Computational Analysis of Perfect-Information Position Auctions

David Robert Martin Thompson
Department of Computer Science
University of British Columbia
201-2366 Main Mall, BC V6T 1Z4, CANADA
daveth@cs.ubc.ca

Kevin Leyton-Brown
Department of Computer Science
University of British Columbia
201-2366 Main Mall, BC V6T 1Z4, CANADA
kevinlb@cs.ubc.ca

ABSTRACT

Position auctions were widely used by search engines to sell keyword advertising before being well understood (and, indeed, studied) theoretically. To date, theorists have made significant progress, for example showing that a given auction is efficient or revenue-dominates a benchmark auction such as VCG. This paper augments that line of work, relying on computational equilibrium analysis. By computing Nash equilibria and calculating their expected revenue and social welfare, we can quantitatively answer questions that theoretical methods have not. Broadly, the questions we answer are: (1) How often do the theoretically predicted "good" (i.e., efficient, high-revenue) equilibria of GSP occur? (2) In models where GSP is known to be inefficient, how much welfare does it waste? We also use our data to examine the larger question of whether GSP is a good choice, compared with the alternatives.

1. INTRODUCTION

Position auctions are a relatively new family of auctions whose distinguishing characteristic is that rather than losing, the second-highest bidder wins the second-most-desirable good. Each year, these auctions are used to sell billions of dollars worth of advertising space on search engine results pages. The search engines have some degrees of freedom within the framework of position auctions: whether advertisers pay every time their ad is shown or only when the user clicks on it, whether ads that attract more clicks are given more favorable rankings, and how prices are set.

Although a few varieties have been tried, the major search engines have converged on a single design: the weighted, generalized second-price auction (wGSP for short). The main question that this paper seeks to address is whether or not wGSP represents a good choice compared to the auctions it has replaced, and as compared to theoretical benchmarks. Specifically, we ask whether it is more economically efficient, and whether it generates more revenue. A range of theoretical work has been published showing properties of wGSP under various preference models, some positive and some negative. These results have raised further questions, not all of which appear to be amenable to theoretical study.

This paper attempts to answer these questions quantitatively. We introduce novel computational and representational techniques that for the first time permit the computation of Nash equilibria of position auctions. This allows us to make "apples-to-apples" comparisons to find out exactly how two different auctions behave in equilibrium, given the same bidder preferences. It also allows us to measure the magnitude and frequency of auction properties that are theoretically possible, but are not proven to always hold, such as the existence of envy-free equilibria.

1.1 Previous Work

Although a variety of position auction variants have been proposed, three have seen large-scale use in practice. We describe each here and also give a short form that we will use througout the paper.

GFP The unweighted, pay-per-click, generalized first-price auction, used by Yahoo! and Overture from 1997-2002.

uGSP The unweighted, pay-per-click generalized second-price auction, used by Yahoo! from 2002-2007.

wGSP The weighted, pay-per-click, generalized second-price auction, used by Yahoo! since 2007, as well as by Microsoft Live and Google.

These auctions have all been subjected to theoretical analysis under a variety of models, typically with the assumption that bidders will converge, in repeated play, to an equilibrium of the full-information, one-shot game.

Edelman, et al. analyzed the uGSP[1] under a preference model (which we denote ESO) in which each bidder's expected value per click is independent of position. The click-through rate is decreasing in position (i.e., ads that appear lower on the screen get fewer clicks) but independent of bidder [15]. They showed that in any locally envy-free equilibrium (one in which no bidder envies the allocation received by a bidder in a neighboring position), uGSP is efficient and revenue dominates the truthful equilibrium of VCG. Given that locally, envy-free equilibria are not guaranteed to exist, a natural question about this model is:

[1] In their model, uGSP and wGSP are strategically equilvalent.

Question 1: Under ESO preferences, how often does wGSP have efficient, VCG-revenue-dominating Nash equilibria? What happens in other equilibria?

Varian analyzed wGSP under a more general model [36]. In his model (which we denote V), each bidder's value per click is still independent of position, but the click-through rate is decreasing and "separable." Separability means that for any position/bidder pair, the click-through rate can be factored into a position-specific component that is independent of bidder identity and a bidder-specific component that is independent of position. The weights of wGSP are assumed to correspond to these bidder-specific factors. Varian showed that in any "symmetric equilibrium" (i.e., globally envy-free) wGSP is efficient and revenue dominates VCG. Because symmetric equilibria are similarly not guaranteed to exist, it is natural to ask a second question analogous to our first:

Question 2: Under V preferences, how often does wGSP have efficient, VCG-revenue-dominating Nash equilibria? What happens in other equilibria?

Blumrosen, et al. proposed an even more general model (which we denote BHN). In this model, click-through rate is still decreasing and separable [8]. However, a bidder's expected value per click is increasing in rank in a separable fashion, subject to the constraint that a bidder's expected value *per impression* is weakly decreasing. The authors support their generalization by describing empirical data that shows that conversions (e.g., sales) are more likely for clicks in lower positions. They show that preference-profiles exist where wGSP has no efficient, pure-strategy Nash equilibrium. However, while we know that such preference profiles exist, we do not know how much of a problem they pose on average.

Question 3: Under BHN preferences, how often does wGSP have no efficient Nash equilibrium? How much social welfare is lost?

Benisch, et al. proposed another model (which we denote BSS), generalizing ESO. In this model, click-through rate is decreasing in position but independent of bidder. However, bidders' values are single peaked in position and strictly decreasing from that peak (e.g., "brand" bidders prefer the prestige of top positions, while "value" bidders prefer positions near the middle) [6]. They analyze this model in an imperfect-information game, and show both that GSP ranking rules can be arbitrarily inefficient for such models and that more expressive bidding languages can improve efficiency. For different distributions over their model, they bounded the loss of efficiency in the best-case Bayes-Nash equilibrium.

Question 4: Under BSS preferences, how often does wGSP have no efficient perfect-information Nash equilibrium? How much social welfare is lost?

Yet another family of models are the "cascade models" [18, 22, 2] which seek to capture externalities in the bidders' preferences. Specifically, in these models a bidder's value and click-through rate depend on which bidders win the other positions. The motivation for this model is that users tend to scan and click ads in the order they appear; a good ad can make lower ones appear less desirable, while a bad ad can cause a user to give up on the ads entirely. Although this is an important and plausible preference model, it is not tractable within our current representation, and so we defer its study to future work.

Having posed the questions that we will address, the rest of the paper takes the following form. In Section 2, we explain the representation that makes our computational approach possible.[2] In Section 3, we describe our specific experimental method: problem distributions, game solving algorithms and statistical techniques. In Section 4, we address our questions, presenting the results of our experimental study.

2. POSITION AUCTIONS AS AGGS

This section describes our preference model and shows how that model can be represented compactly to allow for more efficient computation of Nash equilibria. The essential insight that makes this possible is that these games obey "context-specific independence:" for example, in GFP a bidder's payoff depends only on his own bid and on how many bidders bid higher than he did, but not on who they are or exactly what their bids were. Action-graph games (described below) exploit this independence to represent games more compactly and make them more computationally tractable.

2.1 Preference model

In our evaluation of auction types, we use the following model. As in previous work [36, 15, 24], we look for the equilibria of the one-shot, full-information game.

Definition 1 (Ad auction setting). *An* ad auction setting *is given by a 6-tuple* (N, K, M, P, V, β)*:*

1. *N is the set of bidders;*
2. *K is the set of possible positions;*
3. *M is the set of possible bids;*
4. *P is a matrix of click-through rates, where $P_{j,i}$ is the probability that bidder i will receive a click when his ad is in position j;*
5. *V is a matrix of expected values for a click, where $V_{j,i}$ is the value that bidder i has for a click when his ad is in position j; and*
6. *β is a vector of "quality scores" or weights, where β_i/β_j denotes the quality of bidder i relative to bidder j.*

Because our model allows for arbitrary click-value and click-through rates for every bidder, position pairing, it strictly generalizes all the models described earlier: ESO, V, BHN and BSS.

2.2 AGGs

Since analyses of position auctions have relied on perfect-information games, in principle we could discretize bid amounts and then represent the auction as a normal-form game. This would have the advantage that Nash equilibria of the auction

[2] We introduced this representation in a previous workshop paper [35]. Section 2 essentially follows that earlier work (albeit incorporating some new advances). Otherwise, the results in the current paper are entirely new.

could be identified by standard computational tools such as Gambit [28]. The catch, of course, is that the normal form representation of a realistic ad auction problem is unmanageably large. For example, the normal form representation of a relatively small game with 10 agents and 10 bid amounts per agent consists of 100 billion values, too many to store even on the hard drive of many modern computers—let alone in RAM. Thus, to have any hope of tackling position auctions computationally, it is necessary to work with a representation language that allows the game to be compactly described.

We chose to use the action-graph game (AGG) representation [7, 20]. Action-graph games are similar to the more widely-known graphical game representation [21] in that they exploit utility independencies. AGGs are strictly more powerful than graphical games, however. This is because AGGs are compact not only for games with "strict utility independencies" (the property that one agent's payoff never depends on some second agent's action) but also "context-specific independencies" (one agent's payoff is independent of a second agent's action, at least for some action of the first agent and some set of actions of the second). This distinction is important for modeling position auctions. Note that any bidder can affect any other bidder's payoff (e.g., by outbidding him); hence the graphical game representation of a perfect-information position auction is a clique, meaning that it is no more compact than the normal form. However, position auctions have considerable context-specific independence structure. To give one simple example, in a GFP auction, bidder i's utility is independent of bidder j's bid, conditional on j bidding less than i. This is the sort of structure that can be captured by AGGs.

The core idea behind action-graph games is the action graph, so called because nodes in this directed graph represent actions. Each agent is allowed to choose his action from an arbitrary subset of the nodes; agents' subsets are allowed to overlap or coincide. Play of the game can be visualized as each agent simultaneously placing a single token on one of the nodes in the graph. Given the locations of all the tokens, an agent's utility can be computed by referring only to the *number* of tokens in the *neighborhood* of his chosen node. (The neighborhood of a node v is the set of all nodes having outgoing edges that point to v; self-edges are allowed, and so a node can belong to its own neighborhood.) Figure 1 gives an example of an action-graph game taken from [7]. Observe that there are two action sets consisting of four actions each; unlike in a graphical game, the number of agents cannot be inferred from the graph.

Their compact size is not the only interesting thing about AGGs. More importantly, AGG structure can be leveraged computationally, and hence game-theoretic computations can be performed dramatically more quickly for AGGs than for games represented in normal form. For example, given action graphs with bounded in-degree, a polynomial-time dynamic programming algorithm can be used to compute an agent's expected utility under an arbitrary mixed strategy profile [20]. (Observe that this is interesting because the standard method of computing expected utility has running time polynomial in the size of the normal form, but potentially exponential in the size of more compact representations like AGGs.) This computational problem is important because it constitutes the inner loop of many game-theoretic algorithms, including state-of-the-art algorithms for computing Nash equilibria like Simplicial Subdivision [33] and Govindan-Wilson [19]. An exponential speedup to the solution of the expected utility problem therefore translates directly to an exponential speedup of such algorithms, without altering the solution obtained.

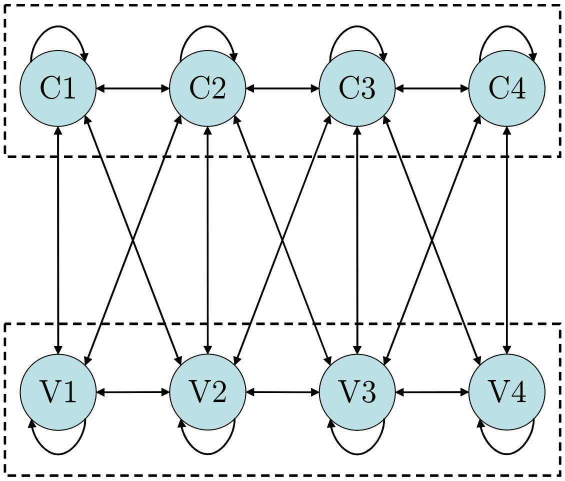

Figure 1: The ice cream vendor game as an AGG (from [20]). Chocolate and vanilla ice-cream vendors must choose which block to open their stores on, given that their payoffs will depend on how many other vendors are within one block of their location. (Circular nodes represent actions, dotted boxes represent action sets for a group of players and arcs represent payoff dependencies.)

Although it is beyond the scope of this paper to describe AGGs in detail, there *is* one further element of the representation that we must describe here. Specifically, it is possible to add so-called *function nodes* to the action graph, which are nodes that belong to no agents' action sets. Instead, the "action count" at a function node is calculated as an (arbitrary) deterministic function of the counts at the function node's parents. For example, when an agent's payoff for playing a depends on how many agents play any of b, c or d, we can add a summation node to reduce the in-degree of a. Function nodes can dramatically reduce representation size when (for example) many actions affect a given action in the same way. As long as the functions are well-behaved ("contribution-independent"; roughly, commutative and associative) function nodes can be used with the dynamic programming algorithm from [20]. Since the variable most important to the asymptotic running time of this algorithm is the maximal in-degree of action nodes, and since this quantity can be drastically reduced by the introduction of high-in-degree function nodes, function nodes can also lead to substantial computational savings.

2.3 Representing GFPs as Action-Graph Games

Having described the auction types and model that we intend to solve, our next step is to show that they can be compactly represented. This section will present algorithms for representing an ad auction type (for example, weighted per-click GSP) and auction setting as an action-graph game

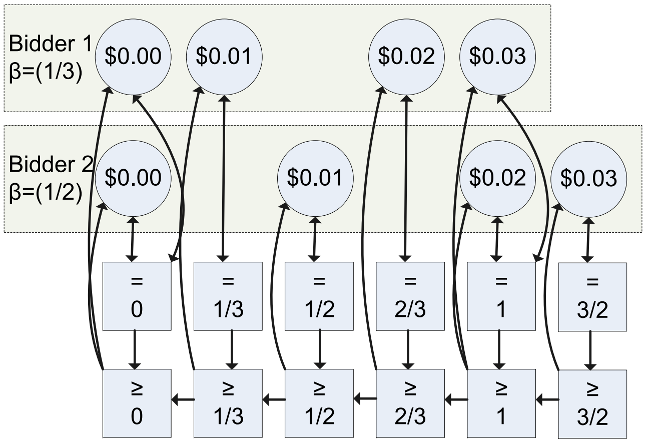

Figure 2: A weighted GFP represented as an AGG. (Square nodes represent summation function nodes.)

and bounds on the size of those representations.

To get a suitably compact representation, one of our biggest concerns is the maximum in-degree of our action graph. Every action node must have a table representing that action's payoff function, and this table will grow exponentially in the in-degree of that action node. An arc (u, v) in an AGG denotes that the payoff for playing action v depends on the number of agents playing u. However, note that an agent's payoff depends only on the position he is awarded and the price he is made to pay. In the case of a GFP, an agent's price is determined by his bid while his position is determined by the number of bids above his and the number of bids equal to his. (We assume that ties are broken randomly.) His position can be affected by many different actions by other bidders, leading to a very large in-degree. However, if we introduce function nodes corresponding to summation (hereafter *summation nodes*) to keep track of how many bids are equal to or greater than each possible bid value, we need only two in-arcs to capture the two values.

Algorithm 1 converts a weighted (or, as a special case, unweighted[3]) GFP to an AGG. An example AGG is shown in Figure 2.

For each action node, we must have a payoff function mapping from the inputs to that node to the payoff an agent playing that action will get. For a bid of b by agent i, we denote this as $\gamma^{1,I}_{i,b}$ for a pay-per-impression GFP. Because of the configuration of the summation nodes, the two inputs to this function are the number of effective bids that are equal to i's bid of b and the number that are greater than or equal. The payoff function for any action node in a pay-per-impression GFP is given by

$$\gamma^{1,I}_{i,b}(e,g) = \frac{1}{e}\sum_{j=g-e+1}^{\min(g,k)} (P_{j,i}V_{j,i} - b).$$

Similarly, the payoff function for any action node in a pay-per-click GFP is given by

$$\gamma^{1,C}_{i,b}(e,g) = \frac{1}{e}\sum_{j=g-e+1}^{\min(g,k)} P_{j,i}(V_{j,i} - b).$$

This representation results in a graph containing nm action nodes, each of which has an in-degree of two. Each node has a payoff table with at most $O(n^2)$ relevant entries. Thus, this representation requires $O(n^3m)$ space.

```
foreach agent $i \in N$ do
    foreach bid $m \in M$ do
        create an action node representing $i$ bidding $m$;
$E \leftarrow \{m\beta_i | \forall i \in N, \forall m \in M\}$;
foreach effective bid $e \in E$ do
    create a summation function node, $(=, e)$ representing
    the bidders bidding exactly $e$;
    create a summation function node, $(\geq, e)$ representing
    the bidders bidding above $e$;
    add an arc from $(=, e)$ to $(\geq, e)$;
    if $e>0$ then
        add an arc from $(\geq, e)$ to $(\geq, e')$ (where $e'$ is the
        next largest effective bid);
foreach action node $a$ do
    $e \leftarrow$ effective bid of $a$;
    add an arc from $a$ to $(=, e)$;
    add an arc from $(=, e)$ to $a$;
    add an arc from $(\geq, e)$ to $a$;
```

Algorithm 1: An algorithm for converting an auction setting into an action graph representing a GFP.

2.4 Representing GSPs as Action-Graph Games

GSPs are similar to GFPs in that each agent's payoff depends on a small number of values. To determine the position (or range of positions), we use the same graph structure as for GFPs. However, we need to augment the graph to capture the pricing rule of GSPs. This is done by adding "price nodes", function nodes that identify the next-highest bid. We use the term *argmax node* to refer to a function node whose value is equal to the largest (given some arbitrary ordering) in-arc carrying a non-zero value. By ordering action nodes according to the value of their effective bids, an argmax node identifies the highest effective bid among the subset of action nodes connected to it. After running the Algorithm 1, we add argmax nodes as shown in Algorithm 2. An example of the resulting action graph is illustrated in Figure 3. Note that although the in-degree of the argmax nodes can get large ($O(nm)$), the computational complexity of solving an AGG only depends on the in-degree of the action nodes.

As in the case of GSP, we must define a payoff function for each action node. Now, we have a third input (p), which identifies the next-highest effective bid. Let E_p denote this value. The payoff function for any action node in a pay-per-impression GSP is

$$\begin{aligned}\gamma^{2,I}_{i,b}(e,g,p) &= \frac{1}{e}\sum_{j=g-e+1}^{\min(g-1,k)} (P_{j,i}V_{j,i} - b) \\ &\quad +\delta(g \leq k)(P_{g,i}V_{g,i} - \max\{1, \lceil E_p \rceil\})/e,\end{aligned}$$

where $\delta(x) = 1$ iff x is true. Similarly, the payoff function

[3] Trivially, this algorithm can also be used to represent an unweighted GFP by replacing β with a vector of ones, causing the auction to treat bids by different agents equivalently.

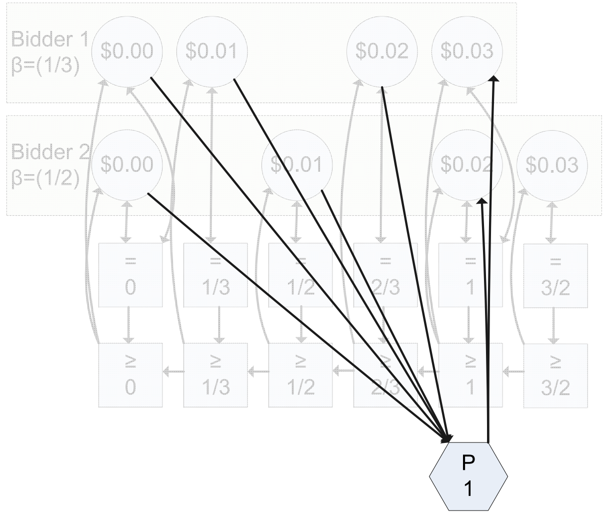

Figure 3: To represent a GSP as an AGG, we add price nodes (argmax nodes denoted by hexagons) to a GFP representation. For clarity only one price node is pictured, while a full GSP representation requires one price node for each effective bid.

```
foreach effective bid $e \in E$ do
    create an argmax function node, $(p, e)$ representing the next highest effective bid below $e$;
    foreach action node $a$ with effective bid $e'$ do
        if $e' < e$ then
            add an arc from $a$ to $(p, e)$;
        if $e' = e$ then
            add an arc from $(p, e)$ to $a$;
```

Algorithm 2: An algorithm for converting an auction setting into an action graph representing a GSP.

for any action node in a pay-per-click GSP is given by

$$\gamma_{i,b}^{2,C}(e,g,p) = \frac{1}{e}\sum_{j=g-e+1}^{\min(g-1,k)} P_{j,i}(V_{j,i}-b) + \delta(g \le k)P_{g,i}(V_{g,i}-\max\{1,\lceil E_p \rceil\})/e.$$

This representation results in a graph containing nm action nodes, each of which has an in-degree of three. Each node has a payoff table with at most $O(n^2|E|)$ relevant entries (where $|E| \le nm$). Thus, this representation requires $O(n^4m^2)$ space. As was the case with GFPs, we can produce an unweighted auction by simply replacing β with a vector of ones. We can find also AGG representations of Lahaie and Pennock's family of ranking rules [25] by adjusting the values of β appropriately.

Although these techniques can represent many different position auctions, we will only focus on the ones used in practice: GFP, uGSP and wGSP.

2.5 Removal of Dominated Strategies

For tractability, we removed weakly dominated strategies while generating action graph games. The strategies we removed fall into two classes. The first class is strategies that require a bidder to bid higher than his maximum value in any position. The second class is strategies where the agent bids the larger of two values that always result in the same allocation, i.e., where every bid by any other agent is either strictly greater than the larger bid or strictly less than the smaller bid. This second class occurs often when bidders have very different weights.

For some games, removing weakly dominated strategies can remove some Nash equilibria; we consider the extent to which this arises for our games. For the first class, we do lose Nash equilibria, but only ones in which at least one agent is playing the obviously dominated strategy of bidding above his value. For the second class, we do not remove any Nash equilibria from GFP auctions, because the dominance is always strict: by bidding the higher of the two values, the bidder gets the same outcome but pays a strictly higher price. However, for GSP auctions removing the second class of dominated strategies does remove some Nash equilibria. Fortunately, these equilibria are redundant from perspective of auction analysis. If the removed strategy is played in a Nash equilibrium, there must be a corresponding equilibrium where the bidder plays the dominating strategy. Each equilibrium will lead to the same outcome (or distribution over outcomes in the case of ties or mixed strategies).

3. METHOD

Broadly speaking, the method of this paper is to generate many specific preference-profile instances from each of the preference models, build an AGG encoding the corresponding perfect-information auction problem, solve the AGG computationally, and then compare the outcomes against both each other and VCG.

3.1 Problem instances

Our preference-profiles come in four sizes and are generated by imposing a probability distribution over the preference-models and then drawing instances from those distributions. For the ESO, V, and BHN preference models we used a probability distribution that was as uniform as possible over the space of possible preferences.

For ESO, each bidder's value was drawn from a uniform $[0, 1]$ distribution. The position-specific click-through rate was drawn from a uniform distribution over the allowed range (at least zero, and at most the click probability of the position immediately above).

For V, the distribution was the same as for ESO, except that each bidder-specific click-through rate factor was drawn from a uniform distribution on $[0, 1]$.

For BHN, the distribution was the same as for V, except that the position-specific click-value factor was drawn from a uniform distribution over the allowed range (at least the value per-click of the position immediately above, at most a value per-click that results in the same value per-impression as the position immediately above).

We handled BSS differently, because in addition to their model, the authors also provided an explicit preference distribution of their own. We followed that model, which we summarize here (a more thorough description is provided in [6]). Click-through rate decreases exponentially. With probability 0.5, a bidder's peak position is drawn from uniform $[0.8, 1.0]$; otherwise this value is drawn from uniform $[0.4, 0.6]$. Conversion probabilities fall off exponentially from the peak. Value per conversion is drawn from uniform $[30, 150]$ while each click costs 1.

In every case, we normalized the values so that the highest value was equal to the highest possible bid, to ensure that the full number of bid increments was potentially useful.

We considered games at four problem sizes:

small: 5 bidders, 3 positions, 5 non-zero bid increments

medium: 10 bidders, 8 positions, 10 non-zero bid increments

large: 10 bidders, 8 positions, 20 non-zero bid increments

extra large: 10 bidders, 8 positions, 40 non-zero bid increments

To allow direct study of the effect of discretization, we generated the medium, large and extra-large problem instance to be identical up to a normalizing constant.

For each preference model and size, we generated 100 preference-profile instances, and then generated games for each of the three position-auction types, GFP, uGSP and wGSP (4800 games, in all).

3.2 Computation

Our experiments were performed using a cluster of 55 machines with dual Intel Xeon 3.2GHz CPUs, 2MB cache and 2GB RAM, running Suse Linux 10.1 (Linux kernel 2.6.16.54-0.2.5-smp).

To compute Nash equilibria, we used "`simpdiv`" and "`gnm`," the simplicial subdivision [33] and global Netwon method [19] implementations provided by Gambit [28], extended to use AGGs and the dynamic programming algorithm of [20].[4]

For each game, we ran `simpdiv` from ten different pure-strategy-profile starting points, chosen uniformly at random. We also ran `gnm` ten times, with random seeds one through ten. Each run was limited to five CPU minutes. `simpdiv` took 895.1 CPU hours and found solutions to 4230 of the 4800 games. `gnm` took 1402.1 CPU hours and found solutions to 3775 games.

To improve the statistical reliability of our results, we did not drop individual games that were not solved by the cutoff. This is because we worried that the very features that made instances hard to solve could also make their equilibrium outcomes qualitatively different those of easier-to-solve games. Instead, we split (preference-model,size,auction) triples into two groups. For those on which we had solved nearly every instance, we re-ran our solvers to completion without a cutoff time. On the other hand, some triples had been too hard to solve reliably; these we abandoned. They were all GFP auctions at large or extra-large sizes: (V,L,GFP), (ESO,L,GFP), (V,XL,GFP), (ESO,XL,GFP) and (BSS,XL,GFP).

3.3 Equilibrium selection

Starting from some arbitrary equilibrium of a GSP, it will often be the case that one or more agents can vary their bids within ranges that do not affect the outcome (the ranking of the ads), but that do affect the amounts that the bidders above must pay [9, 36]. Thus, while social welfare is relatively stable in GSPs, equilibrium revenue depends on which strategy each agent chooses from a set of weak best responses.

We treat this equilibrium selection problem as a constrained optimization problem: maximize or minimize revenue subject to being in a Nash equilibrium. We represent this as a local search through strategy-profile space. Our neighborhood relation is strategies that differ by shifting probability mass from one bid to an adjacent one. Our algorithm is randomized hill-climbing: accepting the first improving move that doesn't violate an equilibrium constraint. We restart twice from each equilibrium found by `simpdiv` or `gnm`. Note that the final maximum or minimum thus found is not guaranteed to be a global optimum.

For consistency, we also use the same local search procedure with GFP problems, and in all auctions when evaluating social welfare as well as revenue.

3.4 Benchmarks: VCG and Discretized VCG

As well as comparing GSP and GFP to each other, we also compared these position auctions to VCG. There are two ways of doing this. First, we used the actual (i.e., non-discretized) preferences of the agents, and computed the truthful equilibrium of VCG. However, when making such a comparison, it is not possible to determine whether differences arise because of the auction mechanism or because of discretization. To answer this question, we also compared against a discretized version of VCG. Bidders' values-per-click were discretized to the same number of increments as in the position auctions, and the optimal allocation was found for these perturbed preferences. We still assume that bidders bid truthfully in this auction. Note that such behavior is in fact only an ϵ-Nash equilibrium, where ϵ is equal to half of one bid increment.

3.5 Statistical Methods

To test whether one auction achieved statistically significantly better performance than another according to a given metric (e.g., revenue), we used a blocking, means-of-means, bootstrapping test [13]. We performed the test as follows:

1. For each setting instance, find the difference in the metric across that pair of auctions on that instance. Each value is normalized by the achievable social welfare in that instance. Call this set of values S.
2. Draw $|S|$ samples from S (with replacement), and compute the mean. Perform this procedure 20,000 times. Let M denote the set of means thus computed.
3. Our estimated performance difference is the mean of M (the mean-of-means of S).
4. This difference has significance level α if the α^{th} quantile of M is weakly greater than zero.

For our hypothesis tests, we use * to denote that a result has a significance level of $\alpha = 0.05$ and ** to denote that a result has a significance level of $\alpha = 0.01$. For each group of data points (i.e., for a specific size and preference model) we must perform 80 simultaneous tests: we compare revenue and welfare in the best and worst equilibrium, from each auction to every other. With this many tests, we must use some multiple-comparison correction. (Otherwise, while every individual significance test will be sound with marginal probability ≥ 0.95, the joint probability of them all being sound could be much lower.) We use a Bonferroni correction (effectively, dividing the desired significance level by the number of tests performed) [30]. Thus, when reporting results for groups of data points, we use * to indicate that a result is individually significant for $\alpha = 0.000625$ and ** to indicate that a result is individually significant for $\alpha = 0.000125$.

[4] The extension of Gambit that allows simplicial subdivision to work with AGGs can be obtained from `http://agg.cs.ubc.ca`.

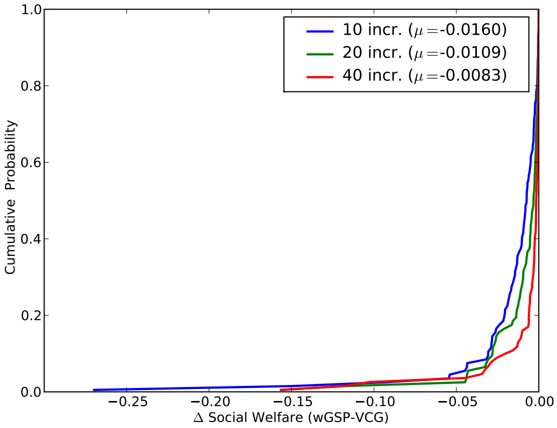

Figure 4: In ESO, wGSP approximated efficiency well. Finer discretization lead to more efficiency.

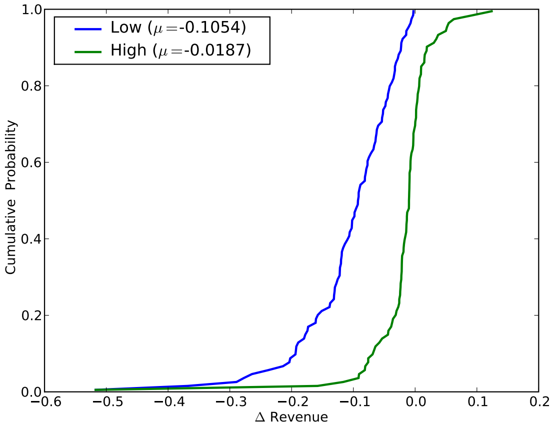

Figure 5: In ESO, wGSP's expected revenue was very sensitive to equilibrium selection.

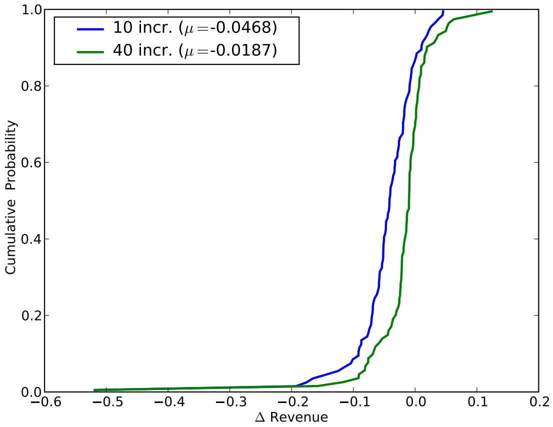

Figure 6: In ESO, even in the best equilibria we found wGSP's did not beat VCG (discrete or continuous).

4. EXPERIMENTAL RESULTS

Now we turn to our experimental findings. We had two main goals: answering the four questions posed in the introduction, and assessing whether wGSP achieved superior performance in equilibrium, as compared to the auctions that it has replaced in widespread practice. This section is structured around the preference models that each question uses. For each model, we answer our corresponding question, and then quantify wGSP's performance compared to that of the other auctions.

4.1 Edelman, Schwarz, Ostrovsky model

Recall that Edelman et al. proved that in "locally envy-free equilibra" wGSP is weakly better than VCG in revenue and welfare [15]. Our question was:

Question 1: Under ESO preferences, how often does wGSP have efficient, VCG-revenue-dominating Nash equilibria? What happens in other equilibria?

We found that wGSP approximated efficiency very well: although wGSP was inefficient (significant** at every size), the welfare loss was small relative to the size of the bid increment. As discretizations got finer, wGSP appeared to approach efficiency (see Figure 4).

On the other hand, our revenue results ran counter to the prediction made by Edelman et al, that wGSP would revenue-dominate VCG. Although wGSP had multiple equilibria with different revenues (see Figure 5), even in its most favorable equilibria wGSP achieved less expected revenue than VCG (see Figure 6, significant** at every size). There are a number of reasons that this could have occurred. First, Edelman et al.'s prediction concerns only envy-free equilibria. These equilibria arise when bidders bid within an interval that may be empty [36]; given instances may therefore have no such equilibria. Second, discretization means that even when the interval is non-empty, it could fall between two bid increments. Third, the only discrete bids on this interval could be higher than the bidder's value, and therefore would removed as weakly dominated.[5]

From the perspective of practice, our efficiency results seem to support Yahoo!'s 2002 decision to switch from GFP to uGSP: uGSP was more efficient (see Figure 7, significant** at small and medium sizes for best and worst equilibria. GFP was too difficult to solve at larger sizes.) Our revenue results were ambiguous: we did not observe a significant revenue difference between GFP and wGSP. Another interesting practical conclusion is that the use of \$0.01 bid increments is not innocuous. For keywords where the value-per-click is small, this amounts to a coarse discretization, which could harm revenue and welfare.

4.2 Varian model

Recall that Varian proved that in "symmetric equilibria" wGSP achieves weakly better performance than VCG according to both revenue and welfare [36]. Our question was:

Question 2: Under V preferences, how often does wGSP have efficient, VCG-revenue-dominating Nash equilibria? What happens in other equilibria?

As with the ESO model, we found that that even in the worst equilibria, wGSP approximated efficiency very well. 95% confidence intervals on the mean welfare loss ranged from $[0.36\%, 1.47\%]$ for the extra-large cases to $[1.40\%, 4.94\%]$ for the small cases.

[5] We are currently running new experiments without removal of dominated strategies to measure this effect; we intend to describe them in the full version of this work.

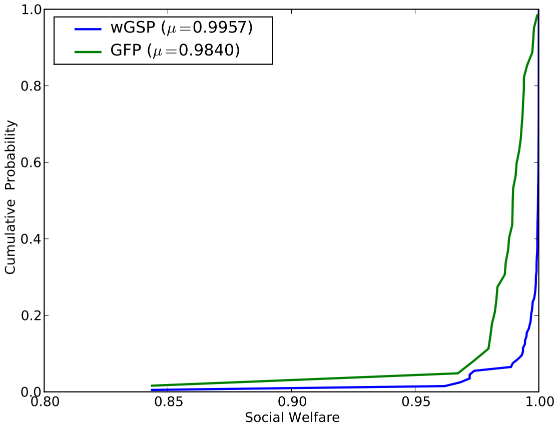

Figure 7: In ESO, uGSP and wGSP are strategically equivalent and were more efficient than GFP.

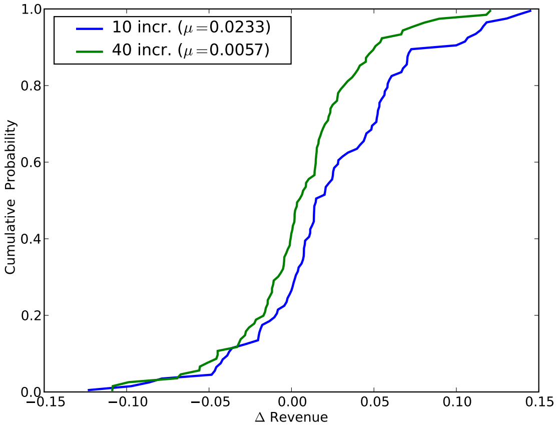

Figure 8: In V, wGSP got more revenue than VCG. As discretization got finer, wGSP's expected revenue gains over VCG got smaller.

Unlike in the case of ESO, our revenue findings were consistent with Varian's. wGSP's best Nash equilibrium did generate more revenue than VCG (significant** for small, medium and large games, not significant for extra-large games). Interestingly, the magnitude of this effect got smaller for finer discretizations (see Figure 8). Also as with uGSP in ESO, revenue was very sensitive to equilibrium selection (see Figure 9).

wGSP clearly dominated GFP and uGSP in terms of efficiency (see Figure 10, significant** at every scale at which data was present). Again, revenue results were ambiguous: wGSP's revenue was only significantly greater than GFP or uGSP for medium-sized instances. Interestingly, finer discretization seemed to help efficiency, but to hurt revenue.

4.3 Blumrosen, Hartline, Nong model

Blumrosen et al. proved the negative result that sometimes wGSP has no efficient equilibrium.

Question 3: Under BHN preferences, how often does wGSP have no efficient Nash equilibrium? How much social welfare is

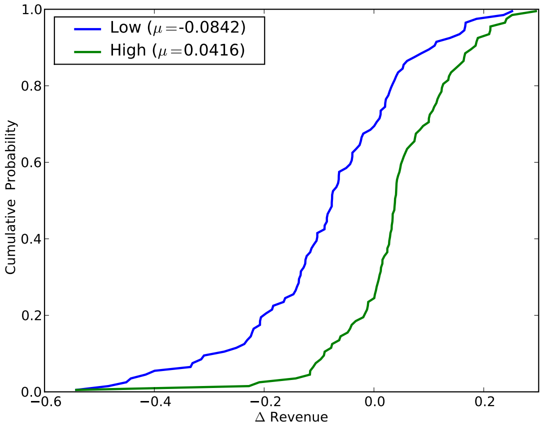

Figure 9: In V, wGSP's revenue was very equilibrium-selection sensitive.

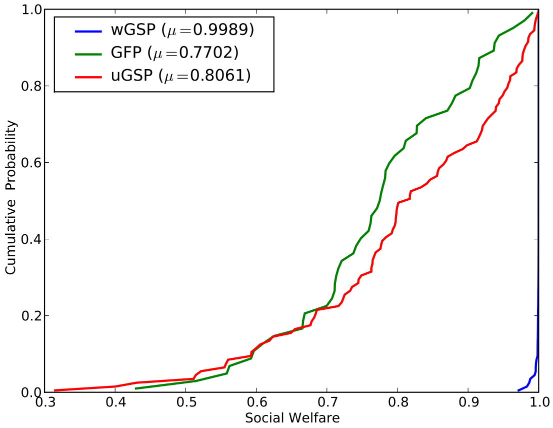

Figure 10: In V, wGSP was more efficient than uGSP and GFP.

lost?

Our efficiency results in this setting were especially interesting. We found that wGSP had frequent, complete failures of efficiency (see Figure 11). Surprisingly, discretized VCG also had massive failures of efficiency. These effects seemed to arise from the fact that the top position, with the highest value-per-impression (and hence then largest contribution to social welfare), had the smallest value-per-click and was therefore very prone to being distorted by discretization.

Despite this, wGSP had higher expected welfare than GFP and uGSP (see Figure 12, significant** for all sizes with sufficient data). Again, revenue results were ambiguous: although wGSP often had higher revenue (see Figure 13), this difference was not always significant.

4.4 Benisch, Sadeh, Sandholm model

Benisch et al. proved the negative result that sometimes wGSP has no efficient Bayes-Nash equilibrium. Quantitatively, they were able to upper-bound social welfare between 70% and 98% depending on bidder values, for instances with three and four bidders.

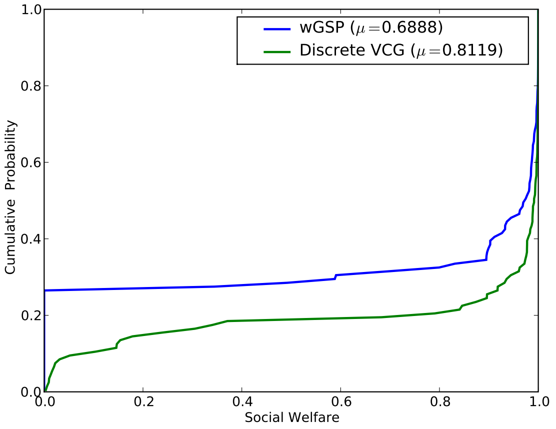

Figure 11: In BHN, wGSP and discretized VCG both have serious failures of efficiency.

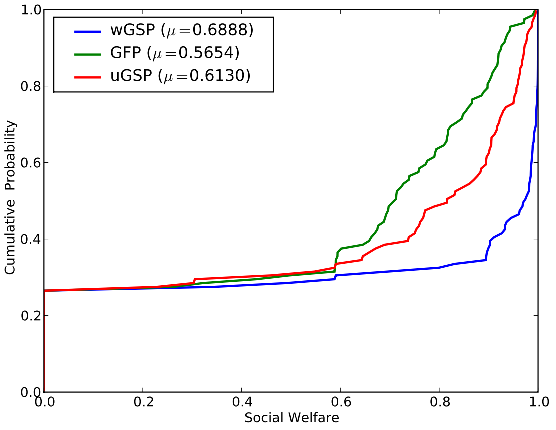

Figure 12: In BHN, wGSP was more efficient than GFP or uGSP

Question 4: Under BSS preferences, how often does wGSP have no efficient perfect-information Nash equilibrium? How much social welfare is lost?

Despite our very different game model (perfect-information, discrete bids), we observed failures of efficiency of comparable magnitude, for five-bidder and ten-bidder instances. Overall, wGSP was more efficient than GFP (see Figure 14, significant** for all sizes except extra large).

Revenue was ambiguous in this setting: both GFP and uGSP were very sensitive to equilibrium selection. Significant differences only existed for the smallest instances, where GFP achieved better revenue than uGSP (see Figure 15.

5. CONCLUSIONS

From a theoretical perspective, we were able to quantify the magnitude of predicted effects and to quantify their sensitivity to equilibrium selection. We found that (in all its equilibria) wGSP was consistently approximately efficient in the models (V and ESO) for which it was already known to (sometimes) have efficient equilibria. However, we found that wGSP's revenue was very equilibrium-selection sensitive; furthermore, even in the best

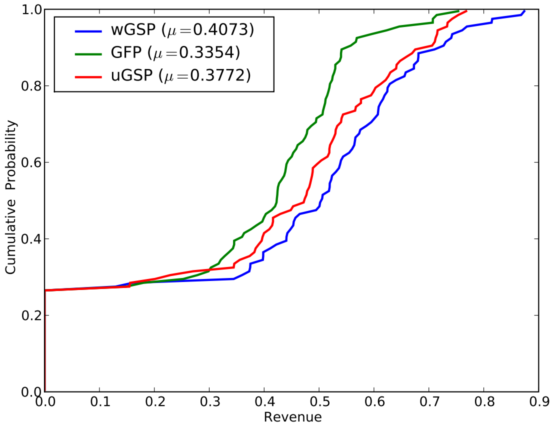

Figure 13: In BHN, wGSP often got more revenue than GFP or uGSP

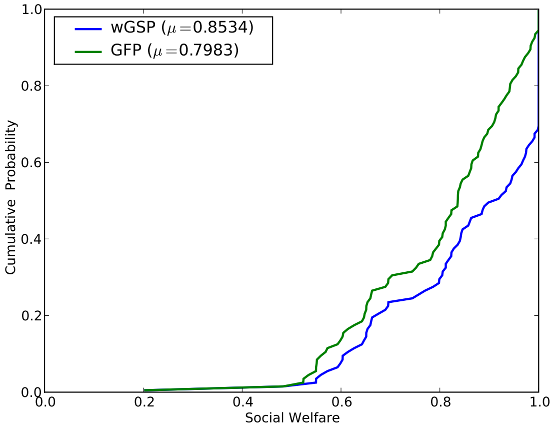

Figure 14: In BSS, wGSP was inefficient, but more efficient than GFP

equilibria, it did not consistently beat VCG. For models where wGSP has been shown to (sometimes) be inefficient—BHN and BSS—we showed that wGSP was *often* inefficient, and often by a large margin.

From a practical perspective, we found that even in settings where it was inefficient, wGSP was consistently more efficient than uGSP or GFP. This supports Yahoo!'s choice to switch to it in 2007. wGSP was a less clear winner in terms of revenue: we did not consistently observe that its best equilibria were significantly better than those of the other auctions.

There is opportunity for future work both on the preferences and on the computational methods. Better distributions of preferences could be learned from real-world data. AGGs (or some other compact representation) may also be applicable to richer preference models, like the cascade model. Better computational methods could also improve our approach: by using an equilibrium-finding algorithm that finds all equilibria (such as the support enumeration methods of [32]), we could make quantitative claims about equilibrium-selection (e.g. what fraction of pure strategy equilibria had revenue greater than VCG?)

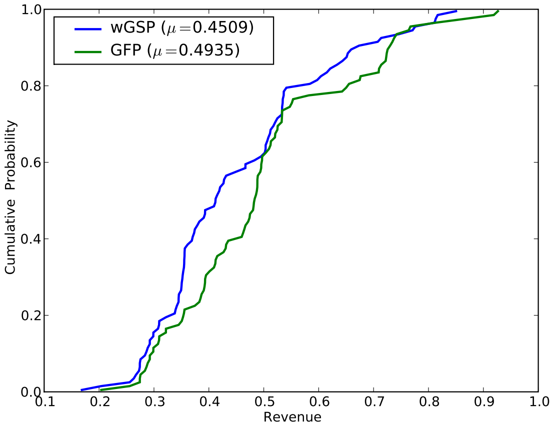

Figure 15: In small BSS problems, GFP generated more revenue than wGSP.

6. ACKNOWLEDGMENTS

This work was supported by a grant from Microsoft under the "Beyond Search" program.

7. REFERENCES

[1] Z. Abrams, O. Mendelevitch, and J. Tomlin. Optimal delivery of sponsored search advertisements subject to budget constraints. In *EC: Proceedings of the ACM Conference on Electronic Commerce*, pages 272–278, 2007.

[2] G. Aggarwal, J. Feldman, S. Muthukrishnan, and M. Pt'al. Sponsored search auctions with markovian users. *ACM EC Workshop on Advertisement Auctions*, 2008.

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

[4] K. Asdemir. Bidding patterns in search engine auctions. In *Second Workshop on Sponsored Search Auctions*, 2006.

[5] I. Ashlagi, D. Monderer, and M. Tennenholtz. Mediators in position auctions. In *EC: Proceedings of the ACM Conference on Electronic Commerce*, pages 279–287, New York, NY, USA, 2007. ACM.

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

[7] N. Bhat and K. Leyton-Brown. Computing Nash equilibria of Action-Graph Games. In *UAI: Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 35–42, 2004.

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

[9] T. Borgers, I. J. Cox, M. Pesendorfer, and V. Petricek. Equilibrium bids in auctions of sponsored links: Theory and evidence, as of November 2006.

[10] C. Borgs, J. Chayes, N. Immorlica, K. Jain, O. Etesami, and M. Mahdian. Dynamics of bid optimization in online advertisement auctions. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 531–540, New York, NY, USA, 2007. ACM.

[11] N. Buchbinder, K. Jain, and S. Naor. Online primal-dual algorithms for maximizing ad-auctions revenue. In *Proceedings of the 15th Annual European Symposium on Algorithms*, 2007.

[12] M. Cary, A. Das, B. Edelman, I. Giotis, K. Heimerl, A. R. Karlin, C. Mathieu, and M. Schwarz. Greedy bidding strategies for keyword auctions. In *EC: Proceedings of the ACM Conference on Electronic Commerce*, pages 262–271, New York, NY, USA, 2007. ACM.

[13] M. R. Chernick. *Bootstrap Methods, A practitioner's guide.* Wiley, 1999.

[14] B. Edelman and M. Ostrovsky. Strategic bidder behavior in sponsored search auctions. *Decis. Support Syst.*, 43(1):192–198, 2007.

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

[16] J. Feng, H. K. Bhargava, and D. Pennock. Comparison of allocation rules for paid placement advertising in search engines. In *EC: Proceedings of the ACM Conference on Electronic Commerce*, pages 294–299, New York, NY, USA, 2003. ACM.

[17] J. Feng, H. K. Bhargava, and D. M. Pennock. Implementing sponsored search in web search engines: Computational evaluation of alternative mechanisms. *INFORMS J. on Computing*, 19(1):137–148, 2007.

[18] A. Ghosh and M. Mahdian. Externalities in online-advertising. In *WWW: International World Wide Web Conference*, 2008.

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

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

[21] M. Kearns, M. Littman, and S. Singh. Graphical models for game theory. In *UAI: Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2001.

[22] D. Kempe and M. Mahdian. A cascade model for externalities in sponsored search. *ACM EC Workshop on Advertisement Auctions*, 2008.

[23] B. Kitts, P. Laxminarayan, B. LeBlanc, and R. Meech. A formal analysis of search auctions including predictions on click fraud and bidding tactics. In *Workshop on Sponsored Search Auctions*, 2005.

[24] S. Lahaie. An analysis of alternative slot auction designs for sponsored search. In *EC: Proceedings of the ACM Conference on Electronic Commerce*, pages 218–227, New York, NY, USA, 2006. ACM.

[25] S. Lahaie and D. M. Pennock. Revenue analysis of a family of ranking rules for keyword auctions. In *EC: Proceedings of the ACM Conference on Electronic Commerce*, 2007.

[26] S. Lahaie, D. M. Pennock, A. Saberi, and R. Vohra. Sponsored search auctions. In Nisan et al. [31], chapter 28, pages 699–719.

[27] M. Mahdian, H. Nazerzadeh, and A. Saberi. Allocating online advertisement space with unreliable estimates. In *EC: Proceedings of the ACM Conference on Electronic Commerce*, pages 288–294, New York, NY, USA, 2007. ACM.

[28] R. D. McKelvey, A. M. McLennan, and T. L. Turocy. Gambit: Software tools for game theory, 2006. `http://econweb.tamu.edu/gambit`.

[29] A. Mehta, A. Saberi, U. Vazirani, and V. Vazirani. Adwords and generalized on-line matching. In *FOCS '05: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pages 264–273, Washington, DC, USA, 2005. IEEE Computer Society.

[30] R. G. J. Miller. *Simultaneous Statistical Inference.* Springer, 1981.

[31] N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani, editors. *Algorithmic Game Theory.* Cambridge University Press, Cambridge, UK, 2007.

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

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

[34] C. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 1948.

[35] D. R. M. Thompson and K. Leyton-Brown. Tractable computational methods for finding Nash equilibria of perfect-information position auctions. *ACM EC Workshop on Advertisement Auctions*, 2008.

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