princeton univ. F'23 cos 521: Advanced Algorithm Design

Lecture 19: Combinatorial Auctions

Lecturer: *Huacheng Yu*

Lectures Notes sourced from the Chapter 11 of the AGT Book (Nisan, Roughgarden, Tardos, Vazirani), cited at end. I strongly recommend visiting the cited reference for details, but these notes are available to recall what was covered in lecture.

In this lecture, we'll take a game-theoretic look at algorithmic problems. Specifically, we'll want to consider optimization problems where the input itself has a stake in the output selected. Consider the following problem, referred to as *combinatorial auctions*. There are n bidders and m items. Each bidder has a *valuation function* for the m items $v_i(\cdot)$. That is, player i receives value $v_i(S)$ for receiving the set S of items. Your optimization problem is to find an allocation, that is a partition of $[m]$ into $S_1, \dots, S_n$ so as to maximize the *social welfare*, $\sum_i v_i(S_i)$.

This is already an interesting algorithmic problem, and we'll see more of it this lecture. But we'll also want to focus on the following: what if you don't know each player's valuation function, but you rely on them to provide you information. Then every algorithm you design imposes a *game*: depending on the behavior (information provided by the other players), and your own behavior (information you choose to provide), you get some payoff (your value for the output of the algorithm). This adds a new angle to the problem that we must consider. In general, we might need to use payments to address this problem.

Example 1 (Single-Item Auction). *Consider the case where $m = 1$. This is an easy algorithmic problem: everyone just has a value v_i for getting the item, and you need to find* $\arg\max_i\{v_i\}$ *and give them the item. However, if this is your algorithm, it induces a game among the bidders where the only equilibrium is to report the highest allowable value (reporting a higher value makes you more likely to get the item and doesn't cost anything). So even though you can easily find the maximum value amongst the provided input, the provided input is garbage and doesn't tell you anything about the actual maximum.*

One alternative is to use instead the first-price auction. *Now, you still give the item to the highest reported bid, but the winner must pay their bid. Now the equilibrium of the induced game isn't quite so bad, but still really hard to reason about (**lots** of work goes into characterizing equilibria of first-price auctions, and it's basically a complete mess - there's not enough information here on a model to see why, but check AGT book for more details). At minimum at least we can conclude that no one will ever bid above their value (instead they would rather bid exactly their value, no matter the other bids), but anything beyond this is hard to come by.*

A third alternative is to use the second-price auction. *Now, you again give the item to the highest reported bid, but the winner pays the* second-highest price. *For this auction, there's an equilibrium that's actually quite easy to reason about. Observe that if the highest bid (aside from bidder i) is p_{-i}, then bidder i will win the item and pay p_{-i} if the bid $b_i \geq p_{-i}$, and lose the item and pay 0 otherwise. So they want to win if and only if their value exceeds*

p_{-i}, and this can be achieved always by submitting a bid of v_i. So it is a Nash equilibrium for everyone to tell the truth in a second-price auction, and the maximum reported bid is selected (so the maximum actual value is also selected in equilibrium). Actually, telling the truth is a dominant strategy*, discussed below.*

So using the single-item auction as an example, we see that algorithms without payments might be really chaotic, but that algorithms with "the right" payments can induce simple equilibria and lead to good algorithms (even when we rely on the bidders themselves to tell their values). The specific notion we saw in the second-price auction is a dominant strategy.

Definition 1. *Strategy s is a* dominant strategy *for player i in a game if for all $s' \neq s$ that i could use, and all strategies $\vec{t}$ that the other bidders might use, $P_i(s, \vec{t}) \geq P_i(s', \vec{t})$, and there exists a $\vec{t}^*$ such that $P_i(s, \vec{t}^*) > P_i(s', \vec{t}^*)$. Here, $P_i(s, \vec{t})$ denotes the payoff enjoyed by player i when they use strategy s and the other players use strategies $\vec{t}$.*

It should be clear that if every bidder has a dominant strategy, then it is a Nash equilibrium for every bidder to play that strategy (in fact, this is a much stronger property: you might reasonably not expect bidders to find an arbitrary Nash equilibrium of a game, but you should reasonably expect them to play dominant strategies if they have one).

1 The Vickrey-Clarke-Groves Auction

The first result we'll discuss is seminal work of Vickrey, Clarke, and Groves (that contributed to a Nobel prize for Vickrey). It's actually a combination of three separate single-author works, and was phrased very differently than the theorem statement below (they are all economists). To state the result, we'll need to be clear about how bidders interact with payments.

Definition 2. *A bidder is* quasi-linear *if their utility for receiving value v and paying p is $v - p$. Bidders always want to maximize their utility.*

Theorem 1. *Let $\mathcal{A}$ be an algorithm that takes as input $v_1(\cdot), \dots, v_n(\cdot)$ and outputs $S_1, \dots, S_n$ maximizing $\sum_i v_i(S_i)$ over all partitions of $[m]$. Then a* dominant-strategy truthful *mechanism exists that requires only $n+1$ black-box calls to $\mathcal{A}$, and selects the partition $\mathcal{A}(v_1, \dots, v_n)$.*

Let's parse this. First, we need to define dominant-strategy truthful. This simply means that telling the truth is a dominant strategy. That is, every bidder prefers to report their true $v_i(\cdot)$ than any other valuation function, no matter what valuation functions the other bidders report. This, together with the final guarantee, that the allocation is $\mathcal{A}(v_1, \dots, v_n)$ means that the actual welfare-maximizing allocation is chosen: because the mechanism is dominant strategy truthful, the reported valuations will be the actual valuations. Because the allocation maximizes welfare on the reported valuations, this means that the actual welfare-maximizing outcome is chosen. The fact that the mechanism only requires $n + 1$ calls to $\mathcal{A}$ defines its runtime. If $\mathcal{A}$ is poly-time, then the mechanism is poly-time too.

Proof. Consider the payment rule where each bidder is charged their *externality* on the other bidders. That is, we will charge every bidder the "harm" they cause the others by existing. More specifically, we look at the total happiness of all other bidders with you in the picture versus the total happiness of all other bidders without you and charge the difference. That is, we do the following payments: below, $v = \langle v_1, \ldots, v_n \rangle$, and $v^*_{-i} = \langle v_1, \ldots, v_{i-1}, 0, v_{i+1}, \ldots, v_n \rangle$. We'll also let $\mathcal{A}_i(v)$ denote the set awarded to bidder i on input v.

$$P_i(v) = \sum_{j \neq i} v_j(\mathcal{A}_j(v^*_{-i})) - \sum_{j \neq i} v_j(\mathcal{A}_j(v)).$$

Again, this is exactly the difference between everyone else's utility without you in the picture versus with you in the picture. Let's now compute bidder i's utility for submitting any bid v'_i (we'll use v' to denote $\langle v_1, \ldots, v_{i-1}, v'_i, v_{i+1}, \ldots, v_n \rangle$).

$$\begin{aligned} U_i(v'_i, v_{-i}) &= v_i(\mathcal{A}_i(v')) - P_i(v') \\ &= v_i(\mathcal{A}_i(v')) - \sum_{j \neq i} v_j(\mathcal{A}_j(v^*_{-i})) + \sum_{j \neq i} v_j(\mathcal{A}_j(v') \\ &= \sum_{j} v_j(\mathcal{A}_j(v')) - \sum_{j \neq i} v_j(\mathcal{A}_j(v^*_{i-1})). \end{aligned}$$

Now let's look at these two terms. The first term is exactly the total welfare generated by the algorithm $\mathcal{A}$ on bids v' *but evaluated according to the real values v.* The second term is completely out of bidder i's control: it only depends on bids submitted by other bidders. So bidder i wants to maximize the first term to maximize their utility. The first term is clearly maximized when $\mathcal{A}$ selects the true welfare-maximizing allocation, which happens when bidder i submits v_i. $\square$

At first this seems great! No matter the valuations, as long as we have an algorithm maximizing welfare, we can turn it into a truthful mechanism maximizing welfare. The drawback is that for basically any interesting class of valuations, maximizing welfare is NP-hard. Also unfortunately, the VCG reduction is incompatible with approximation: if unless $\mathcal{A}$ exactly maximizes welfare on every input, the output mechanism isn't truthful.

2 Truthful Approximation Algorithms

In this section, we'll consider a (very) special case of valuations and derive a truthful approximation algorithm. Here, each bidder's value will be *single-minded.* That is, for each i, there exists a special set S_i, and $v_i(S) = V_i$ if $S \supseteq S_i$, and 0 otherwise. Note that maximizing welfare is NP-hard by a reduction from independent set, even in this super special case.

Consider any graph $G = (V, E)$. We'll make a player for each node in V, and an item for each edge in E. Player i's interest set S_i will be exactly the edges adjacent to them, and $V_i = 1$ for all i. Now it should be clear that we can simultaneously give a set P of players their interest sets if and only if they form an independent set in G. Therefore, maximizing welfare is equivalent to finding a large independent set. Note also that independent set is NP-hard to approximate within $n^{1-\varepsilon}$ for any $\varepsilon > 0$, so welfare maximization for single-minded bidders is also NP-hard to approximate within $n^{1-\varepsilon}$ (or $m^{1/2-\varepsilon}$) for any $\varepsilon > 0$.

Now, we'll show a greedy mechanism that is truthful, and guarantees a $m^{1/2}$-approximation. The mechanism is the following:

- Ask each bidder to report V_i, S_i.
- Sort the bidders so that $V_1/\sqrt{|S_1|} \geq V_2/\sqrt{|S_2|} \dots$
- Initialize $A = \emptyset$ (the set of awarded items). Starting from $i = 1$, visit bidder i and declare them a winner if and only if $S_i \cap A = \emptyset$. If so, update $A := A \cup S_i$.
- Award each winner their declared interest set S_i.
- Charge each winner the minimum V_i they could have reported and still been a winner.

For example, say that the bids are $(1, \{1\}), (1, \{1, 2, 3, 4\}), (4, \{1, 2\}), (4, \{3, 4\})$. Then the bids will be sorted so that the two 4s go first, followed by the two 1s. Both the 4s will win, the other two won't. $(4, \{1, 2\})$ will pay $\sqrt{2}$, because they will win if and only if they appear before $(1, \{1\})$ in the ordering. $(4, \{3, 4\})$ will pay 0, because they would win as long as they bid at least 0.

Now there are two things we want to prove. First, that the mechanism is actually truthful. Second, that it gets the desired approximation ratio. We'll do the approximation ratio first.

Theorem 2. *The greedy algorithm above guarantees a $\sqrt{m}$ approximation.*

Proof. Let OPT denote the true optimal allocation. For each i that wins, let $OPT_i = \{j \in OPT, j \geq i, S_i \cap S_j \neq \emptyset\}$. That is, OPT_i is the players in OPT who are blocked by i (including itself). Clearly, $OPT = \cup_i OPT_i$, as everyone not blocked by any i would have been selected by Greedy. Now we'll prove that for all winners, $\sum_{j \in OPT_i} V_j \leq \sqrt{m} \cdot V_i$.

Note that every $j \in OPT_i$ appears in the greedy order after i, so we have $V_j \leq V_i\sqrt{|S_j|}/\sqrt{|S_i|}$. Summing over all $j \in OPT_i$, we have:

$$\sum_{j \in OPT_i} V_j \leq V_i/\sqrt{|S_i|} \cdot \sum_{j \in OPT_i} \sqrt{|S_j|}.$$

Using the Cauchy-Schwarz inequality ($x \cdot y \leq |x|_2 \cdot |y|_2$, for $x = \langle 1, \dots, 1\rangle$ and $y = \langle \sqrt{|S_j|}\rangle_{j \in OPT_i}$), this is:

$$\sum_{j \in OPT_i} \sqrt{|S_j|} \leq \sqrt{|OPT_i|} \cdot \sqrt{\sum_{j \in OPT_i} |S_j|}.$$

Finally, observe that every $S_j \in OPT_i$ intersects S_i (by definition). Also, since OPT is an allocation, we must have $S_j \cap S_{j'} = \emptyset$ for all $j, j' \in OPT$. Therefore, we have $|OPT_i| \leq |S_i|$. Again since OPT is an allocation, we have $\sum_{j \in OPT_i} |S_j| \leq m$. Therefore, the RHS above is upper bounded by $\sqrt{|S_i|} \cdot \sqrt{m}$. Plugging back into the first inequality yields that:

$$\begin{aligned}\sum_{j \in OPT_i} V_j &\leq V_i/\sqrt{|S_i|} \cdot \sum_{j \in OPT_i} \sqrt{|S_j|} \\ &\leq \frac{V_i}{\sqrt{|S_i|}} \cdot \sqrt{|OPT_i|} \cdot \sqrt{\sum_{j \in OPT_i} |S_j|} \\ &\leq \frac{V_i}{\sqrt{|S_i|}} \cdot \sqrt{|S_i|} \cdot \sqrt{m} \\ &= V_i \cdot \sqrt{m},\end{aligned}$$

as desired. □

Finally, we want to claim that the mechanism is dominant strategy truthful. First, observe that, no matter what set you report, it is a dominant strategy to report your actual value for that set (either 0 or V_i). This is because based on the other bidders, the Greedy ordering induces a minimum bid you can submit (with that set) and still win it. You will win if and only if you bid above that value. If you report your true value for that set, you will always be on the correct side (just like second price).

Now, we want to argue that you should always report your true interest set. It's obvious you should never report an S that doesn't contain S_i, as this guarantees non-positive utility. If you report an S that strictly contains S_i, then this will only increase the bid you'd need to make in order to win (because you conflict with more, and because your bid gets divided by a bigger number). So you both want to bid your true value for the reported set, and report your true interest set.

To see this last claim more formally, consider all bids aside from your own. If you weren't in the picture, then there is some allocation that Greedy would select. Now imagine that you report the bid (V_i', S_i'). Then you will be allocated S_i' if and only if your bid winds up ahead of the first bidder who intersects with S_i'. Call this bidder j. Then you will be allocated if and only if $V_i'/\sqrt{|S_i'|} \geq V_j/\sqrt{|S_j|} \Leftrightarrow V_i' \geq V_j\sqrt{|S_i'|}/\sqrt{|S_j|}$. Note that if $S_i' \supseteq S_i$, the RHS can only go up. This is because first, $|S_i'| \geq |S_i|$, and second, the earliest bidder which intersects S_i' can only be earlier. So the price offered to you can only go up as you report a set strictly containing S_i, and doing so is dominated.

3 Lavi-Swamy Reduction

The reduction is based on tools we've seen throughout class, LP relaxations and rounding. The high-level approach is the following. First, we'll define a class of mechanisms that are "VCG-like," where similar reasoning to last class lets us turn algorithms of a certain form into DST mechanisms.

3.1 Maximal-In-Range and Maximal-In-Distributional-Range

Let's first recall what the VCG reduction did. We took as input any algorithm $\mathcal{A}$ that on input v selected the welfare-maximizing partition $\mathcal{A}(v)$. That is, $\mathcal{A}(v)$ maximizes $\sum_i v_i(\mathcal{A}_i(v))$

over all partitions. Then, we charged each bidder their "externality," how much the other bidders' welfare went down due to their presence: $\sum_{j \neq i} v_j(\mathcal{A}_j(v_1, \dots, v_{i-1}, 0, v_{i+1}, \dots, v_n)) - \sum_{j \neq i} v_j(\mathcal{A}_j(v))$. We then observed that bidder i's utility (value minus price) turned out to be exactly the total welfare minus a term completely out of their control, so bidder i always wants to maximize the welfare (which can be achieved by reporting their true value). Finally, we concluded that because $\mathcal{A}$ maximized welfare on all inputs, that VCG also maximizes welfare on all inputs.

Now we make a key observation: with the VCG payment rule, for any algorithm $\mathcal{A}$, the resulting mechanism is DST if and only if for all v, v' where $v_j = v'_j$ for all $j \neq i$, $\sum_j v_j(\mathcal{A}_j(v)) \geq \sum_j v_j(\mathcal{A}_j(v'))$. That is, given that we're going to run algorithm $\mathcal{A}$, and the other bidders have already reported v_{-i}, welfare is maximized when bidder i reports value v_i. This might seem like an extremely natural property that all approximation algorithms should have, but surprisingly it's not. But if we use an algorithm $\mathcal{A}$ with this property, then the VCG mechanism using $\mathcal{A}$ is still DST. Only the last sentence in the paragraph above isn't necessarily true: $\mathcal{A}$ may no longer maximize welfare on all inputs.

As an example, consider $\mathcal{A}$ that always gives all items to the same bidder. Then the VCG mechanism essentially becomes a second-price auction for the grand bundle $[m]$ of all items, and is DST. But it does a horrible job of maximizing welfare because it never considers splitting the items.

Such mechanisms are called *maximal-in-range*: there exists some set S of possible allocations, and $\mathcal{A}_S(v)$ outputs the best allocation in S (the one that maximizes welfare for v). An extension of these ideas is called *maximal-in-distributional-range* and instead lets S be a set of *distributions over allocations* (or *randomized allocations*). Again, $\mathcal{A}_S(v)$ outputs the welfare-maximizing distribution in S.

For example, maybe S contains all distributions that pick two different bidders for each item, then allocates each item independently to one of the two bidders uniformly at random. Then $\mathcal{A}_S(v)$ would find, over all such $m^{\binom{n}{2}}$ (randomized) allocations the one that maximizes expected welfare (formally: maximizes $\mathbb{E}[\sum_i v_i((\mathcal{A}_S)_i(v))]$). I'm not claiming that this is an interesting class of distributions to optimize over, but it fits the definition.

The remaining goal of this lecture will be to design a maximal-in-distributional-range mechanism that guarantees a good approximation.

3.2 The Configuration LP

Our maximal-in-distributional-range algorithm will use a specific LP relaxation (also used in many other problems) called the configuration LP. It is as follows. Below, think of $x_{i,S}$ as the fraction of set S awarded to bidder i.

$$\begin{aligned}
\text{maximize } & \sum_{i,S} x_{i,S} v_i(S) \\
\text{subject to } & \sum_i \sum_{S \ni j} x_{i,S} \leq 1 \quad \forall j \in [m] \\
& \sum_S x_{i,S} \leq 1 \quad \forall i \in [n] \\
& x_{i,S} \geq 0 \quad \forall i \in [n], S \subseteq [m]
\end{aligned}$$

First observe that this LP is indeed a relaxation: for any allocation $S_1, \ldots, S_n$, we can set $x_{i,S_i} = 1$, and all other variables equal to zero. Observe also that, unfortunately, it has exponentially many variables. That's a shame, so we'll have to be clever in order to solve the LP in poly-time. Fortunately, there aren't too many constraints, so the dual has few variables and exponentially many constraints - this is starting to sound closer to something we can solve. The dual is as follows:

$$\begin{aligned}
\text{minimize } & \sum_i u_i + \sum_j p_j \\
\text{subject to } & u_i \geq v_i(S) - \sum_{j \in S} p_j \quad \forall i, S \\
& u_i, p_j \geq 0
\end{aligned}$$

We can solve the dual as long as we can get a poly-time separation oracle. Whether or not this is possible depends exactly on how each $v_i(\cdot)$ is represented. We'll assume its represented in a way so that we can evaluate a *demand query*. That is, we can take as input a vector of price $p_1, \ldots, p_m$ and output $\arg\max_S \{v_i(S) - \sum_{j \in S} p_j\}$. Notice that this is bidder i's favorite set if the items are priced at $p_1, \ldots, p_m$. It's normally considered a reasonable kind of query, since bidder i themselves is presumably capable of picking their favorite set at some prices in poly-time (philosophical aside: if not, then maybe you should redefine v_i to match whatever bidder i would select?).

observation 1. *With demand query access to each $v_i(\cdot)$, we can implement a poly-time separation oracle.*

Proof. Simply execute a demand query for each v_i at prices $\vec{p}$. If u_i exceeds the resulting $v_i(S) - \sum_{j \in S} p_j$, then all constraints for bidder i are satisfied. If not, we've explicitly found a violated constraint. □

For the rest of this class, we'll assume demand query access to the valuations, and therefore we can get a separation oracle for the dual and solve the configuration LP in poly-time. Note that it's not trivial to take a solution to the dual and transform it into a solution for the primal, but it's not too hard (it uses complementary slackness). Notice though that because the primal only has $n+m$ constraints, there always exists an optimal solution where only $n+m$ coordinates are non-zero (this is because there always exists an optimal solution that is a corner, and all but $n+m$ tight constraints at the corner are setting some variable to zero). So now that we can solve the LP relaxation, we have to figure out what to do with it.

3.3 Rounding the Configuration LP

Definition 3. *We say that a rounding algorithm $\mathcal{A}$ verifies an integrality gap of $c \leq 1$ for the configuration LP if it takes as input v and outputs a (deterministic) allocation such that $\sum_i v_i(\mathcal{A}_i(v)) \geq c \cdot \text{ConfigOPT}(v)$, where $\text{ConfigOPT}(v)$ is the fractional optimum of the configuration LP.*

Proposition 3. *[Lavi/Swamy 2005] Let $\mathcal{A}$ verify an integrality gap of c for the configuration LP. Then for any fractional solution x to the configuration LP with k non-zero coordinates, one can decompose $c \cdot x$ into a distribution over integral allocations in poly(n, m, k) black-box calls to $\mathcal{A}$ (and poly(n, m, k) overhead).*

Proof. (**Note: this was omitted in lecture)** Strongly recommend visiting Chapter 12 of the cited textbook for this since there are some subtleties, but the main ideas are below). Consider the following LP, which tries to write $c \cdot x$ as a convex combination of (exponentially many) integral allocations (let $\mathcal{I}$ denote the set of integral allocations, which is finite):

$$\begin{aligned} \text{minimize } & \sum_{y \in \mathcal{I}} \lambda_y \\ \text{subject to } & \sum_{y \in \mathcal{I}} \lambda_y x^y_{i,S} = c \cdot x_{i,S} \quad \forall (i,S) \text{ such that } x_{i,S} > 0 \\ & \sum_{y \in \mathcal{I}} \lambda_y \geq 1 \qquad \text{(this constraint might seem silly, but it's helpful to keep the dual clean)} \\ & \lambda_y \geq 0 \end{aligned}$$

Quickly observe that we don't need to enforce the constraint $\sum_{y \in \mathcal{I}} \lambda_y x^y_{i,S} = 0$ when $x_{i,S} = 0$, we can simply ignore any integral allocations that award set S to bidder i when $x_{i,S} = 0$ (e.g. hard-code such $\lambda_y = 0$, or just remove these variables entirely). Finally, also observe that if we find a solution to this LP where $\sum_{y \in \mathcal{I}} \lambda_y = 1$, this is exactly a convex combination over integral allocations. So our goal is to find such a solution, via the dual:

$$\begin{aligned} \text{maximize } z + & \sum_{(i,S), x_{i,S} > 0} c x_{i,S} \cdot w_{i,S} \\ \text{subject to } z + & \sum_{(i,S), x_{i,S} > 0} x^y_{i,S} w_{i,S} \leq 1 \quad \forall y \in \mathcal{I} \\ & z \geq 0 \end{aligned}$$

We now want to claim that we can solve the dual in the desired runtime, given black-box access to $\mathcal{A}$, and that the value of the dual must be 1. First, observe that $z = 1, w_{i,S} = 0$ for all (i, S) is a valid dual solution and has value 1. So the dual has value at least one (this is why it was helpful to write the superfluous constraint in the primal, although we could have drawn the same conclusions without it and a little extra reasoning. Also observe that the dual has only $k + 1$ variables, so if we can get a separation oracle, we can solve it in time poly(k).

Now we want to show that the dual has no feasible solutions with value > 1. Assume for contradiction that (w, z) was such a solution. Then consider running algorithm $\mathcal{A}$ on input valuations with $v_i(S) = w_{i,S}$. Then this produces some integral allocation y with $\sum_{i,S} x^y_{i,S} w_{i,S} \geq c \sum_{i,S} x_{i,S} w_{i,S} > 1 - z$. The last inequality follows from hypothesis that we started with a feasible solution of value > 1. But now we can rearrange this into a violated constraint in the dual, and therefore the solution was in fact not feasible.

There are some subtleties this time for recovering the optimal primal from the optimal dual, but we'll again omit this. So we can again recover a solution to the primal with only k integral allocations, and this is the convex combination we desire.

$\square$

3.4 Verifying the Integrality Gap

Finally, we'll observe (without proof, it's nearly identical to the proof of Theorem 2 — check the cited book chapter for a proof) that the same greedy algorithm from Theorem 2 verifies an integrality gap of $1/\sqrt{2m}$. That is, on input v, sort all (i, S) in decreasing order of $v_i(S)/\sqrt{|S|}$, then greedily assign sets that don't conflict (but now two sets conflict if they're of the same bidder as well).

3.5 Putting everything together

So the complete picture looks like this:

1. Let $\mathcal{A}$ denote the algorithm that decomposes a point $\frac{1}{\sqrt{2m}} \cdot x$, where x is feasible for the configuration LP into a distribution over integral allocations. This is based on the greedy algorithm, plugged through Proposition 3.

2. Let S denote the set of all distributions that $\mathcal{A}$ might ever output on input x, where x is feasible for the configuration LP. Let $\mathcal{B}$ be the algorithm that on input v, solves the configuration LP, then runs $\mathcal{A}$ to get a distribution over integral allocations. Note that $\mathcal{B}$ finds the welfare maximizing distribution in S.

3. Use the VCG payments with maximal-in-distributional-range algorithm $\mathcal{B}$. This is a DST mechanism. Observe that it guarantees a $\frac{1}{\sqrt{2m}}$-approximation.

4 An $O(\sqrt{m})$-approximation for subadditive buyers

First, we'll provide an $O(\sqrt{m})$-approximation for subadditive buyers due to [?]. Recall that a valuation function is subadditive if $v(X \cup Y) \leq v(X) + v(Y)$ for all X, Y. Note that this implies that $v(X) \leq \sum_{i \in X} v(\{i\})$.

Consider the following algorithm: query each buyer i for $v_i(M)$, and $v_i(\{j\})$ for all items j. Then, do the following:

- Let $i^* := \arg\max_i\{v_i(M)\}$.
- Create a bipartite graph with bidders on the left and items on the right. Put an edge between node i and j of weight $v_i(\{j\})$ for all i, j. Find the max-weight matching in this graph, and let $j(i)$ denote the item matched to bidder i in this matching (or null if no item is given to bidder i).
- If $v_{i^*}(M) \geq \sum_i v_i(j(i))$, then give all items to bidder i^*. Else, give each bidder i the item $j(i)$.

Theorem 4 ([?]). *The algorithm above guarantees an $O(\sqrt{m})$-approximation whenever all valuations are subadditive.*

Proof. Let $S_1, \dots, S_n$ denote the welfare-maximizing partition. Further relabel the bidders so that $|S_1| \geq \dots \geq S_n$. Let i' denote the smallest i such that $S_i > \sqrt{m}$ (or null if no such i exists). There are two cases to consider.

First, maybe $\sum_{i \leq i'} v_i(S_i) \geq OPT/2$. Note that $i' \leq \sqrt{m}$ (as each $i \leq i'$ receives $\sqrt{m}$ items). Therefore, there exists an $i \leq i'$ such that $v_i(S_i) \geq OPT/(2\sqrt{m})$. Therefore, $v_{i^*}(M) \geq v_i(M) \geq v_i(S_i) \geq OPT/(2\sqrt{m})$, and our algorithm achieves at least $v_{i^*}(M)$, so we get our approximation.

Next, maybe $\sum_{i \leq i'} v_i(S_i) < OPT/2$. Then $\sum_{i > i'} v_i(S_i) \geq OPT/2$. Now we wish to invoke subadditivity: we know that $v_i(S_i) \leq \sum_{j \in S_i} v_i(\{j\})$. Therefore, for all i, there certainly exists a $j \in S_i$ such that $v_i(\{j\}) \geq v_i(S_i)/|S_i|$. For $i > i'$, this further means there exists a $j \in S_i$ such that $v_i(\{j\}) \geq v_i(S_i)/\sqrt{m}$. Therefore, as the S_is are disjoint, there exists a matching of bidders to items guaranteeing welfare $\geq \sum_{i > i'} v_i(S_i)/\sqrt{m} \geq OPT/(2\sqrt{m})$. □

observation 2. *The above algorithm is* maximal-in-range*. Therefore, it can be turned into a truthful auction.*

Proof. Observe that the algorithm considers every allocation that is either a matching (gives each bidder at most one item) or awards all items to the same bidder. The best allocation from this collection is selected, therefore it is maximal-in-range. □

5 Approximation Algorithms for XOS

Now, we'll provide an algorithm for XOS due to [?] that uses polynomially many demand queries and achieves a $1 - 1/e$ approximation with respect to the optimal welfare. Even more precisely, the guarantee is $1 - (1 - 1/n)^n$, which is tight. Note that this is just an approximation *algorithm* (it's not truthful). Developing a truthful mechanism with comparable guarantees is a major open problem (in fact, the best deterministic, truthful mechanism known achieves only a $\sqrt{m}$-approximation, and is described in the previous section).

The algorithm will proceed by first solving the configuration LP, using polynomially many demand queries. However, we still need to round the fractional solution to a feasible one. The rounding we'll discuss is *oblivious*. This type of rounding uses only the solution to the configuration LP (the $x_i(S)$'s) but not the $v_i(\cdot)$'s. Let's first see how to get a $1-(1/2)^2 = 3/4$ approximation for $n = 2$.

- Let $x_i(S)$ be the fractional solution proposed by the LP. Let $x_{ij} := \sum_{S \ni j} x_i(S)$.
- Draw a random S according to the distribution $x_1(\cdot)$. Draw a random T according to the distribution $x_2(\cdot)$.
- If $j \in S \setminus T$, award item j to bidder 1. If $j \in T \setminus S$, award j to bidder 2. If $j \notin S \cup T$, award j arbitrarily (or to no one).
- If $j \in S \cap T$, award j randomly to bidder 1 with probability $\frac{x_{2j}}{x_{1j}+x_{2j}}$ (and bidder 2 with the remaining probability).

Intuitively, the idea is to try and just allocate the two bidders independently. The problem is that they may both "ask for" the same item. In this case, we randomly award the item, biased towards the bidder who asks for the item *least* often. To prove that the algorithm works, we'll need the following lemma:

Lemma 5. *Let $v(\cdot)$ be an XOS function. Let X be a randomly drawn set satisfying* $\Pr[j \in X] \geq p$ *for all* $j \in S$*. Then* $\mathbb{E}[v(X)] \geq p \cdot v(S)$*.*

Proof. We know that because $v(\cdot)$ is XOS, that there exist some additive function $a(\cdot)$ satisfying $a(S) = v(S)$, and $v(T) \geq a(T)$ for all $T \subseteq M$. Now, to conclude the proof, observe: $\mathbb{E}[a(X)] = \sum_j \Pr[j \in X] \cdot a(\{j\}) \geq p \cdot a(S) = p \cdot v(S)$. □

Theorem 6 ([?]). *The above described rounding algorithm guarantees a* $3/4$*-approximation.*

Proof. Suppose bidder 1 draws set S from their randomized rounding, and consider any $j \in S$. We'll show that bidder 1 keeps item j with probability at least $3/4$ (in expectation over the randomness of the set T drawn by bidder 2, and the randomness of who wins the tiebreaker). We can then directly apply Lemma 5 to claim that bidder 1 gets value at least $(3/4)v(S)$ whenever she is randomly assigned set S, and therefore her total value is at least $3/4$ of what she achieves in the fractional solution.

Now let's compute the probability that bidder 1 keeps item $j \in S$. Observe that in order for bidder 1 to *lose* item 1, first bidder 2 needs to draw a $T \ni j$. This happens with probability x_{2j}. Independently of this, bidder 1 needs to lose the tiebreaker. This happens with probability $\frac{x_{1j}}{x_{1j}+x_{2j}}$. So the total probability that bidder 1 loses item j is $\frac{x_{1j}x_{2j}}{x_{1j}+x_{2j}}$. We want to see how big this term can possibly be, subject to the constraint $x_{ij} \in [0,1]$ and $x_{1j} + x_{2j} \leq 1$.

Simple calculus confirms that this is maximized when $x_{1j} = x_{2j} = 1/2$ and yields a value of $1/4$. Therefore, bidder 1 keeps item j with probabilty at least $3/4$ and the theorem follows. □

The algorithm for $n > 2$ is a bit more complicated, and requires going through a few thought experiments. We will not give a full proof of the bound $1-(1-1/n)^n$, but instead give a proof of the bound $1 - 1/e$ (which is the same as $n \to \infty$). For a proof of the exact bound, see [**?**, **?**].

First, let N be an integer such that $1/N$ divides x_{ij} for all i, j. Such an N certainly exists as long as all values $v_i(S)$ are rational. Now, consider the following flawed experiment for each item j:

- Each bidder i gets $x_{ij}N$ biased coins, which come up heads independently with probability $1/N$.
- Each bidder independently flips all of their coins, and puts any that come up heads into the bag B_j.
- A uniformly random coin is drawn from the bag. The owner of that coin gets item j.

Lemma 7. *The above rounding algorithm awards item j to bidder i with probability at least $(1 - 1/e)x_{ij}$.*

Proof. Observe that the bag is empty with probability $(1 - 1/N)^N \le 1/e$. Therefore, someone wins the item with probability at least $1 - 1/e$. By symmetry, each coin wins the item with probability at least $(1 - 1/e)/N$. Therefore, a bidder with $x_{ij}N$ coins wins with probability at least $(1 - 1/e)x_{ij}$, as desired. $\square$

Of course, there is a problem with plugging the above "rounding" into the previous approach: it doesn't make any reference to the initial S drawn from $x_i(\cdot)$. As such, we cannot use Lemma 5. All we can claim is that bidder i gets item j with probability $(1 - 1/e)x_{ij}$, whereas we want to say that bidder i gets item j with probability $(1 - 1/e)$ *conditioned on drawing $S \ni j$ from $x_i(\cdot)$*. The problem is that we haven't changed the rounding based on S at all.

To fix this, we would like to do the following. First, we want to preserve the distribution of the number of coins that bidder i puts into the bag. But, we'd like to make the distribution no longer independent of S. Rather, we would like to make it so that i only puts coins into bag j when $j \in S$. To do this, consider the following:

- The probability that bidder i puts ℓ coins into bag j is $\binom{x_{ij}N}{\ell}(1/N)^\ell(1 - 1/N)^{x_{ij}N-\ell}$.
- In particular, the probability that bidder i puts 0 coins into bag j is $(1 - 1/N)^{x_{ij}N} \ge 1 - x_{ij}$.
- As such, we can modify the distribution of coins into the bag as follows:
  - First draw S according to $x_i(\cdot)$.
  - If $j \notin S$, put no coins into the bag.
  - If $j \in S$, put exactly ℓ coins into the bag with probability $\frac{1}{x_{ij}}\binom{x_{ij}N}{\ell}(1/N)^\ell(1 - 1/N)^{x_{ij}N-\ell}$, and put 0 coins into the bag with the remaining probability (note that this remaining probability is indeed non-negative, since bidder i puts > 0 coins into the bag with probability that was shown above to be $\le x_{ij}$). The factor $\frac{1}{x_{ij}}$ is needed to normalize the formula for the probability of putting ℓ coins into the bag, since x_{ij} is the probability that bidder i is assigned $S \ni j$.

Theorem 8 ([**?**]). *The above described rounding algorithm guarantees a* $(1-1/e)$*-approximation.*

Proof. We'll again show that whenever bidder 1 draws set S from their randomized rounding, in expectation over the randomness of all other bidders drawing their sets, and the randomness on who wins the tiebreaker, bidder 1 keeps item j with probability at least $1 - 1/e$, for all $j \in S$. We can then directly apply Lemma 5 to claim that bidder 1 gets value at least $(1 - 1/e)v(S)$ whenever she is randomly assigned set S, and therefore her total value is at least $1 - 1/e$ of what she achieves in the fractional solution.

To see this, observe that we indeed kept the distribution of coins in the bag identical for each bidder. As such, bidder i indeed wins item j with probability at least $(1 - 1/e)x_{ij}$. However, we have also modified the distribution so that bidder i puts a coin in the bag *only when* $j \in S$. Therefore, bidder i must win item j conditioned on $j \in S$ with probability at least $1 - 1/e$.

$\square$

Bibliography

1. Algorithmic Game Theory. Nisan, Roughgarden, Tardos, Vazirani (eds.), Cambridge University Press 2007.