Comparing Position Auctions Computationally

David R. M. Thompson and **Kevin Leyton-Brown**
Department of Computer Science, University of British Columbia
2366 Main Mall, Vancouver, B.C., Canada, V6T 1Z4
{daveth,kevinlb}@cs.ubc.ca

Abstract

Modern techniques for representing games and computing their Nash equilibria are approaching the point where they can be used to analyze market games. We demonstrate this by showing how the equilibria of different position auction mechanisms can be tractably identified using these techniques. These results enable detailed and quantitative comparisons of the different auction mechanisms—in terms of both efficiency and revenue—under different preference models and equilibrium selection criteria.

Introduction

As the means by which search engines sell advertising space, position auctions generate billions of dollars of revenue every year. As a result, there is considerable interest in understanding these markets well. It is quite standard to model auctions as noncooperative games; they can then be analyzed using the tools of game theory.

We contrast these tools with those of another framework that is even more common for representing high-stakes business processes: mathematical programming. Both frameworks are able to model a wide variety of interesting problems: in the former case, as normal-form games, Bayesian games, and extensive-form games; in the latter case, as linear programs, mixed-integer programs and quadratic programs. Likewise, powerful theoretical tools exist in both cases for analyzing these models: proving the existence or nonexistence of a solution; bounding a solution; transforming one model into another. However, a key difference lies in the applicability of computational methods to the analysis of a concrete problem instance. In the case of mathematical programming, it is commonplace to use software—such as the ubiquitous CPLEX package—to solve a MIP or LP encoding of a given problem. In contrast, it is very rare in economic analysis to sample from an underlying game distribution (e.g., of bidder valuations) and then to determine the desired solution concept (e.g., Nash equilibrium) computationally. This can be largely explained by two key obstacles: games of interest are enormous when written in a canonical form, and equilibrium-finding algorithms exhibit worst-case exponential runtimes even on these enormous inputs.

Our paper [Thompson and Leyton-Brown, 2009] shows that modern computational techniques have reached the point where they can nevertheless be applied to the analysis of large-scale, realistic auction problems. Specifically, we consider three prominent position auction mechanisms and four widely-studied valuation distributions. Our experimental results show that we can characterize revenue and efficiency properties in (exact) equilibrium even for game instances of practical size (e.g., 10 bidders, 8 slots at auction, 40 bid increments per bidder). While this method is unable to answer some questions that theoretical methods traditionally tackle (e.g., "Is this mechanism efficient?") it is conversely able to answer other questions that appear intractable for existing theoretical methods (e.g., "What fraction of efficiency does this mechanism achieve given realistic bids?") We therefore see it as a useful tool for the analytic toolbox, and hope that it might serve as an early step towards what one might term a "CPLEX for markets."

Position Auctions

The games we work with are specified by a position auction and a instantiation of a preference model. This is an advantage of the computational approach; we can feed our Nash equilibrium algorithms any combination of auction and preferences. We considered three types of position auctions that have seen large-scale use. In all three, ads are ranked by bid (highest bid gets top position) and an advertiser pays whenever his ad is clicked on. In the generalized first-price auction (GFP), advertisers pay their bids. In the unweighted generalized second-price auction (uGSP), they pay the next highest bid. The weighted generalized second-price auction (wGSP) calculates ranks and prices by weighting ads by their relative click probabilities (effectively ranking by expected revenue). None of these auctions is truthful [Aggarwal, Goel, and Motwani, 2006].

Our games must also include bidder preferences, which we generate by sampling from one of four models proposed in the literature. Under the first two models, EOS and V, wGSP has been shown to have efficient equilibria with more revenue than VCG [Edelman, Ostrovsky, and Schwarz, 2007; Varian, 2007]. Our results show how much revenue is gained,

as well as showing the existence of other equilibria. Under the other models, BHN and BSS, wGSP has been shown to sometimes have no efficient equilibrium [Benisch, Sadeh, and Sandholm, 2008; Blumrosen, Hartline, and Nong, 2008]. Our results quantify the magnitude and frequency of these efficiency losses. We also show how wGSP compares with uGSP and GFP in each model.

Representation

Consider again the example of mathematical programming. Large problems can have millions of constraints and variables; any non-sparse representation of the constraint matrix would be prohibitively large. Similarly, if we were to represent our largest position-auction games (with 10 players and 41 actions per player) in the normal form, each one would require over 900,000 terabytes! In order to be able to reason about realistic problems in practice, compact representations are crucial.

Action graph games (AGGs, [Jiang and Leyton-Brown, 2006]) are the critical technology that makes it possible for us to compactly represent our auction games. For example, the game just described has an AGG representation of only tens of megabytes. AGGs are like Bayes nets: they use independence to reduce large tables into smaller factors, whose relationships are represented by a directed graph. AGGs focus on context-specific independence, allowing an agent's choice of action to determine the variables upon which his payoff depends. In an action graph, every node corresponds to an action that one or more players can play. An arc from node a to node b implies that the payoff of any agent playing action b depends on how many agents played action a. When every action node has bounded in-degree the representation size of an AGG scales polynomially, and is exponentially smaller than the equivalent normal-form game.

Representing position auctions as AGGs is not trivial. A naive AGG representation could be nearly as big as the normal form: any bid can affect the price for a higher bidder or the position of a lower bidder, meaning the action graph would have to be almost fully connected. However, since each bidder's payoff is determined only by his position and price, we can use "function nodes" to calculate those values. Function nodes are special nodes in the action graph that do not correspond to any action. The value of a function node (used in computing the payoffs of playing neighboring actions) is determined by a function on the values of its neighbors. These functions can be understood as sufficient statistics over action counts. For example, a summation function node can be used to count the number of players playing a set of actions that all have the same cumulative effect on the other players. This can be used to reduce the in-degree of the action nodes, and therefore the representation size.

To calculate the position a bidder will obtain in an auction, we need only to count how many bids are greater than his and how many are equal. (Because bids are discrete, we must consider ties.) This can easily be done using summation function nodes (see Figure 1). In the case of a GFP, this is sufficient to compute an agent's payoff because his price will be determined by his own bid. In the case of a GSP, we

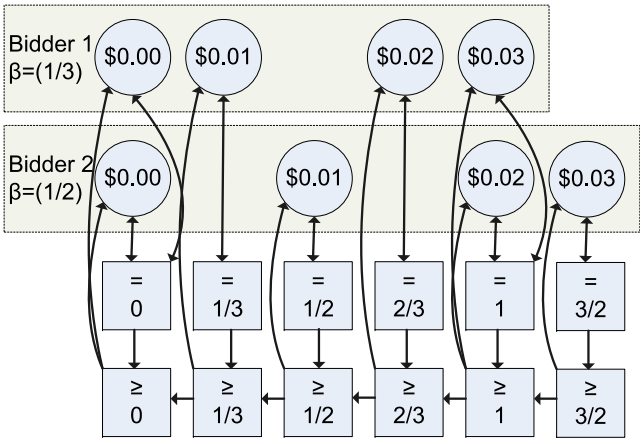

Figure 1: To compute the position a bid will win, we need to know how many bids were equal and how many were greater. This is calculated using summation nodes (squares).

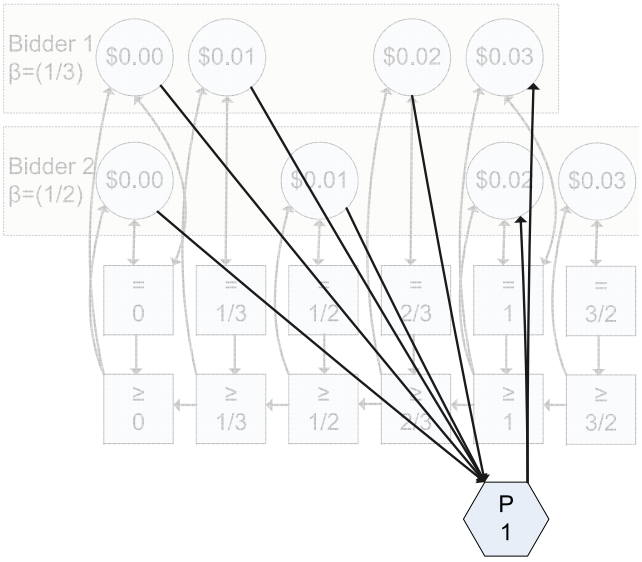

Figure 2: To compute a price, we must know the next highest bid. This is calculated using an argmax node (hexagon), for each bid. (Only one is shown for readability.)

also need function nodes to compute the price an agent will have to pay. We use an "argmax" function node, connected to every lower bid (see Figure 2). The argmax node's value is equal to the highest index of an edge with a non-zero value. This value identifies the next highest bidder and his bid, which is sufficient to compute a price.

We simplify the game by selectively removing weakly-dominated strategies: those in which agents bid above their own values. In general, removing weakly-dominated strategies can eliminate some Nash equilibria. However, in this case, the only equilibria that are removed are (1) those that lead to the same outcome as some equilibrium that was not removed, or (2) those that lead to some outcome that is

only reachable when at least one agent plays the weakly-dominated strategy of bidding higher than his own value.

Computing Nash Equilibria

Many families of mathematical programs are theoretically intractable, but routinely solved in practice by algorithms that achieve good empirical performance on realistic instances. To analyze position auctions computationally, we similarly need algorithms that perform well empirically. There has been recent research into the performance of algorithms on different game distributions [Nudelman et al., 2004; Porter, Nudelman, and Shoham, 2004]. Two of the most powerful algorithms are simpdiv [Scarf, 1967] and gnm [Govindan and Wilson, 2005]. Crucially, these have been adapted to work with AGGs [Jiang and Leyton-Brown, 2006]; as a result, their worst-case performance is exponential in the size of the AGG rather than the size of the normal form. We made a small change to both algorithms, imposing a simple static restart strategy (restarting every 5 minutes). This gained us a substantial overall speedup: for many instances, the fastest run took a few seconds while the slowest runs timed out.

One problem with both of these algorithms is that they return *arbitrary* Nash equilibria. In position auctions, there can be many Nash equilibria that lead to substantially different outcomes. For example, in a GSP, the bidder with the i^{th} highest value can choose any value between the $(i-1)^{th}$ bid and the $(i+1)^{th}$ bid, and the only effect will be to change the price for the $(i-1)^{th}$ highest bidder. There can be whole sets of Nash equilibria differing only in such bids, each yielding different revenue. To address this problem, while exploiting the fact that equilibria appear in connected regions of strategy space, we apply local search. Starting from each Nash equilibrium, we use a randomized hill-climbing algorithm to search through strategy space for equilibria with maximal/minimal revenue/social-welfare (four searches with different objectives). Considering all of these equilibria, we get a clearer picture of the range of equilibrium outcomes.

Results

One of the benefits of a computational analysis is that it allows us to make apples-to-apples comparisons between different auctions given the same bidder preferences. For example, we found that wGSP was the most efficient of the three position auctions, even in settings where it was known to be inefficient (see Figure 3). We also found that efficiency was relatively unaffected by equilibrium selection. (That is, the efficiency of wGSP in the EOS and V settings was not merely a property of envy-free equilibria; instead, all equilibria were very close to efficient.)

We also compared auctions in terms of revenue; here, the differences were less consistent. Theoretical results show that (with continuous values, and with bidders allowed to bid above their own values) wGSP has Nash equilibria with strictly more revenue than VCG, in the EOS and V models. The best equilibria we found did not consistently or substantially yield greater revenue than VCG, and the worst equilibria achieved substantially less revenue (see Figure 4). We also found that wGSP did not clearly outperform the other

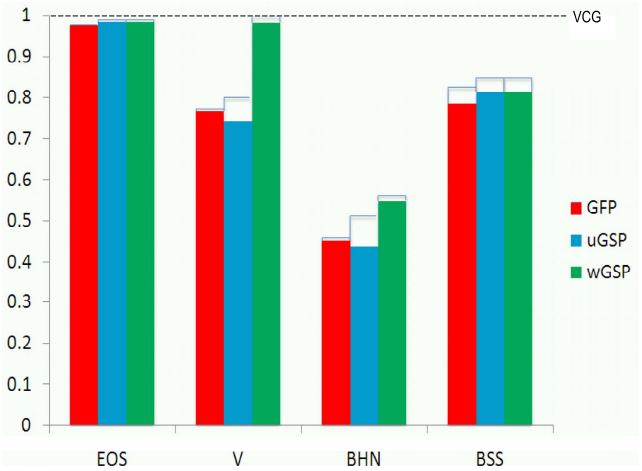

Figure 3: Average efficiency of the worst equilibria (solid bars) and best equilibria (outline bars). Equilibrium selection had little effect. wGSP is consistently best.

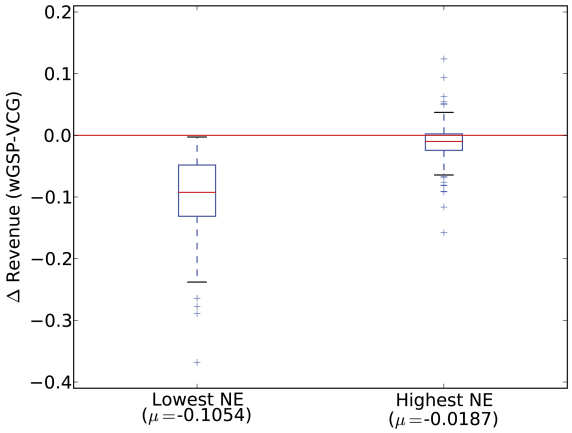

Figure 4: Revenue of wGSP relative to VCG in best and worst equilibria (for ESO preferences). In the best equilibria, wGSP does slightly worse than VCG on average. In the worst equilibria, wGSP does substantially worse. For V preferences, wGSP behaves similarly with slightly better best-case revenue.

position auctions; within a single preference distribution, the relative revenue could vary greatly from one instance to the next (see Figure 5). In short, we could not conclude that wGSP was a clear winner in terms of revenue; its revenue appeared to be very sensitive to equilibrium selection and details of the problem instance.

Discussion and Conclusions

Overall, we have demonstrated that it is now possible to compute the exact Nash equilibria of large position auction games, and to use those equilibria to provide quantitative an-

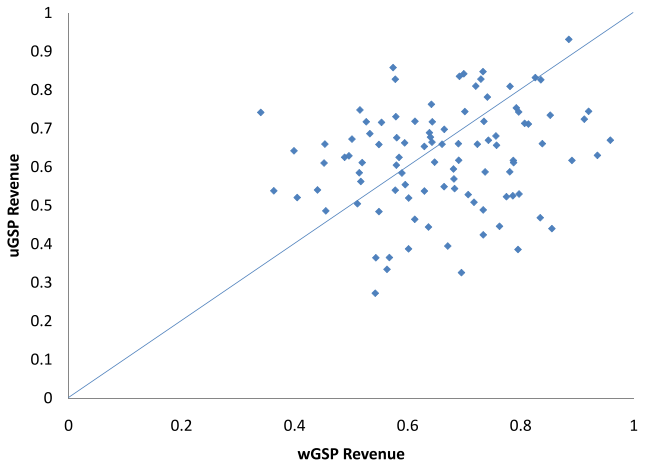

Figure 5: Per-instance (from distribution V) revenue of wGSP versus uGSP (in the best equilibrium). There is no clear winner.

swers to previously-open questions from the literature. This advance depended substantially on the recent development of action graph game, which greatly expanded the set of games that can be represented in a computationally-useful, polynomially-scaling form. We found that the widely-used weighted GSP auction was consistently the most efficient position auction, but that its revenue performance was highly sensitive to equilibrium selection and bidder preferences.

Our computational approach to studying position auctions offers several important advantages over standard methods of theoretical analysis. We can easily relax the common assumptions about value distribution (e.g. uniform value-per-click in every slot) and equilibrium concept (e.g. locally envy-free, pure strategy). It is also straightforward to manipulate auction parameters like reserve prices or rounding behavior. Finally, we can address quantitative questions, like "how much social welfare is lost in the worst equilibrium?" or "which auction generates the most expected revenue?"

Our approach also has several limitations. Chiefly, a computational approach is necessarily tied to the investigation of specific value distributions. Also, our games are discrete with relatively few bid increments, which makes issues of rounding and tie breaking extremely important. This can be viewed either as a drawback or as a benefit: in fact, tie breaking *is* important in position auctions, which are typically run with $0.01 bid increments and clear with prices less than $0.50. Other limitations offer opportunities for future work. We are currently restricted to studying full-information, one-shot games. Although it has been argued in the economic literature that this model is the most appropriate for position auctions (e.g. [Edelman, Ostrovsky, and Schwarz, 2007]), it could also be valuable to compute Bayes-Nash equilibria of imperfect-information position auction games. Progress on this front will depend on the development of computationally-motivated compact representations that can represent large Bayesian games. Finally, while our local search algorithm was effective at exploring regions of connected equilibria, it is not guaranteed to identify the globally-optimal Nash equilibrium. Other game-solving algorithms could help with this, such as the support enumeration method for finding all Nash equilibria [Porter, Nudelman, and Shoham, 2004], or algorithms for finding optimal correlated equilibria [Papadimitriou, 2005].

Acknowledgments

This work was supported by a grant under Microsoft's "Beyond Search" program.

References

Aggarwal, G.; Goel, A.; and Motwani, R. 2006. Truthful auctions for pricing search keywords. In *EC: Proceedings of the ACM Conference on Electronic Commerce*, 1–7. New York, NY, USA: ACM.

Benisch, M.; Sadeh, N.; and Sandholm, T. 2008. The cost of inexpressiveness in advertisement auctions. *ACM EC Workshop on Advertisement Auctions*.

Blumrosen, L.; Hartline, J.; and Nong, S. 2008. Position auctions and non-uniform conversion rates. *ACM EC Workshop on Advertisement Auctions*.

Edelman, B.; Ostrovsky, M.; and Schwarz, M. 2007. Internet advertising and the generalized second price auction: Selling billions of dollars worth of keywords. *American Economic Review* 97(1):242–259.

Govindan, S., and Wilson, R. 2005. Essential equilibria. *Proceedings of the National Academy of Sciences USA* 102:15706–15711.

Jiang, A. X., and Leyton-Brown, K. 2006. A polynomial-time algorithm for Action-Graph Games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 679–684.

Nudelman, E.; Wortman, J.; Shoham, Y.; and Leyton-Brown, K. 2004. Run the GAMUT: A comprehensive approach to evaluating game-theoretic algorithms. In *AAMAS: Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems*, 880–887.

Papadimitriou, C. H. 2005. Computing correlated equilibria in multi-player games. In *STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, 49–56. New York, NY, USA: ACM.

Porter, R.; Nudelman, E.; and Shoham, Y. 2004. Simple search methods for finding a Nash equilibrium. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 664–669.

Scarf, H. 1967. The approximation of fixed points of continuous mappings. *SIAM Journal of Applied Mathematics* 15:1328–1343.

Thompson, D. R. M., and Leyton-Brown, K. 2009. Computational analysis of perfect-information position auctions. In *EC: Proceedings of the ACM Conference on Electronic Commerce*, 51–60.

Varian, H. 2007. Position auctions. *International Journal of Industrial Organization* 25(6):1163–1178.