obtained using the branch-and-bound procedure introduced in Section 5. Standard deviations are given between parentheses.

It can be seen that the procedure is not very sensitive to values of dev_1 and dev_2, with deviations amounting to less than 1 or 2 percent. Even deterministic variations of the heuristic, where dev_1=dev_2=0 often generate fairly good solutions.

Table A1. Sensitivity of the ET/LL search procedure to different values of dev1 and dev2 (ml/wp/wd).

	Dev1=0	0.1	0.2	0.3
Dev2 =0	15.13(5.36)	15.13(5.36)	15.13(5.36)	15.13(5.36)
0.1	14.77(5.27)	13.69(5.79)	13.68(5.92)	13.64(5.84)
0.2	14.57(5.06)	13.44(5.99)	13.08(5.86)	$\underline{11.13}$(5.75)
0.3	14.20(4.89)	13.21(5.43)	12.43(5.67)	11.84(5.13)

Table A2. Sensitivity of the ET/LL search procedure to different values of dev1 and dev2 (hl/wp/wd).

	Dev1=0	0.1	0.2	0.3
Dev2 =0	16.12(3.98)	16.12(3.98)	16.12(3.98)	16.12(3.98)
0.1	15.23(3.55)	15.21(3.58)	15.21(3.54)	14.65(4.19)
0.2	14.39(3.58)	14.38(3.59)	14.39(3.58)	14.08(3.60)
0.3	14.28(3.57)	14.03(3.99)	13.81(3.95)	$\underline{13.62}$(3.95)

Appendix 3: SA – Parameter Sensitivity

Tables A3 and A4 look at the sensitivity of our SA search procedure to different values of α, *K* and *N*. Results are reported as distance from the optimum on problems with 10 orders in category ml/wp/wd. Here again, it can be seen that the procedure is not extremely sensitive to the particular values selected for these parameters as far as solution quality is concerned. Higher values of K result however longer CPU times.

Table A3. SA search: distance from the optimum for K=40 (ml/wp/wd).

α \ N	**80**	**70**	**60**	**50**
0.98	-	39.87%	-	-
0.95	38.67%	-	<u>31.44 %</u>	32.32%
0.90	-	35.53%	-	-
0.85	34.59%	-	32.36%	32.68%

Table A4. SA search: distance from the optimum for α =0.95, N=60 (ml/wp/wd).

K	**50**	**45**	**40**	**35**	**30**	**25**
DFO	(Running over 1 hour with no result)	33.43%	<u>31.44 %</u>	31.57%	32.63%	34.38%

Acknowledgements

The research reported in this paper has been conduced as part of the MASCHINE project, a joint research effort between Carnegie Mellon University and the University of Michigan funded by the National Science Foundation under ITR Grant 0205435.

References

Arunachalam, R., N.M. Sadeh. 2003. Design of the supply chain trading competition. To appear in *Workshop on Trading Agent Design and Analysis*, *IJCAI-03, Acapulco, Mexico, August 2003*.

Babaioff, M., N. Nisan. 2001. Concurrent auctions across the supply chain. *ACM Conference on Electronic Commerce (ACM-EC'01), Tampa, FL, October 2001*. 1-10.

Collins, J., C. Bilot, M.L. Gini, B. Mobasher. 2001. Decision processes in agent-based automated contracting. *IEEE Internet Computing* **5** 61-72.

Collins, J., M. Tsvetovat, B. Mobasher, M. Gini. 1998. MAGNET: a multi-agent contracting system for plan execution. *Workshop on Artificial Intelligence and Manufacturing: State of the Art and State of Practice, Albuquerque, NM, August 1998*. 63-68.

Davis, R., R.G. Smith. 1983. Negotiation as a metaphor for distributed problem solving. *Artificial Intelligence* **20** 63-109.

Du, J., J.Y.T. Leung. 1990. Minimizing total tardiness on one processor is NP-hard. *Mathematics of Operations Res.* **15** 483-495.

Faratin, P., M. Klein. 2001. Automated contract negotiation as a system of constraints. *Proceedings of the Workshop on Distributed Constraint Reasoning, IJCAI-01, Seattle, WA, August 2001*. 33-45.

Gallien, J., L.M. Wein. 2002. A smart market for industrial procurement with capacity constraints. Submitted to Management Science.

Kjenstad, D. 1998. Coordinated supply chain scheduling. Ph.D. dissertation, Norwegian University of Science and Technology. http://www-2.cs.cmu.edu/~dag/.

Ow P.S., T.E. Morton, 1989. The single machine early/tardy problem, *Management Sci.* **35** 177-191.

Pinedo, M. 1995. *Scheduling: Theory, Algorithms, and Systems*, Prentice Hall, NJ.

Sadeh, N.M., R. Arunachalam, R. Aurell, J. Eriksson, N. Finne, S. Janson, 2003. TAC'03: a supply chain trading competition, *AI Magazine* **24** 92-94.

Sadeh, N.M., D.W. Hildum, D. Kjenstad, A. Tseng. 2001. MASCOT: an agent-based architecture for dynamic supply chain creation and coordination in the Internet economy. *J. Production Planning and Control* **12** 21-223.

Sandholm, T.W. 1993. An implementation of the contract net protocol based on marginal cost calculations. *The Eleventh National Conference on Artificial Intelligence (AAAI-93), Washington DC, July 1993.* 256-262.

Vepsalainen, A., T.E. Morton. 1987. Priority rules and lead time estimation for job shop scheduling with weighted tardiness costs. *Management Sci.* **33** 1036-1047.

Walsh, W.E., M.P. Wellman. 1998. A market protocol for decentralized task allocation. *The Third International Conference on Multi-Agent Systems, Paris, France, July 1998*. 325-332.