Algorithmic Game Theory

Autumn 2021
Exercise Set 6

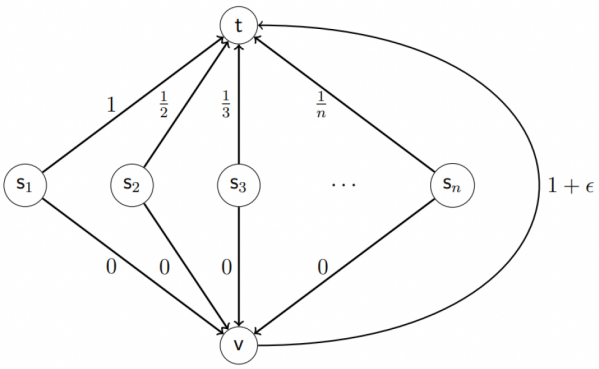
Elisabetta Fedele

Contributors
Elisabetta Fedele, Massimo Albarello, Julian Gleissner, Davide Maioli

Exercise 1

Our task is to show that in the game described in the figure below the following equalities hold:

$$PoS_{PNE} = PoS_{MNE} = PoS_{CCE}$$



As we have already seen in Lecture 5 this game has only one PNE, when all the players choose the direct edge. The minimum social cost possible will be always $min_{s \in S} cost(s) = 1 + \epsilon$, and the price of stability is defined as $PoS = \frac{min_{p \in Eq} cost(p)}{min_{s \in S} cost(s)}$, thus, we only need to prove that

$$min_{c \in PNE} cost(c) = min_{p \in MNE} cost(p) = min_{p \in CCE} cost(p)$$

Our task becomes then to prove that the social cost of the PNE is equal to the minimum expectation of social cost in a MNE and in a CCE.
In order to do so, we need to prove that the only probability distribution p on the set of states S such that for every player i and for every deviation $s'_i \in S$ we have

$$\mathbf{E}_{s \sim p}[c_i(s)] \leq \mathbf{E}_{s \sim p}[c_i(s'_i, s_{-i})]$$

is the one which gives probability 1 to the state in which all the players use the direct edge and a null probability to all the other states. As a result, we need to prove that all the other probability distributions over the states would not be a CCE for at least one player and, thus, that all the players reject the probability distributions in which the have a probability to take the detour greater than 0.
We will prove this thesis by induction on the number of players. Let us start from the player n.
First, we observe that the expected value of the cost of player n choosing the direct path will be always lower than the expected value obtained using the detour, regardless of the probability distribution chosen on the set of state:

$$\mathbf{E}_{s \sim p}[c_n(\text{"direct path"})] = \frac{1}{n}$$

$$\mathbf{E}_{s \sim p}[c_n(\text{"detour"})] \leq max(c_n(\text{"detour"})) = \frac{1+\epsilon}{n}$$

for all $p \in P$, where P is the set of all the possible probability distributions over S, the set of all possible states. As a consequence, the player n will for sure reject all the probability distributions over the set of states, which give a probability greater than zero to states in which the player n chooses to use the detour, as in this cases we have

$$\mathbf{E}_{s \sim p}[c_n(s)] \geq p(\text{"detour"}) \cdot \frac{1+\epsilon}{n} + p(\text{"direct edge"}) \cdot \frac{1}{n} =$$

$$= p(\text{"detour"}) \cdot \frac{1+\epsilon}{n} + (1 - p(\text{"detour"})) \cdot \frac{1}{n} =$$

$$= p(\text{"detour"}) \cdot \frac{1}{\epsilon} + \frac{1}{n} > \frac{1}{n} = \mathbf{E}_{s \sim p}[c_n(s'_n, s_{-n})]$$

and thus

$$\mathbf{E}_{s \sim p}[c_n(s)] > \mathbf{E}_{s \sim p}[c_n(s'_n, s_{-n})]$$

for all the probability distributions p in which the player n takes the detour path with probability greater than 0 and for all the strategies s'_n in which the player n takes the direct path.
As a result all the probability distributions over the set of states which give a probability greater than zero to states in which the player n chooses to use the detour, cannot be a CCE. Now, we will assume that our thesis is true for all the players with index greater than i (with $0 < i < n$) and we will prove it for the player i. As a consequence, all these players have rejected (since they were not a CCE for them) all the probability distributions over the set of strategies which give them a probability greater than zero to choose strategies in which they choose to use the detour.
We now consider the expected costs of player i, considering that now, for hypothesis of induction, at most i players can take the detour in the probability distributions we are still considering

$$\mathbf{E}_{s\sim p}[c_i(\text{"direct path"})] = \frac{1}{i}$$

$$\mathbf{E}_{s\sim p}[c_i(\text{"detour"})] \geq \frac{1+\epsilon}{i}$$

for all $p \in P'$, where P' is the set of all the probability distributions that we are still considering.
Since for all $p \in P'$ it holds that

$$\mathbf{E}_{s\sim p}[c_i(\text{"direct path"})] < \mathbf{E}_{s\sim p}[c_i(\text{"detour"})]$$

the player i will for sure reject all the probability distributions over the set of strategies, which give a probability greater than zero to strategies in which the player i chooses to use the detour.
Thus, for induction on the number of players, we have proved that all the players will reject all the probability distributions over the set of states, which give them a probability greater than zero to choose states in which they choose to use the detour. As a consequence, the only probability distribution feasible in order to create a CCE is the one that assigns probability one to the state in which all the players choose to use the direct edge. Then we have that

$$min_{p\in CCE} cost(p) = cost(\text{"all players taking their direct edge"}) = min_{c\in PNE} cost(c)$$

As a consequence:

$$PoS_{PNE} = PoS_{CCE}$$

and, since

$$PoS_{PNE} \leq PoS_{MNE} \leq PoS_{CCE}$$

we have proved that:

$$PoS_{PNE} = PoS_{MNE} = PoS_{CCE}$$

Exercise 2

Part 1

Our task was to design a truthful mechanism for the problem proposed. In order to do so, we have designed a VCG mechanism composed by the following algorithm and payments.
Our algorithm among all the feasible trees in the set $\mathcal{T} = \{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_M\}$, chooses the first one which minimizes the sum of reported weights. Let us call this tree T_{min}.
Now we will define the payments. We define the payment to an edge e of the graph as follows:

$$PAY_e = Q_{-e} - \sum_{j \in T_{min}, j \neq e} c_j$$

Let us call G' the graph obtained removing e from the given one. We define Q_{-e} as

$$Q_{-e} = \begin{cases} cost(\text{minimum spanning tree G'}) & \text{if the removing of } e \text{ does not disconnect the graph} \\ 0 & \text{otherwise} \end{cases}.$$

Part 2

We can observe that:

- our algorithm always chooses a solution which minimizes the social cost
- the payments satisfy the payments form of a VCG

Thus, our mechanism is a VCG and for the Theorem 8 (Vickrey-Clarke-Groves) this is a truthful mechanism.

Part 3

We now have to show that our mechanism guarantees the voluntary participation for all the edges whose removal does not disconnect the graph. In order to do so we must be able to show that when a player is *truth-telling* in this game, he will always get a non-negative utility.
We must analyze two different situations.

Case 1: $e \in T_{min}$

In this case, $e \in T_{min}$, thus, all the other possible paths (at least one for hypothesis) which link the vertexes connected by e have not be included in T_{min} since, otherwise T_{min} would not have been a tree. Thus, the utility of e is

$$u_e = PAY_e - c_e = cost(\text{minimum spanning tree G'}) - \sum_{j \in T_{min}, j \neq e} c_j - c_e =$$

$$= cost(\text{minimum spanning tree G'}) - cost(\text{minimum spanning tree})$$

We can observe that it must hold $cost(\text{minimum spanning tree G'}) \geq cost(\text{minimum spanning tree})$. Otherwise, we could have obtained a minimum spanning tree with a minor cost excluding e from T_{min}. But this is not possible since we have assumed that $e \in T_{min}$.
Hence, it must hold that $cost(\text{minimum spanning tree G'}) - cost(\text{minimum spanning tree}) \geq 0$.
Rearranging all the terms we obtain

$$u_e = cost(\text{minimum spanning tree G'}) - \sum_{j \in T_{min}, j \neq e} c_j - c_e \geq 0$$

We proved that our algorithm guarantees voluntary participation to all $e \in T_{min}$ whose removal does not disconnect the graph.

Case 2: $e \notin T_{min}$

If the edge e is not included in the chosen T_{min} the cost for this player will be 0 and thus his utility will be:

$$u_e = cost(\text{minimum spanning tree G'}) - \sum_{j \in T_{min}, j \neq e} c_j$$

But if e is not included in T_{min} we can observe that $cost$(minimum spanning tree G') $= \sum_{j \in T_{min}, j \neq e} c_j$ and that thus $u_e = 0$.

Hence, we have proved that our mechanism guarantees the participation of all the players corresponding to edges which are not included in T_{min} and whose removal does not disconnect the graph.

If we run our mechanism on the given graph we obtain a T_{min} composed by the following edges: (a, b), (a, c), (c, d).
The computed payments are:

- $PAY(a, b) = 10 - 7 = 3$
- $PAY(a, c) = 9 - 6 = 3$
- $PAY(b, c) = 8 - 8 = 0$
- $PAY(c, d) = 0 - 3 = -3$

Exercise 3

Part 1

Consider the set of all allocations, sorted in some order, defined independently from the costs declared by the machines.

$$\mathcal{A} = \{a^1, a^2, ..., a^N\}$$

The algorithm returns the first (lexicographically minimal) allocation among those minimizing the makespan with respect to the input cost $c_1, ..., c_n$. That is the minimum s such that

$$makespan(a^s, c) \leq makespan(a^h, c)$$

for all h. We will prove that this *brute-force* algorithm is monotone.

According to the definition of monotone algorithm, we say that A is monotone, for a given one-parameter problem, if for all i, for all c_{-i}

$$w_i(A(c_i, c_{-i}))$$

is monotone non-increasing in c_i. Hence, we need to prove that for every increment of c_i our algorithm would never give more workload (w_i) to the machine i.
We will prove it by contradiction.
In particular we will start from an allocation a which has been chosen by our algorithm and we will increase the declared cost of a fixed machine i, keeping the other costs unchanged.
Suppose that, fixed one machine i, there exist an increment of his declared price from c_i to $c_i' > c_i$ such that the algorithm chooses a new allocation a' in which a workload given to the machine i has increased to the value $w_i' > w_i$.
Let us define c and c' as follows:

$$c = (c_1, ..., c_i, ..., c_n)$$
$$c' = (c_1, ..., c_i', ..., c_n)$$

In the transition from (a, c) to (a', c') we have to consider mainly three different situations.

Case 1: makespan(a, c) < makespan(a', c')

Since $w_i < w_i'$ and from the definition of makespan we can observe that

$$wc' < w'c' \leq makespan(a', c') \tag{1}$$

Moreover, since the costs of the other machines have remained unchanged, if we keep the same configuration a after that the machine i has changed its cost, all the $w_j \cdot c_j$ will remain unchanged forall $j \neq i$. Hence, we can define $makespan(a, c')$ as follows:

$$makespan(a, c') = max(wc', makespan(a, c))$$

Since we are in the case in which $makespan(a, c) < makespan(a', c')$ and from (1) $wc' < makespan(a', c')$, we can rearrange all the terms and obtain that:

$$makespan(a, c') < makespan(a', c')$$

As a result, we have found an allocation a with regard to c' which has a smaller makespan than the one suggested by our algorithm. But this is absurd since our algorithm always chooses the solution with the minimum makespan. Thus, after a change of the cost of the machine i, it is impossible that our algorithm chooses an allocation a' in which $w_i' > w_i$ and $makespan(a, c) < makespan(a', c')$.

Case 2: makespan(a, c) = makespan(a', c')

Let us analyze the case in which the makespan remains unchanged. Our algorithm will find an allocation a' with respect to c'.
We have three main cases:

- Case 1: a' comes before a in the set of allocations
  If $a' < a$ then we can apply the same configuration a' also to the initial costs combination c. Since no cost will increase, the $makespan(a', c)$ can only be smaller than $makespan(a, c)$, but this is absurd, since if this was the case, our algorithm would have picked the solution a' also before when the player i was declaring a cost of c_i.
- Case 2: a' is in the same position of a
  As a consequence we would have $a = a'$. But this is absurd since we assumed that in a' the workload given to the player i has increased and, thus, that $a \neq a'$.
- Case 3: a' comes after a in the set of allocations
  We know now that $makespan(a, c') > makespan(a', c')$, otherwise I would have chosen the solution a, which is lexicographically smaller. Thus, $makespan(a, c') > makespan(a, c) = makespan(a', c')$. But from c to c' only the cost of the player i has changed and, since it made the makespan grow keeping the same allocation as before, it must hold that $c'w > makespan(a, c)$. Moreover, from the definition of makespan we can write that $makespan(a', c') \geq c'_i w'_i$.
  Rearranging the terms derived from the previous two observations and knowing that $w'_i > w_i$ we obtain that

$$makespan(a', c') \geq c'_i w'_i > c'_i w_i > makespan(a, c)$$

  But this is absurd because we assumed that $makespan(a, c) = makespan(a', c')$

In conclusion, it is not possible to find any allocation a' which maintains the makespan unchanged incrementing the workload of the player i, the only one who have increased his declared cost.

Case 3: makespan(a, c) > makespan(a', c')

Suppose that the maxspan is allowed to decrease through the allocation a' with regard to the costs c'. Since all the costs of the machines have remained unchanged or have grown (in the case of the machine i), if we apply the allocation a' with regard to c, we would obtain a solution with a $makespan(a', c) \leq makespan(a', c')$. Since we are assuming that $makespan(a', c') < makespan(a, c)$, we obtain:

$$makespan(a', c) \leq makespan(a', c') < makespan(a, c)$$

Thus, there exists an allocation (with regard to the cost c) that is different from a and guarantees a lower makespan. But this is absurd since $a' \neq a$ ($w'_i > w_i$) the solution initially chosen by A. Thus, after a change of the cost of the machine i, it is impossible that our algorithm chooses an allocation a' in which $w'_i > w_i$ and $makespan(a, c) > makespan(a', c')$.

After having analyzed all the possible situation we can conclude that after the increment of the cost of a player i, my algorithm will never choose any allocation a' that would increase the workload of i. Thus, my algorithm is monotone and there exists a truthful mechanism for the problem of scheduling selfish related machines which minimizes the makespan or maximum cost.

Part 2

Our task is to show that there is no truthful mechanism (**greedy**, P) for the problem described. In order to do that we will first prove that the algorithm proposed is not monotone.
Then, using the Myerson Lemma we will be allowed to say that A is not truthful, regardless of how we choose to define the payment P.

Let us consider a game with two machines, let us call these machines 1 and 2. Suppose that machine 1 has a true cost $c_1 = 7$ while the machine 2 has a true cost of $c_2 = 8$. Consider the following queue of jobs that must be allocated to the machines: $J_1 = 6, J_2 = 4, J_3 = 3$. They would be processed in this order, since the algorithm proposed process the jobs one by one in decreasing order of size.
In particular, the algorithm would make the following choices:

- J_1 will be assigned to the first machine

$$c_1 \cdot (J_1) = 7 \cdot 6 = 42 < 48 = 8 \cdot 6 = c_2 \cdot (J_1)$$

- J_2 will be assigned to the second machine

$$c_1 \cdot (J_1 + J_2) = 7 \cdot 10 = 70 > 32 = 8 \cdot 4 = c_2 \cdot (J_2)$$

- J_3 will be assigned to the second machine

$$c_1 \cdot (J_1 + J_3) = 7 \cdot 9 = 63 > 56 = 8 \cdot 7 = c_2 \cdot (J_2 + J_3)$$

Thus, we obtain $w_1 = J_1 = 6$ and $w_2 = J_2 + J_3 = 7$.
Suppose now that the first machine cheats and declare a cost $c_1' = 9$.
Let us analyze again the choices that would be made by the greedy algorithm.

- J_1 will be assigned to the second machine

$$c_1' \cdot (J_1) = 9 \cdot 6 = 54 > 48 = 8 \cdot 6 = c_2 \cdot (J_1)$$

- J_2 will be assigned to the first machine

$$c_1' \cdot (J_2) = 9 \cdot 4 = 36 < 80 = 8 \cdot 10 = c_2 \cdot (J_1 + J_2)$$

- J_3 will be assigned to the first machine

$$c_1' \cdot (J_2 + J_3) = 9 \cdot 7 = 63 < 72 = 8 \cdot 9 = c_2 \cdot (J_1 + J_3)$$

Thus, we would have $w_1' = J_2 + J_3 = 7$ and $w_2' = J_1 = 6$.
Hence, since for a $c_1' > c_1$ we have obtained a $w_1' > w_1$ we have proved that the greedy algorithm A is not monotone and, thus, for the Myerson Lemma, that (A, P) is not truthful, no matter P.