Matching Auctions for Search and Native Ads

RUGGIERO CAVALLO, Yahoo Research
MAXIM SVIRIDENKO, Yahoo Research
CHRIS WILKENS*, Facebook Research

Unit demand auctions power today's search and native ad marketplaces. Traditional implementations make an extreme "separability" assumption: the relative value of any two ad slots is the same for all advertisers. Under this assumption, the optimal assignment problem can be conveniently solved simply by sorting; without it, efficient allocation requires solving a full-blown weighted matching problem. Motivated by prior work and our own empirical evidence against separability, we abandon that assumption and tackle the algorithmic problems of assignment and pricing for general unit demand ad auctions. Instead of computing prices directly, we take a novel approach and compute bidders' full allocation curves—complete mappings from each agent's bid space to their allocation under the optimal assignment function—from which it is trivial to compute most prices of interest, like those of the Generalized Second Price (GSP) or Vickrey-Clarke-Groves (VCG) auctions. Remarkably, we show that these full allocation curves (and therefore prices) can be computed in the same asymptotic runtime required to compute the optimal matching alone.

1 INTRODUCTION

Consider the following auction setting that is ubiquitous in computational advertising. A set of bidders $I = \{1, \ldots, n\}$ is to be matched to a set of slots $J = \{1, \ldots, m\}$ (where $n \geq m$); when bidder $i \in I$ is matched to slot $j \in J$, a desired event (e.g., a click on i's ad) happens with probability $\pi_{i,j}$, generating expected value $v_{i,j} \equiv v_i \pi_{i,j}$ for the bidder. The auctioneer's job is to take bids b_i, each purportedly representing i's value v_i, and compute an optimal matching of bidders to slots.

Practitioners often assume the probabilities $\pi_{i,j}$ have structure that both makes them easier to learn with scalable Machine Learning methods, and also greatly simplifies the auctioning process. Specifically, it is assumed that the event-probabilities are *separable* in the following sense: there exist $\alpha = (\alpha_1, \ldots, \alpha_m)$ and $\beta = (\beta_1, \ldots, \beta_n)$ such that $\forall i \in I, \forall j \in J, \pi_{i,j} = \beta_i \cdot \alpha_j$; i.e., the likelihood that the event of interest (typically click) occurs when ad i is placed in slot j can be decomposed into the product of two terms, one depending only on the ad and one depending only on the slot.

In the separable model, the optimal allocation can be found by sorting. If $\sigma(j)$ is the ad placed into slot $j \in \{1, \ldots, n\}$,[1] the optimal allocation should maximize $\sum_j \alpha_j \beta_{\sigma(j)} b_{\sigma(j)}$. Assuming WLOG that $\alpha_1 \geq \alpha_2 \cdots \geq \alpha_m$, the optimum can be computed simply by assigning the ad with highest $b_i \beta_i$ to slot 1, the ad with the second-highest $b_i \beta_i$ to slot 2, and so on — i.e. one simply needs to rank ads by $b_i \beta_i$.

The *generalized second price* (GSP) auction—so central to slot auctions since their inception in the previous decade—fits with the separability assumption like a hand in a glove. GSP sorts ads in

*Wilkens was employed by Yahoo during the time of research for this paper.
[1]To contrive a number of slots equal to the number of ads (n), we can tack on dummy slots with $\pi_{i,j} = 0$ as necessary.

Authors' addresses: Ruggiero Cavallo, Yahoo Research, New York, NY, cavallo@yahoo-inc.com; Maxim Sviridenko, Yahoo Research, New York, NY, sviri@yahoo-inc.com; Chris Wilkens, Facebook Research, Menlo Park, CA, c.a.wilkens@gmail.com.

ACM EC'18, June 18–22, 2018, Ithaca, MA, USA. ACM ISBN 978-1-4503-5829-3/18/06…$15.00
https://doi.org/10.1145/3219166.3219191

decreasing order of $b_i\beta_i$, then charges bidder i based on the next ad in the sorted list. If $\beta_{\sigma(j)} > 0$, the GSP price of ad $\sigma(j)$ is:

$$p_{\sigma(j)} = \frac{b_{\sigma(j+1)}\beta_{\sigma(j+1)}}{\beta_{\sigma(j)}}$$

Otherwise, $p_{\sigma(j)} = 0$. This price corresponds to the lowest bid that gets bidder $\sigma(j)$ his slot assignment j.

GSP is a workhorse of computational advertising and much research has focused on understanding its properties [7, 21]. Its elegance facilitates implementation and has led to widespread adoption, but the entire framework is critically based on the separability assumption. That assumption has always been a stretch ([8] provides good evidence against it in the case of Bing search ad auctions), and it is becoming a worse approximation as ad formats and pricing types evolve.

Consider Yahoo's Native Ads marketplace. In this setting, ads are often interspersed within a very long content stream, such that among the many ads allocated in the auction, a user will only see one or two at a time. The auction in place is GSP with the following application of separability: the slot-specific parameter α_j captures the probability that the user views slot j (independent of which ad is shown in it), which decreases with j since scrolling through the content stream is required to see ads in lower slots. Complicating matters, the "payment event" varies across ads. Advertisers can choose to pay per-impression (CPM), per-click (CPC), or in some cases even on a conversion or app-installation basis. Mapping this into the separable framework, the ad-specific parameter β_i equals 1 for any CPM ad, while β_i can be taken as the ad-specific "probability of click conditioned on view" for CPC ads, and similarly for other pricing types.

For CPM ads the separable model collapses to the unobjectionable assumption that whether or not an ad appears on the page is independent of the ad itself. But injecting CPC ads into the mix makes the model highly suspect; there is no a priori reason to believe that click probability conditional on view remains constant as one moves down the page. In fact empirical data strongly suggests otherwise.

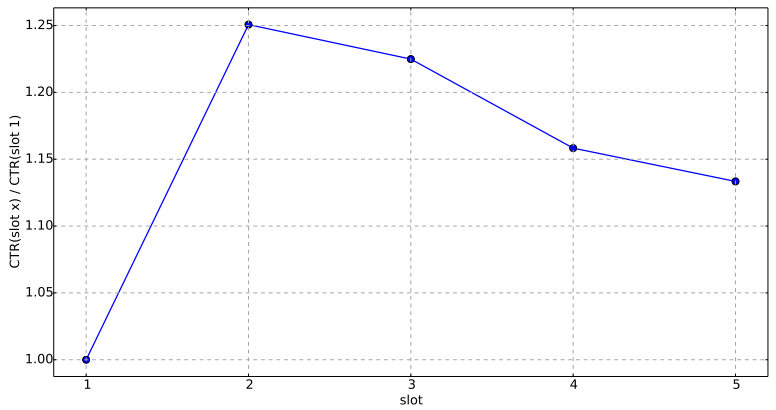

Fig. 1. Empirical click-through-rate (probability of a click conditioned on the ad being viewed) as a function of slot, aggregated over a dataset of approximately 17M pageviews with randomly selected ad impressions from Yahoo's Native Ads marketplace.

As Figure 1 demonstrates, for this Yahoo Native Ads dataset the second slot has a markedly higher click-through-rate than the first (conditioned on view), and from there on click rates slowly

decline as one goes down the page. Thus, in essence, *at a minimum* different α vectors are called for depending on whether the ads are CPM or CPC.[2] If the two ad types are mixed, the separable model becomes a very poor fit, opening room for a more specialized model to yield efficiency improvements. A similar, if less stark, story applies in the search advertising context. Certain types of ads may hold the user's attention more consistently than others as they are moved up or down the page; a separable model precludes such effects and fails to optimize accordingly.

1.1 Main questions

These critical issues with the separable model motivate us to consider the following question:

> *How do we compute efficient ad allocations when click prediction does not satisfy the separability assumption?*

A natural candidate for the ad selection algorithm is to compute a maximum weighted bipartite matching in the complete bipartite graph $G = (I, J, K_{I\times J})$ with edge weights $v_{i,j} = b_i\pi_{i,j}$. This solution maximizes total advertiser value, and can be implemented in a way that meets the severe constraints on run-time inherent to the online auction domain. In our Yahoo Native Ads setting, $m \leq 21$, $n \approx 100$, and the run-time constraint is roughly a few milliseconds. The Maximum Weighted Bipartite Matching problem is a very well-studied optimization problem with a large set of algorithms to choose from. Currently, the best approximate algorithms run in almost linear time [6].[3] Run times usually depend on the number of vertices on each side (n and m in our case), the number of edges (nm in our case), and the maximum integer weight (for cases where edge weights are integers). For our purposes the classical Hungarian algorithm extended to handle imbalanced graphs has run time $O(nm^2)$ [19] and is adequate, with empirical run time under `1ms` on production instances. So far so good. This narrows our focus to the central question of this study, where our contributions lie:

> *How do we price ads that have been assigned to slots by the Maximum Weighted Bipartite Matching algorithm?*

A satisfactory pricing scheme will have desirable equilibrium properties and also be computable under the run-time constraints.

1.2 The VCG (Non) Solution

One attractive answer is to use externality prices as in a *Vickrey-Clarke-Groves* auction (VCG). Indeed, this is an elegant and principled solution, not to mention nearly the only solution offered in the theoretical literature. It has been the go-to implementation for new marketplaces where separability is inconceivable, and there are many methods to compute VCG prices.

Unfortunately, VCG is not a practical option for marketplaces already built on GSP, which comprise the majority of those that presently rely on some form of the separability assumption. VCG prices are quite different from GSP prices, and switching from GSP pricing to VCG pricing—a quick exercise at Yahoo calculated immediate expected revenue more than 20% lower than GSP in one of its marketplaces—would create a disruption that is likely unacceptable in terms of revenue impact, confusion, and general explainability to advertisers. Case in point: engineers at Google independently discovered VCG pricing mere months after implementing GSP in their Adwords auction, but it was already deemed too late to switch [22].

---

[2] It may additionally be the case that the shape of click-rate (conditioned on view) functions varies considerably amongst the CPC ads, in which case one could not effectively apply a separable model even within the subset of ads that are priced CPC.
[3] The introduction in [6] also contains a quick survey of known exact algorithms for the problem.

1.3 Allocation Curves

Unable to use VCG, it is natural to shift focus towards generalizing GSP. Unfortunately, what "generalizing GSP" means is itself an open question that would be impossible to address adequately in the present work. A variety of methods for generalizing GSP have been tried in the literature that have vastly different approaches and lead to substantially different prices, e.g. [4, 5, 12, 16, 24]. Fortunately, this discussion can be set aside: instead of basing our implementation on a specific definition of GSP, we show how to compute full "allocation curves" that capture what an advertiser would have achieved with an alternate bid, which can accomodate them all.

The allocation curves we want to compute are the same functions that form the foundation of most single-parameter auction theory. The allocation curve $a_i : \mathfrak{R}_+ \to [0, 1]$ for bidder i given bids b_{-i} gives i's click probability in the allocation that results as a function of his submitted bid b_i. Possessing these curves confers substantial advantages over common auction implementations:

(1) *Flexibility and abstraction.* Explicitly computing allocation curves decouples allocation and pricing in the auction system, allowing nearly any desired pricing scheme to be implemented quickly and making it easy to refine the pricing mechanism later. For example, it is easy to compute simple GSP "threshold" prices (as we will discuss below) or more complex generalizations of GSP from [4] that limit the marginal costs associated with higher slots; it is similarly easy to start with threshold pricing and move to a more complicated method when the need arises. VCG prices can even be computed using a formula due to [17], as an area above a_i. Note the contrast with traditional externality pricing, which confers little or no information that would be useful for computing anything other than VCG prices.

(2) *Transparency.* By capturing counterfactual information at auction time, allocation curves aid in validation and forecasting. Advertisers often inquire about why their prices have changed and platforms often need to understand why prices are low/high in a particular auction. Additionally, advertisers often want to know how things will change if they modify their bids. In theory it is simple to compute such counterfactuals; unfortunately, real auctions are constantly evolving and, even from logged information, it is often difficult or impossible to precisely and consistently reconstruct the auction logic. Computing and logging the allocation curve explicitly obviates the need for reverse-engineering the auction and substantially facilitates ex-post analysis.

Since the allocation is deterministic, a_i is characterized by the m thresholds at which the advertiser moves up a position. In a separable world, computing these thresholds is again as easy as sorting — ad i gets the j-th slot (so its event probability is $\pi_{i,j} = \beta_i \alpha_j$) when it has the jth-highest value of $b_i \beta_i$. Thus, we can read off a_i after sorting ads by $b_i \beta_i$.

For example, consider the following setting with separable event probabilities and $\alpha_1 = 1$, $\alpha_2 = 0.9$ and $\alpha_3 = 0.1$:

Bidder	b_i	β_i	$\pi_{i,1}$	$\pi_{i,2}$	$\pi_{i,3}$
1	\$4	0.1	$0.1 \times 1 = 0.1$	$0.1 \times 0.9 = 0.09$	$0.1 \times 0.1 = 0.01$
2	\$3	0.2	$0.2 \times 1 = 0.2$	$0.2 \times 0.9 = 0.18$	$0.2 \times 0.1 = 0.02$
3	\$2	0.1	$0.1 \times 1 = 0.1$	$0.1 \times 0.9 = 0.09$	$0.1 \times 0.1 = 0.01$

To find the allocation curve for bidder 2, we consider the $b_i \beta_i$ values $b_1 \beta_1 = 0.4$, $b_2 \beta_2 = 0.2 b_2$, and $b_3 \beta_3 = 0.2$. When $0.2 b_2 \geq 0.4$, bidder 2 will win the top slot, when $0.4 > 0.2 b_2 \geq 0.2$ it will win the second slot, and when $0.2 > 0.2 b_2$ it will win the third slot. Thus, the allocation curve for bidder 2

will be as follows:

$$a_2(b_2) = \begin{cases} 0.2 & b_2 \geq 2 \\ 0.18 & b_2 \in [1, 2) \\ 0.02 & b_2 < 1 \end{cases}.$$

Computing a_2 is easy because the thresholds at which the allocation changes ($b_2 = 1$ and $b_2 = 2$) can be found by inspection once we have $b_1\beta_1$ and $b_3\beta_3$ in sorted order.

However, computing allocation curves is more complicated when event probabilities lack structure. Each point on an allocation curve represents a counterfactual, and the counterfactual optima may be quite different from the true optimum. Consider the following example:

Bidder	b_i	$\pi_{i,1}$	$\pi_{i,2}$	$\pi_{i,3}$
1	\$4	0.1	0.09	0.01
2	\$3	0.1	0.09	0.01
3	\$2	0.1	0.02	0.01

The optimal *bidder* $\longrightarrow$ *slot* allocation is $1 \longrightarrow 1, 2 \longrightarrow 2, 3 \longrightarrow 3$ with a total value of 0.69. optimal allocation would instead be $1 \longrightarrow 2, 2 \longrightarrow 3, 3 \longrightarrow 1$; not only did bidder 2 move down to slot 3 (naturally, since its bid was lowered), the relative order of bidders 1 and 3 changed. In order to compute allocation curves, we will need to find all the counterfactual optimal matchings that would result from alternative bids. The allocation curve for bidder 2 in this case will be

$$a_2(b_2) = \begin{cases} 0.1 & b_2 \geq 4 \\ 0.09 & b_2 \in [1.75, 4) \\ 0.01 & b_2 < 1.75 \end{cases}$$

One straightforward algorithm for computing a_i repeatedly runs the core allocation algorithm in a binary search. The thresholds can naïvely be found in time $O(nm^3 \sum_{i \in I} \log 100b_i)$, assuming that bids are expressed to the penny. Roughly speaking, this algorithm iteratively picks counterfactual bids for bidder i, holding all others constant, re-runs the bipartite matching algorithm, and checks the slot that bidder i receives; this is done repeatedly until the threshold bids are identified. This is sound but not fast enough in practice.

Our central result in this paper is an algorithm that computes the curves a_i substantially faster, in the same $O(nm^2)$ time as the core allocation algorithm. The example above hints at the first problem we need to solve: for any bidder x and slot y, compute the optimal matching in which x is assigned to slot y. Given the true optimal allocation, we show that these counterfactuals can be done quickly by computing short cycles in an appropriate graph. Given these optima, we must identify the thresholds at which we transition from one optimum to another, which can be formulated as an upper envelope computation.

1.4 Generalizing GSP

Though what precisely a generalized GSP looks like is out of scope, we will adopt a simple definition for the sake of exposition. The following folklore definition is broadly applied for less-substantial generalizations of GSP (e.g., [12, 16]) and theoretically rationalized for general contexts in [24]: if a bidder i is assigned to slot j, we define the GSP price as the smallest (infimum) bid it can report and still be assigned into slot j or higher by the ad selection algorithm. Note that when separability holds, these prices will be the same as in a traditional GSP auction.

Perhaps the most immediately practical result of our work is a fast algorithm for computing these GSP prices. We present a simplification of our general algorithm for computing a_i that computes GSP prices and runs in time $O(nm^2)$ (Theorem 3.8). In practice, our implementation of the algorithm

leads to overall run time for assignment and pricing together of less than 1ms on the instances we see in production, which is more than adequate for our goals.

1.5 Multi-dimensional bidding

Moving from the separable model to one in which bidders can have completely idiosyncratic click probabilities across slots, as discussed above, brings a heavy dose of realism to the stylized models applied to ad auctions. But in some cases there is reason to go further yet. Even in the model we proposed above, bidders can still communicate their full valuations across slots in a single number, since the click (or impression, or conversion, etc.) probabilities are known to the auctioneer. But what if bidders' values across slots differ in ways that are known only *privately*?

Motivated by this concern, we go on to consider a more general setting where advertisers are allowed to submit separate bids for each slot. That is, we assume that bidder $i \in I$ gives us a bid $b_{i,j}$ for each $j \in J$. In this case, i's (expressed) expected value for slot j is defined to be $v_{i,j} = b_{i,j} \cdot \pi_{i,j}$. While this is not currently supported by major online ad publishers, we argue that such a system has the potential to significantly increase overall advertiser value in some marketplaces. Indeed, advertisers often choose the cost-per-click (CPC) payment model simply due to the ease of tracking clicks, despite clicks only being a proxy for their real "event of interest". Most advertisers actually care about *conversions*, i.e., various user actions on the advertiser's website like buying an item, downloading an app, creating an account, or just spending enough time browsing. Often conversions are observed privately by the advertiser, and thus are impossible for the auctioneer to accurately predict. If every click yields an equal probability of conversion, then a single per-click bid remains perfectly sufficient, but what about otherwise?

We examined the following hypothesis about conversion rates in Yahoo's Native Ads: users who choose to scroll down the stream are more engaged with the content and, as a result, on average clicks in lower positions will yield higher conversions than those above. In Figure 2 we plot the average rate of conversion conditioned on click for each slot position from 1 to 10 on a subset of Yahoo Native Ads data where conversions were observable. Indeed, this rate grows significantly from slot 1 to slot 8 or 9 before leveling off. An analogous and more principled study for Search ads was performed in [3].

On the other side of the equation, click-rates degrade sharply as one moves down the page, so if an advertiser is looking for wide exposure it cannot just bid for lower positions in order to reap the higher value clicks. Additionally, some advertisers (e.g., those publicizing a brand) are more or less only interested in high position slots due to their higher exposure and general viewability. This diversity of preferences across the slot positions further supports the idea that an ad marketplace with slot-specific bidding has its place in the online advertising ecosystem, or is at least worthy of further study.

Efficient allocation of bidders to slots can, again, be achieved via a maximum weighted bipartite matching algorithm here; but pricing becomes much more complex. While the generality of VCG makes it an always-ready candidate pricing mechanism, the notion of GSP-like pricing is not immediately well-defined in this context. Fortunately, we can expand our notion of allocation curves to this setting, which will allow us to recover some sensible "in spirit" generalizations of GSP. The marginal social choice function f_i of bidder i given bids b_{-i} is a function $f_i : \mathfrak{R}_+^n \to J$ that outputs the slot j that i wins as a function of his bid vector $b_i \in \mathfrak{R}_+^n$. We design an algorithm to compute marginal social choice functions for all bidders with run time $O(nm^2)$ (Theorem 4.1), and discuss various options for GSP-like pricing in Section 4.

While we show that computation of the marginal social choice functions can be done efficiently, there are many practical challenges to using multiple bids per bidder. First, just carrying around a vector of many bids instead of one bid per ad creates significant challenges at the web scale. Second,

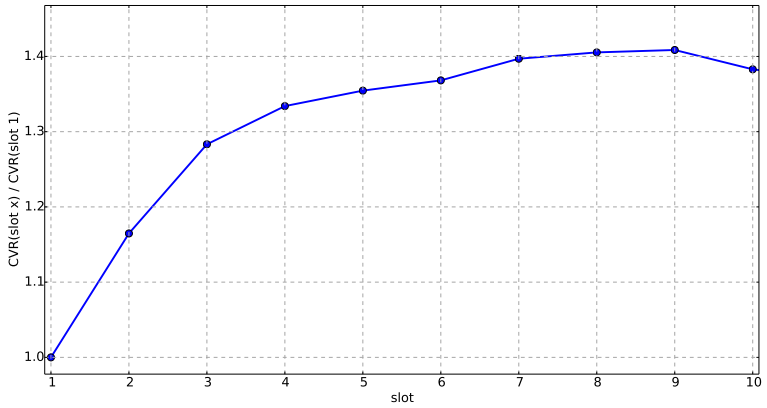

Fig. 2. Conversion-rate (probability of a conversion conditioned on the ad being clicked) as a function of slot, aggregated over all ad impressions on a set of pageviews that yielded over 300K clicks in slot 1, descending to roughly 30K clicks in slot 10.

the "equilibrium features" GSP possesses in the single-dimensional setting (e.g., local truthfulness) do not readily extend. The question of designing a GSP-like mechanism with provable equilibrium properties remains open.

2 RELATED LITERATURE

In computational advertising, two main research threads relate to our work: papers that analyze the separability assumption [3, 8–11] and papers that generalize GSP [1, 4, 5, 12, 16, 24]. Three natural intuitions underly the work critiquing separability. One intuition is that ads belonging to major brands benefit less from being placed in a higher slot. This is explored and generally confirmed in [8, 11]. Another idea is that user behavior follows a cascade model—a user looks at the first item, then conditionally decides whether to click or continue down the page—leading to substantial effects between ads that are not captured by an independent (separable) model; [9] estimates parameters for such a model. A third intuition is that a top slot may attract more accidental or exploratory clicks, whereas clicks on a less-prominent slot signal a stronger intent. This leads to non-separability in the conversion rate (likelihood that a click leads to a purchase, sales lead, etc.) as shown in [3].

A variety of papers use different techniques to generalize GSP, including definitions based on equilibrium properties [12], heuristic properties of the pricing scheme [16], structure of the allocation algorithm [5], or truthfulness for alternative models of advertiser preferences [1, 4, 24]; however, a definitive generalization is still an open question. Since the prices we compute as "GSP prices" can be derived as the truthful solution for preferences alternative to quasilinear utilities, our work is also related to the corresponding literature on stable matchings in the assignment model when preferences are not quasilinear. [1] builds on the Hungarian Method to design a generalized auction framework that computes a stable matching for a broad class of preferences that can include our model of GSP prices and runs in $O(nm^3)$ time. While stability is related to truthfulness, the concepts diverge in general models, so the outcome chosen by [1] would not generally be a maximum weighted matching and the outcome/prices from our auction would not generally be a

stable matching. For information on stable matchings with non-quasilinear utilities, see the related work section of [1]. For an overview of the breadth of work on matching markets, see [20].

Outside advertising, our work has ties to competitive equilibria and the assignment model in matching markets. In the language of bidders and slots, letting $v_{i,j}$ be expected value to i for winning slot j, a competitive equilibrium for quasilinear bidders is a matching with slot prices p_j such that each bidder is matched to a slot that maximizes $v_{i,j} - p_j$. The minimum competitive equilibrium is always well-defined and always corresponds to VCG (a maximum weighted matching with VCG prices) [13]. Moreover, the VCG prices are the dual values associated with an optimal matching in a standard LP formulation of the assignment problem [2]. As a result, they naturally occur implicitly or explicitly in algorithms and mechanisms that have a primal-dual structure, such as the Hungarian Method and related ascending price auctions [13]. Additionally, some of our techniques are closely related to the more general literature on competitive equilibria with gross substitutes. The augmentation graph we use as a starting point for our algorithm is effectively equivalent to the exchange graph of [14], which is used as a method of computing equilibrium prices (see also [18]). We leverage the structure of this graph to identify configurations that are optimal with one fixed assignment, allowing us to recover the allocation curve.

3 CLASSICAL CASE: ONE BID PER BIDDER

Our main result is an algorithm that computes bidders' full allocation curves in time $O(nm^2)$, the same time required to compute a maximum weighted matching using the Hungarian Method.

We first recall our model. We have n bidders $I = \{1, \ldots, n\}$ who are to be matched to m slots $J = \{1, \ldots, m\}$. Each bidder $i \in I$ reports a bid b_i and has an event probability $\pi_{i,j}$ for each $j \in J$. For convenience, we add dummy slots $\{m+1, \ldots, n\}$ with $\pi_{i,j} = 0$ and call the result $J' = \{1, \ldots, m, m+1, \ldots, n\}$. For a matching $\sigma : J' \to I$, we write the total value as $V^\sigma = \sum_j b_i \pi_{\sigma(j),j}$. Given bids, the auction computes a maximum weighted matching σ^* (maximizing V^σ) and we assume WLOG that bidders are labeled so that bidder i is matched to slot i under this matching ($\sigma^*(j) = j$). Let $x_i(b)$ denote the event probability for bidder i as a function of all the bids. For a fixed bidder i and bids b_{-i}, we say that i's allocation curve is the function $a_i(b_i) = x_i(b_i, b_{-i})$. Since the allocation curve is a step function with at most $m+1$ steps we represent it as list of bid-value pairs, one for each step.

Algorithm 1 describes our main algorithm and Algorithm 2 an optimized implementation. Both algorithms implement the same logic; the second achieves better asymptotic run-time by exploiting structure. First, we apply the Hungarian Method to compute a maximal matching. From this matching, we construct a weighted graph in which the shortest cycle through edge $j \to i$ identifies the best matching that assigns bidder i to slot j. Computing all-pairs shortest paths identifies all such cycles, from which we are able to infer the allocaiton curve for each bidder. Algorithm 1 treats dummy nodes as real slots and uses a naive implementation of the Hungarian Method for matching and the Floyd-Warshall algorithm for all-pairs shortest paths. Algorithm 2 is aware that some nodes are just dummy nodes. It uses an optimized version of the Hungarian Method and a modified version of Floyd-Warshall that exploits structure in the paths we need to compute.

Our central theorem is that these algorithms quickly compute a maximal matching as well as all n allocation curves:

THEOREM 3.1. *Algorithms 1 and 2 compute a maximal matching σ^* and one allocation curve $a_i(b_i)$ for each bidder. Algorithm 1 runs in time $O(n^3)$ and Algorithm 2 runs in time $O(nm^2)$.*

Our analysis begins with a series of claims and lemmas about matchings and paths that apply equally to Algorithms 1 and 2. The proof of Theorem 3.1 connects these claims to the algorithms. The following graph is effectively equivalent to the exchange graph of [14]. The subsequent observations

ALGORITHM 1: $O(n^3)$ Algorithm for Maximum Weighted Matching with Allocation Curves

Input: Bids b_i and event probabilities $\pi_{i,j}$
Output: Maximum weighted matching σ^*; allocation curves $a_i(b_i)$.

1 Run the Hungarian Method to compute a maximum matching $\sigma^* : J' \to I$.
Relabel bidders so that $\sigma^*(i) = i$ and let $V^* = \sum_j b_j \pi_{j,j}$ denote the matching's value.

2 Let G be a complete directed graph on vertices $J' = [n]$ with edge weights $w_{i,j} = b_j \pi_{j,j} - b_j \pi_{j,i}$.
Run Floyd-Warshall to compute all-pairs shortest path distances $d_{i,j}$ in G.

3 **for** $i \in I$ **do**
  for $j \in J'$ **do**
    Define a linear function $l_{i,j}(z) = \pi_{i,j} z + (V^* - \pi_{i,i} b_i - d_{i,j})$.
  end
  Compute the upper-envelope of the lines $l_{i,j}(z)$ as a sequence of segments. Call it $A_i(z)$.
  Compute $a_i(z) = \frac{dA_i}{dz}$ (i.e., one slope per segment, with the starting point of the segment). This is bidder i's allocation curve.
end

are slightly stronger than their natural parallels in the gross substitutes literature owing to the unit-demand preferences in our setting [14, 15, 18]:

Definition 3.2 (Augmentation Graph). The *augmentation graph* G^σ is a directed graph on $|J'| = n$ nodes with edge weights $w^\sigma_{i,j} = v_{\sigma(j),j} - v_{\sigma(j),i}$. When σ is not specified we assume $\sigma = \sigma^*$ (the optimal matching chosen by the graph), so $w_{i,j} = w^{\sigma^*}_{i,j} = v_{\sigma^*(j),j} - v_{\sigma^*(j),i} = v_{j,j} - v_{j,i}$ (bidders are labeled so that $\sigma^*(j) = j$).

Observation 1. *The negative weight $(-w^\sigma_{i,j})$ is the amount the objective value would change if bidder $\sigma(j)$ were matched to slot i instead of slot j.*

Definition 3.3 (Augmenting Subgraph). For two matchings σ and σ', the *augmenting subgraph* $H^{\sigma \to \sigma'}_v$ is a subgraph of G^σ_v. This graph has a directed edge (i, j) if for some $t \in I$ the edge (t, i) belongs to matching σ' and the edge (t, j) belongs to matching σ.

Observation 2. *Suppose we are given two matchings σ and σ' of value V^σ and $V^{\sigma'}$ in the original bipartite graph. The total negative weight of edges in $H^{\sigma \to \sigma'}_v$ is equal to the change in objective value from σ to σ', i.e.,*

$$V^{\sigma'} = V^\sigma - \sum_{(i,j) \in H^{\sigma \to \sigma'}_v} w^\sigma_{i,j} \ .$$

Claim 1. *For any σ and σ', the connected components of $H^{\sigma \to \sigma'}_v$ are directed cycles and disconnected vertices; any subgraph H of G^σ_v consisting only of directed cycles and disconnected vertices corresponds to a matching σ'.*

Proof. First, we show that $H^{\sigma \to \sigma'}_v$ is as described. If bidder i is matched to the same slot in σ and σ', i has degree 0 in $H^{\sigma \to \sigma'}_v$ and is a disconnected vertex. For all remaining nodes, each node has exactly one incoming edge from the slot that bidder i gets and exactly one outgoing edge to the bidder that gets slot i. A connected component whose nodes have in-degree=outdegree=1 is a cycle.

Second, we show how to construct σ' from a given H: if j is a disconnected vertex, leave bidder j matched to slot j; otherwise, let (i, j) denote the incoming edge to j in H and match bidder j to slot i. □

ALGORITHM 2: $O(nm^2)$ Algorithm for Maximum Weighted Matching with Allocation Curves

Input: Bids b_i and event probabilities $\pi_{i,j}$
Output: Maximum weighted matching σ^*; allocation curves $a_i(b_i)$.

1 Run the Hungarian Method to compute a maximum matching $\sigma^* : J \to I$. Assign unmatched bidders to slots $\{m+1, \ldots, n\}$ arbitrarily to get a matching $\sigma^* : J' \to I$. Relabel bidders so that $\sigma^*(i) = i$ and let $V^* = \sum_j b_j \pi_{j,j}$ denote the matching's value.

2 Let G be a complete directed graph on vertices $J' = [n]$ with edge weights $w_{i,j} = b_j \pi_{j,j} - b_j \pi_{j,i}$ (do not store G explicitly). Compute shortest paths $d_{i,j}$ that contain at most one node in $\{m+1, \ldots, n\}$ as follows:

`/* Standard Floyd-Warshall initialization over the` i, j `pairs we will need. */`

for $i \in I, j \in J$ **do**

$$d_{j,i} \leftarrow \begin{cases} 0 & i = j \\ w_{i,j} & otherwise. \end{cases}$$

end

`/* Standard Floyd-Warshall update step using only intermediate nodes in` $[m]$`. After this step, each` $d_{i,j}$ `will contain the shortest path distance from` i `to` j `along paths whose intermediate nodes are all in` $[m]$`. */`

for $k \in \{1, \ldots, m\}, i \in I, j \in J \cup \{m+1\}$ **do**

$d_{i,j} \leftarrow \min(d_{i,j}, d_{i,k} + d_{k,j})$

end

`/* When` $i, j \in [m]$ `we do extra Floyd-Warshall style update steps to consider paths from` i `to` j `that contain exactly one node` k `from` $\{m+1, \ldots, n\}$`. */`

for $k \in \{m+1, \ldots, n\}, i \in \{1, \ldots, m\}, j \in J$ **do**

$d_{i,j} \leftarrow \min(d_{i,j}, d_{i,k} + d_{k,j})$

end

3 **for** $i \in I$ **do**

  for $j \in J \cup \{m+1\}$ **do**

    Define $l_{i,j}(z) = \pi_{i,j} z + (V^* - \pi_{i,i} b_i - d_{i,j})$.

  end

  Compute the upper-envelope of the lines $l_{i,j}(z)$ as a sequence of segments. Call it $A_i(z)$.

  Compute $a_i(z) = \frac{dA_i}{dz}$ (i.e., one slope per segment, with the starting point of the segment). This is bidder i's allocation curve.

end

Observation 3. *A negative weight cycle in G_v^σ corresponds to an augmenting path in the original bipartite graph that can be used to generate a matching with a higher value.*

Observation 4. *G_v^σ has a negative weight cycle if and only if it is* not *optimal.*

Claim 2. *Let σ^* be an optimal matching for $v_{i,j}$ and fix a bidder x and slot y. There exists σ^+ such that:*

(1) *the matching σ^+ is optimal among matchings that assign slot y to bidder x,*
(2) *the augmentation graph $H_v^{\sigma^* \to \sigma^+}$ contains exactly one cycle C if x is not matched to y in σ^* (C must contain the edge (y, x)) and contains no edges otherwise, and*
(3) *If C exists then $|C \cap \{m+1, \ldots, n\}| \leq 1$, that is, at most one vertex in C corresponds to a dummy slot.*

Proof. If x is matched to y in σ^* we can trivially take $\sigma^+ = \sigma^*$, whose augmentation graph has no edges. Thus, assume x is not matched to y in σ^*.

We first show that an appropriate σ^+ exists that contains only one cycle C. For any constrained-optimal matching σ' we can write

$$V^{\sigma'} = V - \sum_{(i,j)\in H_v^{\sigma^*\to\sigma'}} w_{i,j} \ .$$

Suppose that $H_v^{\sigma^*\to\sigma'}$ contains cycles $C_0, C_1, \ldots, C_k$ where C_0 includes the edge (y, x). Let $C = \bigcup_{l=1}^{k} C_l$ (i.e., all edges in H except the cycle C_0). We abuse notation and write $\sigma' - C$ as the matching implied by the augmentation graph $H_v^{\sigma^*\to\sigma'}$ with edges from C removed. We can write

$$\begin{aligned} V^{\sigma'} &= V^{\sigma^*} - \sum_{(i,j)\in H_v^{\sigma^*\to\sigma'}\setminus C} w_{i,j} - \sum_{(i,j)\in C} w_{i,j} \\ &= V^{\sigma'-C} - \sum_{(i,j)\in C} w_{i,j} \ . \end{aligned}$$

If $\sum_{(i,j)\in C} w_{i,j} > 0$, then $V^{\sigma'-C} > V^{\sigma'}$. However, $\sigma' - C$ still matches x to y, contradicting optimality of σ'. On the other hand, if $\sum_{(i,j)\in C} w_{i,j} < 0$ then we have a negative-weight-cycle in G_v, contradicting optimality of σ^*. Thus,

$$\sum_{(i,j)\in C} w_{i,j} = 0$$

for any cycle C that does not contain the edge (y, x), and

$$\begin{aligned} V^{\sigma'} &= V^{\sigma^*} - \sum_{(i,j)\in H_v^{\sigma^*\to\sigma'}} w_{i,j} \\ &= V - \sum_{(i,j)\in H_v^{\sigma^*\to\sigma'}\setminus C} w_{i,j} = V^{\sigma'-C} \ , \end{aligned}$$

implying $\sigma' - C$ is also constrained-optimal, so we take $\sigma^+ = \sigma' - C$.

Finally, we show that we can choose C so that it contains at most one dummy slot. Suppose we are given a constrained-optimal σ' with a single cycle

$$C = \{x, \ldots, w_0, z_1, w_1, \ldots, z_2, w_2, \ldots, y\}$$

where $x, \ldots, w_0 \in [m]$ and $w_2, \ldots, y \in [m]$ (possibly $x = w_0$ and/or $y = w_2$) but $z_1, z_2 \notin [m]$. Split C into two cycles as follows:

$$C_0 = \{x, \ldots, w_0, z_1, w_2, \ldots, y\} \quad \text{and} \quad C_1 = \{w_1, \ldots, z_2\}$$

and call the resulting matching σ''. We have

$$V^{\sigma''} = V^{\sigma'} + v_{w_2,z_1} - v_{w_2,z_2} + v_{w_1,z_1} - v_{w_1,z_2} \ .$$

However, since z_1 and z_2 are both dummy slots we have $v_{w_2,z_1} = v_{w_2,z_2} = v_{w_1,z_1} = v_{w_1,z_2} = 0$ and so $V^{\sigma''} = V^{\sigma'}$. Now, our earlier argument says that only the cycle C_0 is required to construct σ^+, so we take $\sigma^+ = \sigma'' - C_1$. Remark that C_0 contains only one node not in $[m]$ (specifically z_1) as desired.

This construction for splitting C works as long as $x, y \in [m]$. When $x \notin [m]$, take the cycles $C_0 = \{x, w_2, \ldots, y\}$ and $C_1 = \{w_1, \ldots, z_2\}$ instead. □

Lemma 3.4. *Fix b, a bidder indexed by x and a slot indexed by $y \neq x$. Let σ^* be an optimal matching (note that due to our relabelling $\sigma(x) = x$) and let σ' be any matching that is optimal subject to the constraint that bidder x is matched to slot y. Then,*

$$V^{\sigma^+} = V^{\sigma^*} - w_{y,x} - d_{x,y}$$

where $d_{x,y}$ is the shortest path distance from x to y in $G_v^{\sigma^*}$. Moreover, there is a path from x to y of length $d_{x,y}$ containing at most one vertex not in $[m]$.

Proof. Notice that all constrained-optimal matchings will have the same value. By the preceding claim, there exists a constrained-optimal matching σ^+ that has an augmenting subgraph $H_v^{\sigma^* \to \sigma'}$ consisting of a single cycle C_0 (and disconnected nodes), where C_0 contains edge (y, x). We can write

$$V^{\sigma^+} = V^{\sigma^*} - \sum_{(i,j)\in C_0} w_{i,j} \ .$$

Moreover, any cycle C including the edge (y, x) will define a matching σ' that assigns x to y, so we can conclude from optimality of σ^+ that C_0 minimizes $\sum_{(i,j)\in C_0} w_{i,j}$ over all cycles containing (y, x). (Note that $G_v^{\sigma^*}$ cannot have negative-weight cycles because σ^* is optimal.)

Finally, given that C_0 minimizes $\sum_{(i,j)\in C_0} w_{i,j}$ over all cycles containing (y, x), we know that

$$\sum_{(i,j)\in C_0} w_{i,j}^{\sigma^*} = w_{y,x} + d_{x,y}$$

where $d_{x,y}^{\sigma^*}$ is the shortest path from x to y. Since we know that we can always choose C_0 so that it contains at most one vertex not in $[m]$, it follows that there is a shortest path containing at most one vertex not in $[m]$ and the lemma immediately follows. □

Next, we introduce the notion of configuration lines, which have been put to use in prior work [4].

Definition 3.5. The *configuration line* for bidder i in matching σ is a function of b_i given by $l_{i,\sigma}(b_i) = \pi_{i,\sigma^{(-1)}(i)} \cdot b_i + V_{-i}^{\sigma}$, where $\sigma^{(-1)}(i)$ is the slot to which i is matched in σ and $V_{-i}^{\sigma} = \sum_{j\neq i} \pi_{j,\sigma^{(-1)}(j)} b_j$.

Claim 3. *Bidder i's allocation curve can be defined as the derivative of the upper envelope of the configuration lines for all σ.*

Proof. Indeed, the upper envelope of configuration lines defines the value of the optimal matching as a function of bidders i bid b_i. For each b_i the maximum is achieved on some matching σ and corresponding line $l_{i,\sigma}(b_i) = \pi_{i,\sigma^{(-1)}(i)} \cdot b_i + V_{-i}^{\sigma}$ is a linear function of b_i. The derivative at point b_i has value $\pi_{i,\sigma^{(-1)}(i)}$ which is "event probability" if we allocate bidder i to the spot $\sigma^{(-1)}(i)$, which matches the definition of the allocation curve. □

We can now leverage the structure of the matching problem to restrict the set of lines we need to consider when computing the upper envelope or its derivative:

Lemma 3.6. *Fix bidder x. For each slot y define the line $l_{x,y}(z) = \pi_{x,y} \cdot z + (V^* - \pi_{x,x} b_x - d_{x,y})$. The derivative of the upper envelope of these lines is bidder x's allocation curve.*

Moreover, when $m < n$, it is sufficient to compute the upper envelope of the lines $l_{x,y}$ where $y \in \{1, \ldots, m, m+1\}$.

Proof. All configuration lines $l_{x,\sigma}$ where $x = \sigma(y)$ will have the same slope. Thus, when computing the upper envelope, of all σ that match x to y, only the ones that maximize V_{-x}^{σ} will be relevant. Consequently, it suffices, for each y, to find one line with maximal value of V_{-x}^{σ}.

Fixing slot y, maximizing V^{σ}_{-x} is equivalent to finding the constrained-optimal matching σ' that matches bidder x to slot y. As shown earlier, this matching has value $V^{\sigma'} = V^{\sigma^*} - w_{y,x} - d_{x,y}$ and thus

$$\begin{aligned} V^{\sigma'}_{-x} &= V^{\sigma^*} - w_{y,x} - d_{x,y} - \pi_{x,y} b_x \\ &= V^{\sigma^*} - (\pi_{x,x} b_x - \pi_{x,y} b_x) - d_{x,y} - \pi_{x,y} b_x \\ &= V^{\sigma^*} - d_{x,y} - \pi_{x,x} b_x \ , \end{aligned}$$

from which the first part of the lemma follows.

Finally, remark that dummy slots are interchangeable, so $\pi_{x,y}$ and $d_{x,y}$ will be the same for all $y \in \{m+1, \ldots, n\}$. Thus, the lines $l_{x,y}$ will be the same and it is sufficient to take only one when computing the upper envelope (we take $y = m+1$). □

Proof of Theorem 3.1. For Algorithm 1, correctness follows because the algorithm directly computes allocation as per Lemma 3.6. Steps (1) and (2) take $O(n^3)$ time due to the classical run time bounds on the Hungarian and Floyd-Warshall algorithms. Once we have computed V^* and the table with all pair shortest paths, it takes $O(n^2)$ to run step (3), for an overall runtime of $O(n^3)$.

Algorithm 2 is an optimization of Algorithm 1. The Hungarian Method is easily modified to run in time $O(nm^2)$ [19]. Step (2) is simply an optimization of the Floyd-Warshall algorithm that takes into account two facts: (1) we only care about paths from $x \in I$ to $y \in J \cup \{m+1\}$ (dummy nodes are equivalent) and (2) we only care about paths that contain at most one node in $\{m+1, \ldots, n\}$ as per Lemma 3.4. The first loop in step (2) initializes the nm distances we care about, the second loop computes the shortest paths where all intermediate nodes are in $\{1, \ldots, m\}$ in the same pattern as the original Floyd-Warshall algorithm, and the third loop considers paths where both endpoints are in $\{1, \ldots, m\}$ and exactly one intermediate node is in $\{m+1, \ldots, n\}$. The second and third loops run in $O(nm^2)$ time. Finally, step (3) is optimized to only use lines for $J \cup \{m+1\}$ as per Lemma 3.6 and requires $O(nm)$ time, for a total asymptotic runtime of $O(nm^2)$. □

3.1 GSP pricing

Our GSP prices are an important special case that can be computed with a slightly simpler algorithm (Algorithm 3). We use the following definition of GSP prices:

Definition 3.7 ("GSP" Price). The GSP price for bidder i is defined as the minimum (infimum) bid that i can submit that will maintain the slot he received.

$$p_i^{GSP} = \inf\{z \mid x_i(b_{-i}, z) \geq x_i(b)\}$$

This definition has long existed in the folklore and is commonly applied in practice. More recently, [24] justified them theoretically as being truthful for a "value maximizing" bidder who tries to maximize x_i while keeping $p_i \leq b_i$ (any outcome with $p_i \leq v_i$ is preferred to any outcome with $p_i > v_i$). The following claim is a direct application of the characterization from [24]:

Claim 4. *The "GSP" price is truthful for value maximizers (almost everywhere[4]).*

If we ran Algorithm 1 we could easily compute GSP prices from the allocation curves a_i, but if we know we only want GSP prices we can leverage the same ideas in a simpler algorithm, shown in Algorithm 3 and proven below.

[4]Truth-telling is a best-response as long as there wouldn't be a tie under truthful bidding, which happens almost everywhere in a continuous bid space (see [24]).

Theorem 3.8. *Letting $d_{i,j}$ be the shortest path from i to j in the augmentation graph for the optimal matching σ^*, the "GSP" price for bidder i is equal to*

$$p_i^{GSP} = \max_{j|\pi_{i,j}<\pi_{i,i}} \frac{-d_{i,j}}{\pi_{i,i} - \pi_{i,j}}$$

if there is at least one index j such that $\pi_{i,j} < \pi_{i,i}$, or equals 0 otherwise.

Algorithm 3 computes a maximal matching σ^* and GSP prices p_i^{GSP}. It can be implemented in time $O(nm^2)$ using the same tricks as Algorithm 2.

Proof. We want to find the smallest $b_i \geq 0$ such that the augmentation graph for σ^* does not have a negative-weight cycle, since this is the smallest b_i for which σ^* is optimal.

When we change b_i, only the weights on edges incoming to node i change, i.e., only $w_{j,i}$ change. If any negative cycle exists, there exists a simple negative cycle that passes through node i at most once and therefore only includes one edge with a modified weight. It thus suffices to consider the edges (i, j) individually to see if they are part of a negative-weight cycle.

An edge (i, j) will be part of a negative-weight cycle if $d_{i,j} + w_{j,i} < 0$ where $d_{i,j}$ is the shortest path from i to j in the augmentation graph. Thus, to ensure no negative weight cycles we need

$$d_{i,j} \geq -w_{j,i} = -(v_{i,i} - v_{i,j}) = -b_i(\pi_{i,i} - \pi_{i,j}) \ .$$

When $\pi_{i,i} = \pi_{i,j}$ this is irrelevant. When $\pi_{i,i} > \pi_{i,j}$ this is equivalent to

$$b_i \geq \frac{-d_{i,j}}{\pi_{i,i} - \pi_{i,j}} \ ,$$

and when $\pi_{i,j} > \pi_{i,i}$ this is equivalent to

$$b_i \leq \frac{-d_{i,j}}{\pi_{i,i} - \pi_{i,j}} \ .$$

Since we want a lower-bound, we take the maximum over the lower-bounds to get the characterization of p_i^{GSP} in the theorem.

The run time analysis for the Hungarian Method and Floyd-Warshall is the same as for computing allocation curves and the same tricks from Algorithm 2 can be applied to achieve $O(nm^2)$ run time (see the proof of Theorem 3.1). Computing the n prices p_i^{GSP} requires $O(nm)$ time.

□

ALGORITHM 3: Algorithm for Computing Maximum Weighted Matching with GSP Prices

Input: Bids b_i and Event Probabilities $\pi_{i,j}$
Output: Allocation Curves $a_i(b_i)$ as

1 Run the Hungarian Method to compute a maximum matching $\sigma^* : J \to I$.
Relabel bidders so that $\sigma^*(i) = i$ and let $V^* = \sum_j b_j \pi_{j,j}$ denote the matching's value.

2 Let G be a complete directed graph on vertices $J' = [n]$ with edge weights $w_{i,j} = b_j \pi_{j,j} - b_j \pi_{j,i}$.
Run Floyd-Warshall to compute all-pairs shortest path distances $d_{i,j}$ in G.

3 **for** $i \in I$ **do**
  $p_i^{GSP} = \max_{j|\pi_{i,j}<\pi_{i,i}} \frac{-d_{i,j}}{\pi_{i,i}-\pi_{i,j}}$
end

4 GENERAL CASE: SEPARATE BIDS FOR EACH SLOT

While using a single bid is convenient, there is often reason to believe that bidders' values for an event may be substantially different for different slots, as discussed earlier. Allowing each bidder $i \in I$ to report a separate bid $b_{i,j}$ for each slot $j \in J$ allows them to express this. In this section we seek algorithms for this general setting that are parallel to our single bid algorithms.

We first recall our model for the general case. We have n bidders $I = \{1, \ldots, n\}$ who are to be matched to m slots $J = \{1, \ldots, m\}$. Each bidder $i \in I$ reports a bid $b_{i,j}$ for each slot $j \in J$ and has an event probability $\pi_{i,j}$ for each $j \in J$. For a matching $\sigma : J \to I$, we write the total value as $V^\sigma = \sum_j b_{i,j}\pi_{\sigma(j),j}$. Given bids, the auction computes a maximum weighted matching σ^* (maximizing V^σ) and, WLOG, we assume that bidders are labeled so that $\sigma^*(j) = j$.

The allocation curve a_i cannot be cleanly defined in a multi-parameter bid space. Instead, we define the *marginal social choice function* for a fixed bidder i and bids b_{-i} as the function $f_i : \mathfrak{R}_+^m \to J \cup \{m+1\}$ yielding the slot bidder i gets when it reports bids $b_{i,j}, j \in J$. The dummy slot $m+1$ corresponds to the event when a bidder does not get any slots in J. The marginal social choice function can be understood as a partition of $\mathfrak{R}_+^n$ (the bid space of bidder i) into $m+1$ regions, one for each slot in $J \cup \{m+1\}$. Let $B_{i,j} \subseteq \mathfrak{R}_+^n$ denote the set of bid vectors $b_i \in \mathfrak{R}_+^n$ that would cause bidder i to get slot j (given the bids of others b_{-i}).

In multi-parameter settings, another useful piece of information for counterfactual analysis is the full menu of VCG prices. That is, for each slot $j \in J \cup \{m+1\}$, what would the VCG price have been if bidder i had submitted bids $b_{i,j}$ such that $f_i(b_i) = j$? For reasons that we will see, the VCG price is relevant even when mechanisms do not aim to use VCG pricing.

Algorithm 4 computes a concise representation of the marginal social choice functions f_i, as well as the full menus of VCG prices. This all assumes that behavior in the case of ties is straightforward and/or irrelevant (e.g., if ties are broken by solving an NP-hard problem, we haven't completely recovered f_i). The proof of the following theorem can be found in the online version of the paper:

Theorem 4.1. *Algorithm 4 computes an optimal matching σ^* along with the n marginal social choice functions. It can be implemented in time $O(nm^2)$. The space required to store these functions is $O(nm)$.*

4.1 GSP prices

While we presented a fast algorithm to compute marginal social choice functions, the question of GSP-like pricing still stands. A natural starting point is to ask: given the optimal weighted bipartite matching where bidder $\sigma^*(j)$ is assigned to slot j, what would be the smallest bid that allows bidder $\sigma^*(j)$ to keep his slot in the optimal assignment? When bids are multi-parameter, even this intuition is not fully specific and lends itself to two possibilities (recall bidders are labeled such that $\sigma^*(j) = j$): (1) what is the smallest bid if i strategizes over all bids $b_{i,j}, j \in J$, or (2) what is the smallest bid $b_{i,i}$ if $b_{i,j}, j \neq i$ are naively held fixed? Call the former *Weak Generalized Second Price* (WGSP) and the latter *Aggressive Generalized Second Price* (AGSP).

Unfortunately, while both WGSP and AGSP prices will be easy to compute, neither will be immediately satisfactory as a generalization of GSP, as we will discuss below. The notion that GSP prices are truthful for value maximizers offers a lens to understand why—if we insist on computing a maximum weighted matching—there will be no prices that make the auction truthful for value maximizers, suggesting that any method of generalizing GSP will sacrifice some of the elegance of a simple GSP auction. Recovering all the properties we desire may require picking a different optimization algorithm, e.g., a greedy algorithm would admit truthful prices for value maximizers [23] (inducing the stable matching computed by [1]).

ALGORITHM 4: Algorithm for Computing Maximum Weighted Matching with Marginal Social Choice Functions

Input: Bids $b_i \in \mathfrak{R}_+^n$ and event probabilities $\pi_{i,j}$
Output: Maximum weighted matching σ^* and marginal social choice functions $f_i(b_i)$ as

Run the Hungarian Method to compute a maximum matching $\sigma^* : J \to I$.
Relabel bidders so that $\sigma^*(i) = i$ and let $V^* = \sum_j b_{j,j}\pi_{j,j}$ denote the matching's value.

Let G be a complete directed graph on vertices $J' = [n]$ with edge weights $w_{i,j} = b_{j,j}\pi_{j,j} - b_{i,j}\pi_{j,i}$.
Run Floyd-Warshall to compute all-pairs shortest path distances $d_{i,j}$ in G. Define $V_{-i}^{-j} = V^* - v_{i,i} - d_{i,j}$.

for $i \in I$ **do**
  Compute $f_i(0) = \text{argmax}_j V_{-i}^{-j}$ breaking ties arbitrarily.
end

for $i \in I$ **do**
  for $j \in J \cup \{m+1\}$ **do**
    Define $p_{i,j}^{VCG} \cdot \pi_{i,j} = V_{-i}^{-f_i(0)} - V_{-i}^{-j} = d_{i,j} - d_{i,f_i(0)}$.
  end
  The marginal social choice function for i is $f_i(b_i) = \text{argmax}_{j \in J \cup \{m+1\}} \left\{ b_{i,j} \cdot \pi_{i,j} - d_{i,j} \right\}$.
end

WGSP. WGSP is the natural generalization of the minimum bid property we used in the single-parameter regime. Unfortunately, it turns out to be degenerate — the WGSP price is simply the VCG price:

Theorem 4.2. $p_i^{WGSP} = p_i^{VCG}$

Proof. First, observe that since the auction will maximize the sum of values, reporting $b_{i,r} = 0$ for $r \neq j$ can only reduce the minimum report b_j required for bidder i to be matched to slot j. Thus, WLOG we can fix $b_{i,r} = 0$ for $r \neq j$ and find the minimum (infimum) $b_{i,j}$ such that i gets matched to slot j.

Recall our notation V_{-i}^{-j} for the maximum total value attainable by bidders other than i when they are allocated to slots other than j (i.e., the maximum total value attainable by bidders other than i if we fix i to be matched to slot j). In this notation, we observe that VCG externality prices can be written as $p_{i,j}^{VCG}\pi_{i,j} = V_{-i}^{-f_i(0)} - V_{-i}^{-j}$ where f_i is the marginal social choice function of bidder i given v_{-i}.

Suppose that the mechanism assigns i to a slot $k \neq j$. We know that the mechanism will get zero value from i and therefore will pick the outcome that maximizes value from bidders $-i$, thus achieving an objective value of $V_{-i}^{f_i(0)}$. On the other hand, when i wins slot j the objective value will be $V_{-i}^{-j} + b_{i,j}\pi_{i,j}$. Thus, bidder i will be assigned to slot j if $V_{-i}^{-j} + b_{i,j}\pi_{i,j} > V_{-i}^{f_i(0)}$, and if $V_{-i}^{-j} + b_{i,j}\pi_{i,j} < V_{-i}^{f_i(0)}$ then it will not be assigned to slot j. We can thus conclude that $V_{-i}^{-j} + p_{i,j}^{WGSP}\pi_{i,j} = V_{-i}^{f_i(0)}$ and it follows that

$$p_{i,j}^{WGSP}\pi_{i,j} = V_{-i}^{-f_i(0)} - V_{-i}^{-j} = p_{i,j}^{VCG}\pi_{i,j} \ .$$

A detailed proof can be found in the online version of the paper. □

AGSP. The following theorem says that AGSP prices are easy to compute in the same manner as the GSP prices. They will generally be higher than VCG prices, which is good in that it differentiates

them from VCG (unlike WGSP) but bad in that an advertiser can reduce its price simply by reducing its bid on other outcomes.

THEOREM 4.3. $p_i^{AGSP} = \max\{0, \max_j \frac{b_{i,j}\pi_{i,j} - d_{i,j}}{\pi_{ii}}\}$

PROOF. We want to find the smallest $b_{ii} \geq 0$ such that the augmentation graph for σ^* does not have a negative weight cycle. Analogously to Lemma 3.7 we need to ensure that $d_{i,j} \geq -b_{i,i}\pi_{ii} + b_{i,j}\pi_{i,j}$. Therefore,

$$b_{i,i} \geq \frac{b_{i,j}\pi_{i,j} - d_{i,j}}{\pi_{ii}}.$$

□

5 CONCLUSION

This paper studies ad auctions when parameters may be slot-specific. The algorithms herein are simultaneously eminently practical and theoretically intriguing. From a practical perspective, our algorithms are all fast enough to be implemented under the severe run time constraints of the online advertising domain, while offering significantly more optimization power than the status quo separable model. From a theoretical perspective, our work takes a novel approach and explicitly computes bidders' allocation functions. While this may initially seem counterintuitive and excessive, it introduces a layer of abstraction between allocation and pricing that confers substantial power to the auctioneer and permits implementations that would otherwise be effectively impossible [4]. This new approach inspires a variety of questions for future work — e.g., what are other settings where allocation functions can be computed efficiently and/or practically, and what are the benefits? Is computing allocation functions provably harder than computing truthful prices in interesting settings, or provably equivalent? Given that the allocation curve is ubiquitous in proofs, the fact that the existing literature never computes it directly suggests that our approach is ripe for further study.

REFERENCES

[1] Gagan Aggarwal, S. Muthukrishnan, Dávid Pál, and Martin Pál. 2009. General Auction Mechanism for Search Advertising. In *Proceedings of the 18th International Conference on World Wide Web (WWW '09)*. ACM, New York, NY, USA, 241–250. DOI: http://dx.doi.org/10.1145/1526709.1526742

[2] Sushil Bikhchandani and Joseph M. Ostroy. 2006. *Combinatorial Auctions*. MIT Press, Chapter From the assignment model to combinatorial auctions.

[3] Liad Blumrosen, Jason D. Hartline, and Shuzhen Nong. 2008. Position Auctions and Non-uniform Conversion Rates. In *ACM EC Workshop on Advertisement Auctions*.

[4] Ruggiero Cavallo, Prabhakar Krishnamurthy, Maxim Sviridenko, and Christopher A. Wilkens. 2017. Sponsored Search Auctions with Rich Ads. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 43–51. DOI: http://dx.doi.org/10.1145/3038912.3052703

[5] Ruggiero Cavallo and Christopher A. Wilkens. 2014. *Web and Internet Economics: 10th International Conference, WINE 2014, Beijing, China, December 14-17, 2014. Proceedings*. Springer International Publishing, Cham, Chapter GSP with General Independent Click-through-Rates, 400–416. DOI: http://dx.doi.org/10.1007/978-3-319-13129-0_32

[6] Ran Duan and Seth Pettie. 2014. Linear-Time Approximation for Maximum Weight Matching. *J. ACM* 61, 1, Article 1 (Jan. 2014), 23 pages. DOI: http://dx.doi.org/10.1145/2529989

[7] Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. 2007. Internet Advertising and the Generalized Second-Price Auction: Selling Billions of Dollars Worth of Keywords. *American Economic Review* 97, 1 (2007), 242–259.

[8] Matt Goldman and Justin M. Rao. 2017. Position Auctions in Practice. (2017). DOI: http://dx.doi.org/10.2139/ssrn.2524688

[9] Renato Gomes, Nicole Immorlica, and Evangelos Markakis. 2009. Externalities in Keyword Auctions: An Empirical and Theoretical Assessment. In *Internet and Network Economics*, Stefano Leonardi (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 172–183.

[10] Przemyslaw Jeziorski and Sridhar Moorthy. 2017. Advertiser Prominence Effects in Search Advertising. *Management Science* 0, 0 (2017), null. DOI: http://dx.doi.org/10.1287/mnsc.2016.2677 arXiv:https://doi.org/10.1287/mnsc.2016.2677

[11] Przemyslaw Jeziorski and Ilya Segal. 2015. What Makes Them Click: Empirical Analysis of Consumer Demand for Search Advertising. *American Economic Journal: Microeconomics* 7, 3 (August 2015), 24–53. DOI: http://dx.doi.org/10.1257/mic.20100119

[12] Sébastien Lahaie and David M. Pennock. 2007. Revenue Analysis of a Family of Ranking Rules for Keyword Auctions. In *Proceedings of the 8th ACM Conference on Electronic Commerce (EC '07)*. ACM, New York, NY, USA, 50–56. DOI: http://dx.doi.org/10.1145/1250910.1250918

[13] Herman B. Leonard. 1983. Elicitation of honest preferences for the assignment of individuals to positions. *The Journal of Political Economy* 91, 3 (1983), 461–479.

[14] Kazuo Murota. 1996. Valuated Matroid Intersection I: Optimality Criteria. *SIAM J. Discrete Math.* 9 (1996), 545–561.

[15] Kazuo Murota. 1996. Valuated Matroid Intersection II: Algorithms. *SIAM J. Discret. Math.* 9, 4 (Nov. 1996), 562–576. DOI: http://dx.doi.org/10.1137/S0895480195280009

[16] S. Muthukrishnan. 2009. Bidding on Configurations in Internet Ad Auctions. In *Computing and Combinatorics*, Hung Q. Ngo (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 1–6.

[17] Roger B. Myerson. 1981. Optimal Auction Design. *Math. Oper. Res.* 6, 1 (Feb. 1981), 58–73. DOI: http://dx.doi.org/10.1287/moor.6.1.58

[18] Renato Paes Leme. 2017. Gross substitutability: An algorithmic survey. *Games and Economic Behavior* 106, C (2017), 294–316. https://EconPapers.repec.org/RePEc:eee:gamebe:v:106:y:2017:i:c:p:294-316

[19] Lyle Ramshaw and Robert E. Tarjan. 2012. On Minimum-Cost Assignments in Unbalanced Bipartite Graphs. (2012).

[20] Uriel G. Rothblum. 1992. Two-sided matching: A study in game-theoretic modeling and analysis: By Alvin E. Roth and Marilda A. Oliveira Sotomayor, Econometric Society Monographs, Cambridge Univ. Press, Cambridge, MA, 1990. 265 + xiii pp. *Games and Economic Behavior* 4, 1 (1992), 161–165. https://EconPapers.repec.org/RePEc:eee:gamebe:v:4:y:1992:i:1:p:161-165

[21] Hal R. Varian. 2007. Position auctions. *International Journal of Industrial Organization* 25 (2007), 1163–1178.

[22] Hal R. Varian and Christopher Harris. 2014. The VCG Auction in Theory and Practice. *American Economic Review* 104, 5 (2014), 442–45. DOI: http://dx.doi.org/10.1257/aer.104.5.442

[23] Christopher A. Wilkens, Ruggiero Cavallo, and Rad Niazadeh. 2016. Mechanism Design for Value Maximizers. *CoRR* abs/1607.04362 (2016). http://arxiv.org/abs/1607.04362

[24] Christopher A. Wilkens, Ruggiero Cavallo, and Rad Niazadeh. 2017. GSP — The Cinderella of Mechanism Design. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*.