Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

An Auction-based Framework for Integrated Due Date Management

Chun Wang
Concordia Institute for Information Systems Engineering, Concordia University
Montreal, Canada
cwang@ciise.concordia.ca

Hamada H. Ghenniwa
Department of Electrical and Computer Engineering, University of Western Ontario
London, Canada
hghenniwa@eng.uwo.ca

Weiming Shen
Centre for Computer-assisted Construction Technologies, National Research Council Canada
London, Canada
weiming.shen@nrc.gc.ca

***Abstract*—This paper proposes an auction-based multilateral negotiation mechanism for collaborative integrated Due Date Management decision making. We assume a specific supply chain setting, in which a supplier supplies a group of customers. Due to limited production capacity within a scheduling time window (e.g. a season of the year), not all customers' due date requirements can be accommodated. This competition for the use of production resources among self-interested customers poses a strategic challenge in the design of due date management systems. To make the resource allocation efficient, we present an iterative auction framework to coordinate the allocation process and to align customers' self-interests with the overall system performance. Our purpose is to provide a decentralized decision support tool which enables the integration of key due date management decisions.**

I. INTRODUCTION

One of the major challenges facing organizations today is the demand for ever-greater levels of responsiveness and shorter defined lead times for deliveries of high-quality goods and services. Customers are looking for both a faster turnaround and on-time delivery, and these, rather than price, are becoming key differentiators. In order to gain an edge over their competitors, firms need to gear their management toward time-based competition, i.e. providing competitive and reliable lead times. However, shorter lead times are not always translated into profits. Given a firm's existing production and supply chain management processes, shorter lead times usually incur higher costs due to expediting the job. In many cases the premium that a customer is willing to pay to get a shorter lead time cannot compensate for the cost of expediting, and accepting the short-lead-time orders becomes unprofitable. One of the challenges in Due Date Management (DDM) is to strike a balance between shorter lead times and anticipated profits.

In supply chain management, the due date of an order is the promised date that the supplier will deliver the product(s). The task of DDM is to determine, in a timely manner, if and when an order can be profitably fulfilled. The main decisions in DDM are order acceptance, due date setting, and scheduling. As these decisions are tightly coupled, ideally, one would consider them simultaneously. However, given the complexities of DDM decisions, in practice, they are often made sequentially. A common practice in DDM is to simply promise the delivery for all orders after a given number of working days. Typically, the sales department would promise anything to get an order and then rely on plant managers to produce it. Although, the fixed lead time policy is simple and easy to implement, it has certain negative impacts. Since due dates are set without considering the characteristics of the order and the current status of the production, they may be unrealistic in terms of production scheduling, thereby worsening due date performance, leading to disappointed customers, and/or inflicting higher costs due to expediting. On the other hand, the due dates will be overstated when the demand is low and some customers may choose to go elsewhere. In addition, a large number of orders may be scheduled in a single period, which may lead to choppy capacity utilization.

Integrated DDM policies usually focus on internal due date performance, such as earliness, tardiness, or the number of early/tardy orders. These policies ignore the impact of quoted due-dates on the customers' decisions to place the orders. Works that consider order acceptance decisions usually assume that customers are indifferent as to when an order is completed (i.e., due-date indifferent) as long as it is within the specified deadline[2]; or model the probability of a customer placing an order as a decreasing function of the quoted due-date[3]. The above mentioned approaches do not assume iterative interactions between customers and the firm during the due date decision making process. Some solutions [6] incorporate the bargaining process into bilateral due date decision making. In their model, both the customer and the firm have a reservation tradeoff curve between price and due date, which is private information. Although, this model provides a negotiation mechanism in DDM, it is a bilateral-bargaining model, which is not practical for negotiation situations, where the firm needs to optimize the order selection, due-date setting and scheduling decisions across a batch of customers for a production time period. In the case of a group customers involved in the negotiation, a multilateral model is required. The purpose of this research is to develop an auction-based multilateral negotiation mechanism for collaborative integrated DDM decision-making.

The rest of the paper is organized as follows: Section 2 gives a brief description of the integrated due date management problem; Section 3 introduces the structure and components of the auction-based decentralized due date management framework; Section 4 describes a case study demonstrating the

978-1-4244-2794-9/09/$25.00 ©2009 IEEE

application of the auction-based framework to the due date management of a job shop; Section 5 analyzes computational evaluation results and Section 6 concludes the paper.

II. THE INTEGRATED DUE DATE MANAGEMENT PROBLEM

The Integrated DDM is a decision making process, which combines order selection, due-date setting, and scheduling decisions. The process involves two types of participants: the firm and the customers. The firm has limited manufacturing capacity that can be used to process job orders from customers. The primary objective of the firm is to maximize the market efficiency (the sum of the values on a solution across all customers). The customers have job orders to be processed by the firm. Each order has a release time, a "preferred" due-date, and a deadline. The value that a customer places on an order (the price that she/he is willing to pay) declines with each day's delay beyond the preferred due-date, until it reaches zero when it passes the deadline. The customers' value functions are their private information.

The Integrated DDM problem involves the allocation of the manufacturing resources of the firm to the orders so that all the deadline requirements are met and market efficiency is maximized. As an example, we present a hypothetic integrated DDM problem based on a case study from [6] as follows1.

As shown in Fig.1, a firm manufactures windows and doors for home builders as well as individual home owners. The products are customized based on the requirements from the customers, which may include different types and quantities, preferred due dates, and deadlines. In this setting, the integrated DDM problem facing the firm is to coordinate the decisions regarding which order to accept, at what price, and with what delivery date. In the following section, we will demonstrate how the decisions of the integrated DDM can be coordinated through the auction-based framework.

III. THE AUCTION-BASED FRAMEWORK

The auction-based framework contains three components: requirement-based bidding languages, constraint-based winner determination algorithms, and an iterative bidding protocol. The bidding languages allow an agent's bid to be expressed by a requirement of processing a set of jobs with constraints. The winner determination algorithm is designed to effectively solve the winner determination problems formulated using the bids expressed in the requirement-based languages. The iterative bidding protocol further reduces the computational complexities and adds the potential of accommodating dynamic changes during the auction process.

A. Requirement-Based Bidding Languages

In Integrated DDM, customers derive values based on how their jobs are scheduled according to their objectives. The quality of a schedule can be measured by time related parameters, e.g. completion times, tardiness. We propose requirement-based languages (L_R) for concise representation of customers' preferences in integrated DDM.

We first define the atomic bid (C-Bid) based on a domain specific characteristic. In many real world situations, a customer could be indifferent about the completion times as long as they are before a specific due date. If the due date cannot be met, the customer's value placed on the job decreases, say 20% every 24 hours for example, until the deadline is reached when the customer has no value on the job. Based on the fact that a customer's valuation often stays unchanged with a certain range of completion times, we define the atomic bid as follows to capture customers' preferences concisely.

C-Bid is a 4-tuple $\langle R^g, eft, lft, p \rangle$ where R^g is the requirement of scheduling a set of jobs which contain the natural descriptions of the jobs and job-related constraints in terms of the classical scheduling theory, *eft* is the earliest finishing time of the jobs and *lft* is the latest finishing time. *p* is the price that the customer is willing to pay for $eft < C^g_{\max} \le lft$ where $C^g_{\max}$ is the completion time (makespan) of a schedule for R^g, which is the measure of schedule quality. We refer the time period (eft, lft] in a C-Bid as a makespan range.

C-Bids can be connected by XOR connective as a XOR-C-Bid to represent values that a customer has on different makespan ranges.

XOR-C-Bid is a collection of C-Bids in which a customer can submit an arbitrary number of $\langle R^g, eft_i, lft_i, p_i \rangle$. p_i is the price that the customer is willing to pay for the completion of J^g with $eft_i < C^g_{\max} \le lft_i$. Any two C-Bids $\langle R^g, eft_i, lft_i, p_i \rangle$ and $\langle R^g, eft_k, lft_k, p_k \rangle$ are not overlapped, that is, either $lft_i \le eft_k$ or $lft_k \le eft_i$ is true. Implicit here is that the customer is willing to obtain at most one of these C-Bids. The following proposition shows the completeness of an XOR-C Bid in terms of representing the customers' valuations.

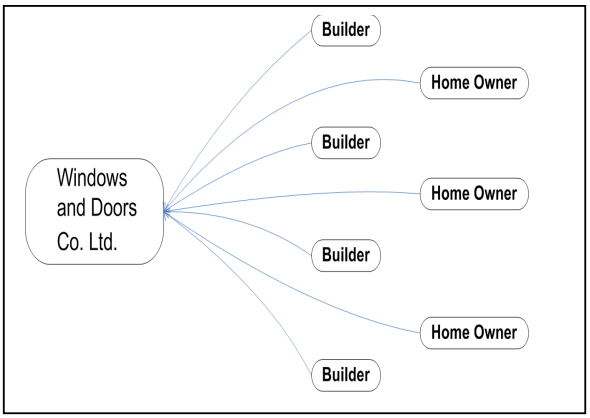

Figure 1. The DDN problem setting in a windows and doors company

[1] In [6], a case study, based on data collected from Gienow Windows and Doors Co. Ltd. (Calgary, Alberta) is presented to verify the effectiveness of an adaptive scheduling algorithm at shop floor level. While our scope is at supply chain level, we use this manufacturing setting to demonstrate the integrated DDM problem in mass-customization manufacturing environments.

Proposition 1 All customers' valuations for a set of jobs in integrated DDM can be represented by an XOR-C Bid.

Proof. Suppose that an arbitrary value function for a set of jobs is $v:\left(r^g,d^g\right]\rightarrow R^+$. For any makespan $c^g_{\max}$ ($r^g < c^g_{\max} \le d^g$), we can construct a unique C-Bid $\left\langle R^g, eft, lft, p\right\rangle$ where $lft = c^g_{\max}$, and $p = v\left(c^g_{\max}\right)$. By including C-Bids for all $c^g_{\max}$ ($r^g < c^g_{\max} \le d^g$), we can construct a XOR-C Bid in which each C-Bid bids the price that the customer is willing to pay (according to its value function) for completing the jobs at a specific time $c^g_{\max}$ ■

B. Constraint-Based Winner Determination

A branch bound algorithm for winner determination problems can branch on items [4] or bids [10]. Since distinct items are not modeled in requirement-based bidding languages, we propose an algorithmic framework which branches on bids. Branch-on-bids starts with an empty temporary solution and gradually adds bids to it along the search path. To detect the unfeasible branches at early stages, feasibility of the temporary solution needs to be checked when a new bid is added in at each node. For the combinatorial auctions with distinct items, feasibility checking is easy since as long as any two winning bids do not share an item, the solution is feasible. However, since requirement-based language is used in the auction, validating the feasibility of a solution is equal to answering the question: given a collection of jobs belong to different agents does a schedule exist that allocates the jobs on the resources, such that all constraints are satisfied? This decision problem is actually a job shop Constraint Satisfaction Problem which is known to be NP-complete [5]. For this problem, constraint-directed search algorithms, which are guided by the problem knowledge represented by constraints, provides strong results [1] . Given that branch-on-bids is now the fastest and most prevalent formulation for winner determination [10] and constraint-directed search has been experimentally proved to be effective in job shop constraint satisfaction[1], we argue that branch-on-bids embedded with constraint-directed feasibility validation can be a promising framework for winner determination in auction-based decentralized scheduling using requirement-based bidding languages.

The actual algorithm designed is a depth-first tree search. The search starts with an empty temporal schedule, called *TEMP* . Along the path *TEMP* is expended by adding more bids form AV , which is a set that constraints available (not winning) bids. The best *TEMP* found so far is $TEMP^*$. sum is the revenue of *TEMP* , which is the summation of prices of bids in *TEMP* , and sum^* be the revenue of $TEMP^*$. h is an upper bound on how much the bids in AV can contribute. The search is invoked by calling Branch-Bound-Scheduling (bids).

Algorithm 1

function Branch-Bound-Scheduling (bids) returns solution

$TEMP = \varphi$, $TEMP^* = \varphi$, and $sum^* = 0$

Recursive-Branching ($bids$, 0)

return $TEMP^*$

function Recursive-Branching (AV , sum)

if $sum > sum^*$ then $sum^* \leftarrow sum$, $TEMP^* \leftarrow TEMP$

if $AV = \varphi$ then return

$h \leftarrow \sum_{bid \in AV} priceOf(bid)$

If $sum + h \le sum^*$, then return

$bid \leftarrow$ Select-Unassigned-Bid (AV)

$TEMP = TEMP \cup \{bid\}$, $AV = AV - \{bid\}$

$sum \leftarrow sum + priceOf(bid)$

if Check-Feasibility ($TEMP$) returns pass

then Recursive-Branching (AV , sum)

$TEMP \leftarrow TEMP - \{bid\}$, $sum = sum - priceOf(bid)$

Recursive-Branching (AV , sum)

add bid to AV , return

The function CHECK-FEASIBILITY used in Algorithm 1 validates the feasibility of each node (a set of winner bids) along the search patch. This checking process is equivalent to solving a job shop constraint satisfaction problem. We have implemented the CHECK-FEASIBILITY function using a constraint-directed backtrack search procedure. Usually, a constraint-directed search procedure consists of propagators, heuristic-commitment techniques and retraction techniques. The feasibility validation algorithm integrates Constraint-Based Analysis (a propagator, developed in [1]), Precedence Constraint Posting (a commitment heuristic, developed in [11]), and chronological backtracking.

C. The Iterative Multi-Attribute Bidding Protocol

This section presents an iterative bidding protocol which utilizes the requirement-based bidding languages and constraint-based winner determination algorithms proposed in previous sections. We say that this is a multi-attribute bidding procedure because it allows the negotiation over a non-price attribute: the makespan of a schedule.

1) Bidding

Each customer has a valuation function expressing the values (prices) for different makespan ranges. We assume that agents have some common knowledge about the minimum reserve prices of their service requirements. Before the auction starts, an agent can calculate the reserve prices for each of the makespan ranges in its valuation and use them as the asking prices in the first round of the auction. At the beginning of each round, each customer selects the set of makespan ranges that maximize its utility function given the asking prices.

At round t a customer can submit an XOR-C-Bid that describes its utility maximization makespan ranges set and the prices it is willing to pay for the requirement R^g to be scheduled with different makespans. Note that all C-Bids in

the utility maximization XOR-C-Bid equally maximize the utility function of customer g given their ranges of makespans and the prices associated. That is for any two C-Bids at round t, $p_j - p_{bid,t}\left(eft_j, lft_j\right) = p_k - p_{bid,t}\left(eft_k, eft_k\right)$, where p_k is the agent's valuation on the makespan range $\left(eft_k, lft_k\right]$, and $p_{bid,t}\left(eft_k, eft_k\right)$ is the bidding price on $\left(eft_k, lft_k\right]$ at round t. Given an asking price, if the customer is included in the current provisional schedule, it repeats its bid for the makespan ranges in the current schedule; if the customer is not included in the current provisional schedule, two types of bids are allowed: (1) the customer can bid either at or greater than the asking prices; (2) the customer can also bid within ε below the asking price where ε is the minimum price increment in the auction. This is called "ε-discount" in [9]. However, if a customer takes this discount, the firm will never increase the asking price on the makespan range again and the customer is also forbidden from bidding at or greater than the asking price in future rounds.

2) Winner Determination

After receiving bids from customers, the firm solves the winner determination problem, computing a schedule that maximizes revenue. The firm first rejects bids from customers with invalid bid prices, and checks that customers repeat bids for makespan ranges received in the current provisional schedule. The new provisional schedule S_{t+1} solves:

$$\max_{S_{t+1}} \sum_{\substack{S^g \in S_{t+1} \\ eft_k < C_{\max}\left(S^g\right) \le lft_k}} p^g_{bid,t+1}\left(eft_k, lft_k\right)$$

where $p^g_{bid,t+1}\left(eft_k, lft_k\right)$ is the bidding price of customer g at round $t+1$ for the makespan range $\left(eft_k, lft_k\right]$.

3) Price Update

Prices in the proposed auction are discriminatory since it is reasonable to assume that customers' job requirements and valuation functions differ. There is a set of asking prices for every customer. Price increases to each customer are based only on bids received from that customer. If a customer is included in the current provisional schedule, the prices to this customer are not increased. For a customer that is not included in the current schedule, if the customer has bid the asking prices in the current round, the asking prices to the customer for the next round will increase by ε. On the other hand, if the customer has taken an "ε-discount" on a makespan range, the price for that makespan range will no longer be increased.

4) Termination

The auction terminates when all customers that bid are included in the current provisional schedule or all customers have submitted the same bids in two consecutive rounds.

IV. A Case Study

We demonstrate the integrated DDM decision-making facilitated by the auction-based framework in a setting similar to that depicted in Fig. 2. In this setting, a manufacturing firm produces a variety of products for its customers, which can be its upstream partners in a larger supply chain. The customers

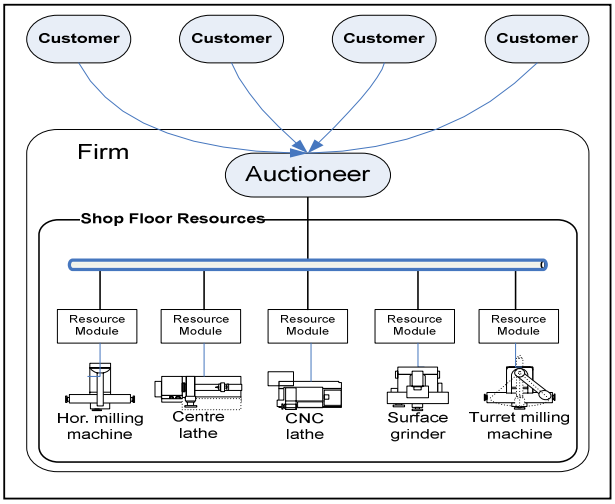

Figure 2. Integrated DDM in a job shop setting

have multiple valuations on the job orders. The firm needs to coordinate its decisions on the order acceptance, prices and the due dates of the orders, as well as on its internal manufacturing resource scheduling. We briefly describe the bidding process as follows:

- The auctioneer first collects the capability information of the firm's resources and their availability information within the time window to be scheduled. Then, it sends messages to all of its potential customers indicating that the firm is now open for receiving orders. The bidding process follows in an iterative manner.
- At the beginning of each round, customers send their orders to the auctioneer. As an order captures the requirements of the customer, it can be expressed by the requirement-based bidding language. While a customer may have multiple- valuation on his/her jobs, only those that maximize the utilities are included in the bid.
- Based on the bids and the resources' capability, capacity, and availability, the auctioneer computes a provisional schedule which includes the winning bids. The asking prices for losing bids are increased by a minimum increment.
- Losing customers can still bid in the subsequent rounds by adjusting their bids, i.e. gradually increasing prices or extending due-date requirements or a combination of both.
- If a provisional schedule includes all customers, or losing customers cannot extend their due-date requirements or increase prices anymore, the bidding terminates and the current provisional schedule is implemented.

Through this iterative bidding procedure, the auctioneer has coordinated the order selection, due-date setting, and scheduling decisions in the DDM through negotiation with

customers. Order selection decisions are made based on the overall capacity demand of job orders in the system, the due-date requirements of the job orders, and the current shop floor workload. If the due-date requirement of an order is too tight to be fulfilled profitably, the auctioneer will indicate this to the customer by the means of excluding the order from the provisional schedule in a round of the bidding process. On the other hand, if a customer cannot make the firm compromise with his/her deadline at an acceptable price, he/she will withdraw from the bidding process.

The uniqueness of this auction-based framework is that it unifies the exploration of the customers' due-date flexibility and integration of DDM decisions within a bidding procedure, which accommodates the self-interested nature of real world parties in supply chain environments.

V. COMPUTATIONAL EVALUATION

This section evaluates the auction framework through computational analysis. We use common assessing metrics in the literature and nine randomly generated problem sets to evaluate the performance of the framework.

A. Metrics and Problem Sets

We use three metrics Efficiency, Revenue, and Information Revelation as the performance measures in the evaluation. These metrics were developed in [8] for testing the performance of iBundle, an iterative combinatorial auction for general combinatorial auction problems. We redefine them in the context of DDM:

Efficiency of Scheduling, $eff(S)$, is measured as the ratio of the value of the final schedule S to the value of the optimal schedule that maximizes total value across the customers:

$$eff(S) = \frac{\sum_{g \in N,\, S^g \in S} v^g\left(S^g\right)}{\sum_{g \in N,\, S^g \in S^*} v^g\left(S^g\right)}$$

Revenue of Auction, $rev(S)$, is measured as the ratio of the auctioneer's income to the value of the optimal solution:

$$rev(S) = \frac{\sum_{g \in N,\, S^g \in S} p^g\left(S^g\right)}{\sum_{g \in N,\, S^g \in S^*} v^g\left(S^g\right)}$$

Information Revelation, $inf(g)$, for customer g is measured as the sum of the final price bid by the customer for all makespan-intervals in its valuation function, as a fraction of the sum of the true value of each makespan-interval.

$$inf(g) = \frac{\sum_{(eft,lft) \in Bid^g} p^g_{bid}(eft,lft)}{\sum_{(eft,lft) \in val\_fun^g} v^g(eft,lft)}$$

Where $p^g_{bid}(eft,lft)$ is the maximum bid from customer g for the makespan-interval $(eft,lft]$ during the auction; Bid^g is the set of makespan-intervals that agent g has placed bids on; and val_fun^g is the set of makespan-intervals with positive value in agent g 's valuation function. The overall auction information revelation, inf, is computed as the average information revelation over all agents. The auction often terminates before customers have revealed complete information about their values for makespan-intervals. The Information Revelation metric is designed to measure the extent to which a customer has revealed its value for each makespan-interval to the auctioneer during the auction.

We construct the multiple-makespan-interval problem sets by adding two more makespan-interval valuations to each problem instance of a randomly generated single makespan-interval set. The first makespan-interval added represents a delay up to 20% and the customer's value on the delayed makespan decreases 20%. The second makespan-interval added represents a 20% to 40% delay and, accordingly, the customer's value decreases by 40% on the delayed makespan. For example, if the single makespan-interval valuation of customer g on a set of jobs J^g can be represented as a CBid $\langle J^g, 8, 10, \$10\rangle$, the multiple-makespan-interval valuation of the customer can be represented as an XOR-CBid $\langle J^g, 8, 10, \$10\rangle XOR \langle J^g, 10, 12, \$8\rangle XOR \langle J^g, 12, 14, \$6\rangle$

B. Comparison Results

We compare the auction framework with a Generalized Vickery Auction (GVA) in which customers report their complete valuations over different makespan-intervals at the beginning of the auction and the auctioneer computes the optimal schedule for the customers. We have used the same constraint-based winner determination algorithm in both cases. Fig. 3 plots the efficiency of the iterative auction over the nine problem sets with a bid increment $\varepsilon = 4$. Compared to GVA (100% efficiency), on average, the auction can achieve more than 90% efficiency. Fig. 4 plots the Information Revelation performance of AdSCHE. Compared to GVA which requires 100% Information Revelation, the auction requires less than 50% at increment=2 and increment=4. Bigger increment values require slightly more Information Revelation. This makes sense because bigger increments may pass some low price equilibrium point which smaller increments may find.

Fig. 5 compares the run time increases between the iterative auction, and GVA as the number of bids increases from 6 to 8. We classify the problem sets into 3 groups as shown in (a), (b), and (c). All 3 groups demonstrate a similar run time increasing pattern as problem difficulty (number of bids) increases. On average, the iterative auction is more than 10 times faster than GVA with the cost of losing 6%-10% efficiency as shown in Fig. 3. We have set an 8000 second time limit. The computation times of GVA for half of the 8-bid instances in group (b) and all the 8-bid instances in group (a) have reached the limit. We did not test instances with more than 8 bids.

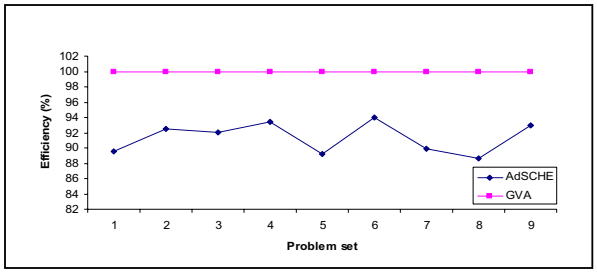

Figure 3. Efficiency performance of AdSCHE over 9 problem sets with bid increment

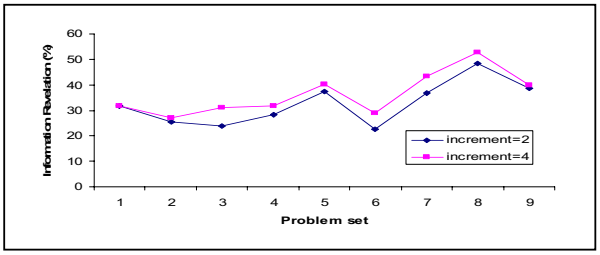

Figure 4. Information revelation performance of AdSCHE over 9 problem sets with bid increment $\varepsilon = 4$ and $\varepsilon = 2$

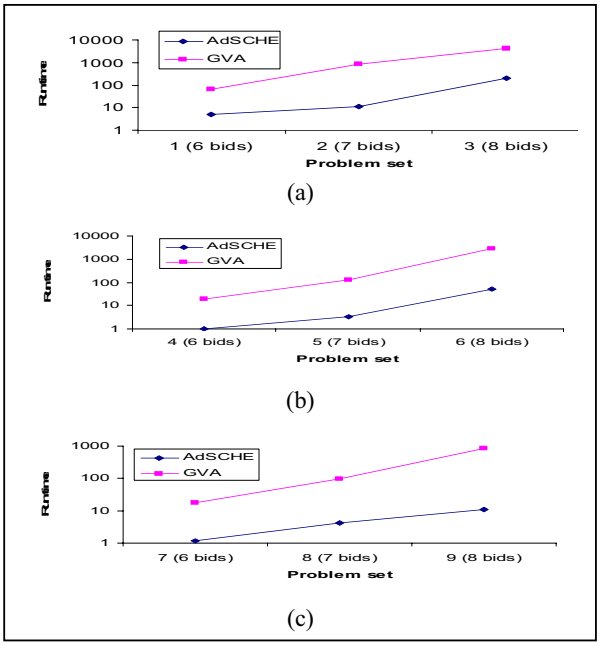

Figure 5. Run time (in seconds) of GVA and AdSCHE as the problem difficulty is increased. (a) problem sets #1, #2, and #3; (b) problem sets #4, #5, and #6; (c) problem sets #7, #8, and #9

VI. Conclusion

Due date management is not a new research topic. Many models and approaches have been proposed. However, these approaches have not taken advantage of the recent developments in automated electronic negotiation systems. In addition, they normally assume a non-strategic behavior of supply chain participants, which is not a typical case in today's globally competitive e-Supply Chain environment. The originality of the proposed approach is in its multilateral automated negotiation mechanism for due date management. It has a two-fold emphasis: 1) integrated due date management decision making and 2) multilateral automated negotiation. In addition to due date management in manufacturing, the approach is general enough to accommodate many other decentralized resource scheduling problems, in which task completion times need to be negotiated amongst participants

References

[1] J.C. Beck and M.S. Fox, "A Generic Framework for Constraint-Directed Search and Scheduling," AI Magazine, vol. 19, No. 4, pp. 101-130, 1998.

[2] J. Chuzhoy, R. Ostrovsky, and Y. Rabani, "Approximation Algorithms for the Job Interval Selection Problem and Related Scheduling Problems," Mathematics of Operations Research vol.31, no.4, pp.730-738, Nov. 2006.

[3] I. Duenyas and W.J. Hopp, "Quoting customer lead times," Management Science, vol.41, no.1, pp.43-57, 1995.

[4] Y. Fujishima, K. Leyton-Brown, and Y. Shoham, "Taming the Computational Complexity of Combinatorial Auctions: Optimal and Approximate Approaches," In Proceedings of 16th International Joint Conference on Artificial Intelligence (IJCAI-99), Stockholm, 1999.

[5] M.R. Garey and D. S. Johnson, Computers and Intractability, a Guide to the Theory of NP-Completeness, W. H. Freeman Company, 1979.

[6] W. Li, X. Luo, Y. Tu, and D. Xue, "Adaptive Production Scheduling for One-of-a-Kind Production with Mass Customization," Transactions of the North American Manufacturing Research Institution of SME, volume 35, 2007.

[7] D.R. Moodie and P.M. Bobrowski, "Due date demand management: negotiating the trade-off between price and delivery," International Journal of Production Research, vol. 37, no.5, pp.997-1021, 1999.

[8] D. C. Parkes, "iBundle: An Efficient Ascending Price Bundle Auction," In Proc. 1st ACM Conf. on Electronic Commerce (EC-99), pp. 148-157, 1999.

[9] D. C. Parkes and L. Ungar, "Iterative Combinatorial Auctions: Theory and Practice," In Proceedings of 17th National Conference on Artificial Intelligence, pp.74-81, 2000.

[10] T. Sandholm, S. Suri, A. Gilpin and D. Levine, "CABOB: A Fast Optimal Algorithm for Winner Determination in Combinatorial Auctions," Management Science, vol.51, no.3, pp.374-390, 2005.

[11] S.F. Smith and C. Cheng, "Slack-Based Heuristics for Constraint Satisfaction Scheduling," In Proceedings of 11th National Conference on Artificial Intelligence. Washington D.C., July 1993.

[12] N. Nisan, "Bidding languages for combinatorial auctions," Combinatorial Auctions, Cramton, Shoham, and Steinberg, eds., MIT Press, 2006.