Construction of Cost-Delivery Date Frontier for Capacitated Multi-Item Lot-Sizing System with Backorders and Integral Production Quantity

Seeronk Prichanont (prichano@cae.wisc.edu)
Department of Industrial Engineering, University of Wisconsin-Madison

Dharmaraj Veeramani (raj@ie.engr.wisc.edu) *
Department of Industrial Engineering, University of Wisconsin-Madison
266B Mechanical Engineering Building,
1513 University Avenue, Madison, WI 53706-1572, USA.
Tel: +1-608-262-0861; Fax: +1-608-262-8454

*Corresponding author

Construction of Cost-Delivery Date Frontier for Capacitated Multi-Item Lot-Sizing System with Backorders and Integral Production Quantity

Abstract

We consider a sourcing process in which price and delivery date are the two dominant decision attributes. Suppliers are allowed to submit a Price-Delivery Date Bid Frontier (PDDF) which contains multiple price-delivery date tuples. In order to develop the PDDF, a supplier needs to determine the Cost-Delivery Date Frontier (CDDF) specifying the minimum cost for each potential delivery date. It is our objective to quickly and accurately find a cost-delivery date tuple. A lot-sizing model with backorders and integral production quantity is used as a representation of the supplier's system. We propose an MIP-based heuristic to solve the problem. This heuristic is so general that it can easily be adapted to solve other MIP problems. Our computational results indicate that our heuristic performs better than other solving strategies employed by a commercial optimization software.

1 Introduction

We consider the sourcing decision system in which price and delivery date are the two dominant decision attributes. We take the delivery date as a relevant product attribute because it affects the production schedules in the supply chain and ultimately the cost. In response to customer's Request for Quotation (RFQ), suppliers are allowed to submit multiple price-delivery date options as a bid. The price-delivery date options presented by a supplier is called Price-Delivery Date Bid Frontier (PDDF). Figure 1 illustrates an example of a PDDF. After gathering PDDFs from suppliers, the customer decides if and which option from a supplier to be selected.

In order for a supplier to determine a price-delivery date option on the

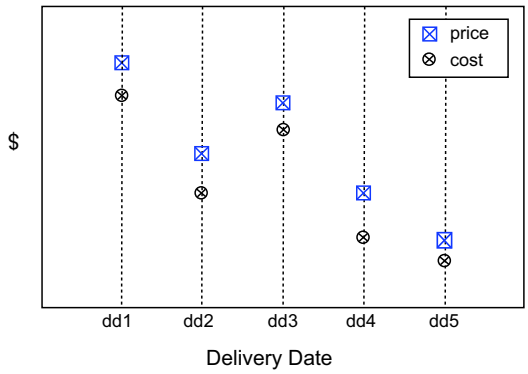

Figure 1: Price-Delivery Date Bid Frontier and Cost-Delivery Date Frontier

PDDF, she should be aware of her cost for that potential delivery date, or the cost-delivery date tuple. The collection of cost-delivery date tuples is called Cost-Delivery Date Frontier (CDDF). This research addresses the problem of how to quickly and accurately determine the CDDF, or equivalently, the cost-delivery date tuple, shown as a circle in Figure 1.

We consider a supplier's manufacturing facility by which products have to be produced in integral units. Such manufacturing systems are not uncommon to see. In the foundry industry, for example, melting metal is poured into moulds. Filling up only a fraction of a mould will just waste the raw material. Production plan has to be prescribed in such a way that the melting metal is just enough to fill up the desired number of moulds. Given a potential delivery date, it is this paper's objective to quickly find the optimal production plan that minimizes the total production cost.

This paper is organized as follows. Section 2 describes the mathematical formulation of the manufacturing system under consideration. Solving the mathematical model will determine the optimal production plan. However, the problem is difficult to solve, we therefore propose an MIP-based heuristic, called the Cut-Relax-Fix-and-Free Heuristic, in Section 3. Implementation issues and computational results are presented in Section 4. We conclude this paper with summary in Section 5.

2 Mathematical Formulation

Consider the situation in which a supplier receives an RFQ contains demand for product j' with the quantity of q' units. Suppose the supplier desires to find a cost-delivery date tuple $(t', z^*(t'))$ on the CDDF, where t' and $z^*(t')$ are the potential delivery date (period) and the corresponding minimum cost, respectively. Let $e_{j't'}$ be the supplier's existing *committed* demand quantity for product j' in period t'. The total demand for product j' in period t' is denoted by $d_{j't'}$, and can be calculated as:

$$d_{j't'} = e_{j't'} + q' \tag{1}$$

The total demand d_{jt}, where $j \neq j'$ and $t \neq t'$, can simply be expressed as:

$$d_{jt} = e_{jt} \tag{2}$$

The considered problem is called the Capacitated Multi-Item Lot-Sizing Problem with Backorder and Integral Production Quantity (CMBI). The notations used in this research are summarized in Table 1. Following is the MIP formulation representing CMBI.

$$\min \quad \sum_{j\in\mathbb{J}} \sum_{t\in\mathbb{T}} (f_j y_{jt} + h_j s_{jt} + b_j u_{jt}) \tag{3}$$

subject to

$$x_{jt} + s_{j(t-1)} + u_{jt} = d_{jt} + s_{jt} + u_{j(t-1)}, \qquad \forall j \in \mathbb{J}, \forall t \in \mathbb{T} \tag{4}$$

$$x_{jt} \leq (\frac{C_t}{p_j}) y_{jt}, \qquad \forall j \in \mathbb{J}, \forall t \in \mathbb{T} \tag{5}$$

$$\sum_{j\in\mathbb{J}} x_{jt} p_j, \leq C_t \qquad \forall t \in \mathbb{T} \tag{6}$$

$$\sum_{t\in\mathbb{T}} x_{jt} = \sum_{t\in\mathbb{T}} d_{jt}, \qquad \forall j \in \mathbb{J} \tag{7}$$

$$y_{jt} \in \{0, 1\}, \qquad \forall j \in \mathbb{J}, \forall t \in \mathbb{T} \tag{8}$$

$$s_{jt}, u_{jt} \geq 0, \qquad \forall j \in \mathbb{J}, \forall t \in \mathbb{T} \tag{9}$$

$$x_{j,t} \in \mathbb{Z}_+, \qquad \forall j \in \mathbb{J}, \forall t \in \mathbb{T} \tag{10}$$

The objective function (3) is to minimize the total setup costs, inventory holding costs, and late penalty costs. Equations (4) represent inventory balance constraints. The setup state variable y_{jt} is controlled by the logical constraints (5). Inequalities (6) limit the total production of a period. Constraints (7) ensure that, eventually, all demands will be fulfilled. Constraints (8) restrict the setup state variable y_{jt} to be binary, while constraints (9) enforce production, inventory and backorder levels to be nonnegative. Lastly, the non-negativity

Sets and Indices

Symbol	Description
$\mathbb{Z}_+$	$= \{0, 1, \ldots\}$, set of nonnegative integral numbers
$\mathbb{T}$	$= \{1, 2, \ldots, T\}$, set of production periods
$\mathbb{J}$	$= \{1, 2, \ldots, J\}$, set of products supplier can produce
t	time period in $\mathbb{T}$
j	product type in $\mathbb{J}$

Parameters

Symbol	Description
h_j	per unit per period inventory holding cost of product j
b_j	per unit per period backorder cost of product j
d_{jt}	units of demand for product j in period t
C_t	production capacity limit of period t (in units of time)
p_j	unit production time required to produce product j

Decision Variables

Symbol	Description
x_{jt}	units of product j produced in period t
y_{jt}	setup status for product j in period t; $y_{jt} = 0$ if product j is not produced in period t, $y_{jt} = 1$ otherwise
s_{jt}	nonnegative inventory of product j in period t
u_{jt}	nonnegative backordered units of product j in period t

Table 1: Notations used in this paper

and integrality properties are imposed on all of the production levels x_{jt}'s in (10). We call the solution set satisfying (4) – (10) as $\mathbb{X}^{CMBI}$.

Florian et al. (1980), Bitran and Yannasse (1982), and Wolsey (1998) show that lot-sizing problems with non-constant capacity restrictions are $\mathcal{NP}$-hard. Despite its prohibitive complexity, enormous research effort has focused on developing exact algorithms (e.g., Constantino (1996), Eppen and Marin (1987), and Thizy and van Wassenhove (1985)) as well as approximate algorithms (e.g., Hindi (1996) and Walser et al. (1998)). A survey and computational comparison of the heuristics developed for multi-item models can be found in Maes and van Wassenhove (1988) and Maes and van Wassenhove (1986), respectively.

In Prichanont and Veeramani (2002), the authors address the capacitated multi-item lot-sizing problem with backorders, called CMB, where production quantities do not have to be in integral units. There, a family of valid inequalities, called LSB inequalities, is introduced and three valid inequality selection strategies are proposed. With integrality constraints on production quantity, CMBI becomes even harder than CMB. To find an optimal solution for CMBI, even for a moderate-sized problem, can take tremendously long computing time. We therefore develop an MIP-based heuristic for CMBI and it will be presented in the next section.

3 Cut-Relax-Fix-and-Free Heuristic

For an IP or an MIP, it is clear that solving a relaxed problem is easier than solving its original one. Also, fixing some of the variables in the original problem using the relaxation's solution usually leads to good-quality solutions with much less computing time requirement. However, by fixing some or all of the variables, the problem's solution set becomes smaller. As a consequence, the solution obtained is likely to be suboptimal to the original problem.

Suppose that we are considering an MIP problem $\mathcal{P}$. Let $\mathbb{N} = \{1, \ldots, n\}$ be the index set for integer variables, and $\mathbf{x} = \{x_i : i \in \mathbb{N}\}$ be the set of integer variables. Also, we define new four sets $\emptyset = \mathbb{F}_A \subset \mathbb{F}_B \subset \mathbb{F}_C \subset \mathbb{F}_D = \mathbb{N}$. We

denote $\bar{\mathcal{P}}$ for the LP relaxation of $\mathcal{P}$, $\mathbf{x}^* = \{x_i^* : i \in \mathbb{N}\}$ and $\bar{\mathbf{x}}^* = \{\bar{x_i}^* : i \in \mathbb{N}\}$ for optimal solutions of $\mathcal{P}$ and $\bar{\mathcal{P}}$, respectively. Now, given positive integers δ_1 and δ_2, consider the following four variable fixing schemes after $\bar{\mathcal{P}}$ is solved.

Scheme A: set $\lfloor \bar{x_i}^* \rfloor - \delta_1 \leq x_i \leq \lceil \bar{x_i}^* \rceil + \delta_2$, for all $i \in \mathbb{F}_A$

Scheme B: set $\lfloor \bar{x_i}^* \rfloor - \delta_1 \leq x_i \leq \lceil \bar{x_i}^* \rceil + \delta_2$, for all $i \in \mathbb{F}_B$

Scheme C: set $\lfloor \bar{x_i}^* \rfloor - \delta_1 \leq x_i \leq \lceil \bar{x_i}^* \rceil + \delta_2$, for all $i \in \mathbb{F}_C$

Scheme D: set $\lfloor \bar{x_i}^* \rfloor - \delta_1 \leq x_i \leq \lceil \bar{x_i}^* \rceil + \delta_2$, for all $i \in \mathbb{F}_D$

Apparently, scheme A does not fix any of the integer variables ($\mathbb{F}_A = \emptyset$), while scheme D fixes all the integer variables ($\mathbb{F}_D = \mathbb{N}$). We denote $\mathbb{X}_i$ and z_i^* as the solution set and the optimal solution of $\mathcal{P}$ when the fixing scheme i is applied, respectively. Since $\mathbb{F}_A \subset \mathbb{F}_B \subset \mathbb{F}_C \subset \mathbb{F}_D$, it implies that $\mathbb{X}_D \subset \mathbb{X}_C \subset \mathbb{X}_B \subset \mathbb{X}_A$ and also $z_D^* \geq z_C^* \geq z_B^* \geq z_A^*$ (for minimization problem). However, in terms of the rate of convergence from the starting solution to the optimal solution, the problem with larger solution set tend to converge slower than the problem with smaller solution set. In other words, if t_i is the time used to solve $\mathcal{P}$ with fixing scheme i, we can generally expect that $t_D < t_C < t_B < t_A$. Regarding the starting solution, it is natural to hypothesize that larger the solution set worse the starting solution.

In Figure 2, we illustrate our hypothesis on the change of the objectives over time when problem $\mathcal{P}$ is solved using the four fixing schemes. There, we assume that we spend t_D time units to solve $\bar{\mathcal{P}}$. It can be seen that, by fixing more number of variables, the starting objective is lower. But over time, the problem with less number of variables fixed will reach better optimal solution. Hence, if we fix some of the variables in the beginning stage and remove the variable fixing constraints in later stage, we are likely to get high-quality solutions in the beginning without giving up the optimal solution in the end. Based on this idea, we develop the Cut-Relax-Fix-and-Free Heuristic (CRFF). The heuristic is so general that it can easily be adapted for any MIP problems.

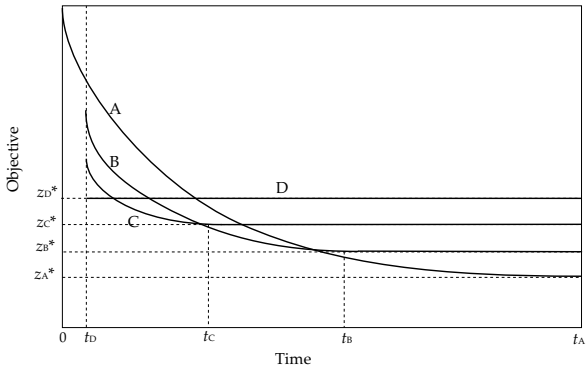

Figure 2: Hypothetical objective change over time

Let $(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{s}}, \bar{\mathbf{u}}) = (\bar{x}_{1,1}, \ldots, \bar{x}_{J,T}, \bar{y}_{1,1}, \ldots, \bar{y}_{J,T}, \bar{s}_{1,1}, \ldots, \bar{s}_{J,T}, \bar{u}_{1,1}, \ldots, \bar{u}_{J,T})$ be a feasible solution for CMB. We now introduce the Cut-Relax-Fix-and-Free Heuristic.

Cut-Relax-Fix-and-Free Heuristic:

Step 1 Select and add valid inequalities into CMB

Step 2 Solve CMB for t_1 seconds and obtain $(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{s}}, \bar{\mathbf{u}})$

Step 3 Convert CMB to CMBI and set $x_{jt} \geq \lfloor \bar{x_{jt}} \rfloor$, for all $1 \leq t \leq T_1$

Step 4 Solve the resulting CMBI using branch-and-bound algorithm for another t_2 seconds, and obtain z

Step 5 Remove the constraints added in Step 3, and set the cut off value (upper bound) at z

Step 6 Solve the resulting CMBI for another t_3 seconds

Four parameters that we need to assign values before running the heuristic are t_1, t_2, t_3, and T_1. With some acquaintance with the considered problem, the appropriate values for those parameters can be set. Also, at Step 1, we need to decide the valid inequalities to be added. The relaxed problem (CMB) is solved in Step 2. At this point, the trade-off between time and quality of the CMB solution have to be made. But generally, any close-to-optimal solution $(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{s}}, \bar{\mathbf{u}})$ would be satisfactory. At Step 3, because we set $x_{jt} \geq \lfloor \bar{x}_{jt} \rfloor$, the setup variable y_{jt} will be automatically affected through constraint (5). Note also, by setting the lower bound of x_{jt} at the floor value of $\bar{x}_{jt}$, the feasible solution will definitely be found if CMBI is in fact feasible. All of the constraints added in Step 3 are removed in Step 5. This enables the solution method to eventually achieve the true optimal solution if t_3 is sufficiently large. Also in Step 5, we set the cut off value at z. In doing so, all the nodes with higher LP objectives will be fathomed, and consequently, less number of nodes will need

to be processed. In the next section, we will examine the performance of CRFF and compare it with other solving strategies.

4 Implementation and Computation Results

It is desirable, of course, to have a solution algorithm that solves problem to optimality quickly. But most of the practical problems are complex and it is usually impractical to solve them to optimality. The solution quality, or the optimality, therefore, has to be traded off with the required computing time. Our goal in this section is to find out if our CRFF heuristic is really efficient in terms of being able to return good solutions quickly, comparing with other solving strategies provided by a commercially available optimization software.

In this research, problems are formulated in GAMS modeling language and solved by CPLEX's MIP solver. Some generic cuts can be produced by CPLEX. However, CPLEX has no ability to generate problem-specific cuts automatically. In this experiment, an 850 MHz Pentium III PC is used as a platform to run the experiments.

4.1 Solving Strategies Considered

We compare an implementation of Cut-Relax-Fix-and-Free Heuristic introduced in Section 3 with the three solving strategies employed by a commercial optimizer. In order to evaluate the four solving strategies, the objective values of the four strategies over time are plotted. Each strategy has an equal run time limit of 300 seconds to solve a problem. In sequence, we now describe CRFF implementation and other three solving strategies as follow.

4.1.1 CRFF(10,140,150)

In Prichanont and Veeramani (2002), the authors introduce the LSB inequalities for CMB. Note that the only difference between CMB and CMBI is that CMBI puts integrality constraints on production quantities, while CMB does not. It

is easy to see that the LSB inequalities are also valid for CMBI, hence the proof for the following proposition is omitted.

Proposition 1 *The inequalities*

$$x_{jt} \leq d_{jt} y_{jt} + s_{jt} + u_{j(t-1)}, \qquad j \in \mathbb{J}, t \in \mathbb{T} \tag{11}$$

are valid inequalities for $\mathbb{X}^{CMBI}$.

The results from Prichanont and Veeramani (2002) indicate that the solution strategy called No-C/All-V is the fastest. Therefore, at Step 1 of the CRFF, we implement No-C/All-V strategy. With No-C/All-V, there will be no CPLEX generated cuts added to the model while the LSB inequalities will be added to the model through the All-V strategy, which can be briefly described as follows.

Let RCMB be the LP relaxation of CMB with optimal solution $(\bar{x}^*, \bar{y}^*, \bar{s}^*, \bar{u}^*)$ $= (\bar{x}^*_{1,1}, \ldots, \bar{x}^*_{J,T}, \bar{y}^*_{1,1}, \ldots, \bar{y}^*_{J,T}, \bar{s}^*_{1,1}, \ldots, \bar{s}^*_{J,T}, \bar{u}^*_{1,1}, \ldots, \bar{u}^*_{J,T})$. The All-V valid inequality selection strategy follows two simple steps described below.

Step 1 Solve RCMB and obtain $(\bar{x}^*, \bar{y}^*, \bar{s}^*, \bar{u}^*)$

Step 2 Add to CMB all the LSB inequalities that $\bar{x}^*_{jt} > d_{jt}\bar{y}^*_{jt} + \bar{s}^*_{jt} + \bar{u}^*_{j(t-1)}$

The numbers 10, 140, and 150 shown in CRFF(10,140,150) represent the t_1, t_2, and t_3 (See CRFF description). In the experiment, we set $T_1 = 10$.

4.1.2 C-Root(300)

This strategy adds CPLEX generated cuts at the root node, then unless it finds optimal solution sooner, it solves the problem for 300 seconds. There will be no LSB inequalities added during the process.

4.1.3 C-Def(300)

This is a CPLEX's default solving strategy for any IP or MIP problems. CPLEX generated cuts are added at any node in the branch-and-bound tree the solver sees appropriate. Unless the optimal solution is found, the strategy is allowed to

run for 300 seconds before being terminated. There will be no LSB inequalities added during the process.

4.1.4 No-Cuts(300)

During the solving process, the strategy simply solves the original MIP formulation without adding any LSB or CPLEX generated cuts. The solving process stops if the optimal solution is found or otherwise when 300 seconds is reached.

4.2 Test Problems

We consider a manufacturing system being able to produce five product types ($J = 5$). Each unit of a product type j requires a constant processing time p_j. In the experiment, we pick p_j from a uniform distribution of the range $[\bar{p}_l, \bar{p}_h]$, where $\bar{p}_l$ and $\bar{p}_h$ are constant numbers and $\bar{p}_l < \bar{p}_h$. It is assumed that the system has already committed to some of demand in the past. The committed demand in a period t is non-negative, and it can be written as a linearly decreasing function over time

$$D(t) = \min\{0, D_1 - \zeta t\} \tag{12}$$

where D_1 and $D(t)$ are the total demands (in integral units) of all the five products in period 1 and period $t > 1$, respectively. The decreasing rate of demand is positive and it is represented by ζ. Please note that $D(t)$ can also be expressed as $D(t) = \sum_{j=1}^{J} d_{j,t}$. To assign a value to a d_{jt}, we pick a random number r_j in $(0,1)$ for each job j, then assign

$$d_{jt} = round(\frac{r_j}{\sum_{j=1}^{5} r_j} D(t)) \tag{13}$$

where the $round(\bullet)$ function returns the closest integer to the argument. Capacity limits (in units of time) of the system are selected from a uniform distribution of the range $[\bar{C}_l, \bar{C}_h]$, where $\bar{C}_l < \bar{C}_h$ and they are constants. For any product type j, the late penalty b_j, the holding cost h_j, and a setup cost f_j

Parameter	Value	Parameter	Value	Parameter	Value
D_1	80	$\bar{C}_l$	120	$\bar{h}_l$	1
δ	4	$\bar{C}_h$	200	$\bar{h}_h$	5
$\bar{p}_l$	1	$\bar{b}_l$	10	$\bar{f}_l$	30
$\bar{p}_h$	5	$\bar{b}_h$	50	$\bar{f}_h$	150

Table 2: Parameter values used in the experiment

are picked from uniformly distributed ranges $[\bar{b}_l, \bar{b}_h]$, $[\bar{h}_l, \bar{h}_h]$, and $[\bar{f}_l, \bar{f}_h]$, respectively. In our experiment, we assign values to all of the mentioned parameters and they are summarized in Table 2. We generate ten test problems to be used in the experiment.

4.3 Computational Results

Each of the ten test problems is solved with the four strategies described in Sections 4.1.1–4.1.4. At every ten seconds elapsed, we record the objective value. For a test problem, a graph showing objectives of all the strategies over time can be produced. While the graph could be useful in tracing the objectives of different strategies, it does not provide information regarding the optimality gap a solving strategy achieves.

Let $\bar{z}_{300}$ be the highest lower bound of the four strategies found after they stop (before or at 300 seconds). We compute ω, the percentage that an objective z is greater than $\bar{z}_{300}$, by

$$\omega = \frac{z - \bar{z}_{300}}{\bar{z}_{300}} \times 100 \tag{14}$$

Obviously, ω is an optimality gap between z and $\bar{z}_{300}$.

In Figure 3–12, we plot ω values of the four strategies against time elapsed for the ten test problems. At ten seconds, CRFF(10,140,150) finishes solving CMB and just starts solving CMBI. It is clear that, starting from 20 seconds until 300 seconds, CRFF(10,140,150) almost always achieves lower optimality gap than other strategies. The only two exceptions are problems in Figure 6 and 8, in which cases No-Cuts(300) and C-Root(300) just marginally achieve lower

ω values than what CRFF(10,140,150) does (i.e., 7.55 vs. 7.67 percent and 4.43 vs. 4.87 percent, respectively). While CRFF(10,140,150) performs consistently well, occasionally, other three strategies can perform really badly. For example, a problem in Figure 8, the ω values of C-Root(300) and C-Def(300) remain at the levels around 28 percent for 280 seconds, while during the same period CRFF(10,140,150)'s ω values stay just above five percent. An important and rather surprising observation though is that No-Cuts(300), in many instances, even outperforms C-Root(300) and C-Def(300). This could be an indication that, as opposed to LSB inequalities, CPLEX generated cuts are not usually helpful in solving CMBI.

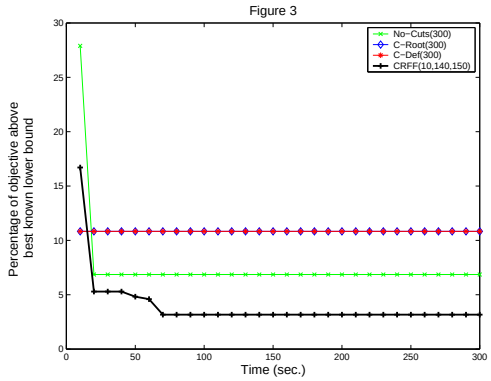
Figure 3
No−Cuts(300)
C−Root(300)
C−Def(300)
CRFF(10,140,150)
Percentage of objective above
best known lower bound
Time (sec.)

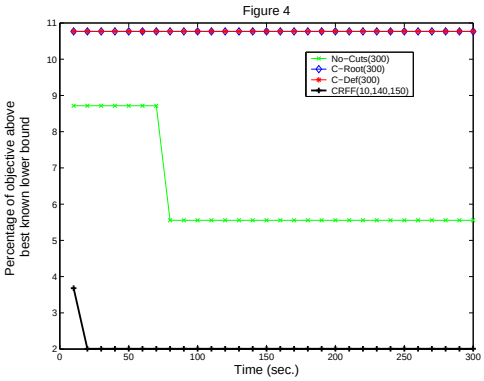
Figure 4
No−Cuts(300)
C−Root(300)
C−Def(300)
CRFF(10,140,150)
Percentage of objective above
best known lower bound
Time (sec.)

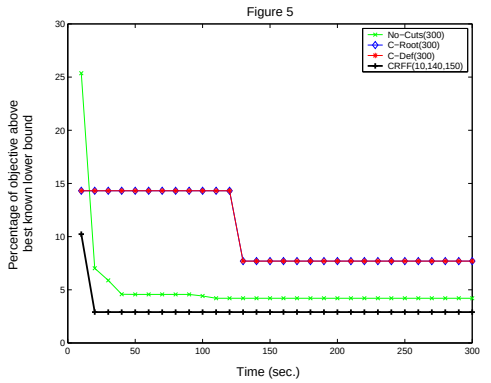
Figure 5
No−Cuts(300)
C−Root(300)
C−Def(300)
CRFF(10,140,150)
Percentage of objective above
best known lower bound
Time (sec.)

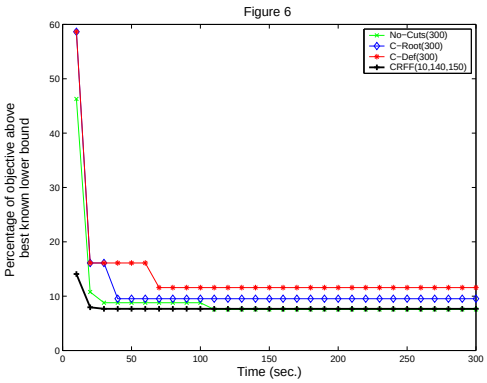
Figure 6
No−Cuts(300)
C−Root(300)
C−Def(300)
CRFF(10,140,150)
Percentage of objective above
best known lower bound
Time (sec.)

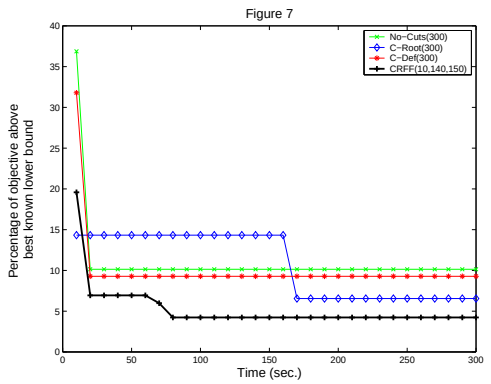
Figure 7
No−Cuts(300)
C−Root(300)
C−Def(300)
CRFF(10,140,150)
Percentage of objective above
best known lower bound
Time (sec.)

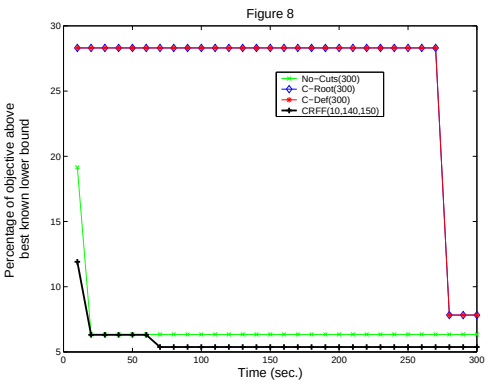
Figure 8
No−Cuts(300)
C−Root(300)
C−Def(300)
CRFF(10,140,150)
Percentage of objective above
best known lower bound
Time (sec.)

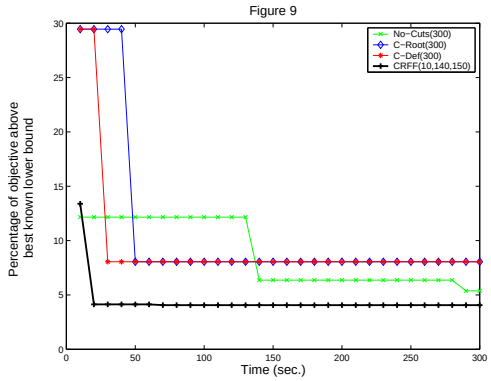
Figure 9
No−Cuts(300)
C−Root(300)
C−Def(300)
CRFF(10,140,150)
Percentage of objective above best known lower bound
Time (sec.)

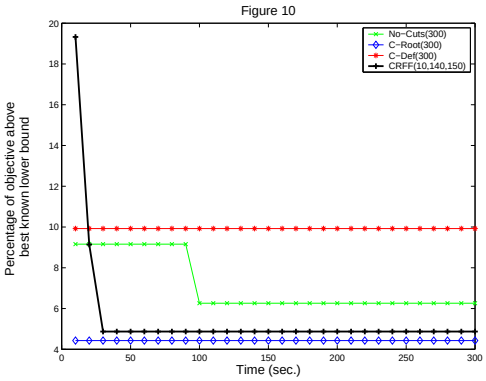
Figure 10
No−Cuts(300)
C−Root(300)
C−Def(300)
CRFF(10,140,150)
Percentage of objective above best known lower bound
Time (sec.)

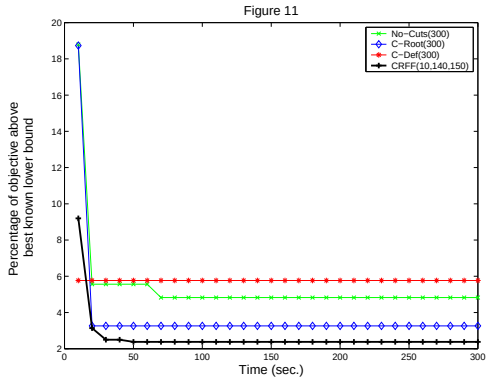
Figure 11
No−Cuts(300)
C−Root(300)
C−Def(300)
CRFF(10,140,150)
Percentage of objective above best known lower bound
Time (sec.)

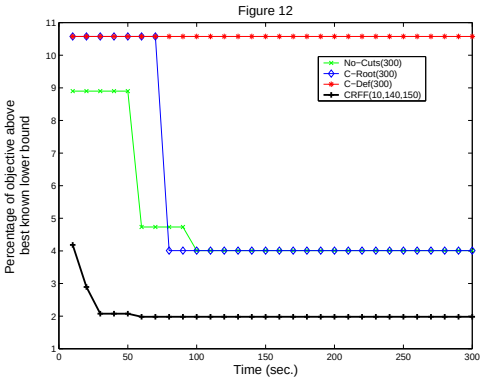
Figure 12
No−Cuts(300)
C−Root(300)
C−Def(300)
CRFF(10,140,150)
Percentage of objective above best known lower bound
Time (sec.)

5 Summary

Cost is usually one of the most important information in determining the price. Therefore, in the process of constructing the Price-Delivery Date Bid Frontier (PDDF), we first determine the Cost-Delivery Date Frontier (CDDF). This paper addresses the problem of how to quickly and accurately develop the CDDF. An MIP-based heuristic is proposed. From the computational results, in most cases, our proposed MIP-based heuristic achieves good solutions faster than other solving strategies employed by a commercial optimizer. For our future research, given that CDDF is known, we will investigate pricing strategies under different scenarios.

Acknowlegements

This material is based upon work supported by a National Science Foundation Grant (DMI 9800449) to the second author.

References

Bitran, G. R. and Yanasse, H. H. (1982) Computational Complexity of the Capacitated Lot Size Problem. *Management Science*, **28**, 1174–1186.

Constantino, M. (1996) A Cutting Plane Approach to Capacitated Lot-Sizing with Start-Up Costs. *Mathematical Programming*, **75**, 353–376.

Eppen, G. D. and Martin, R. K. (1987) Solving Multi-Item Capacitated Lot-Sizing Problems Using Variable Redefinition. *Operations Research*, **35(6)**, 832–848.

Florian, M., Lenstra, J. K., and Kan, A. H. G. R. (1980) Deterministic Production Planning: Algorithms and Complexity. *Management Science*, **26**, 669–679.

Hindi, K. S. (1996) Solving the CLSP by a Tabu Search Heuristic. *Journal of the Operational Research Society*, **47(1)**, 151–161.

Maes, J. and van Wassenhove, L. N. (1986) Muli-Item Single-Level Capacitated Dynamic Lot-Sizing Heuristic: A Computational Comparison (Part I: Static Case). *IIE Transactions*, **18**, 114–123.

Maes, J. and van Wassenhove, L. N. (1988) Multi-Item Single-Level Capacitated Lot-Sizing Heuristics: A General Review. *Journal of the Operational Research Society*, **39(11)**, 991–1004.

Prichanont, S. and Veeramani, D. (2002) Construction of Cost-Delivery Date Frontier for Capacitated Multi-Item Lot-Sizing with Backorder System, Unpublished manuscript, http://www.cae.wisc.edu/~prichano/cmb.ps

Thizy, J. M. and van Wassenhove, L. N. (1985) Lagrangean Relaxation for the Multi-Item Capacitated Lot-Sizing Problem: A Heuristic Implementation. *IIE Transaction*, **17(4)**, 308–313.

Walser, J. P., Iyer, R., and Venkatasubramanyan, N. (1998), In *Proceedings of the 15th National Conference on Artificial Intelligence, Madison, WI, USA*, pp. 373–379, AAAI

Wolsey, L. A. (1998), *Integer Programming* (Wiley-Interscience)