Networks Dynamics and Learning: Homework 1

Valerio Zingarelli
Politecnico di Torino
Student id: s281586
s281586@studenti.polito.it

***Abstract*—In this report I will explain and explore the methods I adopted to solve proposed exercises and the results I got. The essay will be divided into sections, each one corresponding to one of the three assigned tasks. During the resolution of the exercises, Vito Palmisano and I have compared our results and provided advice to each other, especially in the third exercise.**

I. Exercise 1

The first task of the assigned homework is an application of the min-cut theorem. More specifically, given the graph in Fig.1, with capacity C_e on the link e, several tasks required to be solved. It's important to precise that all the points of this exercise has been solved by hand therefore, in order to not weight the essay, only the reasonings, with opportune examples, will be provided instead of all the pictures of the passages.

A. Ex. 1a

In the first point of exercise 1, we were required to find the inf of total capacity that needs to be removed to disconnect o and d (0 and 3 in the figure). To do that, we need to compute all the cuts and apply the min-cut theorem. Let's compute all the cuts:

- U_1 = 0 $\implies$ C$_{U1}$ = C_1+C_2
- U_2 = 0,1 $\implies$ C$_{U2}$ = C_2+C_3+C_4
- U_3 = 0,2 $\implies$ C$_{U3}$ = C_1+C_3+C_5
- U_4 = 0,1,2 $\implies$ C$_{U4}$ = C_4+C_5

As min-cut theorem states, the minimal capacity along the different cuts is equal to maximum τ that can be send from the source 0 to the sink 3. Saying that we want to disconnect 0 and 3, it's the same of saying τ_{max}=0. So, infimum of the total capacity that needs to be removed for no feasible unitary flows from 0 to 3 to exist is:

$$inf = min(C_1 + C_2; C_2 + C_3 + C_4; C_1 + C_3 + C_5; C_4 + C_5) \quad (1)$$

B. Ex 2b

From now on, the value of the capacities will be considered as following:

$$C_1 = C_4 = 3, \quad C_2 = C_3 = C_5 = 2 \quad (2)$$

In the second point of the first task, we were asked to add a unit of additional capacity to one edge and see what happens to the throughput. Calculating the capacity of the cuts, we get:

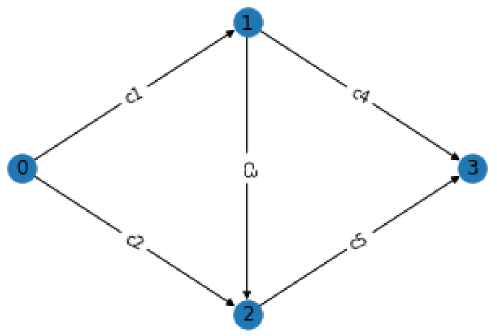

Fig. 1. Graph of exercise 1

- C_{U1} = C_1+C_2 = 5
- C_{U2} = C_2+C_3+C_4 = 7
- C_{U3} = C_1+C_3+C_5 = 7
- C_{U4} = C_4+C_5 = 5

So τ_{max}=5
It's easy to notice that wherever the additional unit of capacity is added, there will always be a cut with capacity not lower then 5.

C. Ex 1c

Now, the additional capacity points that should be added are 2. Since C_3 is not present in any min-cut, no points should be added to it. The possible combination of point assignments that maximize the troughput are the following:

- C_1 $\implies$ C$_1$+1 and C_4 $\implies$ C$_4$+1
- C_1 $\implies$ C$_1$+1 and C_5 $\implies$ C$_5$+1
- C_2 $\implies$ C$_2$+1 and C_5 $\implies$ C$_5$+1

All the possible combinations have been tested. Each time, the min-cut has been computed and the solutions above are the ones that obtained the maximum τ. To be more precise:

$$\tau_{max} = 6 \quad (3)$$

D. Ex 1d

Same reasonings as before have been used again in this point. The obtained combination are the following:

- C_1 $\implies$ C$_1$+2 and C_4 $\implies$ C$_4$+2
- C_1 $\implies$ C$_1$+2 and C_5 $\implies$ C$_5$+1 and C_4 $\implies$ C$_4$+1
- C_1 $\implies$ C$_2$+2 and C_5 $\implies$ C$_5$+2

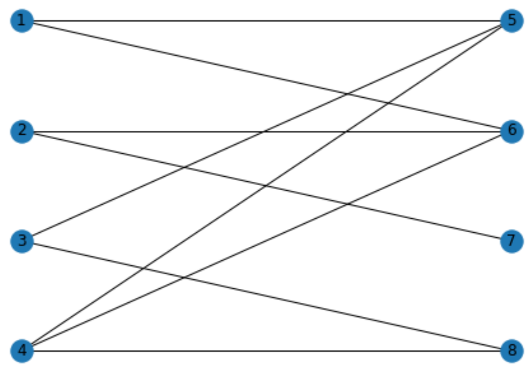

Fig. 2. Graph modelling the situation described in the second exercise

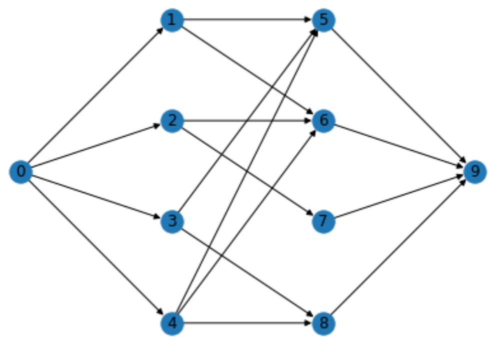

Fig. 3. Graph constructed to solve the perfect matching problem in ex.2

- $C_1 \implies$ C$_1$+1 ; $C_2 \implies$ C$_2$+1 ; $C_4 \implies$ C$_4$+1 and $C_5 \implies$ C$_5$+1
- $C_4 \implies$ C$_1$+1 ; $C_2 \implies$ C$_2$+1 and $C_5 \implies$ C$_5$+2
- $C_2 \implies$ C$_2$+2 and $C \implies$ C$_5$+2

In this last case, the maximal obtained throughput is:

$$\tau_{max} = 7 \tag{4}$$

II. EXERCISE 2

In this exercise, we have a set of people (p1,p2,p3,p4) and a set of books (b1,b2,b3,b4). Each person is interested in one or more books according to the following:

$$p1 \to (b1, b2), p2 \to (b2, b3), p3(b1, b4), p4 \to (b1, b2, b4) \tag{5}$$

A. EX 2a

It's possible to model situations like that using a bipartite graph with V_0=people and V_1=books. Of course,the edges represent the preferences. Fig.2 offers a representation of the problem.

B. Ex 2b

In this point, we were required to find a perfect matching between people and books. In other words, what we had to do is to make all the people happy with one book. As it has been shown in the code, a greedy approach (e.g. assign to each person the first "free" book in his/her preference list) has been tested and it's failed. So, we have to solve this problem using an analogy with max-flow problems. In particular, a source 0 and a sink 9 have been added to the graph as it's shown in Fig.3.
Then, the maximum flow between the source and the sink has been computed and the perfect matching has been deduced. For implementations details, I refer to the code.

$$matching = (p1, b2), \quad (p2, b3), \quad (p3, b1), \quad (p4, b4) \tag{6}$$

C. Ex 2c

Now, there are some more copies of the books. To be more precise, the copies are: (2,3,2,2). Since now people can have more then one book, we have to set the capacities of the edges between the source and the people to infinity. In addition to that, to model the fact we have more than one copy of each book, we need to modify the capacities of the edges between the books and the sink. Exploiting the same max-flow problem as before, we obtain the following matching:

(p1, b2), (p2, b2), (p2, b3), (p3, b1), (p3, b4), (p4, b4), (p4, b2), (p4, b1)

It's easy to notice that, with this distribution of books, there is one book left and so, as a consequence, not all the books are sold.

D. Ex 2d

The only book that is not sold is b3 and, in addition to that, three people are interested in b1 but only two of them can have a copy. So, the best solution for the library is to sell a copy of b3 and buy a copy of b1. As it has been shown in the code, with this solution, all the books are sold and all the people have all the books they are interested in.

III. EX 3

In this exercise, a more complicated graph has been provided. In fact, we have to deal with a simplified version of the highway network in Los Angeles. During the resolution of the proposed tasks, built in functions of networkX and CVXPY have been used.

A. Ex 3a

From the incidence matrix B, the graph has been created using networkX. Then, using the built-in function of networkX, the shortest path between given nodes has been found. The shortest path is provided as the list of successive nodes:

$$shortestPath = (1, 2, 3, 9, 13, 17) \tag{7}$$

B. Ex 3b

Using the built-in function of networkX, the maximum flow has been found:

$$maxFlow = 22448 \tag{8}$$

C. Ex 3c

In this point, we were required to find $\nu = Bf$

$$\nu = \begin{bmatrix} 16806 & ,..., & -23544 \end{bmatrix} \tag{9}$$

D. Ex 3d

In the following, a new vector ν_d will be used. This new ν_d is composed by all 0s, except for the first and the last element that are equal to ν[0] and -ν[0], respectively. We are required to find the social optimum f^* and, to do that, we need to optimize the following function:

$$\sum_{e \in \mathscr{E}} f_e d_e(f_e) = \sum_{eE} (\frac{l_e C_e}{1 - \frac{f_e}{C_e}} - l_e C_e) \tag{10}$$

The main idea behind all the procedures that will be used to solve the following tasks is that, given two vectors $v \in \mathrm{R}^n$ and $w \in \mathrm{R}^n$:

$$v @ w.T = \begin{bmatrix} v_1 w_1 & & \\ & \ddots & \\ & & v_n w_n \end{bmatrix} \tag{11}$$

It's possible to notice that, on the diagonal of that matrix we have the product element-wise of the two initial vectors. So, it's possible to use this property to compute all the products we need to obtain the objective function in cvxpy.
To do that, all opportune computations with matrices and vectors have been performed, as it's written and commented in the code.
The obtained social optimum is the following:

$$f^* = [6.642300e + 03; ...; 4.886435e + 03] \tag{12}$$

I refer to the code for the complete vector.

E. Ex 3e

Now, we were required to compute the Wardrop optimum equilibrium $f^{(0)}$. To do that, we need to optimize the following cost function

$$\sum_{e \in \mathscr{E}} \int_0^{fe} d_e(s)\, ds = \sum_{e \in \mathscr{E}} C_e l_e \log(\frac{C_e}{C_e - f_e}) \tag{13}$$

In this case too, all the opportune operations between matrices and vectors have been performed. The obtained Wardrop equilibrium is:

$$f^{(0)} = [6.71564887e + 03, ..., 4.93333897e + 03] \tag{14}$$

As usual, I refer to the code for the complete vector $f^{(0)}$.

F. Ex 3f

Now we are asked to add tolls. The main idea behind tolls is that if a toll is added to all the edges, then selfish users should pay not only for their delay, but also for the cost that make other users pay due to their choice. Specifically, the added tolls are:

$$\omega_e = f_e^* d_e'(f_e^*) \tag{15}$$

where f_e^* is the social optimum flow. So, now, the delay is equal to

$$d_e = d_e + \omega_e \tag{16}$$

The new Wardrop equilibrium $f^{(W)}$ has been computed using the same method as before but with the new delay. The obtained vector result, as expected, is very very close to the social optimum. In fact, if we subtract element-wise the new Wardrop vector and the optimal-flows one, we obtain a vector composed by very low decimal numbers

G. Ex 3g

During this point an important modify has been applied: instead of the total delay, the cost is set to be the total additional delay compared to the total delay in free flow be given by:

$$c_e = f_e(d_e(f_e) - l_e) \tag{17}$$

Then, the new social optimum has been computed using the same reasoning as before but replacing the old cost with the one above.
After that, we were required to compute the new tolls ω_e^* such that the Wardrop equilibrium will be equal to the new social optimum.
In this case, we need to optimize the function

$$\sum_{e \in \mathscr{E}} \int_0^{fe} (d_e(s) - l_e + \omega_e)\, ds$$

(18)

Applying the same methods used before, the tasks have been completed (I refer to the code for implementation details).
As expected, the new Wardorp equilibrium, subtracted to the new optimal flow, will produce a vector composed by low decimal numbers.