Game Theory & Control
Assignment 3

S. Bolognani

28 May 2022

Please submit your solutions on Moodle.

You can either type them (in LaTeX, for example), or scan your handwritten solutions.

If you are scanning your handwritten solutions, please create a PDF that is readable and small in size. You can use any smartphone app that allows to scan documents to PDF (this feature is present in both the Dropbox app and the Google Drive app, plus many others). Do not send plain photos of the pages.

The deadline for this assignment is 19 June 2022, at 23:59.

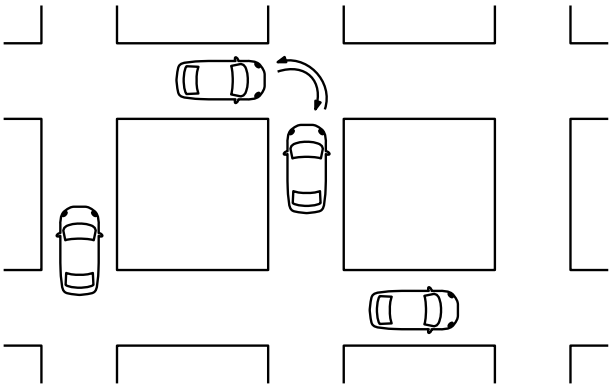
Please write your solutions in a legible form, include all the steps of your reasoning, indicate what question your are answering, and don't include any solution attempt that you don't want us to grade.

Verify your answers by trying to get to the same results in different ways, checking if they are compatible with the theoretical results that you know and with your intuition. Typos and errors by distraction are not acceptable.

This assignment contributes towards 10% of the final score for the course, and must be done independently by each student. No group work is allowed.

Each "letter" question is worth 1 point, unless stated otherwise.
Make sure you answer everything to get the full point for each question. No fraction of points are awarded.
Take your time to double-check that you have not missed any part of each question.

Exercise 1 Karma Game



Consider the problem of regulating traffic intersections. We focus on the scenario where pairs of cars arrive at the same time at an intersection. A mechanism is needed in order to decide which of the two cars goes first, and which one yields. This mechanism needs to be used at every intersection, every time a pair of cars arrive simultaneously.

To study the problem via game theory, we consider each car a player, which has a preference for going first. More precisely, yielding comes with a cost that depends on whether the driver is in a rush or not. For simplicity, we assume that the *urgency* of a player at each intersection is an independent and identically distributed (i.i.d.) process. The urgency is HIGH with probability $P_H = 40\%$ and is LOW with probability $P_L = 60\%$. The level of urgency is a private piece of information.

Yielding when urgency is HIGH has a cost of 10, yielding when urgency is LOW has a cost of 1. The total cost for a player is the cost at each intersection, summed over time.

Consider the following **Karma protocol**:

- each car is associated to a *karma* counter k_i: a non-negative integer quantity of "karma points", which have no monetary value and cannot be exchanged for money;
- when the game starts, each car has 10 karma points;
- when two cars i and j meet, they *bid* an integer amount of karma points (b_i and b_j), between 0 and their currently available karma budget;
- when they bid, each car knows their own karma and urgency, but does not know the karma and urgency of the other car;
- the car that bids the highest bid (denoted b_{win}) goes first, and that amount of karma is transfered to the other car;
- in case of identical bids, the winner is selected with a fair coin toss and the bid is transfered to the car yielding;
- each car's karma counter saturates at $k_{\max} = 20$, i.e., if the yielding car's karma exceeds $k_{\max}$ after the transfer, it is capped at $k_{\max}$ and the excess is distributed to the entire population (see later for an example of how this can be done in practice).

You (player) enter the game in a population of cars that are already playing. You don't know the other players' strategies, but you assume that their bids are *uniformly distributed* between 0 and 20 (included).

First, consider only one intersection (assume that you are going to leave the game right after that).

a) Answer the following questions to compute the optimal bidding strategy in this simple case.
   - What is the probability of yielding, as a function of your bid b_i?

- What is your optimal bidding when your urgency is HIGH? And when your urgency is LOW?
- What is your expected cost?

Now consider the case in which you are going to go through multiple intersections. More precisely, we consider the case in which you want to play strategically with respect to the cost over the next 20 intersections. We consider this an approximation for the infinite horizon case.

Find the optimal strategy by backward induction by answering the following questions.

When answering these questions – from b) to f) – ignore the possibility for an agent of receiving karma form a redistribution of excess karma from cars in other intersections.

b) Find the optimal bidding strategy for the last (20th) stage. Is it a function of the karma counter at that stage?

c) Give an expression for the value function $V^{(20)}(k_i)$ at the last stage, i.e. the expected cost of the last stage of the game when the player arrives there with karma $k_i^{(20)} \in \{0, \dots, 20\}$ and bids optimally.

d) Give an expression for the probability of transitioning to karma level k_i^+ after an intersection, given that the player's current karma is k_i and it bids b_i, i.e.,

$$\mathbb{P}[k_i^+ \mid k_i, b_i].$$

Hint. *It may be easier to consider the cases when* $k_i + b_i < k_{\max}$ *and when* $k_i + b_i \geq k_{\max}$ *separately.*

e) Write down the minimization problem that you need to perform the backward induction step, i.e. to compute the value function $V^{(t)}$ from $V^{(t+1)}$.

f) (2 points) Compute numerically what is your optimal bidding strategy at the first stage (for any value of initial karma and for the two different urgencies). Plot the bidding strategy. **You also need to submit your code (a zip archive if it's made of multiple files).**

Hint. *As a sanity check, you should verify that at the 19th stage with* $k = 10$ *you have*

$$b^{(19),HIGH*} = 10, \qquad b^{(19),LOW*} = 1.$$

We are now going to simulate the effect of the strategy that you just computed. More precisely:

- we have a population of $N = 100$ cars, which start with the same karma level 10 at the beginning;
- at every step in time, cars are randomly paired;
- all cars bid according to the optimal bidding strategy that you derived in step f) (the one at the first step, repeatedly);
- the protocol is implemented as described at the beginning;
- all the interactions between the $N/2$ pairs happen at the same time;
- every time, the karma of all agents is capped at $k_{\max}$ and the excess is redistributed; to do that, you can use the following function (in Matlab) or implement something similar.

```
function newk = capAndRedistribute(k, maxvalue)

        % capAndRedistribute receives a vector of length N with the karma of
        % all players, caps the karma at maxvalue, and redistributes the
        % excess to the other players. The function returns the resulting
        % vector of karma counters.

        newk = min(k,maxvalue);
```

```
excess = sum(k) - sum(newk);
while excess > 0
        recipients = find(newk < maxvalue);
        if excess >= length(recipients)
                newk(recipients) = newk(recipients) + 1;
                excess = excess - length(recipients);
        else
                recipients = recipients(randperm(length(recipients)));
                newk(recipients(1:excess)) = newk(recipients(1:excess)) + 1;
                excess = 0;
        end
end
```

Answer the following questions.

g) Let the simulation run for 1000 iterations and consider the last 100 iterations, so that the distribution of bids can be considered stationary. How does the distribution of bids look like, across the entire population? Does it correspond to the assumption that we used to compute the optimal bidding strategy?

h) Let the *social cost* be defined as

$$SC := \lim_{T\to\infty} \frac{1}{T}\,\mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{N}\frac{c_i(t)}{N}\right],$$

where $c_i(t)$ is the cost incurred by car i at time t. This expresses the long run expected average cost per interaction.

- What is the lowest possible social cost that could be achieved (assuming that all urgency information is public and that a benevolent dictator can decide which car must yield)?
- Provide an empirical approximation of the social cost under the optimal bidding strategy that you derived by considering the stationary regime of the simulation in g) (the last 100 iterations). How does it compare to the lowest possible social cost?
- In those last 100 iterations, what is the highest cumulative cost incurred by a player? What is the lowest?

i) How would you compute a Nash Equilibrium bidding strategy, based on the tools that you have derived so far? Describe in words the procedure that you would use. A rigorous proof of the correctness of your approach is not needed.

4 BONUS POINTS

On Moodle you can find the code that allows to compute the Nash equilibrium bidding strategy for this problem. The code solves the more complicated infinite-horizon case, which however is quite similar to your 20-stage approximation. The provided script `karma_main.m` computes and outputs:

- the Nash equilibrium bidding strategy for HIGH and LOW urgency levels;
- the stationary distribution of karma states;
- the stationary distribution of bids, and;
- the social cost achieved at the Nash equilibrium, compared to the lowest possible social cost.

You can get **1 bonus point** if you run `karma_main.m`, report the Nash equilibrium bidding strategy, and compare it to the bidding strategy derived in f), in terms of the bidding behavior and of achieved social cost.

Moreover, the code allows to alter the karma protocol by modifying the following two functions:

- `get_payments`: implements the rule by which karma is transferred. In the presented protocol the bid of the winning player is transferred to the yielding player. The function allows to encode more general rules, where two payments are collected (usually a non-positive one for the yielding player) and the surplus is redistributed uniformly to all the players;
- `get_karma_tax`: implements a "karma tax" rule, which allows to collect a karma-dependent tax at every iteration and to redistribute it uniformly to all the players.

Refer to the function comments for more details. You should think of different karma protocol designs (payment and tax rules). To assess a protocol, implement it in `get_payments` and `get_karma_tax` functions and run the script `karma_main.m` (do not modify the script). You can also inspect the computed Nash equilibrium bidding strategy (stored in variable `pi_down_u_k_up_b`). Then select a design that you consider a "good" protocol. If you explain two sensible reasons why you think that your design is a good one, you will get **1 bonus point**. Please keep your explanation shorter than half a page and include plots/charts/tables if you think that they can support your case. Notice that it is up to you to define what makes a protocol "good".

Finally, the 5 students that present the most convincing karma protocols (according to the course TAs) will get **2 additional bonus points**.