Game Theory and Control - Assignment 3

Pagani Leonardo

June 2022

1 Karma Game

a)
Assuming that the bids are uniformly distributed in the interval [0,20], we can compute the probability of yielding when bidding b_i. When bidding b_i, we lose the bid (and therefore yield) if the other player bids higher than us or if we lose the coin toss to break the tie. Let us define b_{opp}, the bid of the other player at the intersection. We have assumed that b_{opp} is a random variable following a uniform distribution in the interval [0,20]. Therefore:

$$P(yielding \mid b_i) = P(b_i < b_{opp}) + P(b_i = b_{opp}) * P(losing - coin - toss) = \frac{20-b_i}{21} + \frac{1}{21} * \frac{1}{2} = \frac{41-2b_i}{42} \quad (1)$$

Since we are now assuming that the game is ending after one intersection, there is no point at all in saving the karma, even if our urgency is LOW. Therefore the optimal bidding strategy in this case is to bid all the initial karma $k_i = 10$. This applies both when our urgency is HIGH and LOW.

We can compute the expected cost E(J) by randomizing over the opponent bids and over the urgency. In particular, if we define as P(HIGH) and P(LOW) the probabilities of experiencing high and low urgency at the intersection, we get:

$E(J) = P(yielding \mid 10)[(P(HIGH)*J(yielding, HIGH)+P(LOW)*J(yielding, LOW)] =$

$\frac{1}{2}[0.4 * 10 + 0.6 * 1] =$

2.3

b) Even with a multiple stage game, the optimal strategy at the last (20th) stage is to bid all the available karma. Therefore, if ${k_i}^{20}$ is the karma available to Player i at stage 20:

$\mathrm{b_i}^{*(20,\mathrm{HIGH})} = \mathrm{b_i}^{*(20,\mathrm{LOW})} = {k_i}^{20}$

Again it makes no sense for each Player to try to save karma as it is the last stage of the game. This result applies in both cases of HIGH urgency and LOW urgency.

c) We have already established that at the last stage of the game, the best strategy for each Player is to bet all the remaining karma ${k_i}^{20}$. We can therefore compute the value function $V^{(20)}(\mathrm{k_i})$ as:

$V^{(20)}(k_\mathrm{i}) =$

$P(yielding \mid k_\mathrm{i}) * [(P(HIGH) * J(yielding, HIGH) + P(LOW) * J(yielding, LOW)] =$

$\frac{41-2k_i}{42} * 4.6$

d) We can now study the transition probabilities from one state to another. As suggested in the handout we might distinguish the two cases where $k_i + b_i < k_{max}$ and where $k_i + b_i >= k_{max}$.

Let us start from the simple case where $k_i + b_i >= k_{max}$.This means that when we lose the bid (yield), our karma is capped at k_{max} all the time. In particular, we can write:

$\mathrm{P}({k^+}_i \mid k_i b_i) =$

$\frac{1+2b_i}{42} \quad for\ {k^+}_i = k_i - b_i$

$\frac{41-2b_i}{42} \quad for\ {k^+}_i = k_{max}$
(2)

Basically, in this case, the karma is capped to k_{max} if the player yields, whereas the player pays b_i if he wins.

Let us now consider the case where $k_i + b_i < k_{max}$. This case is a little bit more complicated as the karma may end up in multiple states in case the Player yields. The transition probabilities are computed as follows:

$\mathrm{P}(k^{+}{}_{i}|\ k_i b_i)\ =$

$\frac{1+2b_i}{42} \qquad for\ k^{+}{}_{i} = k_i - b_i$

$\frac{1}{42} \qquad for\ k^{+}{}_{i} = k_i + b_i$

$\frac{1}{21} \qquad for\ k^{+}{}_{i} \in [k_i + b_i + 1, ..., k_{max} - 1]$

$\frac{k_i+1}{21} \qquad for\ k^{+}{}_{i} = k_{max}$

$if\ b_i \neq 0$

(3)

The last probability is simply obtained by setting the sum of all probabilities to 1.

$\mathrm{P}(k^{+}{}_{i}|\ k_i b_i)\ =$

$\frac{1}{21} \qquad for\ k^{+}{}_{i} = k_i$

$\frac{1}{21} \qquad for\ k^{+}{}_{i} \in [k_i + b_i + 1, ..., k_{max} - 1]$

$\frac{k_i+1}{21} \qquad for\ k^{+}{}_{i} = k_{max}$

$if\ b_i = 0$

(4)

e)
To perform backward induction, we have to first make a couple of considerations. First, the state of the system is represented not only by the karma level of the player but also by its urgency. This is due to the fact that at every stage, each player knows whether its urgency is HIGH or LOW. Thus, we have to distinguish among the states of HIGH urgency and LOW urgency.

To perform the backward induction step we have to solve the following optimization problems.

$$V^{(\mathrm{k})}(k_i, H) = \min_{b_\mathrm{i} \in [0, k_\mathrm{i}]} 10 * \frac{41 - 2b_i}{42} + \sum_{{k^+}_{i=0}}^{20} P({k^+}_i|\ k_i b_i)[0.4 * V^{(k\ +\ 1)}({k^+}_i, H) + 0.6 * V^{(k\ +\ 1)}({k^+}_i, L)] (5)$$

$$V^{(\mathrm{k})}(k_i, L) = \min_{b_\mathrm{i} \in [0, k_\mathrm{i}]} 1 * \frac{41 - 2b_i}{42} + \sum_{{k^+}_{i=0}}^{20} P({k^+}_i|\ k_i b_i)[0.4 * V^{(k\ +\ 1)}({k^+}_i, H) + 0.6 * V^{(k\ +\ 1)}({k^+}_i, L)] (6)$$

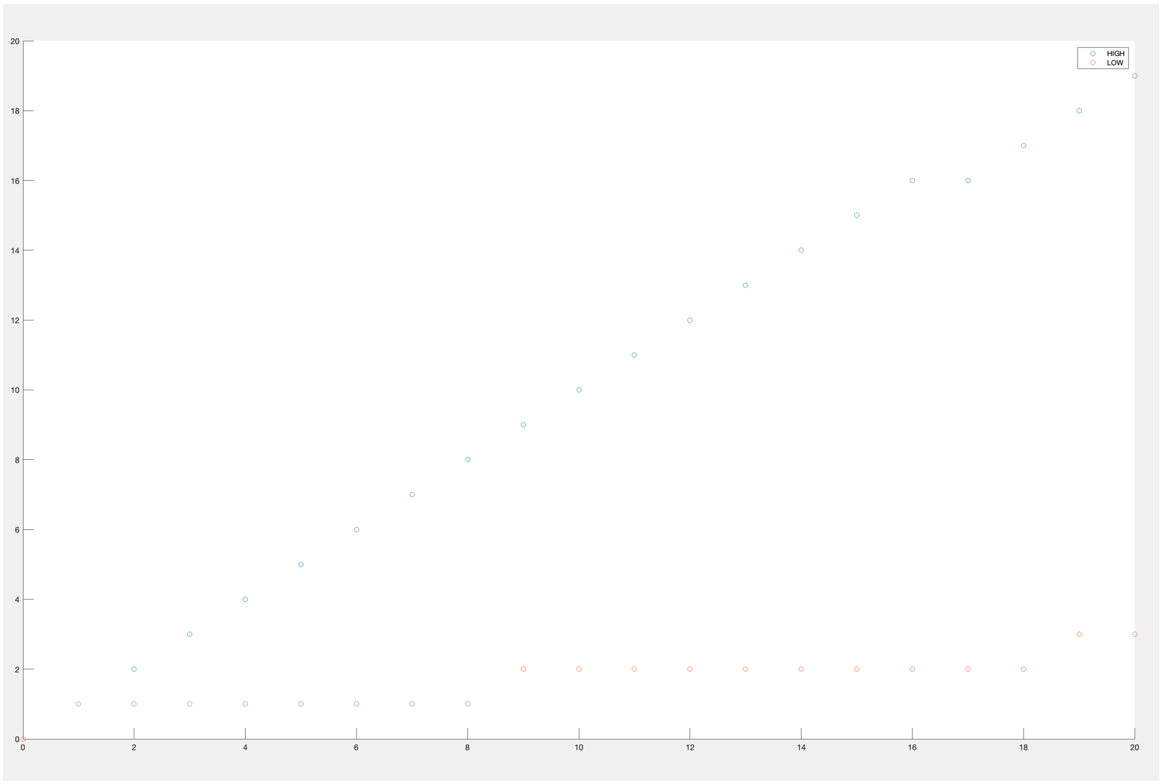

Figure 1: In blue the best bidding strategy for HIGH urgency at stage 1, while in orange the optimal bidding strategy for LOW urgency.

f) After having written the code to perform backward induction, we obtain these results (Figure1) for the optimal bidding strategies at the first stage. The sanity check is also satisfied.

g) We can now perform the simulation suggested in the assignment with the players adopting the optimal strategy computed in point f). If we plot the distribution of the bids in the last 100 iterations, when the setting time is over, we obtain the following histogram, in Figure 2.

We notice that the distribution does not resemble at all a uniform distribution. Instead, the bids are skewed towards the lower end of the [0,20] interval. This is due to the fact that, 60% of the time, players experience low urgency and thus bid a very low amount of karma, according to the optimal strategy in f)

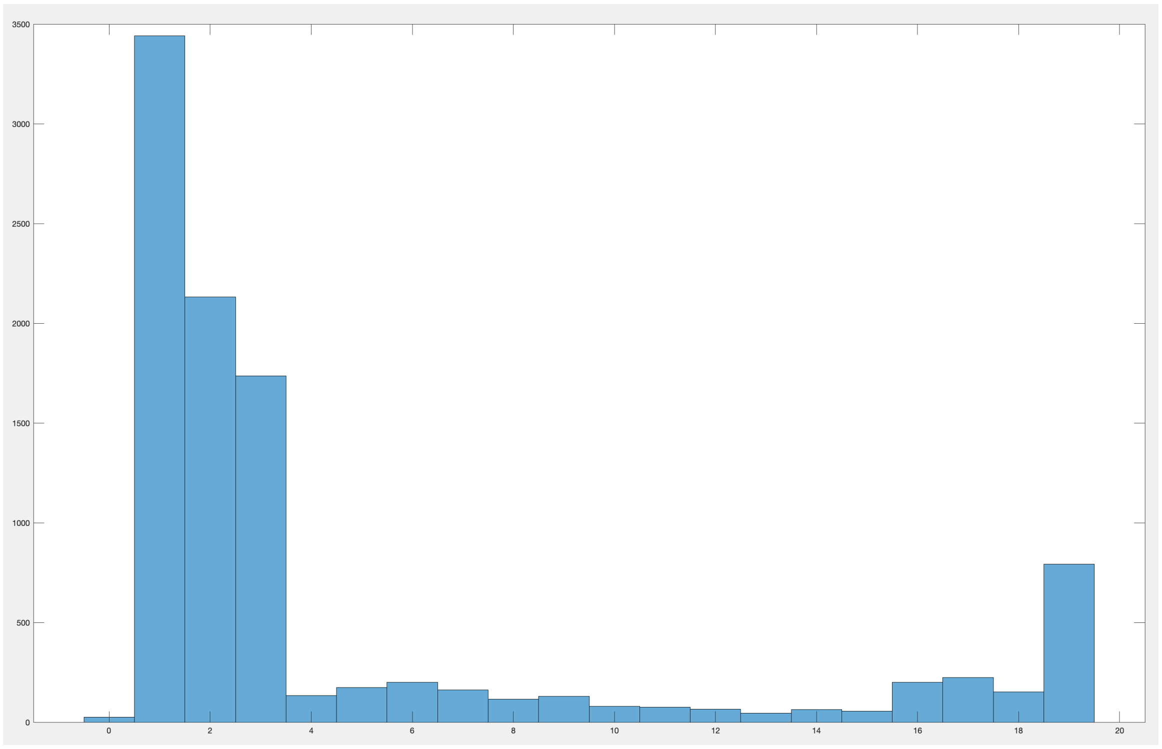

Figure 2: An histogram capturing the distribution of the bids in the last 100 iterations.

h) We can compute the lowest possible social cost assuming a benevolent dictator decides which car yields. Let us denote with P1 and P2 two given players at a given intersection. Four possible situations can arise:

1. Both P1 and P2 have LOW urgency $->$ Benevolent dictator lets either P1 or P2 go first $-->$ Cost 1
2. P1 has LOW urgency while P2 has HIGH urgency $-->$ Benevolent dictator lets P2 go first $-->$ Cost 1
3. P1 has HIGH urgency while P2 has LOW urgency $-->$ Benevolent dictator lets P1 go first $-->$ Cost 1
4. Both P1 and P2 have HIGH urgency $-->$ Benevolent dictator lets either P1 or P2 go first $-->$ Cost 10

We can easily compute the optimal social cost by randomizing over this 4 possible scenarios. If we have N = 100 cars, that means that at every stage *t*, the dictator has to take a decision in 50 different intersection. Each of the 50 different intersections features one of the 4 possible scenarios. In particular, since the urgency of a player at each stage is an independent and identically distributed (i.i.d.) process, we have that:

1. Scenario 1 happens with probability 0.6 * 0.6 = 0.36 and yields a cost of 1
2. Scenario 2 happens with probability 0.6 * 0.4 = 0.24 and yields a cost of 1
3. Scenario 3 happens with probability 0.4 * 0.6 = 0.24 and yields a cost of 1
4. Scenario 4 happens with probability 0.4 * 0.4 = 0.16 and yields a cost of 10

Therefore we can compute the lowest possible social cost SC, as defined in the assignment as:

$SC = \sum_{i=1}^{50} \frac{0.36*1+0.24*1+0.24*1+0.16*10}{100} = 50 * 0.0244 = 1.22$

We can compute an estimate of the social cost in our simulation. The value of this estimate changes for different simulations as there are stochastic processes involved. Generally speaking, we observe that the social cost for our simulation is belongs to the interval [1.45, 1.52], which is considerably higher than the optimal one.

Finally, we can compute, again with Matlab, the highest and the lowest cumulative cost for a player in the last 100 simulations. Again, we find out that

these values are not deterministic, due to the intrinsic randomness of the simulation.
We observe that the highest cumulative cost lies in the interval [200,240], while the lowest cumulative cost lies in the interval [70,90]. This means that in our simulations some players end up in a more favourable situation than others, as they can achieve a lower individual cost.

i) An idea to compute the Nash Equilibrium bidding strategy would be to adopt an iterative algorithm, constituted by two main steps: the estimation of the distribution of bids and the computation of the optimal policy.

1. STEP 0: Hypothesis of an initial distribution of bids (uniform distribution) and computation of the optimal bidding strategy for this hypothetical distribution.

2. STEP 1: Estimation of the new distribution of bids when bidding accordingly to the previously computed optimal policy.

3. STEP 2: If the new distribution differs from the previous one, we compute a new optimal policy for the new distribution of bids.

Repeat STEP 1 and STEP 2 until the new distribution converges to the previous one.

This iterative algorithm alternates between the estimation of the bids' distribution and the computation of the optimal policy. It stops when the optimal policy yields the same distribution upon which the policy itself is optimized on.

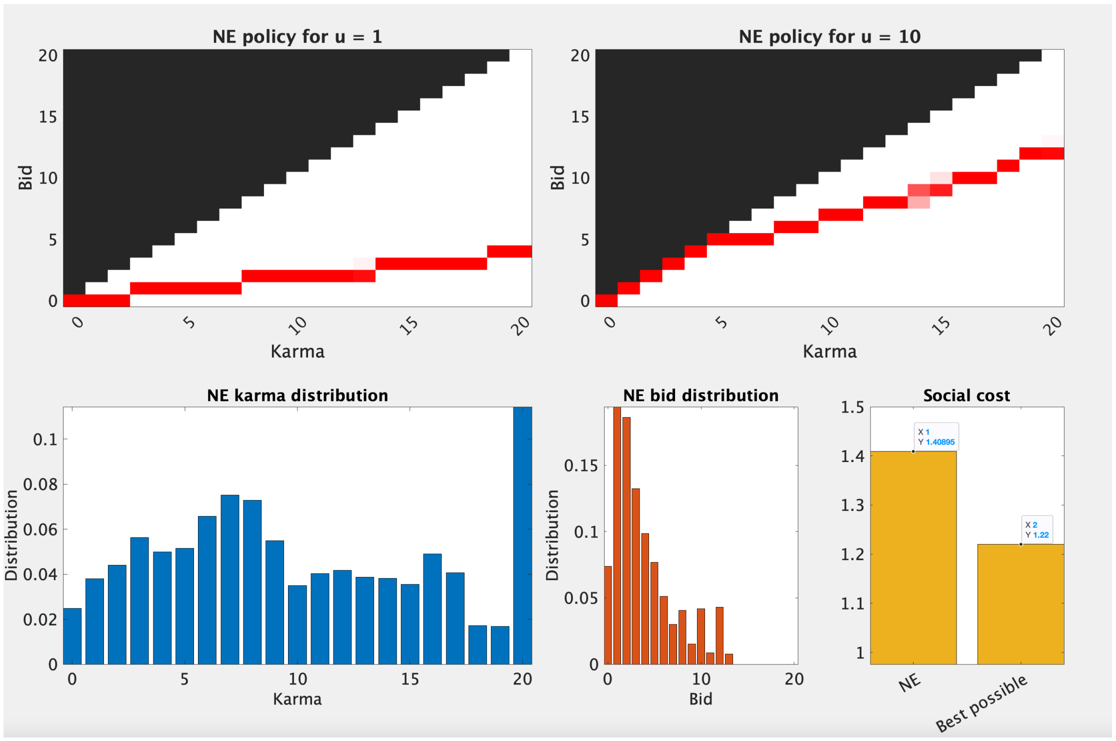

Figure 3: Output of the provided script karma-main.m

BONUS PART I)

If we run the karma-main.m script, we can observe the NE bidding strategy as well as the achieved social cost (Figure 3). We notice that the optimal strategies for HIGH and LOW urgency are different from those we computed in point f). In particular, in case of HIGH urgency, it is recommended to bid all the available karma $\mathrm{k_i}$ for $k_\mathrm{i} <= 5$. For higher values of the karma, however, the best bid in case of high urgency tend to decrease if compared to the strategy computed in f). The idea here is that, in order to win the bid, it is not necessary to bid all the available karma, but bidding less is also fine. This is confirmed by the bid distribution where we notice that there are very few bids greater than 13. Finally, the NE social cost is around 1.40, slightly less than our computed estimate of the social cost in h). This is not a surprising result as the algorithm proposed in the script performs a better optimization of the NE bidding policy.

BONUS PART II)

The idea for this section is to improve the protocol via the get-payments and the get-karma-tax function, possibly achieving a lower social cost. The first idea is to change the get-payments function. After several simulation, it was clear that it is better (in terms of all social cost) to redistribute the winner bid to all the players instead of giving it to the player who yields. Moreover, another idea is to reward the players bidding when their urgency is HIGH and punish them when they urgency is LOW. Therefore my final get-payments-function looks as follows:

```
$\mathrm{p_{win}} = \mathrm{b_{win}};$
$\mathrm{p_{yield}} = 0;$

if($b_{\mathrm{win}} > 7$)
        $p_{\mathrm{win}} = floor(b_{\mathrm{win}}/2);$
```

Of course there is no way to know if a player urgency is HIGH or LOW, but we know that, according to the optimal policy, a player commits to an high bid only when he has HIGH urgency. In this way, we only make the high bidder pay half of what he has bidden. (His high bid is justified by the fact that his urgency is probably HIGH).

The risk of this policy is to encourage players who have a lot of karma at disposal to bid a lot even with LOW urgency (see Figure 4), because they actually pay only half of what they bid. So, to further improve the protocol, we implemented a linear 10% tax on the karma to make sure very few players end up with a lot of karma at their disposal. The results of these implementations are quite encouraging, with the social cost lowered at 1.23 (with the optimal value being 1.22). The achieved distribution of karma levels is promising. Ideally we want the distribution centered in the middle of the interval [0,20], with very low variance. In fact, if everybody all the time had a karma value between 8 and 12 for example, this would guarantee that HIGH urgency player would always win because they would always have enough money to bid, while LOW urgency player would accept to yield, according to the optimal policy.

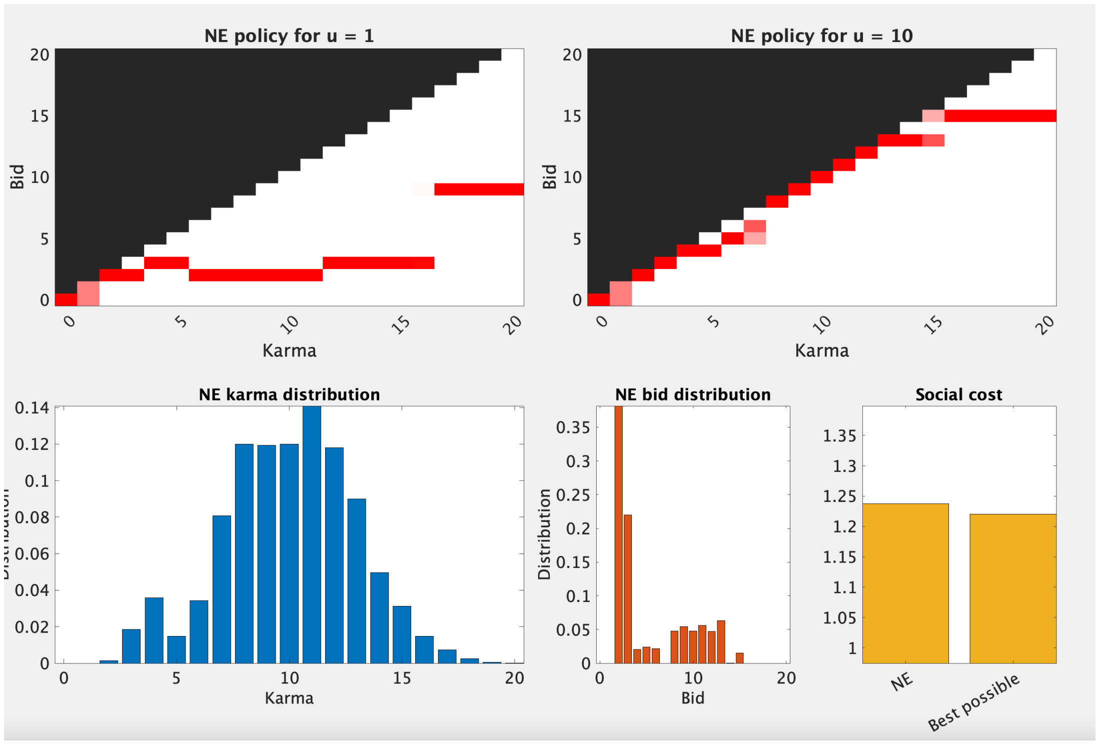

Figure 4: Output of the provided script with the outlined protocol changes