



**HAL**  
open science

## Enchère de type pondéré pour l'acquisition d'articles multiples

Kameshwaran Sampath, Lyes Benyoucef, Xiaolan Xie

► **To cite this version:**

Kameshwaran Sampath, Lyes Benyoucef, Xiaolan Xie. Enchère de type pondéré pour l'acquisition d'articles multiples. [Research Report] 2006, pp.30. inria-00122327v1

**HAL Id: inria-00122327**

**<https://inria.hal.science/inria-00122327v1>**

Submitted on 29 Dec 2006 (v1), last revised 16 Jan 2007 (v2)

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

# *Discount Auctions for Procuring Heterogeneous Items*

Kameshwaran S, Lyès Benyoucef, and Xiaolan Xie

N° ????

December 2006

\_\_\_\_\_ Thème NUM \_\_\_\_\_



*Rapport  
de recherche*





## Discount Auctions for Procuring Heterogeneous Items

Kameshwaran S, Lyès Benyoucef, and Xiaolan Xie

Thème NUM — Systèmes numériques  
Projet MACSI

Rapport de recherche n° ???? — December 2006 — 27 pages

**Abstract:** Advent of business over Internet have given rise to a number of innovative trading mechanisms. In this work we propose a new auction mechanism, called as *discount auctions*, for procuring heterogeneous items. The buyer, who is the auctioneer, has an unit demand for  $M$  distinct items. The suppliers, who are the bidders, specify individual costs for each of the items. In addition, a supplier also specifies a discount function: a non-decreasing function over the number of items. This discount bid, in essence, conveys the individual costs for each of the items and the discount that can be availed based on the number of items bought. The winner determination problem faced by the buyer is to choose the optimal set of winning suppliers and their respective winning items such that the total cost of procurement is minimized. First we show that this problem is  $\mathcal{NP}$ -hard upon reduction from the set covering problem. Next we propose two exact branch-and-bound based algorithms to solve the problem to optimality. Both these algorithms are different from the traditional branch-and-bound techniques. The first generation algorithm, called as *branch-on-supply* (BoS) does not use mathematical programming formulation but rather exploits the embedded network structure. The second generation algorithm, called as *branch-on-price* (BoP) uses a tight LP relaxation with valid inequalities as the lower bounding technique. However, instead of branching on a fractional integer variable, it branches on the current price of an item that is partially supplied by more than one supplier. The design philosophies of the above are different in the sense that BoS searches for the optimal number of items from the suppliers, whereas BoP searches for the optimal price of the items. We compare the performance of these algorithms with extensive computational experiments.

**Key-words:** e-procurement, auctions, integer programming, transportation problem, branch-and-bound, linear programming relaxation, duality, valid inequalities

## Enchère de type pondéré pour l'acquisition d'articles multiples

**Résumé :** L'arrivée de l'Internet dans le monde des affaires a provoqué la naissance d'un certain nombre de mécanismes de ventes (relations client-fournisseur) innovants. Dans cette étude, nous proposons un nouveau mécanisme d'enchère appelé *discount auctions* pour l'acquisition d'articles multiples (de types et de tailles différents). Un acheteur (client dans notre cas) exprime une demande unitaire pour chacun des  $M$  différents articles. Un ensemble de vendeurs (fournisseurs dans notre cas) vont chercher à répondre à ces demandes. L'offre d'un vendeur est composée du prix de chacun des articles séparément et d'une fonction de pondération qui indique la remise offerte sur la base du nombre d'articles achetés (plus d'articles achetés et plus la remise sur le prix de vente est importante). Le problème rencontré par l'acheteur est d'identifier, parmi l'ensemble des tous les vendeurs, les vendeurs dits vainqueurs ainsi que les articles achetés chez chacun d'eux en minimisant le coût d'achat total. Dans un premier temps, nous avons prouvé que ce problème d'optimisation est  $\mathcal{NP}$ -difficile en utilisant une réduction du problème dit de recouvrement. Le problème d'évaluation des offres proposés par les vendeurs est modélisé comme un programme linéaire en nombres mixtes (entiers et réels). Nous avons constaté que différentes structures du problème peuvent être exploitées pour concevoir des méthodes de résolution efficaces. Ainsi, deux méthodes optimales de types banch-and-bound ont été développés. La première, dite *branch-on-supply* (BoS), n'utilise pas la formulation mathématique du problème mais exploite sa structure de type réseau. A l'inverse, la deuxième, dite *branch-on-price* (BoP), utilise la relaxation du problème en un problème en nombre réels seulement (LP relaxation) avec l'introduction d'inégalités dites valides. Les philosophies de conception des deux méthodes sont décrites. Pour comparer les performances des deux méthodes, des expériences numériques sont présentées et analysées.

**Mots-clés :** e-procurement, enchères, programmation en nombres entiers, problème de transport, méthode de séparation et évaluation, relaxation, dualité, inégalités valides

## Contents

<b>1</b>	<b>Introduction</b>	<b>4</b>
<b>2</b>	<b>Discount Auctions</b>	<b>5</b>
2.1	Problem Definition . . . . .	5
2.2	Winner Determination Problem . . . . .	6
<b>3</b>	<b>Complexity of the Winner Determination Problem</b>	<b>7</b>
<b>4</b>	<b>Exploitable Structures</b>	<b>9</b>
4.1	Combinatorial Structure . . . . .	10
4.2	Network Structure . . . . .	10
<b>5</b>	<b>Linear Programming Relaxation</b>	<b>11</b>
<b>6</b>	<b>Branch-and-Bound Algorithms</b>	<b>12</b>
<b>7</b>	<b>Branch-on-Supply</b>	<b>14</b>
7.1	Candidate Problem and the Lower Bounding Technique . . . . .	15
7.2	Branching Strategy . . . . .	15
7.3	Heuristic . . . . .	16
7.4	BoS vs LP Relaxation based B&B . . . . .	16
<b>8</b>	<b>Branch-on-Price</b>	<b>16</b>
8.1	Branch-and-Cut . . . . .	17
8.2	A Tight Formulation with Valid Inequalities . . . . .	17
8.3	Branching Criteria and the Candidate Problem . . . . .	18
8.4	Heuristic Algorithm to Determine Incumbent Solutions . . . . .	20
<b>9</b>	<b>Computational Experiments</b>	<b>20</b>
9.1	Experimental Setup . . . . .	21
9.2	BoP Experiments . . . . .	22
<b>10</b>	<b>Conclusions and Future Work</b>	<b>24</b>

## 1 Introduction

Procurement is the process by which a company obtains materials and services necessary for its manufacturing and/or operations. Traditionally, the procurement process begins with the buyer sending a *request for quotation* (RFQ) to the potential suppliers. The RFQ contains details about the demands of the buyer and the suppliers respond with *bids*. The buyer evaluates the bids and chooses the suppliers, based on the purchasing policies. The dynamics of procurement process, involving suppliers responding with bids and the buyer evaluating the bids, borders on the *auction* mechanism. Auction is a market mechanism with well-defined set of rules for determining the terms of an exchange of something for money [22]. In auction parlance, buyer is the auctioneer and the suppliers are bidders in procurement. The auction mechanism is more structured than the procurement process. However, it forms the core of the many procurement scenarios. For example, consider a government tender for procuring a service like construction of a bridge. The constructors are the suppliers and they respond with sealed bids, quoting the cost of construction, among the other required things. Assuming that the cost is the only negotiable criterion, the constructor with the least quoted cost is chosen as the winner and he is supposed to provide the service at the quoted cost. This is the well known *first price sealed bid* auction, where the lowest bidder is awarded the contract for the quoted price.

The advent of Internet and Internet-based technologies have led to new and innovative auction mechanisms for procurement. Initially, in the past few years, naive use of Internet and information technologies saw complex back-end applications supporting supply chains of large companies, with simple front-end e-catalog systems supporting procurement. Recent trends are focusing on user friendly Business-to-Business e-procurement applications that embed sophisticated business logic and algorithms. With the Internet technologies enabling an interactive front end for human interaction and back end computers that can support complex computations, the research in e-procurement is focused on auction mechanisms, bidding languages, and bid evaluation techniques to make the process computationally and economically efficient. This has led to new generation of procurement techniques: *volume-discount* [9, 12], *combinatorial* [10, 14], and *multi-attribute* [2, 1, 18]. For a more general review of auction techniques in e-commerce and e-procurement, see [11, 4].

The auction mechanism used for e-procurement primarily depends on the type and number of items procured. In the above tender example, the demand was for a single indivisible item and the first price sealed bid auction was used. For an industrial procurement of large quantity of a single good (like raw material), volume discount auctions [12, 20, 19] are appropriate candidates. The bid submitted by a supplier is a cost *function* defined over the quantity. The function, in essence, can capture the discount offered by the supplier based on the quantity procured. Combinatorial auctions (CA) [8], are useful for procuring a set of heterogeneous, but related items. CA allows package bidding, that is, quoting a single cost for a bundle (subset) of items. In this way, the bidders can capture the *complementarity* or *substitutability* existing among the items in a bundle. For  $M$  items, a bidder could thus possibly submit  $2^M - 1$  combinatorial bids, one for each of the possible bundles. The volume discount and combinatorial auctions have led to several profitable industrial procurements

[14, 10, 3]. In this work, we propose a new auction mechanism called as *discount auctions* for procuring heterogeneous items.

The proposed discount auctions (DA) is applicable in scenarios where the items do not exhibit complementarity or substitutability. Consider the procurement of office supplies: *stationary, computers, and furniture*. A supplier has positive cost for each of the items and his profits may not change substantially by selling them separately or together. If he cannot sell the furniture in this auction, he can sell it elsewhere. This is not the case with items that exhibit complementarity. Bundles of items that cannot be split and items that cannot be bundled together are not uncommon in scenarios with complementarity and substitutability, respectively. Given that such conditions do not exist, the supplier is not concerned about *which* bundle of items, but rather about *how much* worth of items he can sell. The proposed DA is useful for such scenarios. The supplier has positive cost for each of the items, and in order to promote sales he gives incentive to the buyer by providing discounts on the number of items procured. The discount bid consists of two parts: (1) individual cost for each of the items and (2) discounts for different number of items (for example, if three items are bought then the discount is 5%, for four items 6%, etc.). The difference between DA and CA is obvious: the costs in the CA are defined over the subsets of the items whereas the discounts in the DA are defined over the cardinality of the subsets of the items.

In this work, we focus on the winner determination or bid evaluation problem of DA. The winner determination problem faced by the buyer is to choose a set of winning bids and a set of winning items for each of the winning bids, such that all demanded items are procured at minimal (total) cost. We show that this problem is  $\mathcal{NP}$ -hard and propose two branch and bound algorithms to solve the problem to optimality. The preliminary versions of this work have been presented in [16, 17].

The rest of the report is organized as follows. In Section 2, we formally introduce the DA and the winner determination problem is modeled as an integer program. The problem is shown to be  $\mathcal{NP}$ -hard, upon reduction from the set covering problem, in Section 3. The exploitable combinatorial and network structures of the problem are studied in Section 4. The linear programming relaxation of the problem and its relation to a network structure are discussed in Section 5. This network structure, in particular, the transportation structure is exploited in Section 7, to develop the first generation of branch-and-bound algorithm, called as *branch-on-supply*. Section 8 proposes the second generation algorithm, *branch-on-price* that uses an improved linear programming relaxation with valid inequalities. Computational experiments are presented in Section 9. Conclusions and future work are discussed in Section 10.

## 2 Discount Auctions

### 2.1 Problem Definition

The buyer is interested in procuring  $M$  different items. Each of the item is indivisible, *i.e.* it can be supplied by only one supplier. An *item* need not refer to a single unit. It can

be a computer or a computer and printer or hundred computers, but it cannot be split and supplied by multiple suppliers. Let there be  $N$  suppliers. An item is denoted by index  $m$  and a supplier by index  $j$ . Each supplier can submit only one discount bid and hence the index  $j$  is used to denote both the supplier and his bid. The *discount bid*  $j$  (i.e. from supplier  $j$ ) consists of two parts: (1) cost  $Q_j^m$  for each item  $m$  and (2) non-decreasing discount  $\theta_j^i$  for  $i$  ( $= 1, \dots, M$ ) number of items. The bid can be compactly expressed as an ordered pair of  $M$ -tuples:  $((Q_j^1, \dots, Q_j^m, \dots, Q_j^M), (\theta_j^1, \dots, \theta_j^i, \dots, \theta_j^M))$ . Note that  $m$  denotes a particular item and  $i$  denotes number of items. If the buyer procures items 2, 4, and 7 from bid  $j$ , then the cost of procurement is  $(1 - \theta_j^3)(Q_j^2 + Q_j^4 + Q_j^7)$ . The  $i = 3$  in  $\theta_j^3$ , as three items 2, 4, and 7 were procured. Note that this is different from the volume discounts, which are used in procuring multiple units of the same item. All the  $Q_j^m$  are positive (possibly infinite for an unavailable item) and the  $\theta_j^i$  are non decreasing over  $i$  (the discount cannot decrease with the number of items bought). One of the main issues in auction design is bid preparation and communication. In CAs, the number of possible combinatorial bids from a supplier is  $2^{M-1}$  (one bid for each subset). Thus both the bid preparation and communication (to the buyer) is costly. In DAs, only one discount bid is submitted from a supplier and its length is linear ( $2M$ ) in the number of items.

## 2.2 Winner Determination Problem

The winner determination problem (WDP) faced by the buyer is to choose a set of winning bids and a set of winning items for each of the winning bids, such that all demanded items are procured at total minimal cost. Without the discount function, the problem can be solved in  $O(NM)$  time (choose the minimum bidder for each of the items). Due to the discount function, the cost of an item bought from a bid depends on the total number of items bought from that bid. Hence, the formulation should also take into account the total number of items bought from a bid. If  $i$  items are bought from a bid  $j$ , then the cost of an item  $m$  is  $(1 - \theta_j^i)Q_j^m$ . With two different decision variables to choose *an* item  $m$  and the *number* of items  $i$  from bid  $j$ , the cost of an item would be a nonlinear function. To linearize the objective function, we define the *effective cost* of an item  $m$  if  $i$  items are bought from bid  $j$  as

$$p_j^{im} = (1 - \theta_j^i)Q_j^m \quad (1)$$

The WDP can be restated as choosing the items with minimal sum of effective costs subject to the demand and discount constraints. Following is an integer programming (IP) formulation for the bid evaluation problem.

$$\min \sum_j \sum_i \sum_m p_j^{im} w_j^{im} \quad (2)$$

subject to

$$\sum_i v_j^i \leq 1 \quad \forall j \quad (3)$$

$$\sum_m w_j^{im} = i v_j^i \quad \forall i, j \quad (4)$$

$$\sum_j \sum_i w_j^{im} = 1 \quad \forall m \quad (5)$$

$$w_j^{im}, v_j^i \in \{0, 1\} \quad \forall j, i, m \quad (6)$$

In the remainder of this report, the notation  $\forall j$  denotes  $j = 1, \dots, N$ . Similarly,  $\forall i$  and  $\forall m$  denote  $i = 1, \dots, M$  and  $m = 1, \dots, M$ , respectively. The binary decision variable  $w_j^{im}$  is to choose an item  $m$  from bid  $j$  with effective cost  $p_j^{im}$  and binary variable  $v_j^i$  is to choose  $i$  items from  $j$ . A constraints in (3) is a multiple choice constraint for variables  $v_j^i$  in  $i$  that determines the number of items  $i$  bought from  $j$ . If  $\sum_i v_j^i = 0$ , then no items are chosen from  $j$  and if  $v_j^i = 1$ ,  $i$  items are bought from  $j$ . Constraints (4) ensure that the items chosen from  $j$  are consistent with their effective cost: if  $i$  items are chosen, then they have the cost with discount for  $i$  items. Constraints (5) ensure that every item is procured from only one supplier. Thus the constraint sets (3) and (4) are for supply and constraint set (5) is for the demand.

### 3 Complexity of the Winner Determination Problem

**Theorem 1** *The WDP of the discount auctions is  $\mathcal{NP}$ -hard.*

**Proof:** We prove the hardness of the WDP by showing that the decision version of the WDP is  $\mathcal{NP}$ -complete upon reduction from the minimum set cover.

**Definition 1 ([DAuc])**

INSTANCE: Set of goods  $G = \{1, \dots, M\}$ , set of discount bids  $J = \{1, \dots, N\}$ , where a discount bid  $j \equiv ((Q_j^1, \dots, Q_j^M), (\theta_j^1, \dots, \theta_j^M))$  with  $Q_j^m \geq 0 \forall m \in G$  and  $0 \leq \theta_j^i \leq \theta_j^{i+1} \leq 1$ ,  $1 \leq i < M$ , and a goal  $K \geq 0$ .

QUESTION: Does there exist a winning set  $J' \subseteq J$ , which defines a partition  $P = \{B_j : B_j \subseteq G, j \in J'\}$  of  $G$ , such that the total cost of procurement  $\sum_{j \in J'} (1 - \theta_j^{|B_j|}) \sum_{m \in B_j} Q_j^m \leq K$ ?

**Definition 2 ([SCov])**

INSTANCE: Collection  $C$  of subsets of finite set  $F$ , positive weight  $w_R \forall R \in C$ , and a goal  $H \geq 0$ .

QUESTION: Does there exist a cover  $C' \subseteq C$  for  $F$  such that  $\sum_{R \in C'} w_R \leq H$ ?

The minimum set cover [SCov] is  $\mathcal{NP}$ -complete [13]. First we note that [DAuc] is in  $\mathcal{NP}$ : given a winning set  $J' \subseteq J$ , one can verify whether it defines a partition and the procurement cost is less than  $K$  in polynomial time. Let an instance of [SCov] be given. We construct an instance of [DAuc] in the following way:

- $G = F$ ,  $|J| = |C|$
- Create a bid  $j$  for each of the subset  $R \in C$  as follows:

$$Q_j^m = \left\{ \begin{array}{ll} w_R & \text{if } m \in R \\ \infty & \text{otherwise} \end{array} \right\}, \forall m$$

$$\theta_j^i = \frac{i-1}{i}, 1 \leq i \leq M$$

- $K = H$

The above reduction can be clearly done in polynomial time. We now show that the reduction is valid by showing that an instance of [SCov] is an *yes* iff if its reduction [DAuc] is an *yes* instance.

( $\Leftarrow$ ) Let there exist an *yes* instance of [DAuc] with  $J' \subseteq J$  defining a partition of  $G$  with procurement cost  $\leq K$ . A cover  $C'$  for [SCov] can be constructed as follows. For every  $j \in J'$ , include the corresponding subset  $R$  in  $C'$ . Note that  $B_j \subseteq R$  as  $m \notin R$  implies  $Q_j^m = \infty$ . The cost of procurement from winning bid  $j$  is given by

$$\begin{aligned} & (1 - \theta_j^{|B_j|}) \sum_{m \in B_j} Q_j^m \\ &= \left( \frac{1}{|B_j|} \right) |B_j| w_R \\ &= w_R \end{aligned}$$

Thus the cost of procurement from each bid is equal to the weight of the corresponding subset in  $C'$ . Since the winning bids partition  $G$ , the collection  $C'$  covers  $F$  with cost  $\leq K = H$ .

( $\Rightarrow$ ) Let there exist an *yes* instance for [SCov] with cover  $C'$ . The solution to [DAuc] can be constructed as follows. For every subset  $R \in C'$ , include its corresponding bid  $j$  in  $J'$ . Since  $C'$  covers  $F$ ,  $J'$  also covers  $G$ . If an item is supplied by more than one supplier then it can be removed from its respective suppliers except one. Note that removing an item from bid will not change the cost because of the assumed discount and cost structure. Hence, we have a partition of  $G$  that satisfies the goal.

**Proposition 1** *Following special cases are solvable in polynomial time:*

1. *Same Cost:*  $Q_j^m = Q_j^{\geq} 0, \forall j, m$
2. *Same Discount:*  $\theta_j^i = \theta_j^*, \forall j, i$
3.  $|M| \leq 2$

(1) This special case is a multi-unit auction of a single good. The requirement of the buyer is  $M$  units of a single good and the suppliers submit a bid with unit cost  $Q_j^*$  and a discount function. The WDP can be formulated as the following integer programming problem:

$$(P) : \quad \min \sum_j \sum_i (1 - \theta_j^i) i Q_j^* v_j^i \tag{7}$$

subject to

$$\sum_i v_j^i \leq 1, \forall j \tag{8}$$

$$\sum_j \sum_i v_j^i = M \quad (9)$$

$$v_j^i \in \{0, 1\}, \forall j, i \quad (10)$$

The above formulation is a *multiple choice knapsack problem* [26] with (8) as the multiple choice constraints and (9) as the knapsack constraint. Though the generic multiple choice knapsack problem is  $\mathcal{NP}$ -hard, the above problem is solvable in linear time due to its cost structure. We show this using the duality theory. Let  $\beta_j$  be the dual variable for the multiple choice constraint for  $j$  in (8) and  $\pi$  be the dual variable for the knapsack constraint (9). The linear programming dual of the above problem is:

$$(D): \quad \max \pi M + \sum_j \beta_j \quad (11)$$

subject to

$$i\pi + \beta_j \leq (1 - \theta_j^i) i Q_j^*, \forall j, i \quad (12)$$

$$\beta_j \geq 0, \forall j \quad (13)$$

Let  $j' = \arg \min_j \{(1 - \theta_j^M) M Q_j^*\}$ . Assign  $\beta_{j'} = (1 - \theta_{j'}^M) M Q_{j'}^*$ , and  $\beta_j = 0, \forall j \neq j'$  and  $\pi = 0$ . This is a feasible dual solution with objective value  $\beta_{j'}$ . A feasible solution to the primal problem can be constructed from this dual solution. Assign  $V_{j'}^M = 1, V_{j'}^i = 0, \forall i \neq M$ , and  $V_j^i = 0, \forall j \neq j'$  and  $\forall i$ . This is a feasible primal solution with the same objective value as of the dual and hence by duality theory it is an optimal solution to the linear relaxation. This optimal solution to the linear relaxation is integral and hence it is also optimal to the original integer programming problem. Thus the optimal solution to this WDP is  $\min_j (1 - \theta_j^M) M Q_j^*$ , which can be solved in linear time.

(2) If the discounts are the same for each bidder, then it is equivalent to no discount, which can be solved in polynomial time ( $O(MN)$ ) by choosing the bidder with minimum cost for each item.

(3) If  $M = 1$ , then WDP can be solved in linear time by choosing the bid with minimum cost. For  $M = 2$ , there are only two possibilities: buy at most one item from each buyer and buy both the items from a single buyer. The first problem is an assignment problem and the second is choosing the minimum cost for both items from  $N$  bidders. The optimal solution is the one with the minimum cost among the above two solutions.

## 4 Exploitable Structures

Knowledge of a structure in a problem can be exploited in designing the solution techniques and algorithms for solving the problem. The WDP is explored in this section for such structures in terms of both problem definition and formulation.

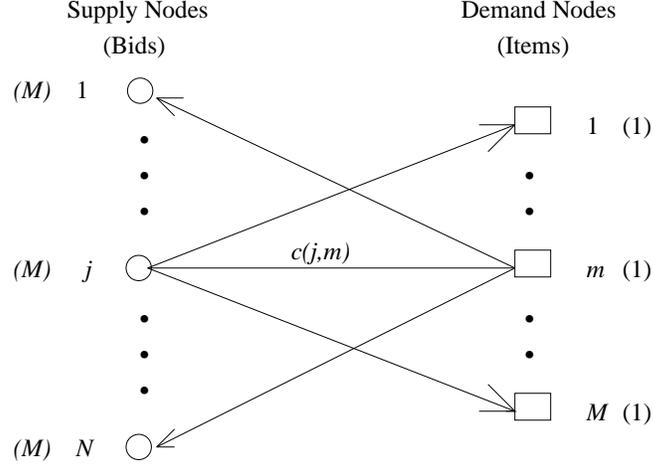


Figure 1: Embedded network structure

#### 4.1 Combinatorial Structure

In CA, the bidder provides a single price for a subset of items, and he can submit different bids for different subsets. The bids submitted in DA can be easily converted into combinatorial bids. A combinatorial bid for bundle (subset)  $B$  from supplier  $j$  has a single price  $P(B, j)$ . From a discount bid  $j$ , the cost of a combinatorial bid for each bundle  $B$  can be easily obtained as follows:

$$P(B, j) = (1 - \theta_j^{|B|}) \sum_{m \in B} Q_j^m \quad (14)$$

Thus given the discount bids, one can construct combinatorial bids. The solution of the bid evaluation problem for the combinatorial bids is also the solution to that of the discount auction. The restriction of choosing at most one bundle from each supplier need not be imposed, as there will always exist an optimal solution with at most one bundle from a supplier due to the non-decreasing discount function. The winner determination algorithms for CAs are well studied [27, 21] and hence one can use them for solving the discount auctions. However, the DA is only an instance of the CA, and hence it is possible to develop more efficient solution techniques by exploiting the other possible structures.

#### 4.2 Network Structure

The WDP of DA can be considered as a transportation network with  $N$  supply nodes (bids) and  $M$  demand nodes (one for each item). Each supply node has a supply of  $M$  units and each demand node has a unit demand. The embedded transport structure is shown in Figure 1. A flow in the network connecting node  $j$  to node  $m$  indicate that bid  $j$  is supplying item

$m$ . Due to the unit demand at each demand node, the flow in any given arc will be at most one. The complicating feature of the model is the cost  $c(j, m)$  of the flow in the arc  $(j, m)$ , which is the function of number of units supplied from node  $j$ . Note that this is different from the conventional nonlinear cost network models, where the cost will vary based on the flow through the arc, whereas in this case the cost varies on the total flow from the supply node. Let  $\delta_j$  be the total supply from node  $j$  in a solution. Then the cost of the flow in an arc  $(j, m)$  is given by  $c(j, m) = (1 - \theta_j^{s_j})Q_j^m$ . The solution is feasible only if the total supply  $\sum_j \delta_j = M$ . It is worth noting that for a given feasible supply, determining the optimal flow is a transportation problem. Thus the problem can be solved without the integer restrictions on the flow. In terms of the IP formulation presented in Section 2, if the binary variable  $\{v_j^i\}$  are fixed in a feasible way,  $\{w_j^{im}\}$  can be easily obtained by solving a transportation problem. Once the  $\{v_j^i\}$  are fixed, the discounts are known and hence the problem is easy. The binary variables  $\{w_j^{im}\}$  can indeed be relaxed to take continuous values, as there will always exist an optimal solution with integer values.

## 5 Linear Programming Relaxation

In this section we study the linear programming (LP) relaxation of the IP formulation presented in Section 2. The LP relaxation provides the lower bound on optimal cost, which is useful in pruning the search space in branch-and-bound algorithms. Moreover, for this problem, the LP relaxation is equivalent to solving a transportation problem. We study the LP relaxation by investigating its dual.

Let  $\{\gamma_j\}$ ,  $\{\lambda_j^i\}$ , and  $\{\gamma^m\}$  be the dual variables corresponding to the constraints (3), (4), and (5), respectively. The dual of the linear relaxation is given by:

$$\max \sum_m \gamma^m - \sum_j \gamma_j \quad (15)$$

subject to

$$\gamma^m + \lambda_j^i \leq p_j^{im} \quad \forall j, i, m \quad (16)$$

$$-\gamma_j - i\lambda_j^i \leq 0 \quad \forall i, j \quad (17)$$

$$\gamma_j \geq 0 \quad \forall j \quad (18)$$

A feasible solution to the above problem can be easily obtained. The nonnegative variables  $\gamma_j$  have negative coefficients in the objective function and hence can be equated to zero. The variables  $\lambda_j^i$  are common to both the constraints and can be eliminated by equating to zero. Thus  $\gamma^m = \min_{j,i} \{p_j^{im}\}$ . The  $\theta_j^i$  are non-decreasing over  $i$  by assumption. Hence the  $p_j^{im}$  are non-increasing and therefore  $\gamma^m = \min_j \{p_j^{Mm}\}$ . From this dual solution one can easily construct a feasible solution to the linear relaxation and these both are optimal solutions to their respective problems.

**Proposition 2**  $\{\pi_j = 0, \lambda_j^i = 0, \gamma^m = \min_j \{p_j^{Mm}\}\}$  is the optimal dual solution and  $v_j^i = 0$  for  $i < M$ ,  $v_j^M = \frac{\sum_m w_j^{Mm}}{M}$ , where

$$w_{j'}^{Mm} = \begin{cases} 1 & \text{if } j' = \arg \min_j \{p_j^{Mm}\} \\ 0 & \text{otherwise} \end{cases} \quad (19)$$

is the optimal solution to the linear relaxation.

It can be easily seen that the solutions are feasible to their respective problems. Moreover, they both have the same objective value. Hence by strong duality theorem, they are optimal solutions. The LP relaxation considers only the maximum discounted cost (with discount for  $M$  items) and allocates the items based on this minimum cost. As a consequence, it violates the discount constraint by procuring less number of items from a supplier but with a maximum discount for  $M$  items. It is worth noting that only variables  $\{v_j^i\}$  take continuous values.

The linear relaxation problem can be solved in  $O(MN)$  by taking the minimum of  $p_j^{Mm}$  over  $j$ , for each  $m$ . If the  $\{v_j^M\}$  are fractional, then it is not an optimal solution to the WDP. One can easily construct a feasible integer solution ( $\{\bar{v}_j^m\}, \{\bar{w}_j^{im}\}$ ) to the WDP from the fractional ( $\{v_j^M\}, \{w_j^{Mm}\}$ ).

1.  $\delta_j = \sum_m w_j^{Mm}, \forall j; \bar{v}_j^m = v_j^m, \forall j, m; \bar{w}_j^{im} = w_j^{im}, \forall j, i, m;$
2.  $\forall j$  do:
  - (a) **if** ( $v_j^M > 0$ )  $\bar{v}_j^{s_j} \leftarrow 1, \bar{v}_j^M \leftarrow 0;$
  - (b)  $\forall m$  **if** ( $w_j^{Mm} = 1$ )  $\bar{w}_j^{\delta_j, m} \leftarrow 1, \bar{w}_j^{Mm} \leftarrow 0;$

## 6 Branch-and-Bound Algorithms

B&B is an exact intelligent enumerative technique that attempts to avoid enumerating a large portion of the feasible integer solutions [5, 25]. It is a widely used approach for solving discrete optimization, combinatorial optimization, and integer programming problems in general. The B&B approach first partitions the overall set of feasible solutions into two or more sets and as the algorithm proceeds the set is partitioned into many simpler and smaller sets, which are explored for the optimal solution. Each such set is represented by a *candidate problem* (CP). A typical iteration of B&B consists of:

- **Selection/Removal** of a CP from the list of CPs
- Determining the **lower bound** of the selected CP
- **Fathoming** or **pruning**, if possible, the selected CP
- Determining and updating the **incumbent** solution, if possible

- **Branching strategy:** If the CP is not fathomed, branching creates subproblems which are added to the list of CPs

The algorithm first starts with a single CP in the list representing the entire feasible set of solutions. As the algorithm proceeds, numerous CPs are added to the list, each containing a set of feasible solutions. The CPs partition the search space and at every iteration, a prospective CP is chosen to search for the optimal solution. The CP though containing less number of solutions than the original problem, could still be hard to solve, and hence a easily solvable lower bounding technique is applied to obtain a good lower bound on the objective value. This lower bound is for the solutions in that particular CP. If a feasible solution had been obtained so far in the algorithm, it can be used to prune a CP with lower bound greater than the cost of the known solution. A CP can be fathomed (removed from further search) if the best solution in that CP is found. If not fathomed, it is then split into smaller CPs and added to the list of CPs. As the algorithm proceeds, the best known feasible solution is maintained and when the list of CPs become empty, the best known feasible solution is the optimal solution.

Although the B&B technique is easy to understand, the implementation for a particular problem is a nontrivial task [5] requiring:

- An efficient lower bounding technique that can be solved with less computational efforts and also guarantee a tight lower bound
- Efficient data structures for handling the rather complicated book-keeping of the list of CPs,
- Clever strategies for selecting promising CPs, and
- Branching strategies that could effectively prune the enumeration tree.

The conventional techniques for implementing the above are to use the LP relaxation as the lower bounding technique with *variable dichotomy* as the branching technique. If the LP relaxation provides a solution with non-integer values for integer variables, then one such variable is chosen and CPs are created by imposing bounds on the variable. The CPs are searched using the *best first search* (BFS). These techniques are generic and are used by many commercial solvers. In this work, we develop two B&B techniques that exploit the structure of the DA. They differ in the lower bounding techniques and the branching strategies. Both algorithms guarantee a feasible solution at every iteration, using heuristic techniques. The BFS is used as the search technique. The higher level algorithm, without the implementation details of the specific strategies is presented next.

1. (Initialize)  
 $cp \leftarrow \{\text{WDP}\}; pq = \emptyset; ls = \emptyset; is = \emptyset; bs = \emptyset;$
2.  $ls \leftarrow \text{LowerBound}(cp); is \leftarrow \text{Heuristic}(ls); bs \leftarrow is;$
3. **if**  $Z(ls) = Z(is)$  **end;**

4.  $pq.insert(cp)$ ;
5. **while**  $pq \neq \emptyset$  **do**:
  - (a)  $cp \leftarrow pq.DeleteMin()$ ;
  - (b) **if**  $Z(LowerBound(cp)) \geq Z(bs)$  **end**;
  - (c) **Create**( $cp-$ );
  - (d)  $ls \leftarrow LowerBound(cp-)$ ;  $is \leftarrow Heuristic(ls)$ ;
  - (e) **if**  $Z(is) < Z(bs)$   $bs \leftarrow is$ ;
  - (f) **if**  $Z(ls) < Z(bs)$   $pq.insert(cp-)$ ;
  - (g) **Create**( $cp+$ );
  - (h)  $ls \leftarrow LowerBound(cp+)$ ;  $is \leftarrow Heuristic(ls)$ ;
  - (i) **if**  $Z(is) < Z(bs)$   $bs \leftarrow is$ ;
  - (j) **if**  $Z(ls) < Z(bs)$   $pq.insert(cp+)$ ;

The list of CPs is stored in a priority queue  $pq$ . The `deletemin` operation of the priority queue returns the CP with the least lower bound. If the selected CP is not pruned, two child CPs are generated and added to the priority queue if they are not infeasible/fathomed/pruned. The priority queue is implemented using the binary heap data structure. Every insertion and deletion is of complexity  $O(\log P)$ , where  $P$  is the current size of the queue [7]. The routine `LowerBound` determines the lower bound and returns the lower bound solution (possibly infeasible to the WDP). The `Heuristic` constructs a feasible incumbent solution to the WDP from the lower bound solution. Using the branching technique two candidate problems  $cp-$  and  $cp+$  are created from  $cp$  using the routine `Create`, partitioning the solution space of  $cp$ . The implementation details of these routines for the proposed algorithms will be presented later. The  $is$ ,  $bs$ , and  $ls$  store the incumbent solution, the best incumbent solution, and the lower bound solution, respectively. The  $Z(\cdot)$  denotes the optimal objective value of the solution ( $\cdot$ ). At the end of the algorithm,  $bs$  contains the optimal solution to the WDP.

## 7 Branch-on-Supply

The first B&B algorithm, called as *branch-on-supply* (BoS) does not use the IP formulation of the WDP, but rather uses the network structure. Note that if the optimal number of winning items for each of the bid is known, then finding the respective winning items is an easy problem. The BoS uses this principle to search for optimal number of winning items.

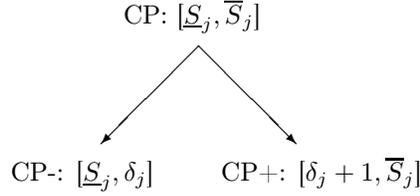


Figure 2: Branching Strategy of BoS

## 7.1 Candidate Problem and the Lower Bounding Technique

The CP represents the set of feasible solutions it contains. BoS uses the following CP: each bid  $j$  has a supply in range  $[\underline{S}_j, \overline{S}_j]$ . Thus the first CP, which contains all the feasible solutions has  $\underline{S}_j = 0$  and  $\overline{S}_j = M, \forall j$ . The CP is feasible iff  $\underline{S}_j \leq \overline{S}_j, \forall j$  and  $\sum_j \underline{S}_j \leq M \leq \sum_j \overline{S}_j$ . The lower bound to the CP is obtained by solving it as an *interval transportation problem* with supply for  $j$  bounded in the interval  $[\underline{S}_j, \overline{S}_j]$  and with cost on the link  $(j, m)$  as  $p_j^{\overline{S}_j m}$ . Clearly the optimal objective value of the interval transportation problem is the lower bound to the associated CP, as the considered cost is the maximum discounted cost.

## 7.2 Branching Strategy

If a CP is not fathomed or pruned or infeasible, then it is partitioned into two subproblems CP- and CP+. Let  $\delta_j$  be the number of items supplied by bid  $j$  in the lower bound solution. Obviously  $\delta_j \in [\underline{S}_j, \overline{S}_j]$ . If  $\delta_j = \overline{S}_j, \forall j$  then the lower bound solution is the optimal solution of that CP and the CP said to be fathomed. If  $\delta_j < \overline{S}_j$  for at least one bid  $j$ , then the bid supplies  $\delta_j$  items for a discount of  $\overline{S}_j$  items, violating the discount constraint. Two subproblems CP- and CP+ are created by branching on a discount violating bid as shown in Figure 2. The upper bound on the supply of CP- is changed to  $\delta_j$  and the lower bound of CP+ is changed to  $\delta_j + 1$ . The supply range of the other bids are not changed. Note that, the solution space of the CP is partitioned into two and the lower bound solution of CP is not feasible to the interval transportation problems of CP- and CP+. Thus, when these problems are solved, the same lower bound solution is not encountered. If there are several such discount violating bids, one of them is chosen to create two subproblems. The branching bid as follows:

$$j' = \arg \min_j \{(\overline{S}_j - \delta_j) : \delta_j < \overline{S}_j\} \quad (20)$$

### 7.3 Heuristic

The lower bound solution with supplies  $\{\delta_j\}$  and the flow is a feasible solution to the WDP as  $\sum_j \delta_j = M$ . However, the cost of the solution is not compatible, as it determined using  $\{p_j^{\bar{S}_j^m}\}$ , instead of  $\{p_j^{\delta_j^m}\}$ . Therefore, the allocation of winning items may not be optimal to the supplies  $\{\delta_j\}$ . To get the optimal allocation, a transportation network is constructed with supplies  $\{\delta_j\}$  and flow costs  $\{p_j^{\delta_j^m}\}$ . The flow to this new network is a feasible solution to the WDP. Thus BoS is an *anytime* algorithm that can be terminated anytime guaranteeing a feasible solution.

### 7.4 BoS vs LP Relaxation based B&B

At the outset, the design philosophy of BoS looks very different from that of conventional LP relaxation based B&B. However, a deep investigation shows that both share many similarities with respect to the lower bounding technique and the branching strategy. As mentioned in Section 5, the LP relaxation is the transportation problem with flow cost equal to the maximum discounted cost. This is the same with the interval transportation problem, which uses the maximum discounted cost ( $p_j^{\bar{S}_j^m}$ ) for item  $m$  from  $j$ . However, there is a lower bound  $\underline{S}_j$  on the number of items that can be supplied from  $j$ . Thus, the problem is more constrained than the LP relaxation and one can expect a more tightened lower bound.

In the LP relaxation of the WDP, only the integrality of the binary variables  $\{v_j^i\}$  are violated. If variable dichotomy branching is used, then a fractional  $v_j^i$  is chosen and two subproblems are created by imposing  $v_j^i = 0$  in of them and  $v_j^i = 1$  in the other. The constraints (3) involving  $\{v_j^i\}$  are *specialty ordered set of type1* (SOS1) constraints. SOS1 constraints impose at most one variable in the constraint to be positive. The effective branching strategy for SOS1 is to choose an index  $i'$  and create two CPs by imposing  $\sum_{i \leq i'} v_j^i \leq 1$  in of them and  $\sum_{i > i'} v_j^i \leq 1$  in the other. This essentially means that in the first CP, bid  $j$  can supply at most  $i'$  items and in the second CP, it can possibly supply more than  $i'$ . This is similar to the branching used in BoS. However, in BoS the respective constraints are of form  $\sum_{i \leq i'} v_j^i = 1$  and  $\sum_{i > i'} v_j^i = 1$ .

## 8 Branch-on-Price

In this section we propose the second B&B algorithm *branch on price* (BoP), which is essentially a *branch-and-cut* (B&C) algorithm with a special branching technique. The LP relaxation of Section 5 is strengthened by adding valid inequalities, which serve as cuts. First we briefly review the B&C technique and its variants.

## 8.1 Branch-and-Cut

A *valid inequality* for an IP problem is an inequality that is satisfied by all the feasible solutions. Valid inequalities that are not part of a formulation are essentially redundant constraints. However, they may serve as *cuts* if they are not satisfied by all feasible solutions of the LP relaxation [29]. A cut that is not satisfied by an optimal solution of the LP relaxation is called as a *violated cut*. Addition of a violated cut to the LP relaxation tightens it and provides a better bound. In this way, the formulation is changed in such a way that the LP feasible region becomes smaller but the IP feasible region is unaffected. Identification of violated cuts, adding them to the LP relaxation, and resolving them to find better bounds, can be iterated till no violated cuts can be found.

Branch-and-cut [23] is a generalization of B&B, which includes the cut routine to identify and add violated cuts. The generation of tighter bounds with addition of violated cuts helps in pruning the B&B nodes, thereby reducing the search space. At each node, the LP relaxation is repeatedly solved with addition of new cuts each time. There are several variations of B&C with respect to cuts generation and addition. For a detailed exposure, see [6, 24]. In this work, we generate the cuts by studying the LP relaxation solution. This does not involve solving separation problems. The generic property of the LP relaxation of the IP formulation given in Section 2.2 is used to generate a family of cuts. Cuts can be added to the B&B tree in various ways, leading to different algorithms.

*Cut-and-branch* (C&B) is a B&C variant where a family of cuts are added to the formulation and B&B is applied to the modified formulation. This technique is useful if generation of cuts are easier and are known a priori (as is the case in our problem). It is advantageous as the cuts added are valid throughout the tree and no further cuts are required to be added. Usually the number of cuts added are very large and many of them may not be useful. With large number of constraints, the time taken to solve the LP relaxation can increase considerably. An alternate approach is to add the cuts that are only violated by the current LP solution. In this way, cuts are progressively added. There are two possible approaches here: *B&C-global*, in which cuts added are valid throughout the search tree and *B&C-local*, where the cuts added are valid only to the subtree of the current node.

## 8.2 A Tight Formulation with Valid Inequalities

The optimal LP solution to the IP formulation (Section 2.2) had binary values for  $\{w_j^{im}\}$  and fractional values for  $\{v_j^i\}$ . Hence the following valid inequalities serve as cuts to the LP relaxation:

$$w_j^{im} \leq v_j^i \quad \forall j, i, m \quad (21)$$

They are obviously valid for the IP formulation and they exclude the optimal solution of the original LP relaxation. Hence, the bounds obtained can be expected to be tighter than the original formulation. The optimal solution of the LP relaxation with cuts satisfies the following properties:

1. If  $v_j^i > 0$ , then number of non-zero  $w_j^{im}$ s are greater than or equal to  $i$ .

2. The  $\{w_j^{im}\}$  take binary values only if  $\{v_j^i\}$  are binary.

The first property is a direct consequence of the valid inequalities and the constraint set (4). The second property follows from the first.

### 8.3 Branching Criteria and the Candidate Problem

Unlike BoS, BoP uses LP relaxation as the branching technique. Hence, variable dichotomy branching technique could be used. According to the above mentioned properties of the optimal LP solution, either all variables are binary (in which case optimal to the IP) or many variables are fractional. The variable dichotomy branching is to chose a particular fractional  $v_j^i$  or  $w_j^{im}$  to create two CPs by imposing the variable to equal to 0 and 1, respectively. The branching on a  $v_j^i$  is more generic than that on a  $w_j^{im}$ . The former splits the solution space based on the number of items supplied by a bid, whereas the latter is more specific about an item, supplied by a particular bid that supplies a certain number of other items. We propose here a novel technique to create candidate problems by branching on the price of an item.

Most of the integer variables in an IP formulation of combinatorial or discrete optimization problems are auxiliary variables created to impose logical constraints or linear constraints. Hence branching on such variables directly may not have logical or natural implication on the way the subsequent candidate problems are created. Further, the size of the solution set of these candidate problems may not be balanced, thus resulting in a unbalanced search tree. For example, a candidate problem created by fixing  $w_j^{im} = 1$  may have very less number of solutions compared with the CP created by fixing  $w_j^{im} = 0$ . Instead of creating CPs by branching on fractional variables, we create by branching on the *price* of an item, which is fractionally supplied by more than one supplier.

Let  $w_j^{im}$  be fractional. Due to the constraint (5), there exist at least one another  $w_{j'}^{i'm}$  with  $i \neq i'$ , which is fractional. Let  $\beta^m$  be the price of the item  $m$  as dictated by the LP solution:

$$\beta^m = \sum_j \sum_i p_j^{im} w_j^{im} \quad (22)$$

We create two CPs, CP- and CP+, by branching on the above price. The CP- is created by adding the following constraints:

$$w_j^{im} = 0 \text{ if } p_j^{im} \geq \beta^m, \forall j, i, m \quad (23)$$

The CP+ is created by adding constraints

$$w_j^{im} = 0 \text{ if } p_j^{im} < \beta^m, \forall j, i, m \quad (24)$$

The two CPs partition the IP feasible solution space. The optimal LP solution (which is infeasible) does not belong to the solution space of the relaxations of the either of the CPs. To facilitate this branching, we represent a CP by using bounds on the prices of each of the items.

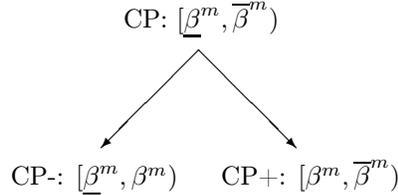


Figure 3: Branching Strategy of BoP

The CP is compactly represented by using an allowable price range  $[\underline{\beta}^m, \overline{\beta}^m)$  for each item  $m$ . Algebraically, this is achieved by imposing the following bounds in the IP formulation:  $w_j^{im} = 0$  if  $p_j^{im}$  is outside the above range. The initial CP contains all the solutions and hence  $\underline{\beta}^m = \min_j \{p_j^{Mm}\}$  and  $\overline{\beta}^m = \max_j \{p_j^{1m}\} + \epsilon$ , for some  $\epsilon < 0$ . If an item  $m$  is chosen for branching with price  $\beta^m$ , then CP- has  $[\underline{\beta}^m, \beta^m)$  and CP+ has  $[\beta^m, \overline{\beta}^m)$ , as the respective price range for  $m$ . This is illustrated in Figure 3. Note that the price range of other items will remain the same as that of the parent CP.

The above branching scheme imposes many variables to be zero, across several bids, rather than just fixing one variable to 0 or 1, as in the variable dichotomy branching. Further, such a branching is more meaningful in terms of the WDP. The  $\beta_m$  can be considered as the price of the item  $m$  and the BoP algorithm is searching for the optimal price from the set of  $\{p_j^{im}\}$ , subject to the discount and demand constraints. Violation of the discount and the demand constraints leads  $\beta^m$  to be a convex combination of some of the prices from the set  $\{p_j^{im}\}$ . The proposed branching scheme partitions the set such that the same convex combination cannot be encountered again, thereby removing the violation in the constraints. In this way, one can expect that the algorithm will converge fast towards the optimal prices. It is worth noting that even though the  $\beta^m$  is a real number and thus the branching could be infinitely divisible, the possible optimal values it can take is  $NM$  and hence the number of branches is finite.

If there are more than one item which has fractional allocations, the algorithm has to choose one item to branch upon. Let  $\beta^m$  be the price of item  $m$  defined by the convex combination (22). The item  $m'$  to branch on is chosen among the following rules:

$$\mathbf{BoP1} : m' = \arg \min_m \{\beta^m - \underline{\beta}^m : \beta^m > \underline{\beta}^m\} \quad (25)$$

$$\mathbf{BoP2} : m' = \arg \min_m \{\overline{\beta}^m - \beta^m : \underline{\beta}^m > \beta^m\} \quad (26)$$

$$\mathbf{BoP3} : m' = \arg \max_m \{\text{Number of bids that partially supply item } m\} \quad (27)$$

The BoP1 chooses the item whose price is closest to its lower bound, but not equal to it. According to the rules of the creation of CPs, branching on an item with  $\beta^m = \underline{\beta}^m$  will create an infeasible CP- with range  $[\beta^m, \beta^m)$  and CP+ with range  $[\beta^m, \overline{\beta}^m)$ , which is same as its parent CP. To avoid an infinite loop, we consider only items whose price is strictly greater than its lower bound. However, a pathological case might arise with all items having their prices equal to their respective lower bounds. In this situation, we chose an item  $m$  randomly and create only CP+ with range  $[\beta^m + \epsilon, \overline{\beta}^m)$ . The  $\epsilon > 0$  is chosen such that it is small enough to exclude just the price  $\beta^m$ . Note that there is no CP- created and with this new CP+ creation, it is possible that a feasible IP solution with prices  $\{\beta^m\}$  might be excluded in the search. However, this does not happen as the BoP will find such an IP solution, if it exists, using the heuristic algorithm to be explained next. BoP2 chooses the item with price closest to its upper bound and BoP3 chooses the item with maximum number of allocated bids.

#### 8.4 Heuristic Algorithm to Determine Incumbent Solutions

Let  $w_j^{im}$  be the optimal LP solution. If all are binary, then it is a feasible solution to the IP. The following heuristic constructs a feasible solution from the fractional LP solution.

1. (Initialize)  $S_j = 0, \forall j$
2. **do**  $\forall m: k = \arg_j \max_{j,i} \{w_j^{im}\}; S_k \leftarrow S_k + 1;$
3. Construct a transportation network with winning bids as sources and items as sinks. The source corresponding to winning bid  $j$  has a supply of  $S_j > 0$  and each sink has a unit demand. The cost of flow from  $j$  to  $m$  is  $p_j^{S_j,m}$ . Let  $x_j^m$  be the optimal flow. Assign  $V_j^{S_j} = 1$  and  $w_j^{S_j,m} = x_j^m$ .

The winning bid for an item  $m$  is chosen as the one with the largest  $w_j^{im}$  value. This is used to determine the number of winning items  $S_j$  for each winning bid. This is in turn used to determine the winning items with the consistent discount prices  $p_j^{S_j,m}$ . This will provide a better IP solution than directly rounding the largest  $w_j^{im}$  to 1. Using this heuristic, an incumbent solution is obtained whenever a new CP is created and the best known solution is updated and stored. Thus like BoS, BoP is also an anytime algorithm that can be terminated anytime guaranteeing a feasible solution.

## 9 Computational Experiments

In this section, we present the results of the extensive computational experiments conducted using the various proposed algorithms across various problem types.

## 9.1 Experimental Setup

A test suite called as *discount auctions test suite* (DATS) was created to randomly generate different types of problem instances. An problem instance is defined by  $M$ ,  $N$ ,  $\{Q_j^m\}$ , and  $\{\theta_j^i\}$ . Given  $M$  and  $N$ , the cost and the discounts can be generated in many ways. The costs  $\{Q_j^m\}$  are normalized and are chosen based using a parameter called as *relative cost*. The relative cost of item  $m$ ,  $rc^m$ , serves two purposes: (1) to denote its cost, relative to the maximum cost of the items, which is 1 and (2) it is the maximum cost that can be quoted for  $m$ , that is  $Q_j^m \leq rc^m, \forall j$ . In a sense, it captures both the relative cost and its market value. The rationale behind such a parameter is to enable the generation of problem instances that reflect varying scenarios. The  $\{rc^m\}$  is chosen randomly in a given range  $[r, 1]$ , with at least two items, each of them taking one of the boundary values is chosen. To model the scenario where the items being procured have only 10% relative difference in the cost, the  $\{rc^m\}$  can be chosen in range  $[0.9, 1]$ . Another parameter is required to know the cost difference quoted among the bids for an item. The *minimum cost*  $mc^m$  of item  $m$  denotes the minimum cost factor for  $m$  for the bid price. The individual cost of  $m$  for each of the bids are chosen in the following way:

$$Q_j^m = \text{Random}[mc^m, 1] \times rc^m \quad (28)$$

The individual costs are chosen in such a way that at least one bid has  $rc^m$  and one has the minimum cost  $mc^m \times rc^m$ . The  $rc^m$  thus captures cost of  $m$  relative to other items and  $mc^m$  denotes the cost variation of  $m$  among the bids.

For generating the discount functions, a discount range is input to the DATS. For example, if the discount range is  $[0.15, 0.2]$ , then all the maximum discounts  $\{\theta_j^M\}$  are chosen randomly in this range. There is no discount for one item:  $\theta_j^1 = 0$ . The intermediate values are chosen according to type of the discount function. The DATS currently supports following types of discount functions:

1. *Linear*:  $\theta_j^i = (i - 1) \times \frac{\theta_j^M}{M-1}$
2. *Marginally Decreasing*:  $\theta_j^i = -\frac{\theta_j^M}{(M-1)^2} \times (M - i)^2 + \theta_j^M$
3. *Marginally Increasing*:  $\theta_j^i = \frac{\theta_j^M}{(M-1)^2} \times (i - 1)^2$
4. *Step*
5. *Arbitrary*
6. *Random*: The discount type for a bid  $j$  is chosen randomly from one of the above.

All functions are strictly increasing, except for the type *Step*. The *Arbitrary* is strictly increasing without any notable structure like the preceding types.

The experiments were carried out on a Windows XP based PC equipped with a 2.8GHz Intel P4 processor with 1GB RAM. The algorithms were coded in Java, and for the model building and solving of LP relaxations and transportation problems, ILOG Concert Technology of CPLEX 10.0 [15] was used.

## 9.2 BoP Experiments

To study the performance of the proposed BoP, we created different algorithms by combining different cut addition techniques and branching rules. We considered following cut addition techniques:

**Cut-and-Branch (C&B)** All the  $NM^2$  cuts ( $w_j^{im} \leq v_j^i, \forall j, i, m$ ) are added to the IP formulation and B&B technique is applied to the new formulation.

**Branch-and-Cut Global- $w$  (B&C-Gbl- $w$ )** After each simplex iteration (in solving the LP relaxation), only those variables that violate the inequality  $w_j^{im} \leq v_j^i$  are added as cuts globally to all the nodes in the search tree.

**Branch-and-Cut Local- $w$  (B&C-Loc- $w$ )** Same as above, except that the cuts are local cuts.

**Branch-and-Cut Global- $V$  (B&C-Gbl- $V$ )** After each simplex iteration, if an  $w_j^{im}$  violates the inequality  $w_j^{im} \leq v_j^i$  then cuts for all items for that  $j$  and  $i$  are added as global cuts.

**Branch-and-Cut Local- $V$  (B&C-Loc- $V$ )** Same as above, except that the cuts are local cuts.

The different branching techniques include the traditional variable dichotomy (Var-Dic) along with the three BoP techniques proposed in Section 8.3. With the above combinations of cut addition and branching techniques, we have twenty different branch-and-cut algorithms. The performance parameters considered were:

- CPU time (in milliseconds)
- number of nodes explored
- number of simplex iterations

The test problem instance was created with following parameters:

- $r = 0.2$
- $mc^m = 0.7, \forall m$
- $\theta_j^M \in [0.15, 2], \forall m$
- $N = 50, 100$
- $M = 10, 15$

Except for problems with discount types *marginally decreasing* and *random*, all others were solved at the root node itself without any need for further branching. The performance of the above two discount types for different  $N$  and  $M$  combinations are shown in tables 1-4.

Table 1: Discount Type: Marginally Decreasing,  $N=50$ ,  $M=10$ 

	<b>Var-Dic</b>	<b>BoP1</b>	<b>BoP2</b>	<b>BoP3</b>
<b>C&amp;B</b>	6094, 8, 17	22296, 14, 29	22266, 14, 29	10312, 6, 13
<b>B&amp;C-Gbl-<math>w</math></b>	6453, 8, 393	16547, 14, 610	16594, 14, 610	8156, 6, 382
<b>B&amp;C-Gbl-<math>V</math></b>	4735, 8, 140	16062, 14, 185	16078, 14, 185	8000, 6, 141
<b>B&amp;C-Loc-<math>w</math></b>	22094, 8, 2889	53296, 14, 5312	53532, 14, 5312	22266, 6, 2007
<b>B&amp;C-Loc-<math>V</math></b>	17297, 8, 1462	41062, 14, 2746	40875, 14, 2746	16157, 6, 1012

Table 2: Discount Type: Marginally Decreasing,  $N=100$ ,  $M=15$ 

	<b>Var-Dic</b>	<b>BoP1</b>	<b>BoP2</b>	<b>BoP3</b>
<b>C&amp;B</b>	472171, 17, 35	1314907, 31, 63	2441734, 73, 147	1090078, 23, 47
<b>B&amp;C-Gbl-<math>w</math></b>	404594, 17, 1301	1078562, 31, 2130	1934906, 73, 2950	1006939, 23, 2269
<b>B&amp;C-Gbl-<math>V</math></b>	458828, 19, 313	1242969, 31, 474	2367922, 73, 635	1064812, 23, 494
<b>B&amp;C-Loc-<math>w</math></b>	1323563, 17, 17802	2866469, 31, 27412	6981784, 73, 59903	2282657, 23, 20225
<b>B&amp;C-Loc-<math>V</math></b>	687441, 17, 7001	1487432, 31, 10657	3358053, 73, 24580	1207507, 23, 8173

Table 3: Discount Type: Random,  $N=50$ ,  $M=15$ 

	<b>Var-Dic</b>	<b>BoP1</b>	<b>BoP2</b>	<b>BoP3</b>
<b>C&amp;B</b>	41344, 13, 27	117594, 15, 31	150735, 29, 59	91047, 10, 21
<b>B&amp;C-Gbl-<math>w</math></b>	35110, 19, 620	76516, 15, 815	110485, 29, 1020	63720, 10, 623
<b>B&amp;C-Gbl-<math>V</math></b>	107188, 38, 239	110657, 15, 195	139391, 29, 272	84204, 10, 178
<b>B&amp;C-Loc-<math>w</math></b>	100735, 15, 5813	131126, 15, 6232	332768, 29, 10561	122907, 10, 3495
<b>B&amp;C-Loc-<math>V</math></b>	147048, 23, 4254	113922, 15, 2764	259439, 29, 4971	94141, 10, 1593

Table 4: Discount Type: Random,  $N=100$ ,  $M=15$ 

	<b>Var-Dic</b>	<b>BoP1</b>	<b>BoP2</b>	<b>BoP3</b>
<b>C&amp;B</b>	163781, 11, 23	553797, 23, 47	322140, 15, 31	315625, 11, 23
<b>B&amp;C-Gbl-<math>w</math></b>	157016, 11, 1196	472875, 24, 1531	340437, 16, 1369	231922, 11, 1061
<b>B&amp;C-Gbl-<math>V</math></b>	140781, 11, 331	500250, 23, 479	287781, 15, 394	261563, 11, 349
<b>B&amp;C-Loc-<math>w</math></b>	580594, 11, 11146	1281421, 23, 17281	1166485, 15, 11995	586390, 11, 9427
<b>B&amp;C-Loc-<math>V</math></b>	382032, 12, 5392	740343, 24, 8501	568016, 15, 5374	385688, 11, 4464

## 10 Conclusions and Future Work

In this work, we proposed a new auction mechanism called as discount auctions for procuring single units of multiple items. This is useful in industrial procurement scenarios, where an organization is interested in buying  $M$  distinct, possibly related, items from a pool of  $N$  suppliers. The  $M$  items are assumed to be independent and compatible with each other even if brought from different suppliers. For this scenario, traditional combinatorial bids for subsets of items are redundant, and the proposed discount bids are more meaningful. Further the bids to be communicated are linear in the number of items, in contrast to exponential number of combinatorial bids. However, despite its simplicity, the winner determination problem was proved to be  $\mathcal{NP}$ -hard upon reduction from the set covering problem.

We proposed two exact solution methodologies based on branch-and-bound techniques. The first generation algorithm, called as BoS (*branch-on-supply*), uses the embedded network structure in the problem to determine the optimal number of items supplied by each seller. The algorithm preserves the combinatorial structure of the problem and uses special branching technique to search for the optimal solution. The second algorithm, called as BoP (*branch-on-price*), uses LP relaxation with strong inequalities and a novel branching technique. The valid inequalities that are added as cuts are not facet-defining. However, the size of the entire family of the cuts is polynomial ( $NM^2$ ) and can be determined directly from the LP solution. Computational experiments showed that the duality gap is considerably less and hence the algorithm is many-folds faster than that of BoS. To further reduce the search space, BoP uses a novel branching technique. It branches on the price of an item, which is allocated to more than one supplier. Computational experiments showed that this branching technique is many-folds faster than that of the traditional variable dichotomy branching.

This work was devoted mainly to the computational aspects of solving the winner determination problem to optimality. There are several interesting research fronts in which the discount auctions could be explored further.

- *Iterative Discount Auctions:* We considered here an one-shot discount auctions with just a single round of bidding. Iterative auctions with several rounds of bidding are preferable for auctions with large number of items. The suppliers need not communicate the entire discount bid, but can progressively express the bids based on the dynamics of the auction. The commonly used approach is the mechanism design technique with duality theory of mathematical programming.
- *Eliciting Bidder Preferences:* One of the important problems in auctions is the bid preparation. For discount auctions, the bidder has to specify individual costs for  $M$  items and a discount function. Determining the discount for the number of items may not be easy and one need a preference elicitation mechanism to determine the discounts.
- *Multisunit Discount Auctions:* In this paper, we considered single unit of multiple units. An useful extension is to look at multiple units of multiple items. Inclusion of multiple

items adds another layer of complexity and the proposed algorithms in this work are not directly extendable.

- *Discounts based on Cost:* The discount structure considered in this paper depends on the number of items bought. Another meaningful perspective is to define the discounts based on the total cost of the items procured.
- *0-1 Knapsack with GUB Constraints:* We added a family of valid inequalities from the LP relaxation. The formulation also has a 0-1 knapsack structure (30) with GUB (generalized upper bound) constraints (30). In particular, the following set of constraints constitute a multiple choice knapsack structure.

$$\sum_i v_j^i \leq 1 \quad \forall j \quad (29)$$

$$\sum_j \sum_i i v_j^i = M \quad (30)$$

Generation of valid cover inequalities for 0-1 knapsacks with generalized upper bound constraints were studied in [28]. The generation of cuts would involve solving the hard separation problem, either optimally or approximately. It is worth investigating the hardness of the separation problem for the above constraints and studying the trade-offs of investing resources in generation of the cover inequalities against the savings in the convergence of the algorithm.

## References

- [1] D. R. Beil and L. M. Wein. An inverse-optimization-based auction mechanism to support a multiattribute RFQ process. *Management Science*, 49(11):1529–1545, 2003.
- [2] M. Bichler, M. Kaukal, and A. Segev. Multi-attribute auctions for electronic procurement. In *Proceedings of the First IBM IAC Workshop on Internet Based Negotiation Technologies*, pages 18–19, Yorktown Heights, NY, USA, 1999.
- [3] Martin Bichler, Andrew Davenport, Gail Hohner, and Jayant Kalagnanam. Industrial procurement auctions. In Peter Cramton, Yoav Shoham, and Richard Steinberg, editors, *Combinatorial Auction*, chapter 23. MIT Press, 2006.
- [4] T.S. Chandrashekar, Y. Narahari, Charlie Rosa, Devadatta Kulkarni, and Jeffrey Tew. Auction based mechanisms for electronic procurement. Technical report, Electronic Enterprises Laboratory, Department of Computer Science and Automation, Indian Institute of Science, July 2005.
- [5] V. Chandru and M. R. Rao. Integer programming. In M. J. Atallah, editor, *Handbook of Algorithms and Theory of Computing*, pages 32.1–32.45. CRC Press, 1999.

- 
- [6] C. Cordier, H. Marchand, R. Laundry, and L.A. Wolsey. bc-opt: A branch-and-cut code for mixed integer programs. Papers 9778, Catholique de Louvain - Center for Operations Research and Economics, 1997. available at <http://ideas.repec.org/p/ftth/louvco/9778.html>.
- [7] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. MIT Press, Cambridge, 1990.
- [8] Peter Cramton, Yoav Shoham, and Richard Steinberg. *Combinatorial Auctions*. MIT Press, Cambridge, 2005.
- [9] A. Davenport and J. Kalagnanam. Price negotiations for procurement of direct inputs. Research Report RC 22078, IBM Research, Yorktown Heights, NJ, USA, 2001.
- [10] W. Elmaghraby and P. Keskinocak. Technology for transportation bidding at the home depot. In *Practice of Supply Chain Management: Where Theory and Practice Converge*. Kluwer Academic Publishers, 2003.
- [11] Wedad Elmaghraby. Auctions and pricing in e-marketplaces. In David Simchi-Levi, S. David Wu, and Z. Max Shen, editors, *Handbook of Quantitative Supply Chain Analysis: Modeling in the E-Business Era*, International Series in Operations Research and Management Science. Kluwer Academic Publishers, Norwell, MA, 2004.
- [12] M. Eso, S. Ghosh, J. Kalagnanam, and L. Ladanyi. Bid evaluation in procurement auctions with piece-wise linear supply curves. Research Report RC 22219, IBM Research, Yorktown Heights, NJ, USA, 2001.
- [13] M. R. Garey and D. S. Johnson. *Computers and Intractability*. Freeman, 1979.
- [14] Gail Hohner, John Rich, Ed Ng, Grant Reid, Andrew J Davenport, Jayant R Kalagnanam, Soo Ho Lee, and Chae An. Combinatorial and quantity discount procurement auctions provide benefits to mars, incorporated and to its suppliers. *INTERFACES*, 33(1):23–35, 2003.
- [15] ILOG CPLEX. *ILOG Concert Technology 1.1 User's Manual*, February 2001.
- [16] S. Kameshwaran, L. Benyoucef, and X. Xie. Discount auctions for procuring heterogeneous items. In *Proceedings of Seventh International Conference on Electronic Commerce (ICEC 2005)*, 2005.
- [17] S. Kameshwaran, L. Benyoucef, and X. Xie. Winner determination in discount auctions. In *Proceedings of Workshop on Internet and Network Economics (WINE 2005)*, Lecture Notes in Computer Science 3828, pages 868 – 877. Springer, 2005.
- [18] S. Kameshwaran and Y. Narahari. e-Procurement using goal programming. In *Proceedings of 4th International Conference on Electronic Commerce and Web Technologies - EC-Web 2003*, Lecture Notes in Computer Science 2738, pages 6 – 15. Springer, 2003.

- 
- [19] S. Kameshwaran and Y. Narahari. A Lagrangian heuristic for bid evaluation in e-Procurement auctions. In *Proceedings of 2005 IEEE International Conference on Automation Science and Engineering*, pages 220–225, 2005.
- [20] A. Kothari, D. C. Parkes, and S. Suri. Approximately strategy proof and tractable multi-unit auctions. In *Proceedings of ACM Conference on Electronic Commerce (EC-03)*, 2003.
- [21] D. Lehmann, R. Müller, and T. Sandholm. The winner determination problem. In Peter Cramton, Yoav Shoham, and Richard Steinberg, editors, *Combinatorial Auction*, chapter 12. MIT Press, 2006.
- [22] R. P. McAfee and J. McMillan. Auctions and bidding. *Journal of Economic Literature*, 25:699–738, 1987.
- [23] J. E. Mitchell. Integer programming: Branch-and-cut algorithms. In C.A. Floudas and P.M. Pardalos, editors, *Encyclopedia of Optimization Vol. 2*, pages 519–525. Kluwer Academic Publishers, 2001.
- [24] J. E. Mitchell. Branch-and-cut algorithms for combinatorial optimization problems. In P.M. Pardalos and M. G. C. Resende, editors, *Handbook of Applied Optimization*, pages 65–77. Oxford University Press, 2002.
- [25] K. G. Murty. *Linear and Combinatorial Programming*. R. E. Krieger, 1985.
- [26] D. Pisinger and P. Toth. Knapsack problems. In Ding-Zhu Du and P. M. Pardalos, editors, *Handbook of Combinatorial Optimization*, pages 299–428. Kluwer Academic Publications, 1998.
- [27] T. Sandholm, S. Suri, A. Gilpin, and D. Levine. Cabob: A fast optimal algorithm for winner determination in combinatorial auctions. *Management Science*, 51(3):374–390, 2005.
- [28] L. A. Wolsey. Valid inequalities for 0-1 knapsacks and MIPs with generalized upper bound constraints. *Discrete Applied Mathematics*, 29:251–261, 1988.
- [29] L. A. Wolsey. *Integer Programming*. John Wiley and Sons, New York, 1998.



---

Unité de recherche INRIA Lorraine  
LORIA, Technopôle de Nancy-Brabois - Campus scientifique  
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes  
4, rue Jacques Monod - 91893 ORSAY Cedex (France)

Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)

Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

---

Éditeur  
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)  
<http://www.inria.fr>  
ISSN 0249-6399