Submitted to *Conference on Information Systems and Technology*
The bulk of the work was done by a student (Andor Goetzendorff).

Core-pricing in large multi-object auctions: a market design for selling TV-ads

Andor Goetzendorff, Martin Bichler, Bob Day*, Pasha Shabalin
Department of Informatics, TU München, Munich, Germany, bichler@in.tum.de
*Operations and Information Management, University of Connecticut, CT, USA, bob.day@business.uconn.edu

We introduce an auction design and computational methods to determine core payments for large markets with many objects such as the market for TV ads. Bidder optimal, core-selecting auctions have been introduced in spectrum auctions worldwide as a means to set incentives for truthful bidding, but at the same time avoid the problems of the Vickrey-Clarke-Groves mechanism. Many real-world markets involve large numbers of objects where such advanced designs cannot be applied any more for computational reasons, but also for the number of possible bundles available to bidders. An enumerative XOR bidding language (widely discussed in the literature and used in recent government spectrum auctions) grows too quickly to be practical. Market designs for large markets with many objects and similar incentive properties have received little attention in the literature. We propose a compact bidding language for coverage or demographic reach of TV ad slots, and investigate the resulting winner-determination problem and the computation of core payments. For realistic instances of the winner determination problem, very good solutions can be found relatively quickly, though closing the integrality gap to find marginally better solutions or prove optimality can take a prohibitively large amount of time. Our subsequent adaptation of a constraint-generation technique for the computation of bidder-optimal core paymants to this environment is a new, practically viable paradigm by which core-selecting auction designs can be applied to large markets with hundreds of items.

Key words: TV ads; core-selecting auction; market design

1. Introduction

The development of the Internet allowed for the exchange of complex preference profiles and laid the foundation for the design of new market mechanisms. The promise of these mechanisms is that by allowing market participants to reveal more comprehensive information

about cost structures or utility functions, they can increase allocative efficiency and lead to higher economic welfare. In recent years, a growing body of literature in the Management Sciences is devoted to the design of such smart markets (Gallien and Wein 2005), with combinatorial auctions (CAs) emerging as a pivotal example (Cramton et al. 2006). A CA allows bidders to bid on combinations of objects, offering protection against the well-known "exposure problem" present in simultaneous auctions for heterogeneous goods, in which a bidder is exposed to winning too few complementary goods to realize synergies, or too many objects which she considers substitutes at a high cost.

In this paper, we approach market design from the specific perspective of auctions for television advertising slots, based on our interactions with European broadcasters and media agencies. These markets involve the sale of air-time in hundreds of different time-slots, weekly or biweekly, where bidding advertisers have differing preferences over slots, based on varying audience demographics and firm-specific needs for sufficient ad reach or coverage. The size of this problem, in terms of the large number of distinct interrelated goods and heterogeneity of bidder preferences, prohibits the application of existing fully enumerative approaches, such as those used in spectrum auctions around the world, in which often only less than a dozen distinct products are sold and a bidder can simply list (an adequately large number of) her bundles of interest. The solution we suggest to this problem is a new compact bidding language, potentially useful in a large variety of market design problems beyond TV ads, including procurement applications and spectrum auctions with a larger number of spectrum products to be sold. In addition, our adaptation of a constraint-generation technique for the computation of bidder-optimal core payments to this environment is a new, practically viable paradigm by which core-selecting auction designs with good incentive properties can be applied to large markets.

1.1. The Ad-Slot Market

The TV ad-slot market will serve as a real-world example in our paper, which shares many of the features with other large markets for the sale of spectrum licenses, in logistics, or in industrial procurement. In what follows, we provide a brief overview of the essential requirements. Parts of the advertisement capacity of a typical TV station are sold via specially-negotiated, large, long-term contracts of about a year, and are not considered in our study. We focus instead on the sale of the remaining ad-slot inventory to specific marketing campaigns that run in the short-term, which in Europe are typically sold via posted prices, one to two weeks in advance of airing. Prices are set by the TV station based on historical demand. Buyers are large media agencies, who purchase a set of slots with the intent to procure the best slots for each of their customers' campaigns. The number of agencies in a particular market depends on the country and the particular station, but a typical short-term market for a TV station in Germany, for example, consists of approximately 50 media agencies, booking slots for several hundred customers in a particular channel.

Because the amount of air-time filled by long-term customers varies, the length of a slot available in the short-term market is not fixed, while the length of an ad also varies considerably, lasting up to 1 minute. For a particular channel in the markets we investigated, there are on the order of 150 short-term slots available during the program per week.

Different slots have a different reach for different customer segments or the population overall. The reach of a particular slot varies over time, but there are estimates based on historical panel data available to clients of the media agencies. Clients use reach per segment (based on gender, age, or other demographics) or per population to determine their willingness-to-pay for different slots. Clearly, the value of some slots, such as those during the finals of the national soccer league, may be difficult to estimate and their valuation varies

considerably depending on the target market of an advertiser. Apart from these high-value slots, there is also typically a segment of low-value slots, which are also difficult to price as the demand is hard to predict. This difficulty in demand or valuation prediction suggests that an auction market would outperform the existing posted-price mechanism.

1.2. Market Design Challenges

An auction mechanism consists of an allocation and a payment rule (Nisan 2007). The allocation rule is largely determined by the bid language used. The compact bidding language we develop allows for the expression of bids on a large number of combinations, while at the same time limiting the number of parameters that a bidder has to specify. Our approach is based on the observation that (in practice) advertisers perceive a large fixed value for an ad campaign as long as its projected coverage of a target audience (demographic) satisfies an adequate threshold. To complete the bidding language, the specification of macro-level parameters (the value of the campaign at one or a few threshold levels for reach) must be accompanied by a vector of micro-level estimates of reach for each time-slot, which firms can compute based on marketing data.

Given this particular bidding language, we investigate the practical solvability of the resulting $\mathcal{NP}$-hard winner-determination problem, a form of multi-knapsack problem. Our simulations use bid-parameter estimates based on empirical observation and guidance from actual media agencies. This specific winner-determination problem results in high-quality solutions rather quickly, but with difficulty closing the integrality gap in a reasonable amount of time. Thus, although very good outcomes can almost always be be found, we usually cannot guarantee actual optimality, a situation given little if any attention in the CA literature.

Core-selecting auctions have been introduced in recent years (Day and Raghavan 2007) as a payment paradigm for multi-object markets in general, designed to balance the incentives

of bidders to reveal bids truthfully (achieved by making the bidders pay the least amount possible) against the perceived fairness of payments (such that payments are adequately large to preclude any set of losing bids from become winning). This auction design computes prices that are "in the core" with respect to submitted bids, stating roughly that no coalition of bidders could claim that their bids offered a mutually preferable outcome that would also raise seller revenue. The game-theoretical properties of bidder-optimal core-selecting auctions have been discussed intensively in the recent years (Day and Milgrom 2007, Goeree and Lien 2013), and core-selecting auction rules have been adopted for spectrum license auctions around the world. The approach seems particularly well-suited to the TV ad market, where auctioneers want to avoid very low revenue outcomes possible with a Vickrey-Clarke-Groves (VCG) mechanism. Also, even if core-selecting auctions do not exhibit a dominant strategy if the VCG outcome is not in the core, bidders typically lack the necessary prior information required to speculate in such markets.

Unfortunately, previous publications on the topic provide no sound method for computing "approximately" core-selecting outcomes when winner-determination is only nearly optimal, which is regularly the case in large markets. Here, we provide the first such demonstration of an approximately core-selecting auction, and compare a simple heuristic trimming of infeasibilities approach to a dynamic update algorithm that stores and re-uses allocations systematically. This new paradigm is tested in the TV ad environment but the approaches are presented in a general form which can be extended to any environment of nearly-optimal winner-determination.

2. Market Design

The allocation of TV ad slots is a multi-knapsack problem, in which each time-slot i in the set $I = \{1, 2, \ldots, N\}$ is treated as a knapsack with a maximum capacity/duration of c_i,

which cannot be exceeded. As mentioned above, each slot can potentially hold a number of ads, though some may have been previously allocated to larger customers, so we assume that c_i reflects only short-term capacity in the current market, making for a potentially heterogeneous list of c_i values, even for a TV station with slots of the same size when considering all ads aired. We also assume that each slot i has a reservation value or minimum price per unit time r_i, which reflects the station's ability to off-load excess capacity at a low price to existing customers if needed. Station call signs and other brief announcements can also be used to fill any excess unused time.

Each bidding advertiser k in the set $K = \{1, 2, \ldots, \mathcal{K}\}$ has an ad of duration d_k to be shown repeatedly (at most once per time slot) if she wins a bundle of time-slots in the auction. To ensure adequate reach, each bidder specifies an abstract "priority vector" or "weight vector" W_k, containing an arbitrary weight value w_{ik} for each time slot. These "priority weights" could represent the expected viewership, expected viewership of a particular demographic, or viewership weighted by expected sales, etc., reflecting the advertiser's performance metric of choice. She can then bound the total priority value in the auction outcome to be greater than or equal to a minimum amount in order to qualify bids of various levels.

Thus after specifying the priority vector and ad duration, a bidder places one or more tuples $(w_j^{\min}, b_j)$ containing the desired sum of priority values $w_j^{\min}$ she wants to obtain for a monetary bid b_j. At most one of the bids placed by a bidder can win, i.e., the bidding language is an XOR of "weight threshold levels." For example, if the bidder sets the priority weights at the expected viewership of each slot, the XOR structure lets her set an exact price for any particular price-points of interest, e.g., a price for $w_j^{\min} = 1$ million viewers, a price for $w_j^{\min} = 2$ million viewers, etc.

The set J_k contains all bid indexes j by a bidder k, and the superset J is defined as $J := \bigcup_{k \in K} J_k$. We assume these bids are submitted in a sealed-bid format.

2.1. Allocation Rule

Formulation WD maximizes the value of accepted bids given that: ad durations do not exceed capacity in any slot (1a); the bid values are not less than the seller's reservation values (1b); the priority threshold level $w_j^{\min}$ of a bid j is met if and only if that bid is accepted (1c,d); at most one bid j is accepted for each bidder k (1e). Decision variables x_{ij} and y_j indicate time if slot i is assigned to bid j and bid j itself is accepted, respectively, while M is a sufficiently large positive constant parameter. This formulation is annotated $WD(K)$ indicating all bidders $k \in K$ are included. Later we will also refer to $WD(C)$ for $C \subset K$, which indicates the same formulation but with all bids made $k \notin C$ set to zero.

$$WD(K) = \max \sum_{j \in J} b_j y_j \tag{WD}$$

$$\text{subject to} \sum_{j \in J} d_k x_{ij} \leq c_i \qquad \forall i \in I, \tag{1a}$$

$$d_k \sum_{i \in I} r_i x_{ij} \leq b_j \qquad \forall j \in J, \tag{1b}$$

$$\sum_{i \in I} w_{ik} x_{ij} \leq M y_j \qquad \forall j \in J, \tag{1c}$$

$$w_j^{\min} - \sum_{i \in I} w_{ik} x_{ij} \leq M(1 - y_j) \qquad \forall j \in J, \tag{1d}$$

$$\sum_{j \in J_k} y_j \leq 1 \qquad \forall k \in K, \tag{1e}$$

$$x_{ij} \in [0,1] \qquad \forall i \in I, j \in J, \tag{1f}$$

$$y_j \in [0,1] \qquad \forall j \in J. \tag{1g}$$

The priority vector W_k provides quite a bit of flexibility to the bidders in expressing their preferences over ad slots. However, for some segments of the ad slot market even a simpler version of the bid language might be sufficient, where a bidder only specifies the number of slots he wants to win from a particular subset of the slots. For example, a bidder might want

his ad to be on the air at least five times within one week between 8 and 10pm. In other words, for this bidder all ad slots between 8 and 10pm are substitutes.

2.2. Payment Rule

We will determine BPOC payments in the following treatment, and compare them to the VCG payments in our experiments. The approach of using constraint separation to find the coalitions defining the core was designed to work in any context where the winner-determination problem could be solved exactly. Here, we quickly reiterate that approach before extending it to situations of nearly-optimal winner determination in the next section. We use the term "nearly-optimal" in a practical sense, indicating that we have set a relative integrality gap parameter to an adequate threshold (e.g., 95%) using a commercial solver and have found a solution at least this close to a theoretical best possible solution.

The approach discussed in the literature is to find core prices by iteratively creating new price vectors p^t and then checking at each iteration t, whether there is an alternative outcome which generates strictly more revenue for the seller and for which every bidder in this new outcome weakly prefers to the current outcome. If such a coalition exists, the alternative winning coalition C is called a *blocking coalition*, and a constraint is added to a partial representation of the core in payment space until no further blocking coalitions can be found. In order to discover the most violated blocking coalition C^t relative to the current payments at iteration t the WD is extended as in SEP^t.

$$z(p^t) = \max \sum_{j \in J} b_j y_j - \sum_{k \in W} (b_k^* - p_k^t)\gamma_k \qquad \text{(SEP}^t\text{)}$$

subject to

$$(1a) - (1g)$$

$$\sum_{j \in J_k} y_j \leq \gamma_k \qquad \forall k \in W,$$

$$\gamma_k \in [0, 1] \qquad \forall k \in W.$$

In it, W is the set of winners from the solution of WD(K), and b_k^* represents bidder k's winning bid. If the sum of the current payments p^t is less than the solution to (SEP^t) then a violated core constraint has been found, and we must add a constraint to our partial representation of the core. Following Day and Raghavan (2007) this partial representation is given in the following linear program, which is then solved to find the next tentative set of payments p^{t+1} until no further constraints can be found.

$$\theta(\epsilon) = \qquad \text{(EBPO}^t\text{)}$$

$$\min \sum_{k \in W} p_k + \epsilon m$$

$$\text{subject to} \sum_{k \in W \setminus C^\tau} p_k \geq z(p^\tau) - \sum_{k \in W \cap C^\tau} p_k^t \qquad \forall \tau \leq t, \qquad \text{(EBPO}^t\text{.1)}$$

$$p_k - m \leq p_k^{vcg} \qquad \forall k \in W,$$

$$p_k \leq b_k^* \qquad \forall k \in W,$$

$$p_k \geq p_k^{vcg} \qquad \forall j \in W.$$

As in SEP^t, b_k^* is the winning bid for k; the parameters $p_k^{vcg} = b_k^* - (\text{WD}(K) - \text{WD}(K_{-k}))$ represent VCG payments, and m represents a maximum deviation from VCG, which is minimized as a secondary objective after minimizing total payments[1]. We then use the value of each p_k in the solution for the next iterate (i.e., set $p_k^{t+1} = p_k$).

3. Core Payments with Nearly-Optimal Allocations

For many combinatorial optimization problems good solutions often can be found quickly, even though finding a provably optimal solution may take a very long time. Figure 1 shows a typical example of WD with 336 slots and 50 bidders, where a feasible solution with 95%

[1] In practice these two minimizations can be handled as separate optimizations but they are presented here as a single optimization using a sufficiently small ϵ for the sake of concise exposition.

optimality could be reached within a few minutes. This is a common phenomenon in many combinatorial optimization problems.

Without the ability to guarantee an optimal solution quickly enough for a practical application, one would naturally consider a provably high-quality solution that can be found quickly. Most industrial MIP solvers provide absolute and relative worst-case optimality gap parameters, allowing the optimization routine to terminate if the optimality gap (difference between the best feasible solution and the theoretical bound) falls below some target or is a small enough percentage of the best feasible solution, respectively. For now, we leave the exact specification of how a "good enough" approximate solution is qualified, but motivated by Figure 1, the reader may assume a 5% optimality gap or an at-least-95%-optimal solution for concreteness. We will thus write WD^a for any qualified approximation of a WD value and consider an implementation using these approximations in place of true WD values. Similarly, we will write $z^a(p^t)$ for separation problem values found using nearly-optimal solutions.

Problems can arise, however, during the VCG and core price calculation when accepting these approximate or nearly-optimal solutions. For example, under truly optimal solutions, with the standard assumption of free disposal, $\text{WD}(K_{-k})$ is always at most the value of $\text{WD}(K)$. But with a series of nearly-optimal computations this is not guaranteed, opening the possibility that one might compute an approximate VCG payment with $b_k^* - (\text{WD}^a(K) - \text{WD}^a(K_{-k})) > b_k^*$. Similarly, under nearly-optimal computation the coalitional value of SEP^t can be higher than the value of the WD. If this happens, the newly generated constraint added to EBPO^t can cause an infeasibility if $\sum_{k \in W \setminus C^t} b_k^* < z^a(p^t) - \sum_{k \in W \cap C^t} p_k^t$. Two different solutions are presented to address this problem that can potentially arise during computations.

3.1. Core Price Calculation – TRIM

With known b_k^* values determining individual rationality (IR) constraints (i.e., payments must not exceed bids), a natural first approach is to adjust each WD-based result so that it fits into the IR region.

For the VCG prices this technique makes use of the fact that:

$$b_k^* \geq p_k^{vcg} \geq 0 \qquad \forall k \in W \tag{2}$$

whereas for the (constant) RHS of the constraints in the EBPO^t, we must always have:

$$\sum_{k \in W \setminus C^\tau} b_k^* \geq z(p^\tau) - \sum_{k \in W \cap C^\tau} p_k^\tau \qquad \forall \tau \leq t \tag{3}$$

Thus our first algorithm[2] is to simply trim the infeasibilities based on known bids, represented in Algorithm 1 in the two steps using min functions.

3.2. Core Price Calculation – REUSE

An alternative to trimming infeasibilities is based on the observation that whenever an infeasibility is found, the validity of expressions (2) and (3) imply that an update can be made to an approximate WD value, from a previously best-known feasible solution to a new tentatively-optimal feasible solution. To implement this change, the storage of any value based on a winner-determination solution can no longer be treated as constant, and must be regenerated at each iteration based on current WD^a values. This includes VCG price estimations and the RHS values for generated core constraints, whose definitions must be reformulated based on current WD^a values.

Thus our second approach is to store a list of all discovered WD$^a(C)$ values, reusing all coalitions found so far and reformulating the entire separation problem and EBPO problem

[2] In all algorithmic implementations that follow, we assume that all feasible integer solutions are stored by the optimizer and used to generate bounds on subsequent optimizations using alternate objective functions.

at each iteration, noting that the set of relevant core constraints, and indeed the set of winners itself, may be changing as new information becomes available. Whenever we run WD^a (the first time, to compute each VCG price, and inside each run of SEP) we get a new collection of feasible bids, representing a coalition of winners, and we check these values against our current list of coalitions and WD^a values. If the coalition has not been found before, our list is extended to include it as among the "potentially important" coalitions to consider, updating any previously found superset coalition with a lower coalitional value to the current $\text{WD}^a(C)$ value.

Because we will now store the blocking coalitions C^t and its value instead of $z(p^t)$ after each SEP^t has been solved, we are forced to work with a reformulation of core constraints based on WD^a values rather than separation levels. For a general winner-determination problem (i.e., with respect to any alternative bidding language) the core constraints can be expressed with the alternative, equivalent expression which can be derived by substitution:

$$\sum_{k \in W \setminus C^\tau} p_k \geq \text{WD}(C^\tau) - \sum_{k \in W \cap C^\tau} b_k^* \qquad \forall \tau \leq t \qquad (\text{EBPO}^t\text{'}.1)$$

Using this formulation of the core, we can generate a constraint in EBPO for any C^τ found so far using the current best-approximation $\text{WD}^a(C^\tau)$ in place of $\text{WD}(C^\tau)$. For bidding languages with only one relevant bid b_k per bidder (as it is the case in the scenarios presented in section 4), this constraint can be further simplified, resulting in the following formulation (4) in place of EBPO^t'.1.

$$\sum_{k \in W \setminus C^\tau} p_k \geq \sum_{k \in C^\tau \setminus W} b_k \qquad \forall \tau \leq t \qquad (4)$$

This new set of constraints provides an intuitively pleasing interpretation of core constraints in the TV-ad context: Any subset of winners pays enough to match the counter-offer including

a set of losing bidders that would otherwise benefit the seller, a direct analogue to second-prices.

While it is not guaranteed that each stored coalition C^t provides a potential maximally-violated coalition at the end of the constraint generation process, the addition of all constraints found at any point drastically improves the overall performance of the algorithm in comparison to having to completely rebuild a set of blocking coalitions after a change in W.

The formulation of the separation problem as an altered WD problem has the additional benefit that all feasible solutions remain feasible for a WD or SEP^t instance. MIP solvers store feasible (integer) bases internally, and if the separation problem is implemented as the same problem instance with some changes to objective coefficients, all feasible bases stored as a branch-and-bound tree remain feasible and thus provide immediate bounds on the SEP^t problem, making efficient use of all information found by the solver. This made it progressively more difficult to find relevant feasible solutions. Therefore, in our experimental evaluation we allowed for longer time limits on the optimization routine for later SEP^t instances.

Algorithm 2 as presented in the appendix keeps a list $Coalitions$, each element being a list winners under some feasible integer solution to WD. For each coalition $C \in Coalitions$ we also store the best known value $val(C)$, which can be revised as the algorithm progresses. Further, for ease of exposition, these algorithms refer to the winning bidders from the most recent optimization run as $optwinners$, and the sum of the (actual, i.e., unaltered) bids of these winners is given as $bidsum$.

4. Experimental Evaluation

In this section, we analyze the solution quality of the presented algorithms under constrained computation time. Due to space constraints, we will only discuss selected experimental results in this section.

Our industry partner, a TV station, had an existing first-come-first-serve booking system, providing a realistic distribution of baseline prices for historical valuations. The analysis of the booking system and interviews with sales managers of the TV station and marketing agencies allowed us generate a series of realistic scenarios for simulations.

Based on this information, a series of parameters and distributions were derived to define the random variables of our experiments (see table 1).

The bid base price β_j can be interpreted as how much a bidder would maximally spend to obtain the right to reach one priority point with his ad for one second. The actual bid price for a campaign is then calculated as follows: $b_j = d_k \beta_j w_j^{min}$.

Using the simulation we analyzed a number of primary attributes to measure overall performance, such as allocative efficiency, revenue, and speed. In order to directly compare the quality of the generated prices, a series of secondary attributes were evaluated. These values allow a comparison on how "fair" an outcome can be perceived and provide bounds on how much a bidder could possibly gain by shading his bid[3] by comparing the ratios of the bid vector to the resulting payment vector and to the generated VCG price vector, respectively.

- Primary variables
  1. The efficiency in terms of the coalitional value
  2. The overall revenue
  3. The duration, measured by the iterations needed to terminate
  4. The number of updates or *switches* to the tentative set of winners (always $= 1$ for TRIM)
- Secondary attributes

[3] See Day and Raghavan (2007) and Goeree and Lien (2013) for more information about bid shading in core-selecting auctions.

5. The ratio of bids to payments
6. The ratio of bids to Vickrey prices

As each scenario had a different set of auctioned slots and bid price/priority tuples, the outcome of each algorithm has to be compared on a within-scenario basis, allowing us to make several comparisons of the TRIM and REUSE algorithms on the same data. For each primary attribute, we therefore computed a REUSE to TRIM ratio. For example, the ratio of *revenue* between the two approaches, with a ratio above 1 indicating greater revenue for REUSE.

In order to compare secondary attributes across different scenarios, such as the ratios p_k/b_k, p_k^{vcg}/b_k and p_k^{vcg}/p_k, a series of identical quantiles were built. Because the values used in this comparison are already relative to the respective prices, each quantile is then compared on a per-scenario basis by subtracting the TRIM value from the REUSE value, meaning that if the resulting value is > 0, the REUSE value is bigger than its TRIM counterpart, and vice versa. An example is shown in table 2, which illustrates typical behavior; REUSE runs longer and generates less revenue than TRIM, but is more efficient.

Based upon the distributional parameters of table 1, 20 scenarios were generated. These scenarios were then run with the optimizer settings as provided in table 3. As the resulting performance should be comparable to a typical personal computer, all experiments were run on Dual Socket Octo Core AMD Opteron 2.4GHz Linux computers with 8GB DDR2 RAM as part of a Linux Cluster.

Next, the results of the two algorithms are compared using the primary and secondary attributes described above (see figures 2 to 3).

As the REUSE algorithm is able to change the winning coalition if an improvement is found, it is not surprising that its median efficiency is 2% higher (see figure 2). What may

be surprising at first sight, however, is that the actual revenue generated by the REUSE algorithm is consistently lower, despite the fact that a better winning coalition yields a higher coalitional value or social welfare, resp. The median it generated only 87% of the TRIM revenue for the same scenario.

The REUSE algorithm typically switched its set of winners 3.5 times (see figure 3). Switching also initiates new automatic VCG calculations, partially explaining why the REUSE algorithm takes, at its median, 63% more iterations than the TRIM counterpart.

5. Conclusion

In many markets, auctioneers would prefer core pricing to VCG mechanisms, in order to avoid non-core outcomes where the bids of losing bidders are higher than the payments of the winners. With the introduction of core-selecting auctions for spectrum licenses in recent years, many stake-holders have developed software to determine winners and core prices based on the use of integer programming to solve a series of winner-determination problems. Extending the use of this software to larger and more complex markets (such as the TV ad context we address here, or the FCC's new incentive auctions for spectrum reconfiguration) cannot be accomplished by merely specifying time limits or optimality-gap thresholds to the solver engine, as it could for the more simple case of a single optimization problem. Doing so would often result in an infeasible pricing problem. This is a general problem in all larger markets using bidder-optimal core-selecting prices.

Here we compare two potential algorithms for dealing with these infeasibilities, finding one faster and higher revenue method (for a fixed set of bids) and one slower but more efficient method. Our results on a simulated TV ad market show that the former TRIM algorithm may be suited to a fast-clearing market in which speculation to lower bids is offset by uncertainty and healthy competition.

Acknowledgments

The authors gratefully acknowledge support of the Deutsche Forschungsgemeinschaft (DFG) (BI 1057/4-1).

References

Cramton, P., Y. Shoham, R. Steinberg. 2006. Introduction to combinatorial auctions. P. Cramton, Y. Shoham, R. Steinberg, eds., *Combinatorial Auctions*. MIT Press, Cambridge, MA.

Day, R., P. Milgrom. 2007. Core-selecting package auctions. *International Journal of Game Theory* **36** 393–407.

Day, R., S. Raghavan. 2007. Fair payments for efficient allocations in public sector combinatorial auctions. *Management Science* **53** 1389–1406.

Gallien, J., L. Wein. 2005. A smart market for industrial procurement with capacity constraints. *Management Science* **51** 76–91.

Goeree, J., Y. Lien. 2013. On the impossibility of core-selecting auctions. *Theoretical Economics* .

Nisan, n. 2007. *Introduction to Mechanism Design (for Computer Scientists)*.

List of Algorithms

Algorithm 1: Core Price Calculation – TRIM

Solve: $\mathrm{WD}^a(K)$;
for $j \in W$ **do**
  Solve: $\mathrm{WD}^a(K_{-k})$;
  $p_k^{vcg} \leftarrow \min(b_k^*,\, b_k^* - (\mathrm{WD}^a(K) - \mathrm{WD}^a(K_{-k})))$;
$p^1 \leftarrow p^{vcg}$;
$\theta^0 \leftarrow \sum_{k \in W} p_k^{vcg}$;
while *true* **do**
  Solve: SEP^t;
  if $z^a(p^t) \leq \theta^{t-1}$ **then**
    Break: 'core' price vector found;
  Generate RHS of new constraint: $\alpha^t \leftarrow \min\Big(\sum_{k \in W \setminus C^t} b_k^*,\, z^a(p^t) - \sum_{k \in W \cap C^t} p_k^t\Big)$;
  Add constraint $\sum_{k \in W \setminus C^t} p_k \geq \alpha^t$ to EBPO;
  $p^t, \theta^t \leftarrow$ Solve: EBPO;
  if $(C^t, \theta^t) = (C^{t-1}, \theta^{t-1})$ **then**
    Break: no better price vector possible;
  Iterate: $t \leftarrow t + 1$;

Function `Core Price Calculation - REUSE - UpdateCoalitions`$(optwinners, bidsum)$

for $C \in Coalitions$ **do**
  if $C \supseteq optwinners$ *and* $val(C) < bidsum$ **then**
    $val(C) \leftarrow bidsum$;
    if $C = K$ **then**
      $W \leftarrow optwinners$;
      EBPO $\leftarrow$ null;
      $reset \leftarrow true$;
for $k \in W$ **do**
  $p_k^{vcg} \leftarrow b_k^* - \mathrm{val}(K) + \mathrm{val}(K_{-k})$;

Algorithm 3: Core Price Calculation – REUSE

Initialize:
Solve: $\text{WD}^a(K)$;
$Coalitions \leftarrow \{K\}$;
$\text{val}(K) \leftarrow bidsum$;
$W \leftarrow optwinners$;
$t \leftarrow 0$;
EBPO $\leftarrow$ null;
$reset \leftarrow false$;
while *true* **do**
  if $EBPO = null$ **then**
    ComputeVCG:
    for $k \in W$ **do**
      Solve: $\text{WD}^a(K_{-k})$;
      $Coalitions \leftarrow Coalitions \cup \{K_{-k}\}$;
      $\text{val}(K_{-k}) \leftarrow \text{WD}^a(K_{-k})$;
      UpdateCoalitions$(optwinners, bidsum)$;
      if $reset = true$ **then**
        Break k loop;
    $p^t \leftarrow p^{vcg}$;
    $\theta^t \leftarrow \sum_{k \in W} p_k^t$;
  if $reset = true$ **then**
    $reset \leftarrow false$;
    Continue;
  Solve: $\text{SEP}^t(p^t)$;
  if $z(p^t) \leq \theta^t$ **then**
    Break: 'core' price vector found;
  $Coalitions \leftarrow Coalitions \cup \{optwinners\}$;
  $\text{val}(optwinners) \leftarrow bidsum$;
  UpdateCoalitions$(optwinners, bidsum)$;
  if $reset = true$ **then**
    $reset \leftarrow false$;
    Continue;
  if $EBPO = null$ **then**
    build EBPO with constraints EBPO^t'.1 for all $C \in Coalitions$ with $\text{val}(C)$ as a best approximation of $\text{WD}(C)$;
  else
    add constraint EBPO^t'.1 to EBPO with $C = optwinners$ and with $\text{val}(C)$ as a best approximation of $\text{WD}(C)$;
  $p^{t+1}, \theta^{t+1} \leftarrow$ Solve: EBPO;
  Iterate: $t \leftarrow t + 1$;

List of Figures

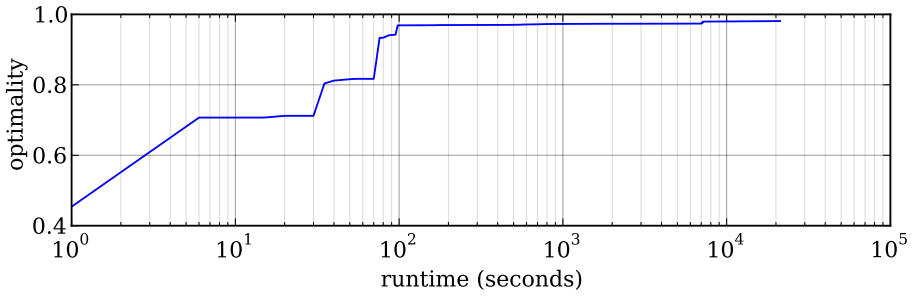

Figure 1 **Average reduction of the integrality gap over time**

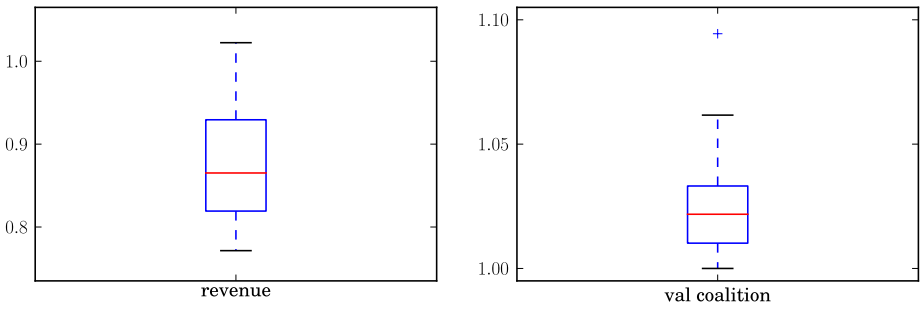

Figure 2 **REUSE to TRIM ratios – revenue, efficiency**

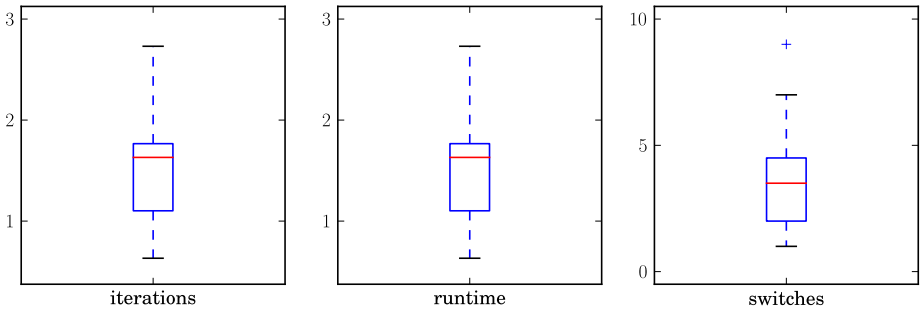

Figure 3 **REUSE to TRIM ratios – iterations, runtime, switches**

List of Tables

	Name	Parameters $\{\mu;\sigma\}$ or $\{\lambda\}$	Distribution
I	slot qty	336	-
J	bid qty	50	-
K	bidder qty	50	-
c_i	slot duration	$\{60;30\}$	Normal
r_i	slot reserve price steps (in €/s) [1, 2, 5, 10, 50, 75]	$\{1.2\}$	Poisson
d_k	advert duration	$\{20;10\}$	Normal
β_j	bid base price (in €/s)	$\{50;25\}$	Normal
$w_j^{min\_rel}$	min $\sum$ of campaign priorities (in %)	$\{30;20\}$	Normal
-	correlation of priority to slot reserve price	-	Linear
-	distribution of priorities around the priority/price value	-	Normal

Table 1 simulation input parameters

example instance	REUSE	TRIM	
			ratio
revenue	36 569 158	42 766 735	0.86
coalitional value	46 073 899	44 590 749	1.03
run time	2.8h	2.0h	1.42
switches	7	1	7.00
			difference
median(p_k/b_k)	0.79	1.00	-0.21
median(p_k^{vcg}/b_k)	0.58	1.00	-0.42
median(p_k^{vcg}/p_k)	0.78	1.00	-0.22

Table 2 scenario comparison – example

Name	REUSE	TRIM
Algorithm	reuse	trim
Runs	20	20
Time Limit – WD	300s	300s
Time Limit – Other	300s	300s

Table 3 runtime specifications