December 2001

Towards a Universal Language for Combinatorial Auctions

Elena Katok
Penn State University

TOWARDS A UNIVERSAL LANGUAGE FOR COMBINATORIAL AUCTIONS

Elena Katok
Penn State University
ekatok@psu.edu

Abstract

A type of auctions that allows bidding on packages, or combinations, is called a combinatorial auction. One reason combinatorial auctions are challenging to implement is that the bidder determination problem is computationally difficult (NP-hard). The second reason is that a if a mechanism is to be useful in practice, a language for expressing bidder valuations has to be relatively simple and the process of bid creation itself should be straightforward. In this paper we present a general formulation of the winner determination problem as an integer program and describe a bidding language capable of expressing a large variety of valuations. The winner determination is done through an SQL-based interface between a database and an algebraic modeling language, making it quick to develop, easy to modify and flexible to implement.

Keywords: Combinatorial Auctions

Introduction

In recent years Internet auctions have received a great deal of attention in the e-business community. Companies like eBay and Amazon made electronic auctions available to millions, but those types of auctions are largely of a Consumer-to-consumer (C2C) and Business-to-Consumer (C2C) variety, where bidders have simple preferences that do not require them to bid on combinations of objects. In this paper we focus on a type of auction more relevant in the Business-to-Business (B2B) setting, called combinatorial auction. In this type of auction, bidders' valuations for objects are not independent—they either have complementarities or they are substitutes. One example of situations where combinatorial auctions are currently used is FCC auction for radio spectrum.

When valuations for objects involve either substitution effects or complementarities, auction designs where bidders bid on individual items give rise to the so-called *exposure problem*. This problem is best explained with an example. Suppose a bidder values objects A and B at $200,000, but either A or B by themselves are worthless. An example of this in practice is airport time slot for a take-off and landing—you need both to be able to fly. Suppose an auction is conducted, where A and B are auctioned off separately. A bidder may start bidding on both, but this involves a risk of winning one and loosing the other one—the exposure problem. A bidder would really like to place a bid of the form: I want either both A and B, or nothing--an example of a combinatorial bid.

The Winner Determination Problem Formulations

Determining winners in a combinatorial auction can be expressed as an integer-programming problem. Let $\mathbf{I} = \{i \mid i = 1, \dots I\}$ be a set of objects, $\mathbf{B} = \{b \mid b = 1, \dots B\}$ be a set of bids, and $\mathbf{P} = \{p \mid p = 1, \dots P\}$ be the set of bidders, with bidders being allowed to place bids on any combination of objects. Standard formulations assume that bidders can place multiple OR bids, and only one bid can be accepted. However, we are not going to make this assumption for reasons that will become clear when we discuss bidding languages. Let $\mathbf{B}_p$ be the set of bidder p's bids, $\mathbf{B}_i$ be the set of bids that include object i, and c_{bp} be the amount bidder

p bid for her bid b. We assume that there is one unit of each object available. If x_b is our decision variable that is 1 if bid b wins and 0 otherwise, the winner determination problem can be stated as follows:

$$\max \quad z = \sum_{p, b \in \mathbf{B}_p} c_{bp} x_b \tag{1.1}$$

subject to

$$\sum_{b \in \mathbf{B}_i} x_b \le r_i \quad \forall\, i \tag{1.2}$$

$$\sum_{b \in \mathbf{B}_p} x_b \le 1 \quad \forall\, p \tag{1.3}$$

$$x_b \in \{0,1\} \quad \forall\, b \tag{1.4}$$

The objective function (1.1) maximizes the total seller's revenue, subject to the capacity constraint on each object (1.2) and the requirement that not more than one bid per bidder is accepted (1.3).

There are several alternative formulations (deVries and Vohra 2000) that require an explicit enumeration of all possible subsets of $\mathbf{I}$. A main difference in our formulation is that we do not actually assume that every possible subset of $\mathbf{I}$ is included in the model. In practice, when $I = |\mathbf{I}|$ is large, the number of subsets of $\mathbf{I}$ is extremely large, and bidders are likely to express valuations for only a small number of subsets. A disadvantage of our formulation is that if some economically important subset is inadvertently omitted from the model, the solution is sub-optimal.

The general case of the winner determination problem is NP-hard, since it is a special case of the set packing problem. There are certain restrictions that can be placed on bids or on assumptions about bidder preferences, where it was shown that the resulting bidder determination problem is easily solvable (see Rothkopf et al 1998.).

Bidding Languages

A bidding language is a way of expressing bidder valuations for bundles (Nissan 2000). A useful language should be able to represent a wide class of valuations, while simultaneously it should be intuitive to use. A good way to evaluate a potential language is to test how it can be used to represent several different valuations. Nissan 2000 shows that most simple languages are not able to simultaneously efficiently represent such common valuations as ones with complementarities and substitutability, simple additive, and some others.

Nissan 2000 summarizes the basic bidding languages. All bidding languages include a concept of an *atomic bid*. An atomic bid is simply one subset of items and a price representing the willingness to pay for that subset. Note that a language that includes only atomic bids cannot represent some very simple valuations, such as additive. In a more general language, atomic bids are combined together using some combination of the XOR/OR logic. The following is a list of some common bidding languages.

- OR Bids: is a language where a bidder can place any number of atomic bids and is willing to accept any combination of these bids.
- XOR bids: here each bidder can submit any number of atomic bids, and only one of them can be accepted. The XOR bids can describe all valuations. However, the number of atomic bids required to describe additive valuation on m items using XOR bids is 2^m.
- OR-XOR bids: a bidder can submit any number of XOR bids, called *clusters*, and is willing to accept any number of these clusters.
- XOR-OR bids: a bidder can submit a number of OR bids and is willing to accept just one of them
- OR* bids, also called OR bids with phantom items was originally introduced by Fujishima 1999. The idea is to allow XOR bids to be expressed as OR bids. A bidder submits any number of OR bids, and is allowed to include an additional constraint on any group of bids, saying that only one of them can be accepted. This constraint is included implicitly by creating a

dummy item and including it into all mutually exclusive bids. The capacity constraint on the dummy item insures that only one of the group of bids can be accepted.

Data Structure

This section lays out the basic structure for a combinatorial bidding system, including the interface between the data and the model. The basic Entity-Relationship diagram is shown in Figure 1(a) and the Data Structure diagram is shown in 1(b).

The auction starts out with the BIDDABLE_OBJECT_INFO table populated. Generally, objects may have a capacity other than 1, although in all previous discussion we were assuming capacities of 1. As bidders come in they register, creating an entry in the BIDDER_INFO table. They may have an option of specifying a budget, and in this case the system insures that the total amount paid will not exceed this budget. Budgets require separate constraints. When bidders place their bids, entries are made in the BID_INFO and BID_COMPOSITION tables. Whenever XOR (mutually exclusive) bids are placed, an entry is also made into the BIDDABLE_OBJECT_INFO table, for the corresponding dummy object, and this object is added to the bids with the appropriate entry into the BID_COMPOSITION table.

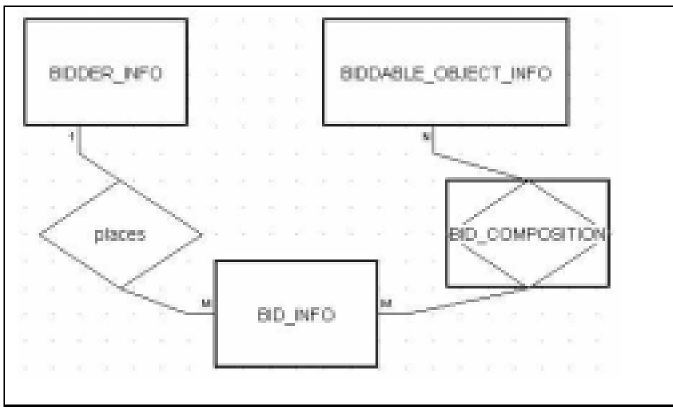

Figure 1(a) The Entity Relationship Diagram

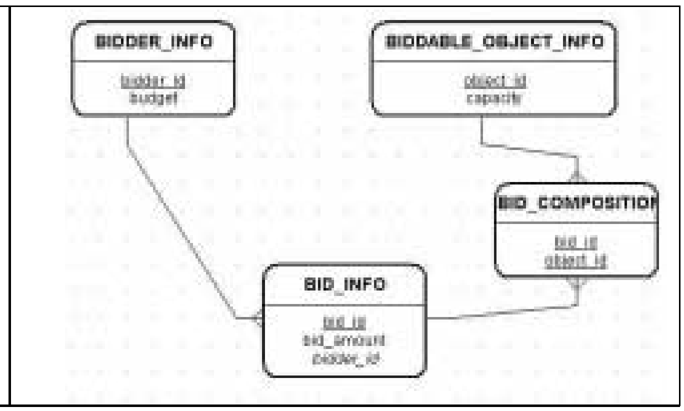

Figure 1(b) The Data Structure Diagram

Data/Model Interface

To clarify the connection between the data structure and the model, we rewrite the model using the algebraic modeling notation, and include the SQL statements that retrieve this data in the form appropriate for the use in the model

Indices

I Items (or objects)
SELECT object_id from biddable_object_info;
P Bidders
SELECT bidder_id from bidder_info;
B Bids
SELECT bid_id from bid_info;

Index Sets

B_p all bids bade by bidder p,
SELECT bid_id, bidder_id from BID_INFO;
$\mathbf{B}_i$ all bids that include item I
SELECT bid_id, object_id from BID_COMPOSITION;

Data

c_{bp} bid amount of bidder p on bid b
SELECT bid_id, bidder_id, bid_amount from bid_info;
r_i the number of units of object i available. This is often 1
SELECT object_id, capacity from BIDDABLE_OBJECT_INFO;

Variables

x_b 1 if bid b wins and 0 otherwise

Objective Function

$$\max \sum_{p \in \mathbf{P}, b \in \mathbf{B}_p} c_{bp} x_b$$

Maximizes the total revenue.

Capacity Constraint for Each Object

$$\sum_{b \in \mathbf{B}_i} x_b \le r_i \quad \forall\, i \in \mathbf{I}$$

The number of winning bids that include object i cannot exceed the number of units of object i available.

Note that since all mutually exclusive bids now include a common dummy object, we do not need a restriction that each bidder can have at most one winning bid.

Conclusion

We described a bidding language capable of expressing a large variety of valuations in combinatorial auction. We also present a design for the user interface for bid creation that allows users to submit a large number of bids with little effort. The bid creation module serves as a front end to a general combinatorial auction system that can be used for conducting multi-round combinatorial auctions. The winner determination problem is solved using an integer programming model implemented through an SQL-based interface with an algebraic modeling language, making it quick to develop, easy to modify and flexible to implement.

References

DeVries, S. and R. Vohra. “Combinatorial auctions: A survey.” *Working Paper*, 2000.
Fujishima, Y., K. Leyton-Brown and Y. Shoham, “Taming the Computational Complexity of Combinatorial Auctions: Optimal and Approximate Approaches", *Proceedings of IJCAI-99*, Stockholm, 1999.
Nisan, N. “Bidding and Allocation in Combinatorial Auctions", *EC2000*, 2000.
Rothkopf, M. H., A. Pekec and R. M. Harstad, “Computationally Manageable Combinatorial Auctions", *Management Science*, vol. 44, No. 8, 1131-1147, 1998.