An Expressive Auction Design for Online Display Advertising

Sébastien Lahaie
Yahoo! Research
New York, NY 10018

David C. Parkes
School of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138

David M. Pennock
Yahoo! Research
New York, NY 10018

Abstract

We propose an expressive auction design that allows advertisers to specify the kinds of demographics and websites they wish to target within an advertising network. The design allows the network to differentiate impressions according to relevant attributes (e.g., geographic location of the user, topic of the webpage). Advertisers can place bids for different kinds of impressions according to their attributes, and can also specify volume constraints to control exposure. The novelty of the design is a bidding language that admits scalable allocation and pricing algorithms. We discuss the incentive properties of different pricing approaches. We also propose a bidder feedback mechanism to mitigate the complexity of expressive bidding.

Introduction

Advertising networks such as those run by Ad.com, Google, and Yahoo! provide matching functions that display banners throughout websites in a network of affiliates. The matching is based on the nature of the advertisement, and a bid provided by the advertiser. The trend in display advertising is to allow advertisers to specify what kinds of users they are targeting, for example by geographic location, interests, and even online behavior (Story 2007). This points to the need for auction mechanisms that can allow bidders to express complicated valuations over user attributes.

The design of expressive auctions is an interdisciplinary endeavor that draws on ideas from microeconomics and computer science, in particular artificial intelligence. AI techniques have found application in two key design areas:

- *Bidding languages.* Bidders cannot exhaustively describe their values for every possible outcome. A language (syntax and semantics) is needed that can succinctly describe realistic valuation functions. The language is usually specified in terms of data structures that allow value information to be efficiently queried (Cavallo *et al.* 2005; Boutilier & Hoos 2001; Nisan 2000).
- *Allocation algorithms.* Given representations of the bidders' preferences, an efficient (value-maximizing) allocation must be computed, and in practical settings such as combinatorial auctions this can be NP-hard. Effective algorithms and heuristics are needed that perform well on realistic problem instances (Boutilier 2002; Sandholm *et al.* 2005).

Our main contribution is to the bidding language literature, because the properties of our auction hinge on the structure of our language. Our language is specifically designed to admit tractable allocation algorithms.

We propose an expressive auction design for online display advertising. The auction allocates a forecasted supply of different kinds of impressions (views of an advertisement); impressions are differentiated according to the conditions under which they occur (e.g., topic of the webpage or geographic location of user). The supply is forecast over a fixed time period such as a week. If market conditions do not change, the computed allocation can be interpreted as the expected amount of impressions that each bidder will receive in the period. At any given time, however, it is best to interpret the allocation as the *rate* at which bidders will receive different kinds of impressions. Our view is that the allocation should be recomputed at fixed time intervals (e.g., daily), and also if there are significant changes in market conditions, such as entry or exit of large bidders.

The auction is meant to be complemented with a scheduling algorithm. The scheduler tries to display advertisements over time in such a way that the realized allocation rate matches the desired rate computed by the auction. This scheduling problem is an orthogonal issue that we do not address in this paper.

The core of our design is a bidding language that allows bidders to specify values for different kinds of impressions, and also to place volume constraints on impressions to control exposure. Besides expressiveness, the language is designed with incentives and computational tractability in mind. It allows for:

1. Efficient and scalable allocation algorithms, so that allocations can be quickly recomputed when needed
2. Unique market-clearing prices that minimize the bidders' incentives to game the auction
3. Efficient, combinatorial algorithms for computing these specific market-clearing prices

We also propose a bidder feedback mechanism that can recommend bid increases to agents who would like to achieve

greater volume of certain demographics.

The auction computes an "efficient" (value-maximizing) rather than "optimal" (revenue-maximizing) allocation.[1] We focus on efficiency because there are currently several competing ad networks, so it is unclear whether it is feasible to exercise monopoly power in this landscape. Still, it may be possible to modify our design to extract more revenue; this would involve weighting bidders so that their bids are not all considered equally. This kind of approach has already been suggested for sponsored search auctions (Lahaie & Pennock 2007), and its application to the display advertising domain is still an open and challenging problem.

Preliminaries

An *impression* occurs when a user observes an ad. The impression is distinguished by the conditions in which the ad was observed, for example:

- the user is in California
- the webpage has political content
- the time of day is "evening"

and so on. Formally, there is a set of *attributes* $\mathcal{A} = \{A_1, \ldots, A_s\}$, and each attribute is a set of *values*; for example, the "state" attribute would be $\{\text{CA}, \text{MA}, \text{NY}, \ldots\}$. Distinct attributes are disjoint.[2] The attributes and their possible values are determined by the seller (the network). An impression is a tuple $\langle a_1, \ldots, a_s \rangle$ such that $a_t \in A_t$ for $t = 1, \ldots s$. Let M be the set of possible impressions (tuples), with cardinality $m = \Pi_{t=1}^{s} |A_t|$.

Let $N = \{1, \ldots, n\}$ be the set of bidders (advertisers). An element of $\mathbf{Z}_+^M$ represents a bundle of impressions. The notation $\mathbf{Z}_+^M$ denotes the set of vectors with entries indexed by elements of M and drawn from $\mathbf{Z}_+$ (the non-negative integers). For $x_i \in \mathbf{Z}_+^M$, we denote the entry corresponding to impression $j \in M$ by $x_i(j)$. Each bidder i has a valuation defined over bundles, $v_i : \mathbf{Z}_+^M \to \mathbf{R}_+$. The total price of a bundle of impressions x_i at prices $p \in \mathbf{R}_+^M$ is the usual scalar product $p \cdot x_i = \sum_{j \in M} p(j) x_i(j)$. Bidders have quasi-linear utilities, so that the utility to bidder i of bundle x_i at prices p is $v_i(x_i) - p \cdot x_i$.

To plan an allocation of impressions, we assume the seller has an estimate of the number of units available of each impression over some fixed time period. Let $z(j)$ be the estimate for the supply of impression $j \in M$. The objective is to compute an efficient allocation of the forecasted impression units together with market-clearing prices. An allocation is a vector of bundles $x = (x_1, \ldots, x_n)$. An allocation is *feasible* if $\sum_{i=1}^{n} x_i(j) \leq z(j)$ for all $j \in M$ (an impression does not have to be allocated: the ad space can be left blank). Let Γ be the set of feasible allocations. An allocation x is *efficient* if

$$x \in \arg\max_{y \in \Gamma} \sum_{i=1}^{n} v_i(y_i).$$

Prices are used to bring a level of stability to the agents' bids. They also provide information on the cost of different kinds of impressions to possible new entrants. Our auction quotes market-clearing prices. At clearing prices, each agent prefers his own bundle of impressions to any other possible bundle, and the allocation maximizes the seller's revenue. In this sense, demand equals supply, and bids remain stable as long as agents act as pure price-takers. Market-clearing prices are called *competitive equilibrium* (CE) prices. Let $D_i(p) = \arg\max v_i(x_i) - p \cdot x_i$ be the set of utility-maximizing bundles for bidder i at prices p, its "demand set." A CE is an allocation-price pair $\langle x, p \rangle$ such that $x_i \in D_i(p)$ for each $i \in N$, and such that each $j \in M$ with $\sum_{i \in N} x_i(j) < z(j)$ has $p(j) = 0$.

The Fundamental Welfare Theorems state that, if a CE exists—a property that depends on the model and price space in general—then (1) if p are CE prices, $\langle x, p \rangle$ is a CE for any efficient allocation x, and (2) if $\langle x, p \rangle$ is a CE, x is efficient (Bikhchandani & Ostroy 2002). Computing an efficient allocation is therefore consistent with the objective of computing market-clearing prices.

It may seem that the number of different impressions m would be too large to make the design practical: m is exponential in the number of attributes. Note though that it does not make sense to let advertisers bid for impressions that only exist in very low volumes, because decent supply forecasts for such impressions would not be possible. This restricts the number of different impressions that can be usefully allowed. (Bidders could be forbidden to bid on impressions j whose $z(j)$ is too small, presumably through the bidding interface.) Viewing the allocation problem as a combinatorial auction, the items here are impressions (combinations of attributes values), hence prices are specified over impressions rather than attributes.

Bidding Language

We first describe the bidding language in terms of the data structures that would be used to encode its instances, and then turn to the properties of the valuations it describes. We do not discuss the actual bidding interface; there are many conceivable interfaces for our language.

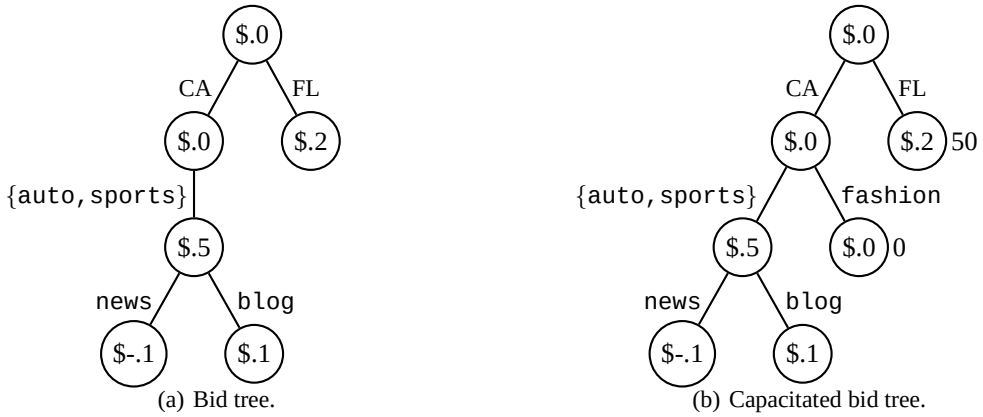
Bid Trees

To encode valuations, we propose "bid trees" that enable advertisers to specify values for various kinds of impressions.[3] Bid trees can be used to encode both the advertisers' true valuations and their *reported* valuations, or "bids". The two

[1] Efficient can mean value-maximizing, when referring to allocations, or polynomial-time, when referring to algorithms. It should be clear from context which definition applies.

[2] Certain attributes could be hard to measure for a given impression; for example, we may not be able to identify the geographic location of a user. Each attribute can have its own "undefined" value to handle such cases.

[3] Tree-based languages have been proposed for combinatorial auctions, such as the $\mathcal{L}_{GB}$ language of Boutilier & Hoos (2001), as well as for combinatorial exchanges, such as TBBL by Cavallo *et al.* (2005). The language proposed here represents a smaller class of valuations than either of those languages, but is specially tailored to the domain of display advertising.



(a) Bid tree. (b) Capacitated bid tree.

Figure 1: Instances of the bidding language.

need not coincide, as we explain later when addressing incentives, although for clarity we assume for now that they do.

For instance, suppose a car manufacturer wishes to run an online campaign to advertise a new truck model. The campaign should only run in California and Florida. In Florida, the campaign should have a limited exploratory run across a variety of sites, to see what demographics respond best to the new model. In California, the campaign should project a "rugged" image for the truck. The company therefore decides to run its banner ad next to content with an "automotive" or "sports" theme, and to avoid any content with a "fashion" theme. It also decides to value exposure on blogs and devalue mainstream news sites, because it would like the campaign to have "grassroots" appeal.

A candidate bid tree for this kind of valuation is given in Figure 1(a). The root node represents all possible impressions over the network. The value of an impression from a certain source is evaluated by traveling down the tree following attributes that apply, starting at the root, and summing the values in the nodes along the way. For instance, impressions on automotive or sports sites in CA are valued at $\$.4$ if they are news sites, $\$.6$ if they are blogs, and $\$.5$ otherwise.

An advertiser constructs its bid tree by starting with a root node, and then creating more nodes by repeatedly branching on an attribute at a leaf. An advertiser may branch on only some values of an attribute A_t rather than all values for A_t, if it chooses. The advertiser may also branch on sets of attributes, as long as the sets are disjoint; for instance, separate branches for $\{\text{CA}, \text{FL}\}$ and $\{\text{FL}, \text{NY}\}$ are not allowed, as this leads to ambiguous semantics. An attribute may not be branched on at a node if it was already branched on at an ancestor node.

But there may still be a disconnect between the bid tree in Figure 1(a) and the company's valuation. Exposure outside of CA or FL is worthless, but does not do any harm if it occurs, so it indeed has value \$0. On the other hand, exposure on fashion sites in CA does do harm, because it goes against the brand image the advertiser is trying to project in that state. Also, there is no telling how many impressions will come from FL, even though the run there should be limited. Hence we further allow advertisers to annotate nodes with capacities. In Figure 1(b), there is a new node for fashion sites in CA, with a capacity of 0 to ensure no impressions are provided.The FL node now has a capacity of 50,000 impressions to ensure the campaign there is limited to this exposure.

Properties

The value functions encoded by bid trees can be described formally as follows. Given bidder i's bid tree, we can define a family $\mathcal{T}_i$ of subsets of M corresponding to each node. For instance, for the tree of Figure 1 we would have the set of all impressions that occurred in CA, the set of all impressions on automotive or sports sites in CA, the set of all impressions on automotive or sports blogs in CA, etc. It is straightforward to see that, because of the tree format, this family is *laminar*: for any $T, T' \in \mathcal{T}_i$, either $T \subseteq T'$, $T' \subseteq T$, or $T \cap T' = \emptyset$. To each $T \in \mathcal{T}_i$ is associated an integral capacity c_{iT} (possibly $+\infty$) and a value b_{iT}. Define the functions

$$v_{iT}(r) = \begin{cases} b_{iT} r & \text{if } 0 \leq r \leq c_{iT} \\ -\infty & \text{otherwise} \end{cases} \quad (1)$$

for each $T \in \mathcal{T}_i$. Here r is a non-negative scalar, representing a certain number of impressions.

The agent's value for a bundle of impressions x_i is then

$$v_i(x_i) = \sum_{T \in \mathcal{T}_i} v_{iT}(x_i(T)) \quad (2)$$

where $x_i(T)$ is shorthand for $\sum_{j \in T} x_i(j)$, namely the total of all impressions from T. The volume constraints make the valuation functions nonlinear. Without loss of generality, we can assume that $\mathcal{T}_i$ contains a node $\{j\}$ for each $j \in M$, whose capacity is no more than $z(j)$.

Danilov *et al.* (2001), generalizing results of Kelso & Crawford (1982), show that valuations of the form (2), where $\mathcal{T}_i$ is laminar and each v_{iT} exhibits decreasing marginal values over some interval (and is $-\infty$ outside this interval), are "M-concave." The actual specification of this condition is not important for our treatment. Intuitively, it

implies that impressions are "substitutes": if the price of an impression $j \in M$ is increased, an agent's demand for the other impressions does not decrease, because the agent substitutes away from j.

This M-concave property is a main motivation for adopting bid trees, besides their natural expressiveness. It is well-known that it implies the existence of a CE (Danilov, Koshevoy, & Murota 2001; Kelso & Crawford 1982), and it also allows for fast allocation and pricing algorithms.

Queries

There are two fundamental queries that are typically made on bidding languages: value and demand queries. Each query is a fundamental subroutine of the allocation and pricing algorithms discussed later, so it is important to understand their complexity.

Value. On a value query, a bundle of impressions $x_i \in \mathbf{Z}_+^M$ is input and the value $v_i(x_i)$ according to the bid tree is output.

Demand. On a demand query, prices $p \in \mathbf{R}_+^M$ are input, and some utility-maximizing bundle at prices p is output.

Clearly, the time to evaluate the response to a value query is at most the depth of the bid tree. This is at most $t_i = |\mathcal{T}_i|$, the size of the bid tree, but can be $O(\log t_i)$ if the tree is balanced.

From a valuation of the form (2) derived from a bid tree, the response to a demand query can be computed using Algorithm 1. For $j \in M$, χ_j denotes the unit vector with entry j being 1 and all others 0. Let $\mathbf{0}$ be the zero vector. Let $w_i(j) = \sum_{j \in T} b_{iT}$ be the marginal value from an impression on site j, and let $\pi_i(j) = w_i(j) - p(j)$ be the marginal surplus.

Input: Prices $p \in \mathbf{R}^M$.
Output: A set of impressions $x_i \in D_i(p)$.
Set $x_i := \mathbf{0}$.
For each $T \in \mathcal{T}_i$, set $d_T := c_{iT}$.
Discard the elements $j \in M$ that have $\pi_i(j) < 0$.
Sort the remaining elements according to π_i. In case of a tie, break arbitrarily.
foreach $j \in M$ *in order* **do**
  Set $k := \min_{\{T \in \mathcal{T}_i \mid j \in T\}} d_T$.
  Set $x_i := x_i + k\chi_j$.
  foreach $T \in \mathcal{T}_i$ *such that* $j \in T$ **do**
    Set $d_T := d_T - k$.
  end
end

Algorithm 1: Greedy algorithm for demand queries on bid trees.

Algorithm 1 is a greedy algorithm that computes a response to a demand query. It considers impressions in decreasing order of marginal utility, and collects as much as possible of each impression until volume bounds are reached. Including the sort, the greedy algorithm has a worst-case running time of $O(m \log m + mt_i)$, where t_i is the size of the bid tree. The correctness of the algorithm is recorded as a lemma. Proofs are collected in an appendix (available from the authors).

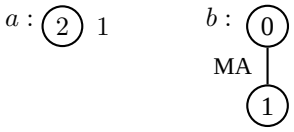
Lemma 1 *Algorithm 1 correctly outputs a set of impressions $x_i \in D_i(p)$.*

Among the common bidding languages surveyed by Nisan (2000), only XOR allows for polynomial-time demand queries; for the others, evaluating a demand query is NP-hard. The XOR language does not seem appropriate here, however. It does not provide the ability to specify volume constraints, and cannot succinctly represent the additive valuation, which is plausible in this domain.

Allocation

A natural way to allocate impressions in an online fashion is to give each arriving impression to the advertiser who values it most among those advertisers for whom the extra impression would not violate any volume constraints. This scheme is not efficient in our context. Volume constraints are the source of the complication, as the following example illustrates.

Example. There are two bidders $\{a, b\}$ and two impressions available, one from Massachusetts (MA) and one from California (CA). The valuations as bid trees are



If the MA impression comes before the CA impression, the greedy scheme assigns MA to a and CA to b, for a value of 2. But it is optimal to assign MA to b and CA to a, for a value of 3.

The scheme fails because it does not take into account forecasted supply. Recall that our auction computes a desired allocation over a fixed time period (equivalently, desired allocation rates), and passes the result to a scheduler. The efficient allocation problem can be formulated as a linear program:

$$\max_x \quad \sum_{i \in N} \sum_{T \in \mathcal{T}_i} \sum_{j \in T} b_{iT} x_i(j)$$

$$\text{s.t.} \quad \sum_{j \in T} x_i(j) \le c_{iT} \qquad (T \in \mathcal{T}_i, i \in N) \quad (3)$$

$$\sum_{i \in N} x_i(j) \le z(j) \qquad (j \in M) \quad (4)$$

$$x_i(j) \ge 0 \qquad (i \in N, j \in M)$$

Here (3) enforces the bidders' capacity constraints and (4) ensures that the number of impressions does not exceed the supply. Because the agents' valuations are M-concave, this linear program in fact has an integer optimal solution. In our context this is not important, because the $z(j)$ are estimates anyway, and presumably large. However, this fact does make the auction design also suitable to situations where there is a small, fixed number of impressions available (e.g., in a television station's daily programming schedule), so it is recorded here as a proposition.

Proposition 1 *When agents submit their valuations as bid trees, the corresponding allocation LP has an integer optimal solution.*

A solution to the allocation LP can be found in polynomial time using the Ellipsoid Method, and standard approaches such as the simplex method should perform well in practice. However, the bidding language was designed for use with the *subgradient method*, which has several advantages in this domain. The subgradient method operates on the dual of the allocation LP, although both primal and dual optimal solutions result. It updates prices according to the rule

$$p^{k+1} = p^k + \beta_k g^k,$$

where p^k is the kth iterate of the prices. Let $x_i^k \in D_i(p^k)$ for each $i \in N$, and y^k be a revenue-maximizing allocation at prices p^k. The "subgradient" g^k can be chosen as $y^k - \sum_i x_i^k$. Note that if $g^k = \mathbf{0}$, demand equals supply and p are market-clearing prices. There are various approaches for selecting the stepsize β_k, which typically depends on the Euclidean norm of g^k (smaller norms imply smaller stepsizes); see for example Chapter 6.3 of Bertsekas (1999). To be clear, the iterates p^k are never actually quoted as prices, except perhaps for the final one. Different iterates should not be confused with different prices quoted over time because of changing market conditions. The final allocation and prices can be implemented directly, or used as inputs to a procedure that computes special CE prices (see the next section). Although Proposition 1 only guarantees the existence of a single integer optimal solution, the subgradient method still converges to such a solution because the iterates x^k are always integer.

The first advantage of this method is that computing the subgradient reduces to a series of demand queries, which could be parallelized, and each demand query can be evaluated efficiently by Lemma 1. A second advantage is that if bid trees change, the price computation can be restarted from the current price vector rather than some default such as $\mathbf{0}$. If the changes to the bid trees are slight, only a few rounds should be required to converge to the new prices.

Pricing

The linear programming approach to the allocation problem is useful because it also provides prices for various kinds of impressions. Let $p(j)$ be the dual variable corresponding to the constraint for j in (4) in the allocation LP.

Proposition 2 *If x is an optimal primal solution to the allocation LP and p is an optimal dual solution, then $\langle x, p\rangle$ is a competitive equilibrium.*

The set of CE prices will typically not be unique. In our model, because each agent's valuation is M-concave, the set of competitive equilibrium prices is in fact a lattice under the usual meet and join operations for real vectors.

Proposition 3 *The set of competitive equilibrium prices is a lattice when valuations are described by bid trees.*

The lattice property is a key feature of our design. It allows the auctioneer to be consistent in his choice of prices. Since a lattice has a unique minimal element $\underline{p}$ and a unique maximal element $\bar{p}$, the auctioneer may choose to consistently implement either of these. The minimal element gives the most possible surplus to the bidders, while the maximal element gives the most possible revenue to the seller, among the set of CE prices. Because the set of dual solutions to an LP is convex, so is the set of CE prices, and so $\alpha\underline{p} + (1-\alpha)\bar{p}$ is also a vector of CE prices, for any $\alpha \in [0, 1]$. This allows the auctioneer to also choose to modulate the allocation of surplus between the bidders and seller. Because bid trees represent M-concave valuations, there exist purely combinatorial algorithms (i.e., that do not use floating-point arithmetic) for computing the minimal and maximal CE prices. Chapter 12 of Murota (2003) shows how the problem of computing either element can be formulated as the dual of a shortest path problem.

To save on bandwidth and lessen the burden on the auction infrastructure, we would like changes in the agents' bid trees to reflect actual changes in their valuations, rather than gaming behavior. Competitive equilibrium prices are useful because they ensure a certain stability in the bids. If agents act purely as price takers, they are satisfied with the given allocation, and no changes in their trees are needed. The auctioneer is also satisfied because no impression that could have generated more revenue goes unallocated.

Of course, agents may realize that they are not in fact price-takers, and that the prices—being dual variables of the allocation LP—should vary as the bid trees are changed, either in structure or value. It is therefore instructive to consider what incentives the agents may have to alter their bid trees. Ideally, we would like to reach a scenario where the agents are satisfied with their current bid trees as far as the allocation and prices that result. The next section addresses this design issue.

Incentives

If no agent would gain by adapting its bid tree, no matter how the other agents behave, this is known as a "dominant strategy equilibrium" in pure strategies. A classic way to achieve this in multi-agent systems is through VCG payments. Unfortunately, it may not be possible to charge VCG payments in a competitive equilibrium in our model, because the price space is not rich enough—we only consider linear prices. Bikhchandani & Ostroy (2002) show that in general, nonlinear and personalized prices may be needed to price the VCG payoff point when agents have M-concave valuations. This kind of pricing does not seem appropriate for this domain, because bidders can enter or leave the system at any time in a typical online ad auction, and so it is necessary to provide informative prices to new arrivals. Personalized prices provide no useful information to new entrants. Laboratory experiments suggest that linear prices are very informative: they are easy to interpret, and apply to everyone simultaneously (Porter *et al.* 2003).

An alternative then is to quote the linear prices that minimize the incentives for bidders to adapt their bid trees.[4] This

[4]Formally, the incentive to deviate for agent i is the maximum increase in utility ϵ_i that can be achieved by switching to another

may be a good compromise if bidders are bounded-rational and would not notice or bother to switch when this would yield just a small improvement in payoff. In this case, Parkes *et al.* (2001) have shown that minimal CE prices maximize the incentives for truthful reporting. Therefore, implementing the smallest linear CE price vector $\underline{p}$ leads to the most "stable" system in a sense, if we restrict ourselves to linear prices. Again, the lattice property proves convenient. If there were multiple minimal CE price vectors, there would be the added problem of choosing among these, and each implies a different distribution of surplus among the bidders. Distributing surplus is a sensitive question: for instance, should the seller favor small or large bidders? The difficulty is compounded by the fact that once a selection is made, algorithms must be developed to compute the desired prices.

Bidder Feedback

The discussion on incentives in the previous section assumed—as is standard in game theory and mechanism design—that bidders are perfectly rational and know their valuation functions exactly. In practice, bidders vary in their level of sophistication. Some bidders may not have exact value information, or may find it costly to place a precise value on different impressions. It may be the case that such advertisers only indirectly know their valuations through *demand* information: given current prices, they know how much volume they would like of different kinds of impressions, but have not converted their demand function into a valuation function. In this section, we describe a simple bidder feedback mechanism that can suggest bid tree updates to bidders who want to increase their volume for different kinds of impressions.

Suppose bidder i wants to receive at least d_{iT} impressions from sites in $T \subseteq M$, where $T \in \mathcal{T}_i$ is a node in bidder i's bid tree. For example, the advertiser with the bid tree in Figure 1(b) may only receive 10,000 impressions from FL, and want to increase FL impressions to 20,000. Naturally, the advertiser should first ensure that the volume constraint for the node is $c_{iT} \geq d_{iT}$. If this change still does not give the desired volume, the bid b_{iT} should be increased.

Suppose we introduce the constraint

$$\sum_{j \in T} x_i(j) \geq d_{iT} \tag{5}$$

into the allocation LP to find an efficient allocation. Let λ be the optimal value of the dual variable corresponding to this constraint. The following proposition confirms that λ is informative feedback to the bidder.

Proposition 4 *Suppose bidder i increases b_{iT} to $b_{iT} + \lambda$ in its bid tree, and leaves the bids in the other nodes unchanged. Then there exists an efficient allocation, with respect to the new profile of bid trees, in which i receives at least d_{iT} units of impressions from $T \subseteq M$.*

bid tree, keeping the other agents' trees fixed. We wish to minimize $\max_{i \in N} \epsilon_i$.

The proposition shows that, just as bids can be specified at various levels of granularity, local updates to the bid tree can affect volumes at different levels of granularity. For instance, if a bidder wishes to increase overall impressions regardless of origin, the feedback scheme would suggest an appropriate increase at the root node.

Conclusion

We proposed an expressive auction design for the domain of display advertising, for use within advertising networks. At the core of our design is a bidding language that allows advertisers to specify values for different kinds of impressions, and that admits scalable allocation and pricing algorithms. The language does not force the advertisers to refine their values according to irrelevant attributes, but rather allows them to specify bids at different levels of granularity. Volume constraints give the advertisers even more control over their campaigns. There is also the possibility of providing feedback (again, at different levels of granularity), to help the advertisers assess the cost of volume increases at different nodes.

References

Bertsekas, D. P. 1999. *Nonlinear Programming.* Belmont, MA: Athena Scientific.

Bikhchandani, S., and Ostroy, J. M. 2002. The package assignment model. *Journal of Economic Theory* 107:377–406.

Boutilier, C., and Hoos, H. H. 2001. Bidding languages for combinatorial auctions. In *18th International Joint Conference on Artificial Intelligence (IJCAI)*, 1211–1217.

Boutilier, C. 2002. Solving concisely expressed combinatorial auction problems. In *19th National Conference on Artificial Intelligence (AAAI)*, 359–366.

Cavallo, R.; Parkes, D. C.; Juda, A.; Kirsch, A.; Kulesza, A.; Lahaie, S.; Lubin, B.; Michael, L.; and Shneidman, J. 2005. TBBL: A tree-based bidding language for iterative combinatorial exchanges. In *IJCAI Workshop on Preference Handling*.

Danilov, V.; Koshevoy, G.; and Murota, K. 2001. Discrete convexity and equilibria in economies with indivisible goods and money. *Mathematical Social Sciences* 41(3):251–273.

Kelso, A. S., and Crawford, V. P. 1982. Job matching, coalition formation, and gross substitutes. *Econometrica* 50:1483–1504.

Lahaie, S., and Pennock, D. M. 2007. Revenue analysis of a family of ranking rules for keyword auctions. In *8th ACM Conference on Electronic Commerce (EC)*, 50–56.

Murota, K. 2003. *Discrete Convex Analysis*. SIAM.

Nisan, N. 2000. Bidding and allocation in combinatorial auctions. In *second ACM Conference on Electronic Commerce (EC)*, 1–12.

Parkes, D. C.; Kalagnanam, J.; and Eső, M. 2001. Achieving budget-balance with Vickrey-based payment schemes in exchanges. In *17th IJCAI*, 1161–1168.

Porter, D.; Rassenti, S.; Roopnarine, A.; and Smith, V. 2003. Combinatorial auction design. *Proceedings of the National Academy of Sciences* 100:11153–11157.

Sandholm, T.; Suri, S.; Gilpin, A.; and Levine, D. 2005. CABOB: A fast optimal algorithm for winner determination in combinatorial auctions. *Management Science* 51(3):374–390.

Story, L. 2007. Online customized ads move a step closer. *New York Times*. July 2: Technology.