Google's Auction for TV Ads
Preliminary version

Noam Nisan Jason Bayer Deepak Chandra Tal Franji Robert Gardner
Yossi Matias Neil Rhodes Misha Seltzer Danny Tom Hal Varian
Dan Zigmond

Google Inc.

Abstract

This document describes the auction system used by Google for allocation and pricing of TV ads. It is based on a simultaneous ascending auction, and has been in use since September 2008.

1 Introduction

While Google is known for its advertising on the web, not many people know that it also allows advertisers to buy TV ads, and do so in a convenient way online. At the time of writing, Google offers TV advertising inventory on over 100 channels in the USA, some national and some local. While this paper will not elaborate on the advantages that this offering can provide to advertisers, broadcasters, cable operators, or Google itself, we will shortly mention the following advantages for advertisers:

- **Automation:** all aspects of of creating, buying, and running the campaign are done via a simple online web interface.
- **Flexibility:** in contrast to the old-fashioned habits of the TV advertising industry where complex deals are manually negotiated long in advance, this system provides a simple, transparent, just-in-time, granular auction model. In particular this allows convenient aggregation of inventory over many small networks.
- **Measurement:** Excellent online measurements and analysis of the campaign are provided. Notably, the TV ads are delivered via set-top boxes that track exact and actual numbers of viewers for each ad.

This document concentrates on the auction mechanism that is used for allocation and pricing of TV ads. We only sketch a high-level description of the whole system. The reader who wishes to learn more about the rest of the system may consult Google's web-sites for TV ads [2], or the adwords "traditional media" blog [1]. A variant of the auction mechanism was also used for allocation of Radio ads from October 2008 to May 2009, at which point Google cancelled its Radio operation.

1.1 How it works — overview

Google has deals in place with various "publishers" of TV content: broadcasters and Cable companies. From the point of view of advertising, these publishers own an inventory of ad-slots: time-slots on various stations, where each of these is (part of) a commercial break within some scheduled program. A typical slot may be between 30 second to 120 seconds long, and there may be many dozens of such slots available daily on each station. Publishers make (part of) their inventory of slots on their various stations available for sale through Google. All day-to-day interaction with publishers is automated: slightly over-simplifying, each day the publisher's IT systems send the next day's inventory (set of slots to be sold by Google) to Google; Google then auctions it among interested advertisers; finally, Google sends back the resulting schedule and ads to the publisher's systems that then insert the scheduled ads at the scheduled commercial breaks. The auction also sets the prices for the advertisers and Google handles their billing.

This paper does not discuss the financial arrangements between Google and the different publishers, which are manually negotiated, but rather focuses on the advertiser-facing side which is goverened by an auction.

1.2 The Advertiser-facing user interface

An advertiser that wishes to set a TV advertising campaign can do so using the respective section of Google's Adwords site [2]. There are basically three steps involved. First, the ad itself — the

"creative" — a video clip, must be produced in standard format. This is the responsibility of the advertiser, but Google offers an online "ad creation marketplace" which helps connecting an advertiser with specialists that can produce a TV ad for him. Once the ad exists, the advertiser simply uploads it to the site.

The second step is targeting where your ad may appear. This is done using a web-page interface that allows targeting by various criteria: stations, days in the week, day parts, demographics, geographic regions, scheduled programs, etc. Much sophistication went into making this interface convenient and powerful, but logically, the output of this stage is simply the set of slots that the advertiser is interested in. Figure 1 gives a screen shot of (part of) the user interface for this.

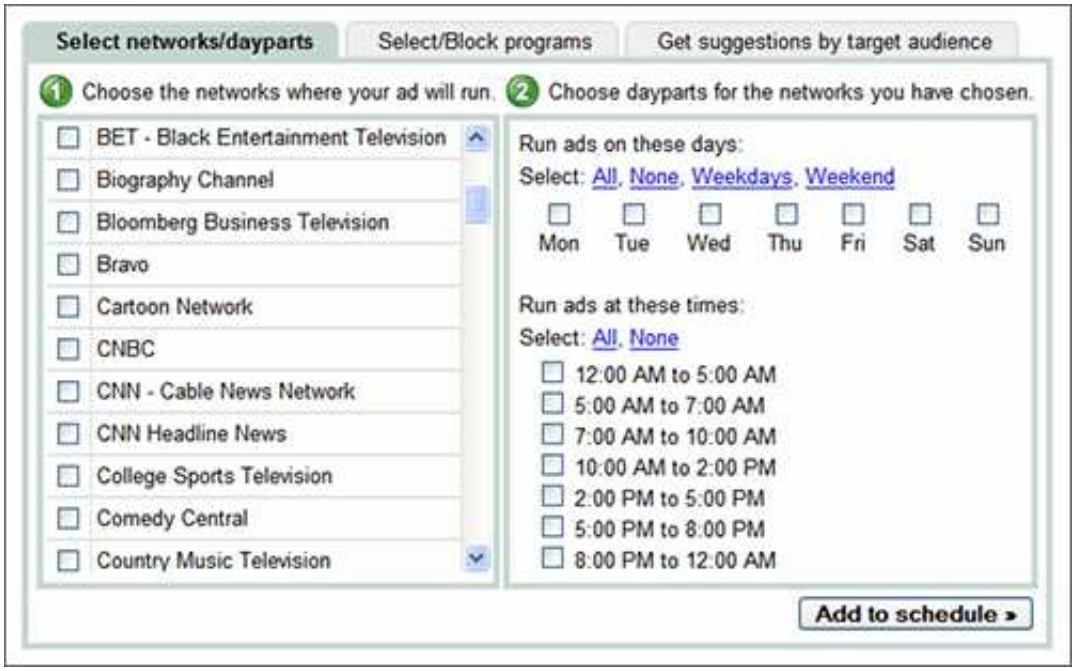

Figure 1: screen shot from TV ads targeting interface

The third step is setting the bid: how much the advertiser is willing to pay. Logically, there are two main conceptual parts to this bid: a total budget and a per-ad bid. First, a daily budget is specified, and then a maximum price that the advertiser is willing to pay for his ad to appear in every targeted slot is specified. The latter is commonly given in terms of "cpm" — cost per Millie — the price for each 1000 viewers. Thus for example a "$5 cpm" ad that is watched by 3,000 viewers will cost $15. Again, a web-interface lets the advertiser specify these "max-cpm" bids. While additional constraints and preferences may be specified, logically, the heart of the specified information here is a real value for each targeted slot. Figure 2 gives a screen shot of (part of) the TV-ads user interface for this.

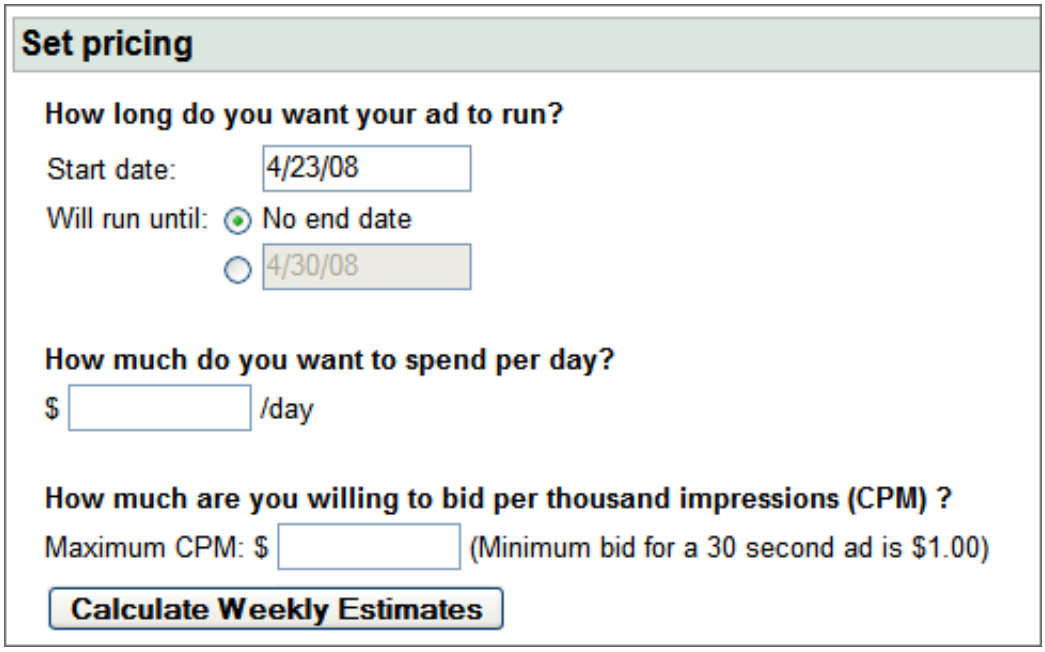

Figure 2: screen shot from TV ads bidding interface

1.3 The Auction goals

Once all the inventory and bids are given, the goal of the auction is to decide on the allocation, i.e. the schedule of ads, as well as the correct pricing. The desired input and output of this auction is clear:

Input:

- The inventory that is available for the next day. This is essentially a set of slots, where each slot is specified by a time-range on a station, as well as its basic parameters.
- The set of all campaigns that are interested in this inventory. The basic information given for each campaign is the daily budget and the cpm for each slot, but there may also be additional constraints and preferences.

Output:

- The schedule: which advertiser gets to air his creative in which slot.
- Pricing: How much is every advertiser charged (in cpm) for each slot that he received.

As opposed to traditional methods in the industry where prices are set by manual negotiations long in advance, the system here allows buying inventory on a daily basis and thus the prices themselves must be determined automatically as manual negotiation is impractical and inconvenient. These prices must be flexible, changing on a day-to-day basis as to reflect the changing market

conditions, and thus must be determined by some auction-like (or market-like) mechanism. This flexibility of prices is needed as to ensure basic economic efficiency in the face of changing market conditions. This automatic setting of prices is a major difference from existing systems [6] that automatically handle scheduling of ads, but are given manually defined prices.

The exact criteria according to which the allocation and pricing should be done is slightly subtle. It is clear that we would want to allocate slots to advertisers in a way that maximizes their value from the ads as implied by their bids, and that we can never charge the advertisers more than what is implied by their budget or bid. It is also clear that we would like to maximize the revenue, which is then split between the publishers and Google (according to the negotiated business terms whose details do not concern us here). What is less clear at first sight is the exact desired trade off between the conflicting goals of the different advertisers, between these and the revenue goal, as well as how all this is implemented in a way that encourages advertisers to bid truthfully and not "shade" their bids. This last consideration strongly suggests that we should not charge advertisers directly according to their bid, but rather in the spirit of the "second price auction".

In section 2.2, we study some of the difficulties in even attempting to pose this as an optimization goal (before even worrying about the practicality of the optimization). Our conclusion is to attempt reaching a "market allocation" with minimum equilibrium prices. In such an equilibrium, each ad slot is priced at the minimum price needed for the winner to "take it away" from the competition, and each advertiser is allocated the most cost-effective set of ads under these prices according to his bids. Such an outcome would, in partuclar, be "Pareto-optimal" as well as fair. Taking minimum market equilibrium prices implies that bidders have little strategic incentive to reduce their bids, as in any case they pay the minimum price needed to win their allocation. This approach focuses on setting up an economic environment—a market—that we anticipate will grow in the future. Making sure that the environment provides an efficient and fair outcome is critical for future growth.

1.4 The Auction logic

The mechanism that was chosen to implement the auction is based on the simultaneous ascending auction with item prices. The theoretical foundations of this ascending approach go back to [8] with a more general point of view taken in [10], and has been famously used in the FCC spectrum auctions [12]. For a background on combinatorial auctions in general and the simultaneous ascending auction in particular we refer the reader to [7, 12]. The basic logic of this auction is as follows: each ad-slot has an associated price that keeps increasing throughout the auction. Prices start at low reserve prices, and rise whenever there is "over-demand" for an ad-slot — i.e. a slot that is currently held by one bidder is desired by another. Such small price increases keep going on until there is no "over-demand", at which point the auction closes. The basic step in the auction is the calculation of the "demand" of a bidder at current prices — i.e. which set of slots would this bidder desire to acquire assuming that slots are priced as given. In its basic form the calculation of this demand is done by a greedy algorithm that chooses slots according to decreasing bid-to-price ratio.

Under certain theoretical assumptions ("gross substitutes") it is known that this procedure ends with a "Walrasian market equilibrium" [10] and under even stricter assumptions these prices are incentive compatible — i.e. give no bidder any strategic reason to under-bid [8]. Of course, these theoretical assumptions do not hold in reality, and thus we can not expect these desired properties to perfectly hold in reality. In fact, we show that it is impossible to reach any of these two conditions even under very restricted cases of our basic setting. However, we do find that this general approach

does work "well" in practice, with various heuristic solutions to various complications[1].

1.5 The auction implementation

The auction system described here has been in operation continuously since September 2008. The complete TV-ads system (which has been in operation longer) is quite sophisticated in terms of architecture involving multiple components that interact with publisher systems, billing systems, databases, monitoring, as well as a web-based front-end. The auction component itself is shielded from this complexity and is quite simple in architecture. The auction is implemented as a single-threaded single-processor program that accepts its input, in one "chunk", in the Google-standard open format of a protocol buffer [3], and produces an output in a similar format. The most notable feature of the internal architecture is the central place of a "Bidder" interface that represents a single advertiser to the auction. The auction itself proceeds by repeatedly asking Bidder objects for their "demand". This internal architecture directly corresponds to the theoretical point of view [4], separates the concerns of each advertiser from those of the auction as a whole, and is very adaptable to new advertiser bidding product features. (The reader may observe the rapid rate of change (product improvements) on the adwords traditional media blog [1].)

The system is considered a success: short-term simulations do show a significant improvement in the quality of allocation (revenue, advertiser value, and other measures) and feedback from the advertiser and publisher directions has been quite positive.

1.6 Rest of the paper

The rest of this paper describes this auction in detail. We believe that it will be more illuminating to start, in section 2, with a simplified description that captures the basic issues, and only then, in section 3, discuss various "complications" that real life brings. This paper does not present any new theoretical results, it does however attempt to provide a theoretical context to the many issues that were faced and dealt with by the auction. As usual, any real implementation faces multiple complications, many of which are handled in an ad-hoc way but really require new theoretical analysis. We attempt pointing out some of the issues that we believe deserve such theoretical treatment. The first author has in fact collaborated in theoretical analysis on one such topic [9].

2 The Basic Problem

This section discusses the basic version of the auction.

2.1 The Formulation

Let us start by introducing basic notation that captures the essence of the issue:

- We will have m abstract slots, numbered $1...m$. For each slot j we are given a reserve price $r_j \geq 0$, as well as basic slot data (station, time, impressions).

[1] We wish we could quantify this last claim by bringing experimental results, but are not able to do so here. Some of this quantification has been done but is confidential data, some of it has not been completely measured as it requires non-trivial effort, and some of it is even not clear how to measure.

- We will have n bidders. Each bidder is specified by his budget $b_i \geq 0$, and his maximum bid for each slot, specified by a "valuation function" $v_i()$, where $v_i(j) \geq 0$ denotes the bid for slot j. We denote $T_i = \{j | v_i(j) > 0\}$ the set of slots that i targets. From the point of view of the auction, we take v_i as given, with entries that have been already calculated according to the advertiser's bid as a function of the slot data.

- The allocation produced is a partition of the slots into disjoint subsets $S_0, S_1 ... S_n \subseteq \{1...m\}$, where each bidder i wins the set of slots S_i, and S_0 are the unallocated slots.

- The pricing produced is a real price p_j for each slot j that satisfy the following properties: (a) at least the reserve price: $p_j \geq r_j$ (b) individual rationality: if $j \in S_i$ then $p_j \leq v_i(j)$ (c) budget constraints: for each bidder i, $\sum_{j \in S_i} p_j \leq b_i$[2].

We first need to model the utilities of the bidders, i.e. what do they desire. While in section 3 we elaborate on additional constraints and preferences that they may express, in the basic formulation described here, we only get from each bidder a bid for each slot. Our basic assumption on the value of a set of items is "additive valuations" *up to the budget limit*. I.e., that the (declared) value that bidder i gets from acquiring a bundle of slots $S \subseteq \{1...m\}$ is simply $v_i(S) = \sum_{j \in S} v_i(j)$. Our assumption is that this is a monetary measure and thus if the bidder pays a total of q for the bundle S then his utility — what we should aim to optimize for him — is $v_i(S) - q$, but this holds only as long as we are within budget $q \leq b_i$. This budget limit takes us out of the usual quasi-linear setting, and is analyzed theoretically in [9].

2.2 What are the goals?

Let us start by informally stating the goals that we would like to get, at first not worrying about exact definitions, whether they are feasible, or how to handle the conflicts between them.

- Efficiency: We should try to maximize the values obtained by the bidders, i.e. the vector $v_1(S_1)...v_n(S_n)$. There will clearly be some trade off, which we should specify, between the values obtained by different bidders.

- Revenue: certainly the auctioneer should aim to maximize the revenue, $sum_{j \notin S_0} p_j$.

- Fairness: We should not discriminate between bidders. The exact meaning of this requires some thought, but lack of fairness is usually quite clear.

- Incentive Compatibility: We should remember that the bid information is given to us by advertisers. These will react strategically to the auction system used, and optimize their bids as to get highest utility from the system. We should ensure that there are no strategic reasons for advertisers to "under-bid" or otherwise strategically declare a bid that is different than their true value.

[2]Less specifically, as in mechanism design theory, we could only ask for the total payments from each bidder i without breakdown by "item price", e.g., as given by the VCG payment rule. This relaxation does not seem to really help, and our subsequent discussion regarding formalizing the auction goals applies also to this less specific "bundle price" setting.

While the reader may certainly see the need for trade offs between these goals, there is even conceptual difficulty in attempting to handle them separately. We encourage the reader to pause for a while and try to formulate for himself what his optimization trade-off approach will be. To our understanding there is no clear mathematical programming formulation of the optimization goal that makes much sense here. The difficulty is inherent in the combination of budget constraints with valuations, as simple attempts to optimize "social welfare" do not take budgets into proper account and simple attempts to maximize revenue do not give reasonable weight to the valuations. This is especially apparent when looking at scenarios where the budget constraints are the significant ones, which is the typical case. In appendix A we use a simple example to discuss why several natural attempts at formalizing the problem do not really make much sense. We also discuss there the problems with auctioning each slot separately, an approach which would be quite appealing due to its simplicity.

We believe that the approach that makes sense is the classic economic goal of reaching a "Walrasian" market equilibrium:

- Unallocated slots remain at reserve price: $j \in S_0$ implies $p_j = r_j$.
- Each advertiser gets his "demand" at the equilibrium prices, i.e. wins the best package for him. Formally, any set S of slots where $\sum_{j \in S} p_j \leq b_i$, we have that $\sum_{j \in S}(v_i(j) - p_j) \leq \sum_{j \in S_i}(v_i(j) - p_j)$.
- Where there is a range of equilibrium prices, we choose the lowest possible equilibrium prices.

Achieving this goal would seem to be natural and desirable in its own right. Let us say a few words on how it addresses our previous informal list of goals.

1. Efficiency: By the first welfare theorem, any equilibrium allocation will be Pareto-optimal. In particular, every bidder gets the bundles of slots that is optimal for him, under given prices, according to his bid.
2. Revenue: This auction does not always maximize revenue among all auctions. However, at a high level, budgets are exhausted for all bidders that bid "high enough", while fairness and incentive constraints limit what can be taken from "low bidders".
3. Fairness: The auction is obviously anonymous, has the "no envy" property, and all prices are justified by the property that at a lower price the slot would be over-demanded.
4. Incentive Compatibility: in general we know that markets are incentive compatible as long as no single participant has non-negligible effect on market prices. Without this assumption, there only are theoretical results showing incentive compatibility of minimum equilibrium prices in some simple cases: unit demand [8] and multi-unit auctions when valuations constraints are significantly weaker than budget constraints [9]. One can not hope for perfect theoretical incentive compatibility as [9] also show that *no* Pareto-optimal auction can be incentive compatible in the presence of budget limits[3].

[3]In particular due to the non-quasi-linear setting, VCG prices are not incentive compatible, even if they could be computed efficiently.

Unfortunately, it is theoretically impossible to always reach such an equilibrium, even in this restricted setting, for two main theoretical reasons: The first reason is the computational difficulty: even when slot prices are given, computing the demand of a bidder is equivalent to a knapsack problem. Computing the equilibrium allocation can only be harder, as computing the demand is a special case. This is addressed by taking account of the fact that usually slot prices are relatively small compared to budgets. In this case, we are close to a fractional setting, where the demand of a bidder is efficiently computed by a greedy algorithm. Not only is this greedy algorithm a good approximation, but when more realistic issues are taken into account, in section 3.1, it may be argued that it represents the demand better than the theoretical optimum.

The second reason is that it is well known that a Walrasian equilibrium may not exist unless all demands are "gross substitutes", which they need not be in our case. Thus even ignoring computational issues an equilibrium may not exist at all. Appendix B gives an example. This is handled in our auction by first relaxing the condition that all non-reserve-priced slots must be allocated, and then allocating the relatively few unsold spots in a sub-optimal "remnant inventory sale" round.

2.3 The simultaneous ascending auction

We describe here the basic version of the auction, still dealing only with the basic scenario formalized above. This basic version is the framework for the complete solution, and in the next section we will describe the various changes and enhancements to the basic algorithm. The overall idea is simple:

Initialization:

1. For all slots j, set price to reserve: $p_j \leftarrow r_j$.
2. Start with an empty allocation: For all bidders $i > 0$, $S_i \leftarrow \emptyset$, and $S_0 \leftarrow$ the set of all slots.
3. For all advertisers i, enqueue i into the bidder queue.

Main loop:

While bidder queue not empty do:

1. Dequeue the next bidder i from the bidder que.
2. Compute the demand D of bidder i greedily as follows:
   (a) Sort all slots in T_i with $\tilde{p}_j \leq v_i(j)$ according to decreasing value of $v_i(j)/\tilde{p}_j$, where $\tilde{p}_j = p_j$ for $j \in S_i \cup S_0$ and $\tilde{p}_j = p_j + \delta$ otherwise.
   (b) For all j according to the sorted order, if taking this slot does not exceed budget, $\tilde{p}_j + \sum_{t \in D} \tilde{p}_t \leq b_i$, then acquire it, $D \leftarrow D \cup \{j\}$.
3. For all slots $j \in D \cap S_k$ for some $i \neq k > 0$ do:
   (a) Increase price of j: $p_j \leftarrow p_j + \delta$.
   (b) Remove j from S_k.

(c) if k is not already in the bidder queue then enqueue k.

4. Update the set of unallocated spots: $S_0 \leftarrow S_0 \cup (S_i - D)$.

5. Update S_i: $S_i \leftarrow D$.

Output:

Each bidder $i > 0$ is allocated all slots j in S_i, at a price of p_j for slot j. Slots in S_0 are left un-allocated at this point.

The key step in this auction is the calculation of the demand D. It is important that this step only depends on bidder i's information as well as the values $\tilde{p}_j$ and not on any other global information. The goal of the step is to find the set of slots that maximize i's utility $\sum_{j \in D}(v_i(j) - \tilde{p}_j)$ subject to the budget constraint that $\sum_{j \in D} \tilde{p}_j \leq b_i$. The greedy algorithm gives the optimal solution to the fractional variant of the problem where a slot may be partially taken at any fraction α for price $\alpha\tilde{p}_j$ and giving value $\alpha v_i(j)$, and thus is practically a good approximation to the optimal set[4]. Notice also the modularity and flexibility of demand computation allowing easy extensions to take into account various other bidder preferences as described in section 3.

This ascending auction algorithm follows the theoretical work starting with [8, 10], and described, e.g., in [5]. It is easy to see that it ends with a Walrasian equilibrium (up to the additive δ) if no slots remain un-allocated. This is known to be the case with "gross substitutes" bidders in which case it ends with the minimal equilibrium prices[5]. In general, however, and especially given the complications discussed in section 3, some slots may remain unallocated at this stage and are allocated in a next "remnant inventory" stage.

2.4 Remnant inventory

The auction main loop ends with some slots unsold, those in S_0. The remnant sale sells these slots. In this stage all bidders participate as before, but their demands take into account the slots that they have already won. The logic of this stage is heuristic, basically attempting to sell as much as possible at prices that are as close as possible to the attempted "equilibrium prices" from the previous round. This stage proceeds by reducing (slightly) all prices p_j of the remaining slots, and re-running the main loop, selling some more slots. This is repeated until all remaining unsold slots are priced at their reserve price which means that they can not be sold at all and are left un-allocated.

This round is certainly a heuristic; a theoretical foundation for handling relatively small deviations from equilibrium would be of considerable interest.

[4] A dynamic programming algorithm could give a provable tighter approximation, but would require more running time, be much less flexible to addition of further constraints, and even more importantly would likely give a worse model of the bidder's true utility as we discuss in section 3.1.

[5] The minimum is well defined as equilibrium prices turn out to be a lattice [10].

3 Some Complications

3.1 The Imprecise Nature of Budgets

Budgets play a critical role in the problem formulation above, as they do in reality: an advertiser's budget is usually the *main* constraint on his allocated set of spots. In reality, however, budgets are not totally well defined as in our formulation for a host of reasons, including the following significant ones:

1. Google charges TV advertisers according to the actual number of people that watched the ad (as reported by their cable boxes) and these real payments need to be constrained by the budget but are not known at allocation time, when only an estimate is available. Thus the algorithm works with estimated payments that may later turn out to be smaller or larger than the real payments.[6] The implication is that the budget should not be treated like a clear-cut constraint but rather as a "band" in which the higher you get the higher the probability that you are over-budget.

2. The hard budget constraints are usually specified by the advertisers for longer periods of time (month, week, or campaign-length), which encompass multiple auctions. While it is expected that the single-auction daily budget is approximately the appropriate proportion, this is not a hard constraint. Indeed the current Google adwords rules allow exceeding a single day's budget by up to 20%, and only treat the longer-term budgets as "hard".

The implication from this is double: first, since the budget should really be treated as a "smooth" constraint, there is some room for optimizations as well as policy decisions in regards to the exact stopping rule in calculating the demand using the greedy algorithm. A simple example of an optimization is for handling the integrality constraint at the "last spot" — the one that just goes over budget. This may be allowed as long as it does not go over-budget beyond some threshold. An example for a policy decision is to be conservative in optimizations and try to stick closely to proportional daily budgets as to simplify advertiser control of their campaign. The smoothness of the budget constraint further justifies the greedy algorithm for computing demand rather than trying some kind of knapsack algorithm, since the whole justification of the latter is dealing with the sharp integrality constraint at the budget limit, without which the greedy algorithm is optimal (i.e. for the fractional knapsack problem.)

We suggest that some more detailed modeling of budget constraints may be of considerable interest in various settings.

3.2 Crowd control

The basic formulation of the problem did not place any structural constraints on the set of slots allocated to a single bidder. In reality there are some constraints of this form, where the most significant ones forbid too much "crowding" of slots on the same station. Such constraints come in different flavors: they may forbid a single ad to to appear twice in the same commercial break or within some predefined time gap, they may place the restriction on a single ad, on all ads by the same advertiser, or even on ads by different advertisers in the same industry. (See section 3.4

[6] In the algorithm above, p_j is the total price of the ad given the estimated number of impressions for it. The actual bids and payments are on CPM basis, i.e. after the ad is aired will be scaled by (actual number of impressions)/(estimated impressions).

below for more sophisticated variants of such constraints.) These kind of restrictions can represent the requirements of publishers or of advertisers, and must be respected by the auction. These constraints are easily incorporated into the auction by modifying the greedy *demand* algorithm to take them into account: in each greedy step we check whether taking the slot would violate "crowding" constraints, and skip the slot if this is the case. While the greedy algorithm is no longer theoretically optimal for calculating the demand even in the fractional case under these constraints, we did not observe a significant sub-optimality in practice. Appendix C shortly provides a theoretical analysis of such constraints that seems of general interest.

A constraint between different advertisers in the same industry is conceptually more problematic: it breaks the basic model of a combinatorial auction since it introduces externalities: whether I can take a slot or not depends also on someone else's allocation. While in principle it is possible to "internalize" these externalities into the model by introducing "crowding tokens" which are also put in auction, this would have significant overhead. A simpler, although not "perfectly correct" solution is to simply incorporate these "industry" crowding constraints into the greedy demand logic, slightly breaking the theoretical contract that the demand is a function of solely the current prices. One significant addition to the basic auction algorithm which is needed here is a mechanism that ensures that an advertiser is re-scheduled for calculating his demand whenever an external constraint on him changes. A theoretical analysis and quantification of the effect of "mild" externalities in combinatorial auctions and of various ways of dealing with them seems to be of interest.

3.3 The Nature of Bidders: Accounts, Campaigns, and Creatives

All of our discussion assumes the atomic notion of the "bidders": the entities among which we allocate the slots and which are the strategic participants in the auction. The reality is more complicated: there is an hierarchy of entities among which we allocate. In the case of Google adwords the hierarchy contains three levels.

1. The *Account:* Represents a single advertiser (company).
2. The *Campaign:* Represents an advertising campaign with its own budget and goals. An account may run multiple campaigns.
3. The *Creative:* Represents a single ad. A campaign may run multiple ads.

Now, which of these entities should a bidder be? From one point of view, the allocation is ultimately between ads, so the creative level seems right. From a different point of view the advertiser is really the strategic player in the auction, so the account level seems right. However, it seems that the campaign level is really the preferred answer. Conceptually, a campaign has a goal that it is trying to achieve, and the significant budget constraint usually is the campaign budget. Indeed, bidding is set on a campaign level. Some modification need to be made to the auction in order to handle the other levels of the hierarchy. First, we must sub-allocate each campaign's allocated slots among its creatives. This allocation is not price-based but rather by non-economic criteria usually, more or less, by rotation. Second, we must take care of special relations between campaigns in the same account, in particular they may share an "account budget" — a limit on the combined expenditure of all campaigns in the same account — which in terms of our campaign-based modeling is an externality. Also, as a matter of policy, it might be required that campaigns

from the same account do not compete with each other, driving prices up without real competition from another advertiser.

We are not aware of theoretical work that attempts to directly model this "fuzziness" in the nature of the agents themselves, but this certainly seems like an interesting research direction.

3.4 Long vs. short ads

All of our discussion so far assumed that all ads are of the same length, and thus all advertisers that target the same slot simply compete against each other. In reality, ads come in several standardized lengths: current TV industry standards use an integer multiple of 15 seconds, and so we need to enhance our setting to allow ads whose length is a small integer number of slots (practically between a single slot and eight consecutive slots). The main difficulty arises when different length ads compete for the same slots: should we prefer a \$5 bid for 2-slots or a \$3 bid for 1-slot? In a totally "liquid" situation, which behaves just like the fractional setting, there would also be another \$3 1-slot bid for these 2 slots and thus taking the two \$3 bids for a total of \$6 is certainly best. In practice this will not always be the case.

At the algorithmic level, the problem is not very difficult due to the consecutive linear nature of the multi-slot bids and can be solved polynomial time using dynamic programming [15][7]. However, the pricing problem here is significant since a bid for two consecutive slots has strong built-in complementarity: a single slot is worthless. The difficulties with such a situation are well known and appear at full strength even with a small example of selling 2 consecutive slots: Assume that Alice has a 2-slot ad at a value of v_A, while Bob and Charlie each have a 1-slot ad at values v_B and v_C, respectively. It terms of maximizing efficiency Alice should win whenever $v_A > v_B + v_C$, and should "logically" pay $v_B + v_C$. But how much should Bob and Charlie pay when $v_B + v_C > v_A$? It is well recognized [16, 12] that in such a case the incentive compatible VCG payments are quite problematic: they would have Bob pay $max(0, v_A - v_C)$ and Charlie pay $max(0, v_A - v_B)$. This is awkward for several reasons, e.g., the payments may well be zero and may certainly be such that the combined payment is much less than v_A. A natural approach would be to let Bob and Charlie share paying a sum of v_A in some manner, e.g. proportionally to their bid. But this is strongly non-incentive compatible as it encourages free-riding: reducing my bid will reduce my payment.

Our approach has been to stick with the price-per-slot auction in the main ascending auction stage and then do a correction in the second remnant sale stage. This fits directly into the basic ascending slot price architecture. In places with sufficient liquidity, it is optimal in terms of allocation and seems "correct" in terms of pricing. Specifically, during the main ascending stage we maintain a price for each slot and bids that require several continuous slots simply see the sum of the slot prices when they compute their demand. When a "short" ad takes a slot from a longer ad, the remaining slots are left un-allocated. This may lead to slots remaining unallocated at the end of the auction at which point they are sold in the "remanent sale". In fact, the competition between different length ads is usually the main source of un-allocated slots at the end of the first ascending stage[8].

In the remnant sale stage we change the pricing rule from being per-slot to taking the whole ad into account. Since we are already in a situation where it is clear that there are liquidity problems, a short ad can replace a longer one only by paying for the complete price of the replaced ad —

[7] In fact, this is true even in conjunction with the crowding constraints described above.

[8] As the complementarity between adjacent slots indeed is "farthest away" from the theoretical "gross substitutes" condition.

even those slots that are not desired by the short one. Theoretically, the allocation achieved is no longer optimal and we also deviate from incentive compatibility, but since typically only a small fraction of slots are left unsold at the remnant stage, this is practically acceptable.

While much analysis of the basic "one two-item bid vs. two one-item bid" has appeared in the literature, we leave a comprehensive theoretic strategic treatment of this scenario when there are many slots and bidders (and so we are "somewhat close" to a fractional setting) as a research problem.

3.5 Auction overlap

Our basic model considers a single fixed auction: all the input is available, then an algorithm that determines the allocation and pricing is run, and then results are reported to the systems that actually run the ads at the allocated times and that bill the advertisers appropriately. Unfortunately, reality has a significant "online" component: the "input" i.e. the inventory for sale arrives from the different publishers at different times. Some may know their inventory for Tuesday a week before, some may only have it late Monday night, and in some cases some preliminary information is available early and may then be updated later. Similarly, some publishers are willing to get their schedule for Tuesday late Monday night while others need it a few days in advance.

The way that this staggered schedule of availability of the input and the output is handled is as follows. Every time some output (i.e. allocation information for some set of slots) is needed, a new auction is run. At that point in time the auction is run on all inventory whose allocation affects the required set of slots. For inventory that is unavailable at that time, the preliminary available information — a prediction, if needed — is used. Only the allocation of slots that needs to be specified at this time is actually used, and the allocation of other slots is discarded, to be finalized in future auctions which may have better information.

As an example (which is similar in spirit though not details to the case in our auction) suppose that publisher A needs to get his schedule 2 days in advance and publisher B requires only 1 day in advance, but then may also report his inventory to Google only 1 day in advance. We then hold two auctions on every day x: one for day $x+1$ and the other day $x+2$. The auction for day $x+2$ will use the best estimates so far for B's inventory and the final inventory of A. The allocation it produces for B will be simply thrown away, and only the allocation for A will be committed. This will exhaust some of the budgets of advertisers for day $x+2$. The auction for day $x+1$ (run on day x) will take into account the budgets that were already spent for day $x+1$ (by the auction run on day $x-1$) and will use the final inventory of B to get the allocation for B.

The quality of results obtained by this staggered online algorithm depends of course on the quality of preliminary information. It is not needed really that the preliminary information or prediction get the exact inventory correctly but rather that prices implied by the predicted inventory are close enough to those implied by the actual final inventory.

While there has been some work on online auctions (see survey in [14]), we did not find any work that is directly applicable to our setting. We leave a disciplined theoretical analysis of this type of "staggered" online problem as an open problem.

3.6 Lack of free disposal

The model of combinatorial auctions implicitly assumes free disposal: an item which is not sold can simply be thrown away. In reality this need not always be the case. For example, a commercial

break *must* be filled — empty airtime is not tolerated. The solution to this is very simple in principle: just have a bunch of "filler" ads ready that can be used to fill any empty slot. Of course preparing, managing, and approving these ads may be non-trivial in practice, but from this paper's auction-centric point of view the problem reduces to filling empty slots with appropriately chosen filler ads. In our auction, Google has "public service" announcements used for this purpose, while providing this public service to the community.

Acknowledgments

Of the many Googlers who contributed to this system, we would especially like to thank the following: Jag Duggal, Daniel Gertsacov, Kaustuv, Ryan Roemer, Steve Stuckenberg, Tal Sela, and Kevin Thompson.

References

[1] Adwords traditional media blog. Web Page: http://google-tmads.blogspot.com/search/label/Google%20TV%20Ads.

[2] Adwords tv ads web site. Web Page: http://www.google.com/adwords/tvads.

[3] Protocol buffers web site. Web Page: http://code.google.com/apis/protocolbuffers.

[4] Liad Blumrosen and Noam Nisan. On the computational power of iterative auctions. In *ACM EC*, 2005.

[5] Liad Blumrosen and Noam Nisan. Combinatorial auctions (a survey). In Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay Vazirani, editors, *Algorithmic Game Theory*. Cambridge University Press, 2007.

[6] Srinivas Bollapragada, Hong Cheng, Mary Phillips, Marc Garbira, Tim Gibbs, and Mark Humphreville. Nbc's optimization systems increase revenues and productivity. In *Interfaces*, pages 47 – 60, January-February 2002.

[7] P. Cramton, Y. Shoham, and R. Steinberg (Editors). *Combinatorial Auctions*. MIT Press, 2006.

[8] G. Demange, D. Gale, and M. Sotomayor. Multi-item auctions. *Journal of Political Economy*, 94:863–872, 1986.

[9] Shahar Dobzinski, Ron Lavi, and Noam Nisan. Multi-unit auctions with budget constraints. In FOCS'08.

[10] Faruk Gul and Ennio Stacchetti. Walrasian equilibrium with gross substitutes. *Journal of Economic Theory*, 87:95 – 124, 1999.

[11] Benny Lehamnn, Daniel Lehmann, and Noam Nisan. Combinatorial auctions with decreasing marginal utilities. *Games and Economic Behavior*, 55(2):270–296, 2006.

[12] Paul Milgrom. *Putting auction theory to work*. Cambridge University Press, 2004.

[13] Noam Nisan. *In P. Cramton and Y. Shoham and R. Steinberg (Editors), Combinatorial Auctions. Bidding Languages.* MIT Press, 2006.

[14] David C. Parkes. Online mechanisms. In Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay Vazirani, editors, *Algorithmic Game Theory*, chapter 16. Cambridge University Press, 2007.

[15] Michael H. Rothkhof, Aleksandar Pekeč, and Ronald M. Harstad. Computationally manageable combinatorial auctions. *Management Science*, 44(8):1131–1147, 1998.

[16] Michael H Rothkopf, Thomas J Teisberg, and Edward P Kahn. Why are vickrey auctions rare? *Journal of Political Economy*, 98(1):94–109, 1990.

Appendix A: Difficulties in Problem Formulation

Let us look at a few natural attempts to formulate the desired goals of the auction and see why they do not make much sense. These will lead us to the market equilibrium formulation that we took. For concreteness, let us consider the following simple example:

Running Example: 100 slots are being auctioned among two advertisers: Alice has a \$60 budget and a \$3 value for each slot and Bob has a \$30 budget and a \$6 value per slot.

3.7 Independent auctions?

The first approach that one may consider is to auction the spots one by one, each time to the highest bidder whose budget has not been exhausted yet. This seems to make much sense as slot values are independent of each other. This approach also has the very strong appeal of simplicity both in terms of implementation and in terms of of explaining it to the advertisers. One may think of various approaches to decide on pricing, but using the second price in each of the slot auctions seems appropriate for incentive compatibility. If we follow what this approach means in our case we see that Bob, the high bidder, will win all the first auctions until his budget is exhausted. Using a second-price rule, he will be charged \$3 per slot, so his budget will run out after winning 10 slots. At this point, Alice will win the remaining 90 spots for free (or whatever low reserve price there is). The extreme un-fairness would be even more apparent if Alice only had a \$10 budget — she would still win these 90 spots for free. Strategic manipulation can be extremely helpful here: had Bob declared a \$2 per-slot bid instead, Alice would win the first slots for \$2 each, exhausting her budget after 30 slots, and then Bob could reap the remaining 70 spots for free!

3.8 Maximize Revenue?

Suppose that are goal is maximizing revenue. This is achieved by charging the two advertisers their full budgets. This will also satisfy individual rationality as long as the allocation gives at least 20 slots to Alice and at least 5 to Bob. Thus revenue maximization provides very little guidance on which allocation to choose. So which other criteria should be use? Fairness and efficiency would seem to suggest allocating a slot to the one that has a higher value for it. Should Bob get all but 20 of the slots? This seems quite unfair. Why should he pay much less per slot? It also

is clearly not incentive compatible since advertisers are strongly motivated to strategically reduce their declaration of the budget. Another difficulty of this approach is to what extent is taking the whole budget is justified: suppose that Bob only puts in a low \$0.1 value for each spot (rather than \$6). Can we still charge Alice her full \$60 budget, even though the "second price" would only be \$10 (10 cents for each of 100 slots)? Should we give some spots to Bob in this case and charge him, hence increasing our revenue further?

3.9 Maximize Efficiency?

Suppose, on the other hand, that our goal is maximizing efficiency. Let us further decide that our resolution of the trade off between different advertisers is the "utilitarian" one of maximizing the sum of values, $\sum_i \sum_{j \in S_i} v_i(j)$. How are the budgets taken into account? As previously, should Bob get all slots? How much should he pay?

Some previous papers have considered the budget limit as an upper bound on the value, $v_i(S) = min(b_i, \sum_{j \in S} v_i(j)$, and thus attempted to maximize $\sum_i (min(b_i, \sum_{j \in S_i} v_i(j))$. If this is done, then again any allocation that gives at least 20 slots to Alice and at least 10 to Bob would achieve this maximization. Again we get very little guidance on which allocation to choose. How much should they pay? VCG payments in this context make no sense since they would be 0 (and in general do not ensure incentive compatibility in our non-quasi-linear setting).

3.10 Market Equilibrium

At this point, we hope that the reader is coming to the realization, like we have, that the correct goal here is a market equilibrium.

Let us see what this equilibrium will look like here. At any price $p \leq 30/34 = 0.882...$ Alice will demand at least 68 slots and Bob at least 34 so there will be over-demand. Just above this price, Bob's demand would be 33 and Alice's 67, so the the lowest equilibrium price would be just above 88.2 cents. Bob would pay \$29.1.. for his 33 slots and Alice would pay \$59.1.. her 67 slots. This is quite efficient, nearly maximizes revenue, and seems to be quite fair. This case turns out to also be incentive compatible: no advertiser can gain by manipulating his bid.

Appendix B: Example with no equilibrium prices

Here is an example for a simple setting where no Walrasian equilibrium exists.

bidder	budget	v(a)	v(b)	v(c)
1	6	5	5	0
2	9	4	4	8
3	7	0	0	7

Assume towards contradiction that an equilibrium exists, and assume without loss of generality that $p_a \leq p_b$. If $p_a + p_b > p_c$ then bidder 2 demands c and $p_c \geq 7$ (otherwise 3 also demands it). But then bidder 1 demands only a, and b is left un-allocated contradicting the requirement that unallocated slots be priced at reserve (0 here). On the other hand, if $p_a + p_b < p_c$ then $p_c \leq 7$ (otherwise neither 2 nor 3 demand c and it is left un-allocated despite its non-zero price). But then both 1 and 2 demand a. Contradiction. The case $p_a + p_b = q_c$ is treated like the first case if c is allocated to 2, and like the second case otherwise, concluding the contradiction.

Appendix C: Additive Valuations with pair-wise constraints

In this appendix we focus, in a general combinatorial auction setting, on constraints that forbid a bidder to win certain given pairs of items. This appendix does not address the interplay with budget constraints (which we have not analyzed and leave as a topic for further study) but rather reverts to the standard quasi-linear setting. A linear valuation with such constraints is given by two elements:

1. A value $v(j)$ for each item $j \in M$, where M is the set of items for sale.
2. A graph $G = (M, E)$, where E are the set of pairs that are forbidden to be taken together.

The valuation of a set is defined as $v(S) = \max_{I \subseteq S} \sum_{j \in I} v(j)$, where I ranges over all sets that are *independent* in G. This is essentially the "OR*" bidding language used on singleton valuations (see [13]). We believe that this is quite a useful "bidding language" in general and should be studied. Here are a few preliminary results, whose proofs are straight forward given the literature and omitted.

- Every linear valuation with pair-wise constraints lies in the class XOS of [11] and hence is sub-additive. There are linear valuations with pair-wise constraints that are not sub-modular and hence not gross-substitutes.
- Answering a value query or a demand query (see [4]) given the description of the valuation as above is NP-hard. The same is true for approximating the value to within $m^{0.5-\epsilon}$ or for finding the welfare-maximizing allocation between such valuations.
- When the graph is restricted to be an interval graph (as it is in our "crowding" constraints) both value and demand queries can be answered exactly in polynomial time (using dynamic programming).