Report Homework 1

Chiara Scagliola
Politecnico di Torino
s303037@studenti.polito.it

Chiara Van der Putten
Politecnico di Torino
s306273@studenti.polito.it

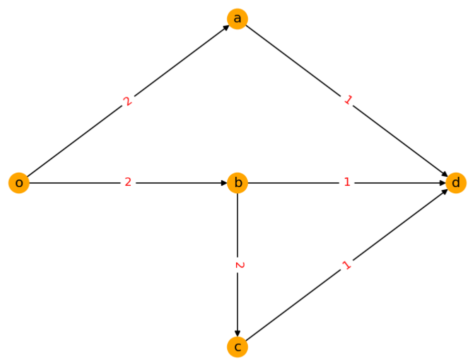
November 2022

1 Exercise

1.1 Disconnect the origin and the destination

According to Menger's theorem, to disconnect the origin o and the destination d, the minimum capacity that needs to be removed is the one associated with the min-cut capacity. Considering the graph shown in Figure 1, we calculate the cut capacities

Figure 1: graph representation



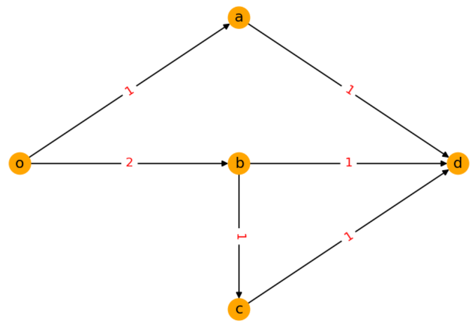
Cuts with capacity:
- $U = \{o\}, U^C = \{a, b, c, d\} \to C_U = 4$
- $U = \{o, a\}, U^C = \{b, c, d\} \to C_U = 3$
- $U = \{o, c\}, U^C = \{b, a, d\} \to C_U = 5$
- $U = \{o, b\}, U^C = \{a, c, d\} \to C_U = 5$
- $U = \{o, a, c\}, U^C = \{b, d\} \to C_U = 4$
- $U = \{o, b, a\}, U^C = \{c, d\} \to C_U = 4$
- $U = \{o, b, c\}, U^C = \{a, d\} \to C_U = 4$
- $U = \{o, b, a, c\}, U^C = \{d\} \to C_U = 3$

According to the definition given above and the calculated cut capacities, the minimum capacity that needs to be removed is 3. The edges with capacity 3 are between the sets $\{o, a, b, c\}$ and the set $\{d\}$ and the one between the sets $\{o, a\}$ and the set $\{b, c, d\}$

1.2 Remove aggregate capacity without affecting the throughput

Considering, the maximum flow function from node o to node d, it's possible to obtain all the capacities used to compute the maximum flow (Figure 2).

Figure 2: Graph after computing maximum flow



The used capacity of edge (o,a) is only 1 instead of the total capacity of 2, and the used capacity of edge (c,d) is 1 instead of 2. It is possible to deduce that the maximum aggregate capacity that can be removed from the links without affecting the maximum throughput from o to d is 2.

1.3 Additional capacity

After calculating the initial min-cuts, is necessary to add a unit of capacity. It's possible to notice that two min-cuts are present:

$$- U = \{o, a, b, c\}, U^C = \{d\} \rightarrow C_U = 3$$
$$- U = \{o, a\}, U^C = \{b, c, d\} \rightarrow C_U = 3$$

both share the link (a,d) so is possible to improve the capacity of the two by adding the capacity on edge (a,d). After that, the min-cuts are recalculated:

$$- U = \{o\}, U^C = \{a, b, c, d\} \rightarrow C_U = 4$$
$$- U = \{o, a\}, U^C = \{b, c, d\} \rightarrow C_U = 4$$
$$- U = \{o, b\}, U^C = \{a, c, d\} \rightarrow C_U = 5$$
$$- U = \{o, c\}, U^C = \{b, a, d\} \rightarrow C_U = 5$$
$$- U = \{o, a, c\}, U^C = \{b, d\} \rightarrow C_U = 5$$
$$- U = \{o, b, a\}, U^C = \{c, d\} \rightarrow C_U = 5$$
$$- U = \{o, b, c\}, U^C = \{a, d\} \rightarrow C_U = 4$$
$$- U = \{o, b, a, c\}, U^C = \{d\} \rightarrow C_U = 4$$

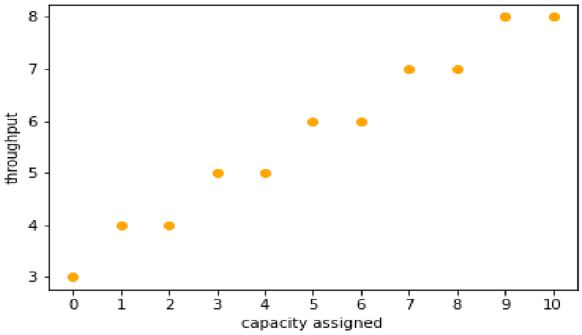
Capacity is now increased to 4 and there are four min-cuts, to decide on which link to go to increase the next capacity is necessary to consider the links present in more min-cuts:

$$- U = \{o\}, U^C = \{a, b, c, d\} \rightarrow C_U = c(o, a) + c(o, b)$$
$$- U = \{o, a\}, U^C = \{b, c, d\} \rightarrow C_U = c(o, b) + c(a, d)$$
$$- U = \{o, b, c\}, U^C = \{a, d\} \rightarrow C_U = c(o, a) + c(b, d) + c(c, d)$$
$$- U = \{o, b, a, c\}, U^C = \{d\} \rightarrow C_U = c(a, d) + c(b, d) + c(c, d)$$

between the selected edges, one capacity is arbitrarily allocated to the edge (o, a), and the min-cuts are recomputed:

$$- U = \{o\}, U^C = \{a, b, c, d\} \rightarrow C_U = 5$$
$$- U = \{o, a\}, U^C = \{b, c, d\} \rightarrow C_U = 4$$

Figure 3: Graph obtained by allocating 10 capacities



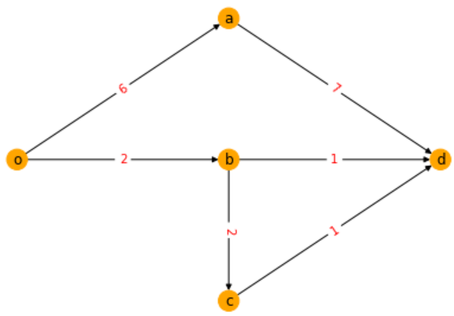
- $U = \{o, b\}, U^C = \{a, c, d\} \rightarrow C_U = 6$
- $U = \{o, c\}, U^C = \{b, a, d\} \rightarrow C_U = 6$
- $U = \{o, a, c\}, U^C = \{b, d\} \rightarrow C_U = 5$
- $U = \{o, b, a\}, U^C = \{c, d\} \rightarrow C_U = 5$
- $U = \{o, b, c\}, U^C = \{a, d\} \rightarrow C_U = 6$
- $U = \{o, b, a, c\}, U^C = \{d\} \rightarrow C_U = 4$

Considering the two min-cut:

- $U = \{o, a\}, U^C = \{b, c, d\} \rightarrow C_U = 4 \rightarrow C_U = c(a, d) + c(o, b)$
- $U = \{o, b, a, c\}, U^C = \{d\} \rightarrow C_U = 4 \rightarrow C_U = c(a, d) + c(b, d) + c(c, d)$

It's possible to notice that the only edge in common is (a, d), by increasing its capacity the throughput increases to 5. It is possible to continue by going to increase the edges in common within the min-cuts. Going forward with the same method, it's possible to deduce that, except for the first allocated capacity, for every two allocated capacities the final throughput increases by 1, as can be seen in Figure 3.

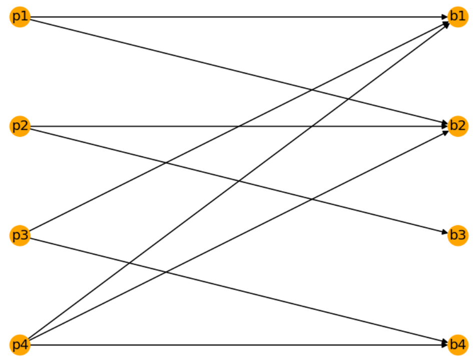
Figure 4: Graph with the 10 allocated capacities



2 Excersise

The graph considered is the one represented in Figure 5. This is a bipartite graph in which the first set represents people while the other is the set of books, and the edges represent a person's interest in a particular book.

Figure 5: Graph representation of people and books



2.1 Perfect matching

To find the perfect matching of the graph, a directed graph in which the edges range from people to books with a capacity equal to is considered.
A person can purchase only one copy of the books in which he or she is interested. Two nodes are then added:

- a source connected to all people with capacity 1
- a target that is reached with an edge by all nodes representing a book with capacity 1.

The resulting graph is represented in Figure 6.

Figure 6: original graph with source and target, all capacities are set to 1

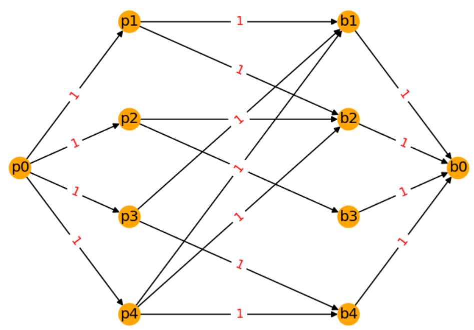


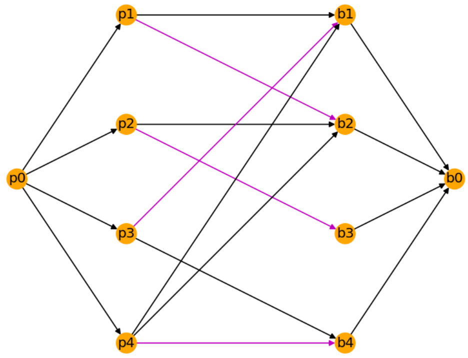
Figure 7 shows the result of the maximum flow algorithm on the graph.

It's possible to notice that a perfect matching is (p1,b2), (p2,b2),(p3,b1),(p4,b4). In this case, there is an analogy between perfect matching and maximum flow since the capacity of the edges connecting people to books is set to 1. Thus we will use an edge completely (sending 1 unit of flow) or not use an edge at all. So if there is a matching of n edges, there is a flow f of value n.

2.2 Matching with multiple copies of books

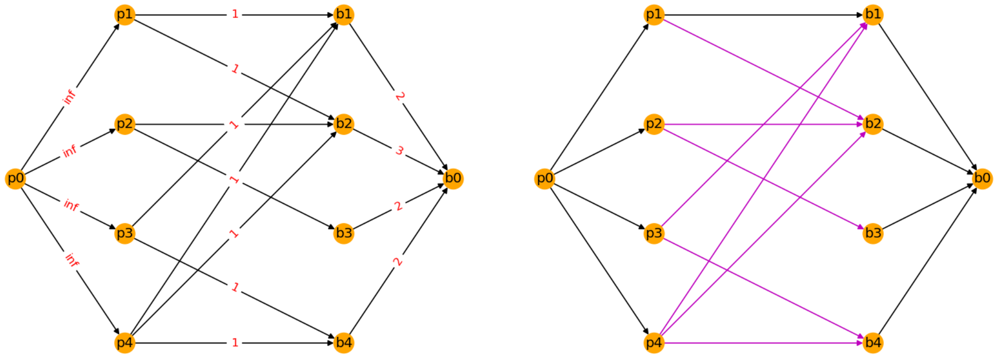
Assuming that a person can take only one copy of any book he or she prefers and that multiple copies of each book exist (respectively 2 copies of b1, 3 of b2, 2 of b3, 2 of b4), the capacities of edges going from the source to people equal to infinity, and those going from books to the target equal to the capacities listed above (Figure 8 on the left). The resulting

Figure 7: graph with the edges used to create a perfect match highlighted in pink



matching will be the one shown in Figure 8 on the right.

Figure 8: total capacities(left), matching(right)



2.3 Sell a copy of a book and buy a copy of another book

As it's shown in Figure 9 on the right, b3 sells only 1 copy even though there are 2 copies that can be sold, while p1 would like to buy a copy of b1 but cannot since there are only 2 copies. Supposing that the library can sell a copy of a book and buy a copy of another book, then the library should purchase one more copy of b1 and one less copy of b3. By implementing these changes, the resulting graph is as in Figure 9 (right)

3 Excersise

In this exercise, a graph that simplifies the highway network in Los Angeles is used. It is represented in Figure 10

3.1 Shortest path

Dijkstra's algorithm can be used to find the minimum path.
The fastest path from node 1 to node 17 is equal to [1, 2, 3, 9, 13, 17], as we can see from the Figure 10

Figure 9: comparison between total capacities (left), representing the number of books that can be sold, and the total capacity used in each edge calculated with the maximum flow (right), representing the number of sold copies

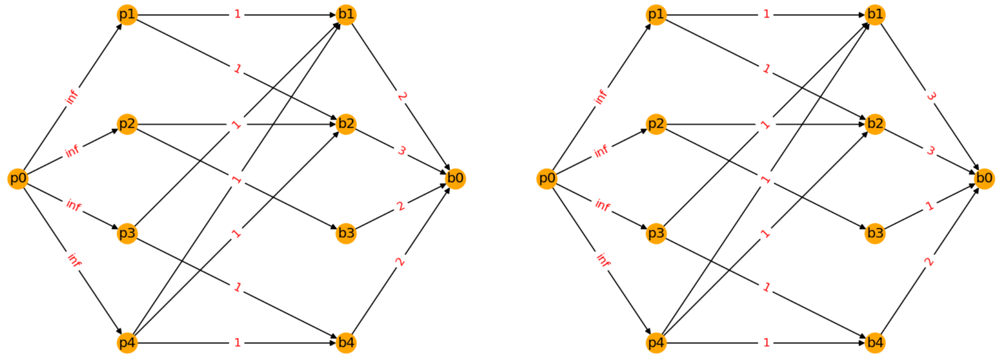
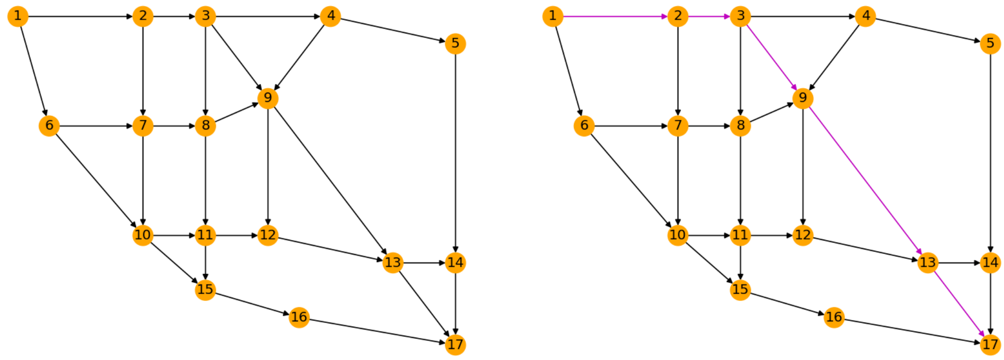


Figure 10: Graph representation (left), graph after computing maximum flow (right)



3.2 Maximum flow

The Ford-Fulkerson Algorithm is used to calculate the maximum flow and the result of the maximum flow is 22448. The flow distribution is represented in Figure 11

3.3 External inflow

Given the flow vector:

$$f = (7524, 6537, 11139, 9282, 9282, 6398, 6728, 5988, 5951, 9557, 7423, 7423, 6814, 8536, 7652,\\ 6537, 11924, 9640, 8161, 8603, 7974, 9446, 5562, 6719, 9455, 6686, 10833, 7403)$$

the external inflow v is calculated, satisfying the equation $Bf = v$, obtaining:

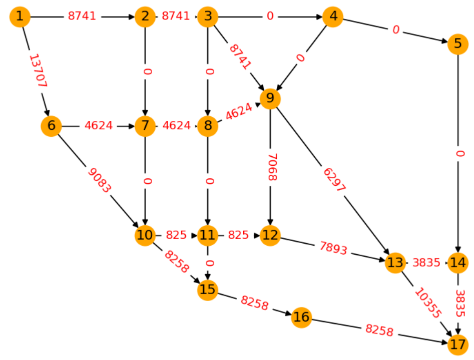
$$v = (16806, 8570, 19448, 4957, -746, 4768, 413, -2, -5671, 1169,\\ -5, -7131, -380, -7412, -7810, -3430, -23544)$$

From now on the following exogenous inflow vector v and delay function $\tau_e(f_e)$ will be considered:

$$v = [16806, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -16806]$$

$$\tau_e(f_e) = \frac{l_e}{1 - f_e/c_e}, \quad 0 \leq f_e < c_e$$

Figure 11: Distribution of maximum flow in the graph



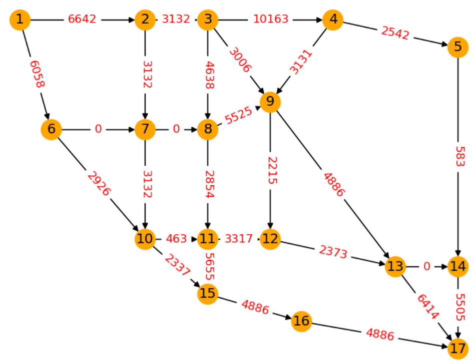
3.4 Social optimum

To find the social optimum, it's necessary to minimize the cost function:

$$\sum_{e \in E} f_e \tau_e(f_e) = \sum_{e \in E} \left(\frac{l_e c_e}{1 - f_e/c_e} - l_e c_e \right)$$

Figure 12 illustrates how the Social Optimum Flow is distributed along the edges of the graph:

Figure 12:



The cost of the social optimum is: 25943.62

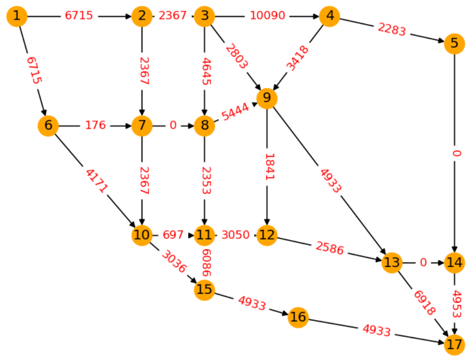
3.5 Wardrop equilibrium

The Wardrop equilibrium is a path flow distribution such that if a path p is used, then the cost of the path is minimal. So a feasible flow vector f is at Wardrop equilibrium if for every commodity $i \in [k]$ and paths $P_1, P_2 \in P_i$ with $f_{Pi} > 0$ it holds that the total latency on path P_1 is less or equal to the total latency on path P_2. To calculate the Wardrop equilibrium it's necessary to minimized the following function:

$$\sum_{e \in E} \int_0^{f_e} \tau_e(u) du$$

obtaining the flow distribution along the edges depicted in Figure 13:

Figure 13:



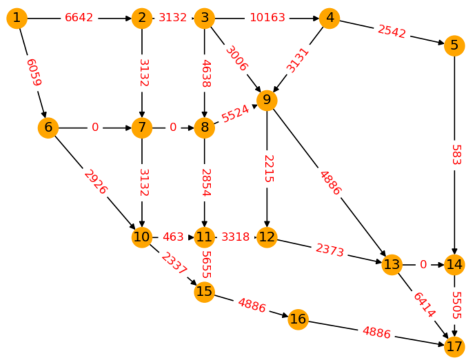
3.6 Wardrop equilibrium with toll

Now adding to each link a tool equal to $\omega_e = f_e^* \tau_e^{'}(f_e^*)$ the delay function change to $\tau_e(f_e) + \omega_e$. Recalculating the Wardrop equilibrium with the use of the new delay function in the following way:

$$\sum_{e \in E} \int_0^{f_e} \tau_e(s) + \omega_e ds$$

is obtained a flow distributed along the edges as in Figure 14:

Figure 14:



The difference between the social optimum flow vector and the Wardrop equilibrium with toll, calculated with the norm of the difference of the two flow vectors, is equal to 1.69. The difference as can be seen is minimal and is due only to small computational and approximation errors.

3.7 Change of the cost function

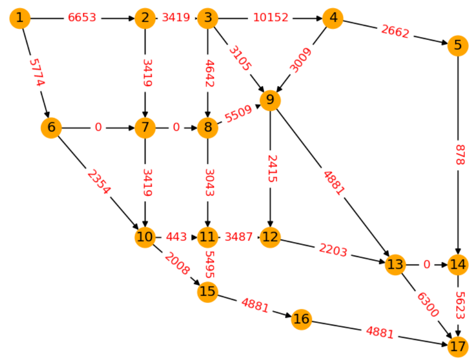
Considering a new cost function for the system:

$$\psi_e(f_e) = f_e(\tau_e(f_e) - l_e)$$

The new social optimum is now calculated by minimizing the function:

$$\sum_{e \in E} f_e \tau_e(f_e) - \sum_{e \in E} f_e l_e$$

Figure 15:



The representation of the new social optimum flow is shown in Figure 15:

To make the Wardrop equilibrium $f^{(w^*)}$ coincides with f^* it's necessary to construct $\omega^* = \psi_e^{'}(f_e^*) - \tau_e(f_e^*)$ where τ_e is $\frac{l_e}{1-\frac{f_e}{c_e}}$ and the derivative of ψ_e is $\tau_e(f_e^*) + f_e^* \tau_e^{'}(f_e^*) - l_e$, so the $\omega^* = f_e^* \tau_e^{'}(f_e^*) - l_e$ becomes equal to:

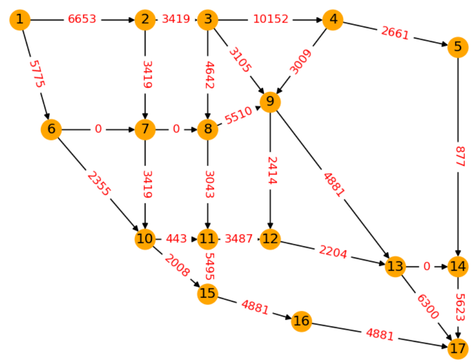
$$\frac{c_e l_e}{(c_e - f_e^*)^2} - l_e$$

So to find the Wardrop flow vector the following function needs to be minimized:

$$\sum_{e \in E} \int_0^{f_e} \tau_e(s) + \omega_e^* ds = \sum_{e \in E} \int_0^{f_e} \frac{l_e}{1 - \frac{f_e}{c_e}} + \frac{c_e l_e}{(c_e - f_e^*)^2} - l_e = \sum_{e \in E} -l_e c_e \log(c_e - f_e) + \omega_e f_e$$

The result obtained is shown in Figure 16:

Figure 16:



The difference between the social optimum flow vector and the Wardrop equilibrium compute with the new toll is 2.18. As it's possible to notice, the Wardrop flow vector is very close to the System Optimum flow, the only differences are due to small computational and approximation errors.