Economics and Computation (Spring 2025)
Assignment #3
— Solutions —

Due: 3/24/2025 11:59pm ET

Problem 1: The VCG Mechanism

[15 points] Under the VCG Mechanism, the allocation rule maximizes (utilitarian) social welfare. Consider, instead, a *weighted social welfare* objective, defined as $\max_{x \in A} \sum_{i \in N} \alpha_i v_i(x)$, for given multipliers $\alpha_i \geq 0$, $i = 1, \ldots, n$.

For an allocation rule optimizing weighted social welfare, design a payment rule such that the mechanism is strategyproof.

Solution:

Proposed Payment Rule: In line with the VCG Mechanism we have studied in class, we want to charge agent i with the externality they impose on others. First, we follow the allocation rule and choose $x^*(v)$ that maximizes the weighted social welfare:

$$x^*(v) \in arg \max_{x \in A} \sum_{i \in N} \alpha_i v_i(x)$$

Then, we define the payment method for each agent i:

$$p_i(v) = h_i(v_{-i}) - \sum_{j \neq i} \frac{\alpha_j}{\alpha_i} v_j(x^*(v))$$

where h is an arbitrary function that does not depend on v_i.

Strategyproofness: We want to show that no misreport will yield a higher utility for agent i.

$$\begin{aligned} u_i(v) &= v_i(x^*(v)) - p_i(v) \\ &= v_i(x^*(v)) - h_i(v_{-i}) + \sum_{j \neq i} \frac{\alpha_j}{\alpha_i} v_j(x^*(v)) \end{aligned}$$

We have designed a function that is simply the maximized weighted social welfare with $\frac{1}{\alpha_i}$. Here, we notice when an agent i reports $\hat{v}_i \neq v_i$, it can only impact the utility through the x^* chosen from the objective function.

$$\alpha_i \left[v_i(x^*(v)) + \sum_{j \neq i} \frac{\alpha_j}{\alpha_i} v_j(x^*(v)) \right] = \alpha_i v_i(x^*(v)) + \sum_{j \neq i} \alpha_j v_j(x^*(v))$$
$$= \sum_{i \in N} \alpha_i v_i(x^*(v))$$

If agent i overstates or understates $\hat{v}_i$, then they risk selecting an outcome that lowers the utility compared to the maximizing choice x^*.

Problem 2: Strategyproof approximation algorithms

[15 points] In Lecture 9, we discussed the greedy mechanism for single-minded bidders. Let us modify the mechanism by tweaking the allocation rule: instead of ordering bids by decreasing w_i, sort the bids by decreasing $w_i/\sqrt{|T_i|}$. It turns out that this gives a $\sqrt{m}$-approximation.

Prove that the modified greedy algorithm is strategyproof.

Guidance: Adapt the lemma on Slide 7 by showing that the critical value of i is $\min\{w_i' : w_i'/\sqrt{|T_i|} \geq \max_{j \in N_i'(T_i)} w_j/\sqrt{|T_j|}\}$. Next, adapt the proof on Slide 8.

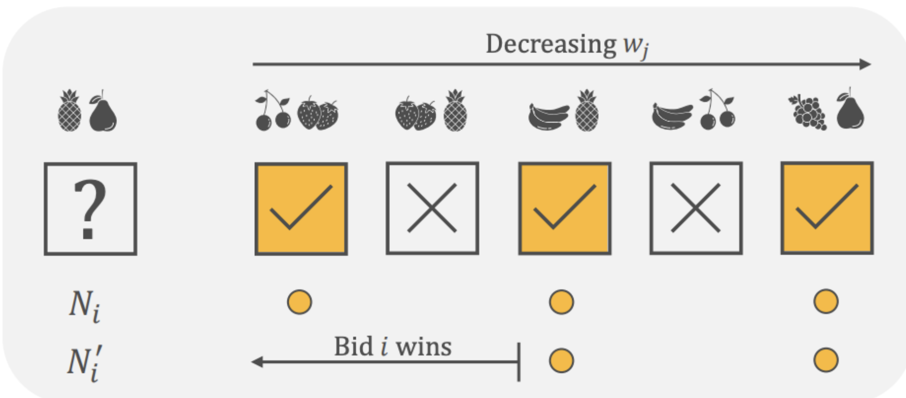

Figure 1: From Slide 7 of Mechanism Design 2 lecture. Given that the target bundles includes the pineapple and the pear, the 3rd and 5th bundles from the right are in the conflict set, and i needs to score ahead of both of them to be allocated.

Solution:

Critical value of bidder in greedy single-minded auctions. Consider the Figure 1. Let N_i be the set of winners if i is removes. Let N_i' be the conflict set of i. First, observe that if $j \geq 0$ bids are allocated before bid i then these are the same bids, and allocated in the same order, as the first j bids allocated in N_i. Thus, a bid with value $\hat{w}_i = w_{crit,i}(T_i)$ wins because any bids allocated before i are in $N_i \setminus N_i'$, and thus don't block i. Also, if $\hat{w}_i < w_{crit,i}(T_i)$, then i is not allocated because at least one of the bidders in N_i' is allocated before i, and blocks i from being allocated.

Strategyproofness of greedy single-minded auctions for a monotone score function. Consider monotone score function $\sigma(T_i, w_i) = w_i/\sqrt{|T_i|}$. For any such T_i, the allocation is monotone weakly increasing in report w_i: if allocated for report w_i' then i is allocated for any $w_i^{"} > w_i'$ since the score of the bid weakly increases (score function is monotone) and so no new bids can be allocated before i at $w_i^{"}$ than w_i' (and i continues to be scored ahead of any bids in $N_i'(T_i)$). From this, and following the above lemma, bidder i is allocated bundle T_i at price $w_{crit,i}(T_i)$ if and only if $\hat{w}_i \geq w_{crit,i}(T_i)$. Since $w_{crit,i}(T_i)$ is self-report independent, bidding $\hat{w}_i = w_i$ is optimal.

Problem 3: Cake cutting

[**20 points**] In class we discussed the Even-Paz Algorithm, which guarantees a proportional allocation of the cake with $O(n \log n)$ queries in the Robertson-Webb Model. However, in the analysis we made the simplifying assumption that $n = 2^k$ for some $k \in \mathbb{N}$.

Generalize the algorithm to an arbitrary number of players n, and prove that your generalized algorithm is proportional and that it requires $O(n \log n)$ queries.

Solution:

Algorithm (divide and conquer):

If $n = 1$: the single player takes the whole cake.

If $n = 2k$ for some $k \in \mathbb{N}^+$: every player draws a vertical line on the cake that cuts it in half according to their valuation function. Cut the cake anywhere between the kth and $(k+1)$th lines inclusive. The players who drew the k leftmost lines recurse on the left piece and the players who drew the k rightmost lines recurse on the right piece.

If $n = 2k + 1$ for some $k \in \mathbb{N}^+$: every player draws a vertical line such that, according to their valuation function, the left side of the line is worth $\frac{k}{2k+1}$ of the total cake and the right side of the line is worth $\frac{k+1}{2k+1}$ of the total cake. Cut the cake on the median line. The players who drew the k leftmost lines recurse on the left piece and the players who drew the $k + 1$ rightmost lines recurse on the right piece.

Proof of proportionality: This algorithm is proportional: for any piece C of cake and for any number n of players, each player will walk away with a slice S_i that they feel is worth at least $\frac{1}{n}$ of the value of C. (Equivalently, $\forall i,\ n \cdot V_i(S_i) \geq V_i(C)$). The proof is by induction.

Base case ($n = 1$): The player walks away with all of the remaining cake. $1 \cdot V_i(C) \geq V_i(C)$.

Induction hypothesis: for all $k \leq n$, this algorithm always achieves a proportional allocation among $(k - 1)$ players.

Induction step ($n = 2k$): Note that after the cake is cut, the k players who recurse on the left piece (call it L) value it at least half as much as they value C. By the IH, each player i will end up with a piece that they think is worth at least $\frac{V_i(L)}{k}$. But since we just showed that $V_i(L) \geq \frac{V_i(C)}{2}$ for these players, this value is at least $\frac{V_i(C)}{n}$. So proportionality is achieved for these k players. An identical argument applies to the other k players who recurse on the right piece.

Induction step ($n = 2k + 1$): Note that after the cake is cut, the k players who recurse on the left piece (call it L) value it at least $\frac{k}{2k+1}$ as much as they value C. By the IH, each player i will end up with a piece that they think is worth at least

$$\frac{V_i(L)}{k} \geq \frac{V_i(C)}{2k+1} = \frac{V_i(C)}{n}.$$

A structurally identical argument applies to the $k+1$ players who recurse on the right piece: by the IH, they each end up with a piece that they think is worth at least

$$\frac{V_i(R)}{k+1} \geq \frac{V_i(C)}{2k+1} = \frac{V_i(C)}{n}.$$

So proportionality is achieved for everybody.

Proof of runtime: The algorithm makes $O(n)$ cuts at each level of recursion. And there are $O(\log n)$ levels of recursion, because at each step, at least $1/3$ of the players are removed, so the depth of the recursion tree is at most $\log_{3/2} n$.

Problem 4: Rent division

[20 points] In class we stated two lemmas in the context of the rent division with quasi-linear utilities. The second lemma can actually be strengthened as follows:

Lemma: If $(\pi, \mathbf{p})$ is an EF solution and σ is a welfare-maximizing assignment, then for all $i \in N$, $v_{i\pi(i)} - p_{\pi(i)} = v_{i\sigma(i)} - p_{\sigma(i)}$.

Using this lemma (without proof), describe a polynomial-time algorithm for computing the maximin solution and prove its correctness.

Guidance: Follow the algorithm sketch on Slide 19 of Lecture 11 but replace the system of linear inequalities with an appropriate linear program. You may assume that a linear program can be solved in polynomial time. Note that a linear program can maximize the minimum of linear functions, similarly to our linear program for computing a maximin strategy in a two-player zero-sum game.

Solution: We first find the matching of players to rooms of maximum weight, where the weight of an edge between a player and a room is the utility that the player has for the room. This can be done in polynomial time, and the resulting allocation σ will be a welfare-maximizing assignment. We then compute the price vector $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$ by solving the following linear program:

maximize t

subject to
$t \leq v_{i\sigma(i)} - p_{\sigma(i)}$ for all $i \in N$
$v_{i\sigma(i)} - p_{\sigma(i)} \geq v_{ij} - p_j$ for all $i, j \in N$
$\sum_{j=1}^{n} p_j = R$

Our computed solution $(\sigma, \boldsymbol{p})$ is clearly EF since we enforced this in the constraints, so all that remains is to show that it maximizes the minimum utility of any player.

Observe that, given a price vector, the optimal t is the minimum utility of any agent under assignment σ according to those prices. Therefore, since we are maximizing t, the optimal price vector $\boldsymbol{p}$ that we compute maximizes the minimum utility of any agent under assignment σ and prices satisfying EF. Suppose the true Maximin solution is $(\sigma^*, \boldsymbol{p}^*)$. Since, by the definition of the Maximin solution, $(\sigma^*, \boldsymbol{p}^*)$ is EF, the lemma implies that, for all $i \in N$,

$$v_{i\sigma(i)} - p^*_{\sigma(i)} = v_{i\sigma^*(i)} - p^*_{\sigma^*(i)}.$$

This means that $\boldsymbol{p}^*$ is a feasible solution to the LP, as $(\sigma, \boldsymbol{p}^*)$ must clearly be EF by the lemma. Therefore,

$$\min_i v_{i\sigma(i)} - p_{\sigma(i)} \geq \min_i v_{i\sigma(i)} - p^*_{\sigma(i)} = \min_i v_{i\sigma^*(i)} - p^*_{\sigma^*(i)},$$

where the inequality follows from the fact that $\boldsymbol{p}$ maximized the LP and $\boldsymbol{p}^*$ is feasible, and the equality follows from the lemma. In other words, $(\sigma, \boldsymbol{p})$ achieves at least as high a minimum utility as the optimal minimum utility (subject to EF), so it must also have an optimal minimum utility (subject to EF). Thus, $(\sigma, \boldsymbol{p})$ is a Minimax solution.

Problem 5: Formulate a research question

[**30 points**] Formulate a research question that is relevant to one of the topics covered in this assignment: the VCG Mechanism, strategyproof approximation algorithms, cake cutting, and rent division. Refer to this document for guidelines.

During the process of formulating your question, keep track of your findings in a "research journal." At a minimum, it should include brainstorming ideas for questions and notes on relevant papers that you have identified.

Please submit the following deliverables:

1. Your research question.
2. A *brief* explanation of why it satisfies each of the following criteria:
   (a) Relevant: Which course topics is the question related to?
   (b) Nontrivial: What is an immediate way of attempting to answer the question and why does it fail?
   (c) Feasible: How would you tackle the question if you had the entire semester?
   (d) Novel: List the 1–3 most closely related papers that you have identified in your literature review and explain how your question differs.

   Note: *Your writeup of all four parts of Item 2 must be at most two pages long overall.*
3. Append your research journal to the PDF that contains your solutions. The research journal will not be graded; it is there to show your work.