Network Dynamics and Learning Homework #1

Lal Akin

Politecnico di Torino
Turin, Italy
s300192@studenti.polito.it

November 15, 2023

Introduction

This report is written with the purpose of delving deeper into the detials of the work done for the first homework of Network Dynamics and Learning lecture of 2023/2024 semester of Politecnico di Torino, Master's of Data Science and Engineering.

The main topics included in this report are: Max-Flow Min-Cut Problems, Bipartite Graphs, Perfect Matching, Network Flow Optimization.

This homework is mainly done by me, with Exercise 3 being a collaboration with my group members, namely Vida Ahmedy 301905, Faeze Saeedian 301304, Reza Barati 301309 and Xiyang Zu 288740.

Exercise 1

1.a

What is the minimum aggregate capacity that needs to be removed for no feasible flow from o to d to exist?

For this question, the answer is found by finding the minimum cut problem. Because to find a setting in which there are no links carrying a feasbile flow from o to d, we must reach the ending step of the Ford-Fulkerson algorithm and see that there is no links left from the origin to the destination. This also corresponds to finding the graph partition cut with the minimum value. In this case after implementing both of the algorithms to double-check the result, I found the maximum flow or minimum cut value to be 5, with nodes 0,1,2,3 on one side and 4 on the other of the graph partition after the cut. This means after the 5 units of flow is sent, there are no links from o to d.

1.b

What is the maximum aggregate capacity that can be removed from the links without affecting the maximum throughput from o to d?

I solved this section by hand. The problem is to find the maximum aggregate capacity that can be removed from the links without affecting the maximum throughput from o to d. Which is easily done after finding the maximum throughput value from Figure 2 as 5. There is 2 units of flow on link e_1, 2 units of flow on e_2, 3 units on e_3, 1 unit on e_5 and e_6 and finally 2 units on e_4. This leaves 1 unit of unused capacity on link e_1 and 2 units unused on e_5. Therefore the maximum aggregate capacity that can be removed from the links without affecting the maximum throughput is 3 units. If 3 units of capacity were removed from the respective links, the same result would still be reached.

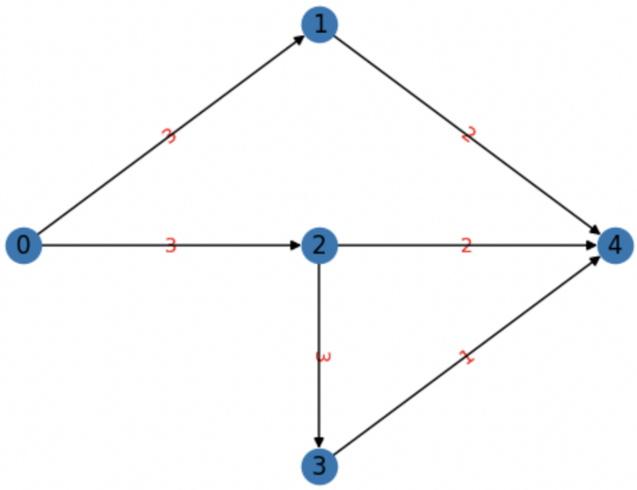

Figure 1: Graph with capacities (0,1,2,3,4) corresponds to (o,a,b,c,d) respectively

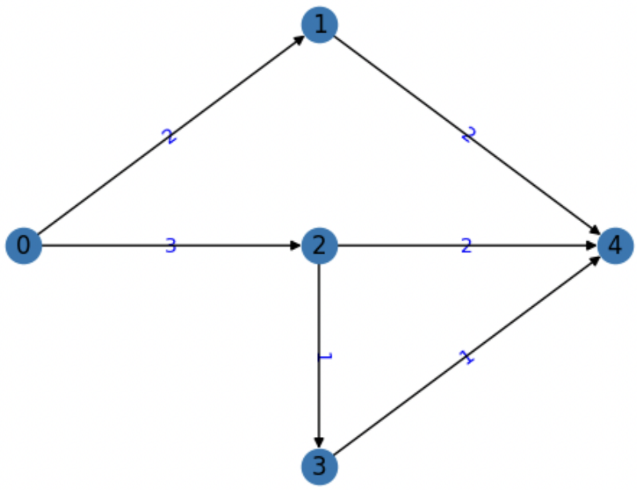

Figure 2: Graph with the units of flow passing from each link written in blue

1.c

You are given $x > 0$ extra units of capacity. How should you distribute them in order to maximize the throughput that can be sent from o to d? Plot the maximum throughput from o to d as a function of x greater than 0.

This question has made me think more than the others, since it is not an intuitive question, or one we already did before. I developed a thought by logic, to consider each scenario in which we add one by one a unit of capacity.

I manually did the steps and observed how the u (extra units of capacity) effects the t (max throughput) value.

By the iterations I concluded in the script, I saw some relationships form.

$$t = \begin{cases} 5 & : u = 0 \\ 6 & : u = 1, 2 \\ 7 & : u = 3, 4 \\ 8 & : u = 5 \\ 9 & : u = 6, 7 \\ \infty & : u, u - 1 \end{cases}$$

This isn't enough evidence to draw a correlation however I think there exists a relationship.

Exercise 2

2.a

Exploit max-flow problems to find a perfect matching (if any).

Perfect Matching means, in a bipartite graph, every node is covered exactly once by one edge. To see if there is a perfect matching we can transform this graph into a graph with an origin and a destination, and then give 1 unit of capacity to each edge to constraint the flow. The edges between people and books that have a flow passing through them will be the edges/relationships that we select. Also if we can reach a maximum flow of 4, which is equal to the amount of matches we are after, we can say that there exists a perfect matching by the definition of the perfect matching.

The matching is shown below in Figure 4

- Person 1 - Book 2
- Person 2 - Book 3
- Person 3 - Book 1
- Person 4 - Book 4

2.b

Assume now that there are multiple copies books, and the distribution of the number of copies is (2; 3; 2; 2). Each person can take an arbitrary number of different books. Exploit

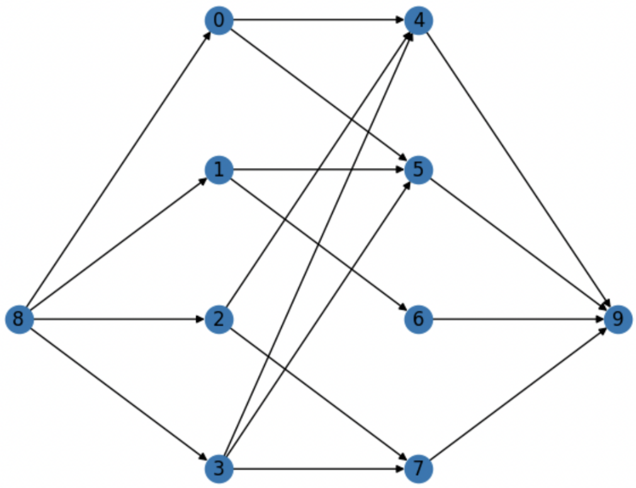

Figure 3: Source and sink addition to exploit Max Flow

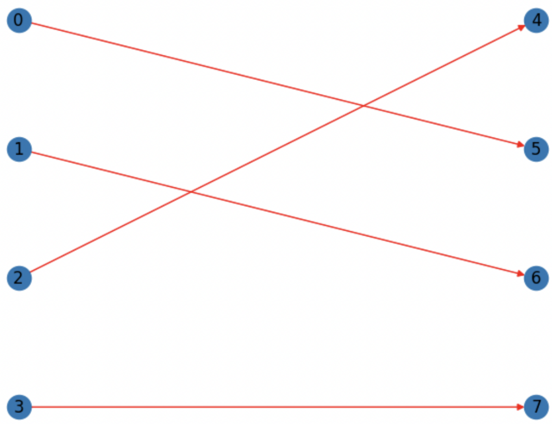

Figure 4: People and Book Matching

the analogy with max-flow problems to establish how many books of interest can be assigned in total.

To find an answer for this question, the graph above is modified slightly. Now, since it is said that every person can take an arbitrary number of any book, the capacity of the links going from the 8th node (origin) to each person must be large enough, namely Big M. I decided 100 >> 3, therefore the capacity value is set to 100 for the first 4 links. After this, to pose a constraint on the number of books that are available, I set the capacities of the links going out from the book nodes and to the destination node as the number of books that are available.

I observed that the throughput is equal to 8 as the result, which means one of the books were not bought. After checking the individual flows it is evident that it was the second of Book 3. It is because there isn't enough demand for this book.

2.c

Suppose that the library can sell a copy of a book and buy a copy of another book. Which books should be sold and bought to maximize the number of assigned books?

I solved this problem by hand mainly because it is very intuitive and it follows up from the last section.

There are (2,3,2,2) copies of each book respectively and I matched the people with the books they were interested in. Also as I found in the previous question, the 2nd copy of the 3rd book is the only one left after matching. So, if we sell one copy of the 3rd book and then buy one copy of the 1st book, we will be able to assign 9 books to 4 people. This way we increased the matching because before this switch there was one extra unit of demand for the 1st book. Now we covered that demand by selling the unused copy. I solved the same graph of above but now the capacities of links going from Book 1 to destination to 3, and from Book 3 to destination to 1. This way we can see the maximum throughput is 9.

Exercise 3

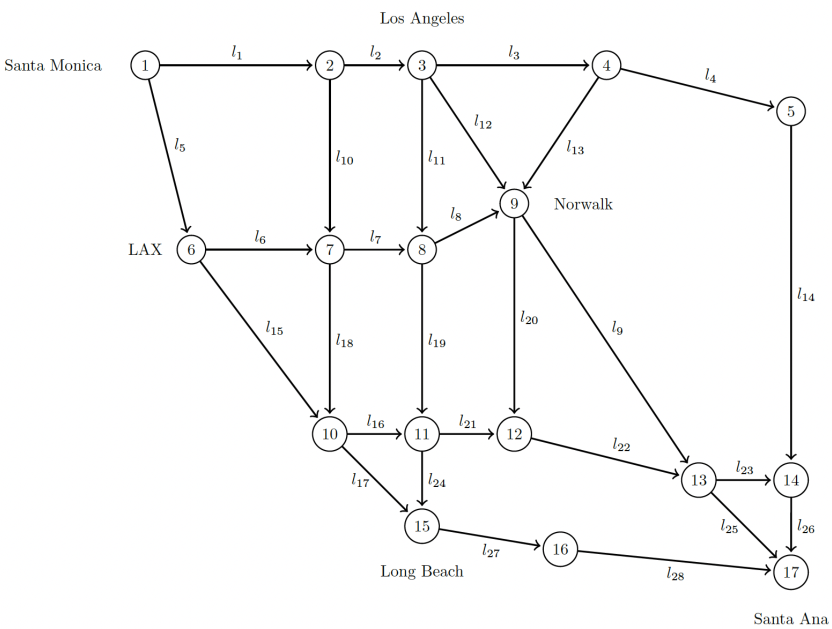

Figure 5: Highway Networks of Los Angeles

We are given the highway network in Los Angeles as it can be seen from 5 with links as highways, and each node is the intersections of the highways and the area surrounding them.

3.a

Find the shortest path between node 1 and 17. This is equivalent to the fastest path (path with shortest traveling time) in an empty network.

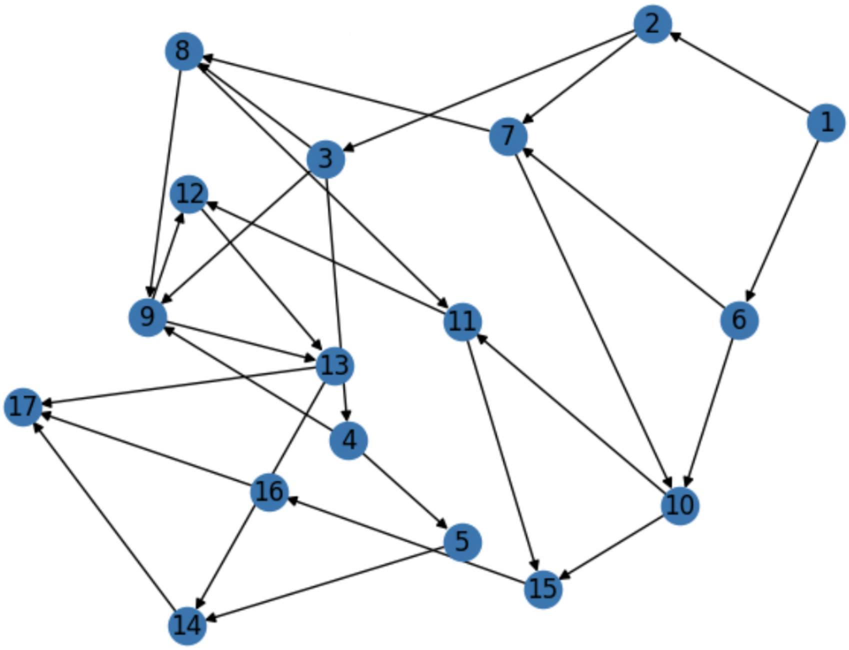

Figure 6: Drawing the graph

It is given in the problem that, to find the shortest path in this context we have to assume an empty network and deploy the shortest path algorithm. I simply utilized the NetworkX method "shortest path". The result is going from nodes: $1 \Longrightarrow 2 \Longrightarrow 3 \Longrightarrow 9 \Longrightarrow 13 \Longrightarrow 17$ will result in the shortest path according to the lengths (travel time) of each link on the path.

3.b

Find the maximum flow between node 1 and 17.

To find the maximum flow, I implemented the "max flow" method. In the end, the result is the flow value sent from nodes to one other. The maximum throughput is equal to 22448 units.

3.c

Given the flow vector in flow.mat, compute the external inflow satisfying $Bf = v$.

B is the node-link incidence matrix or in this context the traffic values. The f is the flow of each link, and v is equal to the external inflow value for each of the 17 nodes. After simply implementing the equation in python language, the resulting vector was found. The resulting matrix contains negative values so that is not representative of the external inflow matrix characteristics, so a mask can be used.

The external inflow vector is as shown in Table 1.

node	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
external inflow	16282	9094	19448	4957	-746	4768	413	-2	-5671	1169	-5	-7131	-380	-7412	-7810	-3430	-23544

Table 1: External Inflow Vector

node	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
external inflow	16282	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-0	-16282

Table 2: Exogenous Network Flow Vector

3.d

Find the social optimum flow with respect to the delays on the different links. For this, minimize the cost function which is subject to the flow constraints.

From this point onward we take the exogenous flow as shown Table 2, also in accordance to the requirement $\sum Vi = 0$

Social optimum represents the optimum reached by the whole of society. The best case scenario for the society in this case. To find this, we are given cost function $\sum_{e \epsilon E} f_e \tau_e(f_e)$ to minimize, which is mainly to minimize the sum of delay times flow on each link.

In CVXPY, we can code a mathematical model by defining the Variables, Constraints and the Objective Function that will be maximized or minimized. We are using the number of links (28) as the variable, since we want to solve this problem for each link. We have the cost function given therefore once we formulate that in a proper way and add the constraints, we can solve this problem.

Social optimum cost of this network is: 23835.48 (this is expressed as the time it takes to travel, more specifically the sum of the delay times flow on each link for the social optimum flow f^*.

3.e

Find the Wardrop equilibrium f(0). For this, use the cost function given.

To calculate the Wardrop equilibrium, we are given the cost function $\sum_{e \epsilon E} \int_0^{f_e} \tau_e(s) ds$. This equilibrium symbolizes each free agent choosing for their benefit, the least travel time option or the fastest path, and resulting in a congestion. Because of this I expect, before any calculation, for the cost of Wardrop equilibrium to be higher than the Social optimum cost.

After modifying the code to this cost function, I find the Wardrop equilibrium cost: 24162.20. As it was mentioned above, this cost is higher than the social optimum cost because by acting in a selfish manner, each agent contributes to having more delay in the end.

3.f

Introduce tolls. Compute the new Wardrop equilibrium f(!). What do you observe?

Now we have the toll on each link e as $w_e = \psi'_e(f^*_e) - \tau_e(f^*_e)$, and the delay on each link e is given by $\tau_e(f_e) + w_e$. According to these alterations, I modified the model objective by adding the toll value. After finding the optimum flow values for this setting of Wardrop Equilibrium, by using those flow values to calculate the cost, I found the value below. After introducing the tolls, it is evident that the cost is reduced exactly to meet the social optimum cost because tolls add a negative effect to the links therefore it spreads the agents to other links, and the total travel time decreases.

- Social optimum cost: 23835.48
- Wardrop equilibrium cost: 24162.20
- Wardrop equilibrium cost after introducing tolls: 23835.48

3.g

Instead of the total travel time, let the cost for the system be the total additional travel time compared to the total travel time in free flow. Compute the system optimum flow for the costs above. Construct a toll vector such that the Wardrop equilibrium coincides with the system optimum flow. Compute the new Wardrop equilibrium with the constructed tolls to verify your result.

This question is an addition of the previous steps, but now with a different travel time definition.

The way to measure the cost is now equal to $\psi_e(f_e) = f_e(\tau_e(f_e) - l_e)$ which takes into account also the total additional travel time compared to the total travel time in free flow.

Firstly we have to calculate the social optimum flow and cost, then calculate the Wardrop equilibrium with tolls added so that we can balance the costs to each other. The tricky part is to be able to transfer the mathematical equations to the python language in a correct way, but I believe it should be correct because as expected, both the costs that I have found are equal to each other after the addition of tolls.

- Social optimum cost: 13334.30
- Wardrop equilibrium cost after introducing tolls: 13334.29

Another observation of this exercise is that both of the costs have decreased, and that is due to the new minus term in the cost function.

Group Work Details

The first and second exercises besides 1.c are my work. For 1.c, we collaborated in thinking how it can be solved The third exercise is done as a collaboration, as a collaborative thought process, with my group members who's names have been given previously.