Harvard CS 1360, Spring 2025
Instructor: Ariel Procaccia
Scribe: Ethan Tan

Strategyproof Approximation Algorithms

—

Lecture 9

1 Introduction

This lecture explores strategyproof approximation algorithms in mechanism design, specifically in cases where computing an optimal solution is computationally infeasible. We focus on single-minded auctions as a case study.

2 VCG Mechanism and Its Limitations

Recall the definition of the Vickrey-Clarke-Groves (VCG) mechanism:

Definition 1 (The VCG Mechanism)**.** The Vickrey-Clarke-Groves (VCG) mechanism is defined by:

- A welfare-maximizing choice rule:
$$f(\mathbf{v}) \in \arg\max_{x \in A} \sum_{i \in N} v_i(x).$$

- A payment rule p, where A^{-i} is the set of alternatives that are available when i is not present:
$$p_i(\mathbf{v}) = \max_{x \in A^{-i}} \sum_{j \neq i} v_j(x) - \sum_{j \neq i} v_j(f(\mathbf{v})).$$

Although it is strategyproof and welfare-maximizing, the VCG mechanism relies on computing the allocation x that satisfies the welfare-maximizing choice rule:

$$f(\mathbf{v}) \in \arg\max_{x \in A} \sum_{i \in N} v_i(x)$$

And this is computationally hard in many settings.

Hence, we turn to strategyproof approximation algorithms, which are computationally feasible mechanisms that allow us to retain strategyproofness while approximating the optimal solution.

3 Single-Minded Auctions

Definition 2 (Single-Minded Auction)**.** A single-minded auction consists of:

- A set G of m goods.
- Each player $i \in N$ has a target bundle $T_i \subseteq G$.
- Player i has a value $v_i(S) = w_i$ if $T_i \subseteq S$ and 0 otherwise.

For instance, consider this scenario with 4 players, goods $\{1, 2, 3, 4\}$, and the following target bundles and values for each player:

Player	1	2	3	4
Bundle	$\{1,2\}$	$\{2,3\}$	$\{1,3\}$	$\{3,4\}$
Value	5	3	4	2

If we assign goods $\{1, 2\}$ to player 1, they receive a value of 5. Player 1 also receives a value of 5 if they get $\{1, 2, 3\}$, because they receive their target of $\{1, 2\}$ and no additional value for the extra good 3. However, if we only assign $\{1\}$ to player 1, they receive a value of 0, because their target bundle $\{1, 2\}$ was not in their set of assigned goods. We also see by inspection that the welfare-maximizing allocation is $\{1, 2\}$ to player 1 and $\{3, 4\}$ to player 4, yielding a total welfare of 7.

3.1 NP-Hardness

Maximizing welfare in single-minded auctions is NP-hard, as shown by a reduction from the Maximum Independent Set problem.

The Maximum Independent Set problem is defined as follows:

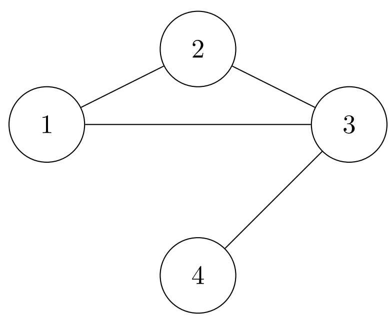
Definition 3 (Maximum Independent Set)**.** Given a graph $G = (V, E)$, find the largest set $S \subseteq V$ such that no two vertices in S are adjacent.

Let us show that any instance of the Maximum Independent Set problem can be reduced to an instance of the single-minded auction problem.

- Given a graph $G = (V, E)$, construct a single-minded auction with players corresponding to vertices and goods corresponding to edges.
- Each player desires the set of edges incident to their vertex, with value 1.
- The welfare-maximizing allocation gives us exactly the maximum independent set, because we choose the maximum number of players such that no two players share an edge, and each player contributes a value of 1.

So any algorithm that finds a welfare-maximizing allocation in single-minded auctions would also solve the Maximum Independent Set problem. Hence, single-minded auctions are at least as hard as the Maximum Independent Set problem, and are thus NP-hard.

For example, let's apply the above reduction to this graph:



Here, our players are the nodes $\{1, 2, 3, 4\}$, the target bundle of any player i is the set of edges connected to node i, and each player has a value of 1 on their bundle.

Player	1	2	3	4
Bundle	$\{(1,2),(1,3)\}$	$\{(1,2),(2,3)\}$	$\{(2,3),(1,3)\}$	$\{(3,4)\}$
Value	1	1	1	1

Then finding the optimal allocation in this auction would give us exactly the maximum set of nodes (players) such that no two nodes share an edge (no two players were assigned the same good).

4 The Greedy Mechanism

The greedy mechanism for single-minded auctions is an approximation algorithm that is also strategyproof. It is defined as follows:

Definition 4 (Greedy Mechanism)**.** Each player i submits a bid (T_i, w_i), where T_i is the target bundle and w_i is the value. Then the greedy mechanism consists of:

- Allocation rule: Sort bids (T_i, w_i) in decreasing order of w_i, breaking ties arbitrarily. Accept bids greedily while feasible, i.e., if T_i is disjoint from the bundles of previously accepted players.

- Payment rule: Each allocated player pays the *critical value*: the smallest w'_i such that their bid (T_i, w'_i) would still be accepted.

Let us show that the greedy mechanism's allocation and payments can be computed in polynomial time. To do this, we define a helpful lemma to calculate the critical value for each player. We start by defining the *conflict set* of a player:

Definition 5 (Conflict Set)**.** Let N_i be the set of winners if player i is not present. The conflict set of player i is:

$$N'_i(T_i) = \{j \in N_i \mid T_i \cap T_j \neq \emptyset\}.$$

Which is the set of players in N_i whose target bundles overlap with player i's target bundle.

Then we can find the critical value of player i as follows:

Lemma 1 (Critical Value Calculation)**.** The critical value of player i is:

$$w_i^c = \max_{j \in N'_i(T_i)} w_j,$$

i.e., the maximum value of any player in the conflict set of player i.

Proof. For player i to be allocated, their bid must be higher than the competing bids in the conflict set $N'_i(T_i)$. We don't have to consider competing bids outside of N_i because those players are not allocated and won't affect whether player i is allocated.

Now consider N_i. Suppose some player $j \in N_i$ has a target bundle that does not overlap with T_i. Player i does not need to outbid player j because even if player j is allocated, player i can still be allocated since their target bundles do not overlap.

Finally, we are left with the players in $N'_i(T_i)$. If player i bids below any of these players, they will not be allocated, since the greedy mechanism allocates players in decreasing order of value. Thus, player i's bid must be at least as high as the highest bid in $N'_i(T_i)$. □

It is then easy to define a polynomial-time algorithm to compute the allocation and payments for the greedy mechanism:

Greedy Mechanism

1. Sort the bids in decreasing order of value.
2. Initialize the set of allocated players $N_a = \emptyset$.
3. For each player i in the sorted order:
   (a) If T_i is disjoint from the bundles of players in N_a, add i to N_a. Else, move to the next player.
   (b) Compute the critical value w_i^c, and assign w_i^c as the payment for player i.

Then where n is the number of players, and m is the number of goods:

- Step 1 takes $O(n \log n)$ time using any choice of efficient sorting algorithm.
- Step 3(a) takes $O(m)$ time to check for disjointedness.
- To compute the critical value in Step 3(b), we first find N_i, which at worst takes $O((n \log n) \cdot m)$ time to sort the bids and find the allocated players. Then we find the maximum value in N_i, which at worst takes $O(nm)$ time to check each player for overlap with T_i in decreasing order of value, returning the maximum value or 0 if none was found.

3(a) and 3(b) are repeated for n players, so the total time complexity is polynomial in n and m.

4.1 Strategyproofness

Theorem 1 (Strategyproofness of Greedy Mechanism)**.** *The greedy single-minded auction is strategyproof.*

Proof. Firstly, note that it is not useful for any player i to misreport a bundle T_i' that does not contain T_i, as they will receive a value of 0 in that case. Thus it must be the case that $T_i \subseteq T_i'$.

Fixing T_i', we now consider possible misreports of value, w_i'. Player i's allocation is monotone weakly increasing in w_i', because as w_i' increases from 0, there will be some point at which $w_i' \geq w_j$ for all players j with bundles that intersect T_i': this is when player i gets allocated. Beyond this point, any increase in w_i' will not affect the allocation of player i, since the greedy mechanism allocates to the highest bids first for which the remaining goods are still allocatable.

Now we will show that the value that should be reported for T_i' is exactly the true value w_i. Suppose player i is currently to pay w_i^c, the critical value. The reasoning here is similar to a second-price auction.

Consider a misreport of $w_i' > w_i$. Player i can only go from not being allocated to being allocated, since allocations are monotone weakly increasing. But if they are allocated, they will either still pay w_i^c, or pay more if some other player j with $T_j \cap T_i' \neq \emptyset$ had $w_j \geq w_i$ and was allocated over player i. In this case, they will pay at least as much as their actual valuation of w_i for the bundle T_i', which is not beneficial.

Next, consider a misreport of $w_i' < w_i$. If $w_i' \geq w_i^c$, player i's allocation and payment will not change (where WLOG, we break ties in favor of player i). If player i was initially allocated but the new $w_i' < w_i^c$, they will lose their allocation to the player whose value was w_i^c. But since the original $w_i \geq w_i^c$, player i was receiving non-negative utility from being allocated, since their payment was at most their true valuation, which they have now lost.

So we see that any misreport of value either does not change player i's value or results in a loss of utility.

Finally, we show that misreporting a larger bundle $T_i' \supset T_i$ is not beneficial. Player i receives the same value from T_i and T_i', but the conflict set $N_i'(T_i') \supseteq N_i'(T_i)$ since there may be more players whose bundles overlap with the larger T_i'. Thus, the critical value w_i^c will either remain the same or larger while player i receives the same value, so this is not a useful misreport either.

Since a misreport in value or bundle is not beneficial, the greedy mechanism is strategyproof. ☐

5 Approximation Guarantees

Definition 6 (c-approximation for maximization problem)**.** An algorithm is a c-approximation for a maximization problem if for all instances I and $c \leq 1$:

$$ALG(I) \geq c \cdot OPT(I),$$

where $ALG(I)$ is the welfare of the solution produced by the algorithm, and $OPT(I)$ is the welfare of the optimal solution.

In words, the worst-case utility produced by the algorithm is at least a fraction c of the optimal solution.

Definition 7 (c-approximation for minimization problem)**.** With the same notation, an algorithm is a c-approximation for a minimization problem if for all instances I and $c \geq 1$:

$$ALG(I) \leq c \cdot OPT(I),$$

where $ALG(I)$ is the cost of the solution produced by the algorithm, and $OPT(I)$ is the cost of the optimal solution.

In words, the worst-case cost produced by the algorithm is at most c times the optimal solution.

5.1 Greedy Mechanism Approximation Ratio

We can now define the approximation ratio of the greedy mechanism:

Theorem 2 (Approximation Ratio of Greedy Mechanism)**.** *The greedy mechanism for single-minded auctions is a $1/d$-approximation, where d is the maximum size of any target bundle.*

A variant of the greedy mechanism sorting by $w_i/\sqrt{|T_i|}$ instead of w_i gives a $1/\sqrt{m}$-approximation for m items.

Theorem 3 (NP-Hardness of Better Approximation)**.** *Finding a better approximation ratio for single-minded auctions than $1/\sqrt{m}$ is NP-hard.*

That is to say, there is no polynomial-time algorithm that can achieve a better approximation ratio than $1/\sqrt{m}$ for single-minded auctions, unless P=NP.

Finally, we prove Theorem 2:

Proof. Let N_{alg} be the set of players allocated by the greedy mechanism, and N_{opt} be the set of players in the optimal solution.

Let's think about the players whose allocations were 'messed up' by the greedy mechanism. First, consider players $i \in N_{\text{alg}} \setminus N_{\text{opt}}$: these are the players who were not allocated in the optimal solution and wrongly allocated by the greedy mechanism.

For each such i, let N_i be the set of players j in the optimal allocation N_{opt} that they blocked from being allocated, i.e. where $w_j \leq w_i$ and $T_i \cap T_j \neq \emptyset$.

For all other players $i \in N_{\text{opt}} \cap N_{\text{alg}}$, i.e. players rightfully allocated by the greedy mechanism, let $N_i = \{i\}$.

Now consider an arbitrary N_i. First observe that:

$$\sum_{j \in N_i} w_j \leq \sum_{j \in N_i} w_i = |N_i| \cdot w_i.$$

Because either $N_i = \{i\}$ and equality holds, or N_i contains players j with $w_j \leq w_i$, so the sum must be at least as big when we replace the w_j with w_i. Next, note that:

$$|N_i| \cdot w_i \leq d \cdot w_i$$

because for each player $j \in N_i$, $T_i \cap T_j$ intersects on at least one item, and the T_j's are disjoint from each other — recall that N_i consists of players from the optimal allocation, and two players cannot be allocated at once. But then since d is the largest bundle size, T_i can have at most d intersections with other bundles. Thus, $|N_i| \leq d$, and:

$$\sum_{j \in N_i} w_j \leq d \cdot w_i. \tag{1}$$

In addition, note that:

$$N_{\text{opt}} = \bigcup_{i \in N_{\text{alg}}} N_i. \tag{2}$$

Let's briefly prove each direction:

($N_{\text{opt}} \subseteq \bigcup_{i \in N_{\text{alg}}} N_i$): For any player $i \in N_{\text{opt}}$, they were either allocated by the greedy mechanism and so they must be in $N_i = \{i\}$, or they were not allocated and so are in some N_j where player j was allocated by the greedy mechanism and blocked i.

($N_{\text{opt}} \supseteq \bigcup_{i \in N_{\text{alg}}} N_i$): This follows immediately from how we defined each N_i. For any player j in some N_i either they were allocated by both the greedy and optimal algorithm and are thus in N_{opt}, or they were not allocated by the greedy mechanism but were in the optimal allocation and are thus in N_{opt}.

Putting it all together, with OPT as the welfare of the optimal solution and ALG as the welfare of the greedy mechanism:

$$
\begin{aligned}
OPT &= \sum_{i \in N_{\text{opt}}} w_i && \text{by definition} \\
&\le \sum_{i \in N_{\text{alg}}} \sum_{j \in N_i} w_j && \text{from Eq. (2)} \\
&\le \sum_{i \in N_{\text{alg}}} d \cdot w_i && \text{from Eq. (1)} \\
&= d \cdot \sum_{i \in N_{\text{alg}}} w_i \\
&= d \cdot ALG && \text{by definition.}
\end{aligned}
$$

For the first inequality, note that we may count some elements of N_{opt} multiple times, which is why it is an inequality instead of an equality. □