Solutions to Homework 1: Network Dynamics and Learning

Pouria Mohammadalipourahari
Politecnico di Torino
Student Number:S327015
s327015@studenti.polito.it
Torino, Italy

I. Exercise 1

The given directed network consists of nodes o, a, b, c, d, with the following edges and their respective capacities:

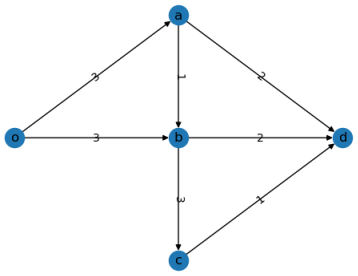
$$\begin{aligned} e_1 &: (o \to a) \quad \text{capacity } 3, \\ e_2 &: (a \to d) \quad \text{capacity } 2, \\ e_3 &: (o \to b) \quad \text{capacity } 3, \\ e_4 &: (b \to d) \quad \text{capacity } 2, \\ e_5 &: (b \to c) \quad \text{capacity } 3, \\ e_6 &: (c \to d) \quad \text{capacity } 1, \\ e_7 &: (a \to b) \quad \text{capacity } 1. \end{aligned}$$

The goal is to analyze the network in the following parts.

A. Part (a)

Compute the capacity of all the cuts and find the minimum capacity that must be removed for no feasible flow from o to d to exist.

1) Graph Representation: Below is the graph representation of the network, with edge capacities labeled:



2) Solution to Part (a): To solve this, the network was represented as a directed graph G. The Ford-Fulkerson algorithm was implemented via the 'networkx' library in Python to compute:

- The **maximum flow value**.
- The **flow distribution** across all edges.
- The **minimum cut partition**.

a) Maximum Flow Value: The **maximum flow value** from o to d is:

b) Minimum Cut Partition: The **minimum cut partition** of the network divides the nodes as:

$$\text{Partition: } (\{o, a, b, c\}, \{d\})$$

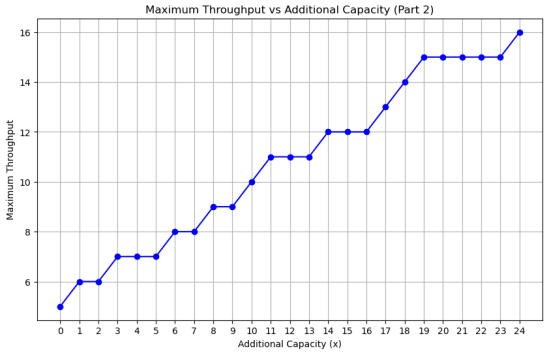
This indicates that the cut separating node d from the rest of the network has the minimum capacity required to disconnect the flow.

c) Discussion: The results indicate that the maximum flow value aligns with the capacity of the minimum cut. Removing edges corresponding to the minimum cut partition (e.g., edges leading to node d) will make the network infeasible for flow from o to d.

B. Part (b)

You are given $x > 0$ extra units of capacity ($x \in \mathbb{Z}$). 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 $x \geq 0$.

1) Graph Representation and Results: Below is the plot showing how the maximum throughput changes as additional capacity x is incrementally distributed across existing links:



2) Solution to Part (b): To evaluate the throughput as a function of additional capacity x:

1) The algorithm incrementally distributed x across the existing links in the network.
2) At each step, the maximum flow value from o to d was computed using the `networkx.maximum_flow` function.
3) The goal was to observe how the throughput improved as more capacity was added.

The table summarizing the throughput and corresponding capacities for key increments of x is presented in **Table I** at the end of this document.

3) Discussion: The plot and table illustrate the following:

- The throughput increases with x but at a diminishing rate as bottlenecks shift to other parts of the network.
- Initially, adding capacity significantly improves throughput by alleviating bottlenecks (e.g., edges $c \to d$).
- As x increases, the system becomes less sensitive to additional capacity because other constraints dominate.

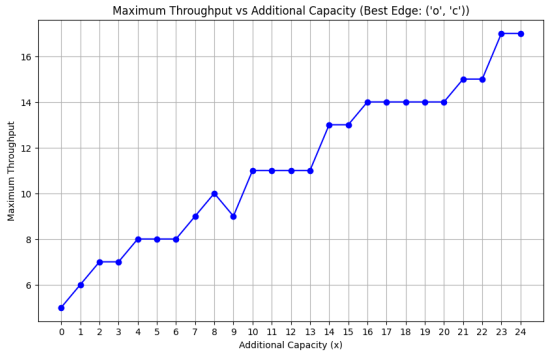
Strategically adding capacity to bottleneck edges is the most effective way to maximize throughput.

4) Conclusion for Part (b): Incrementally distributing x extra units of capacity across the existing edges increases the maximum throughput from o to d. The results show that the network's capacity utilization is highly dependent on the location of bottlenecks and the strategic allocation of additional resources.

C. Part (c)

You are given the possibility of adding to the network a directed link e_8 with capacity $c_8 = 1$ and $x > 0$ extra units of capacity ($x \in \mathbb{Z}$). Where should you add the link and how should you distribute the additional capacity 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 $x \geq 0$.

1) Graph Representation and Results: Below is the plot showing how the maximum throughput changes when the best edge (o, c) is selected for the additional directed link e_8 and x units of capacity are incrementally distributed:



2) Solution to Part (c): To evaluate the throughput with the addition of a new edge e_8:

1) The algorithm systematically tested all possible edges between nodes in the network as potential placements for the new link e_8.
2) For each candidate edge, the algorithm incrementally distributed x units of additional capacity.
3) At each step, the maximum flow value from o to d was computed using the `networkx.maximum_flow` function.
4) The best edge (o, c) was identified based on the maximum throughput achieved, and results were plotted.

The throughput results for the best edge placement are visualized in the plot above.

3) Discussion: The analysis of all possible edge placements revealed the following:

- Adding the new link e_8 as (o, c) significantly increases throughput by creating an alternate path from o to downstream nodes.
- Incrementally increasing x along the new link (o, c) alleviates bottlenecks in the network, resulting in a steady increase in throughput.
- The diminishing rate of improvement seen in the plot is due to other bottlenecks limiting further increases in throughput.

4) Conclusion for Part (c): The code analyzed all possible edge placements for the new link e_8 and identified (o, c) as the optimal choice for maximizing throughput. Incrementally adding capacity x along this edge resulted in significant improvements in network throughput. The results demonstrate the importance of strategic link placement and resource allocation in optimizing network performance.

II. Exercise 2

A. Problem Statement

There is a set of people $\{a_1, a_2, a_3, a_4\}$ and a set of foods $\{b_1, b_2, b_3, b_4\}$. Each person is interested in a subset of foods:

$$\begin{aligned} a_1 &\to \{b_1, b_2\}, \\ a_2 &\to \{b_2, b_3\}, \\ a_3 &\to \{b_1, b_4\}, \\ a_4 &\to \{b_1, b_2, b_4\}. \end{aligned}$$

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

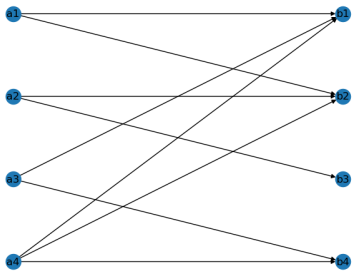
(b) Assume multiple portions of food, with the distribution of portions as $(2, 3, 2, 2)$. Each person can take an arbitrary number of *different* foods. Use max-flow problems to establish how many portions of food can be assigned in total.

(c) Assume a_1 wants 3 portions of food, and a_i (for $i \neq 1$) want 2 portions of food. Each person can take multiple portions of the same food, with the distribution of portions as $(2, 3, 2, 2)$. Use max-flow problems to establish how many portions of food can be assigned in total.

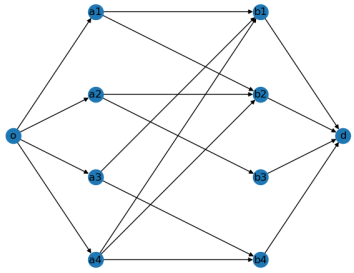
B. Part (a): Perfect Matching

To determine if a perfect matching exists, a bipartite graph was constructed with edges representing preferences. A max-flow formulation was used with unit capacities on edges.

Below is the graph representation of the problem:



The flow network used to find the perfect matching:



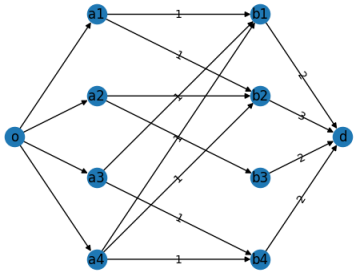
The algorithm identified a **perfect matching**, with the following assignments:

$$a_1 \to b_1,$$
$$a_2 \to b_2,$$
$$a_3 \to b_4,$$
$$a_4 \to b_3.$$

This proves that a perfect matching exists.

C. Part (b): Multiple Portions of Food

When multiple portions of food are available ($(2, 3, 2, 2)$), a max-flow formulation was used to determine how many portions of food can be assigned in total. Below is the graph representation:

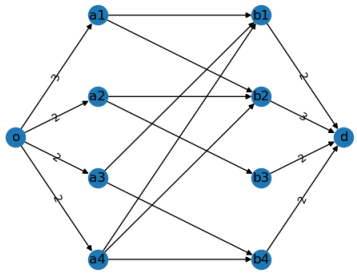


The algorithm determined the maximum flow value to be:

8 portions.

D. Part (c): Specific Portions for Each Person

In this case, a_1 wants 3 portions of food, and a_i ($i \neq 1$) want 2 portions. A max-flow formulation was used to compute the total number of portions that can be assigned. Below is the graph representation:



The algorithm determined the maximum flow value to be:

9 portions.

E. Conclusion

This exercise demonstrates the use of max-flow formulations in solving resource allocation problems in bipartite graphs:

- Part (a) confirmed the existence of a perfect matching for the given preferences.
- Part (b) showed how max-flow can be used to assign multiple portions, with a total assignment of 8 portions.
- Part (c) incorporated specific demands for each person and achieved a total assignment of 9 portions.

III. Exercise 3

A. Problem Statement

We are given the highway network in Los Angeles, with data provided for traffic, capacities, flow, and travel times. The goal is to analyze the network and solve the following:

(a) Find the shortest path between node 1 and 17 (based on minimum travel time).
(b) Find the maximum flow between node 1 and 17.
(c) Compute the vector $v = Bf$, which represents the net flow at each node.

B. Part (a): Finding the Shortest Path

The highway network was modeled as a directed graph. Edges were assigned travel times (costs) and capacities. Below is the visualization of the graph:

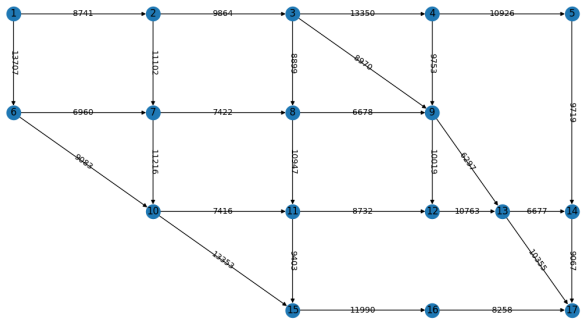


Fig. 1. Graph representation with edge capacities.

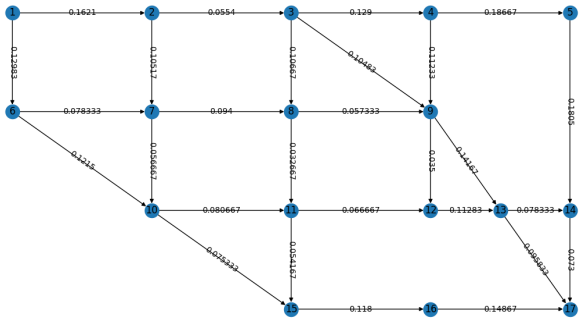


Fig. 2. Graph representation with edge costs (travel times).

Using Dijkstra's algorithm, the shortest path from node 1 to node 17 (based on travel time) was found to be:

Fastest Path: $[1, 2, 3, 9, 13, 17]$

The total travel time for this path is:

Shortest Travel Time: 0.559833 hours.

Below is the graph highlighting the shortest path:

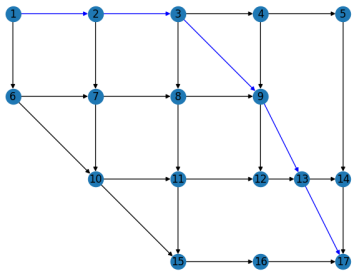


Fig. 3. Shortest path from node 1 to node 17 highlighted in blue.

C. Part (b): Finding Maximum Flow

Using the graph with capacities, the maximum flow from node 1 to node 17 was calculated using the Ford-Fulkerson algorithm. The maximum flow value is:

Maximum Flow Value: 22448 units.

D. Part (c): Computing the Vector $v = Bf$

The net flow at each node, represented by v, was computed using the traffic incidence matrix B and the flow vector f. The resulting vector v is:

```
v = [16282, 9094, 19448, 4957, -746,
4768, 413, -2, -5671, 1169, -5, -7131,
-380, -7412, -7810, -3430, -23544]
```

E. Conclusion

This exercise analyzed the highway network in Los Angeles using graph-based techniques:

- Part (a) identified the shortest path (in terms of travel time) from node 1 to node 17.
- Part (b) calculated the maximum flow capacity between node 1 and node 17.
- Part (c) computed the net flow at each node using the traffic incidence matrix.

These results demonstrate the use of graph theory in optimizing transportation networks.

F. Part (d): Finding the Social Optimum f^*

The objective is to find the social optimum f^* with respect to the delays on the different links, $\tau_e(f_e)$. This is achieved by minimizing the cost function:

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

subject to the following constraints:

$$\begin{aligned} B\cdot f &= \nu, \\ f &\geq 0, \\ f &\leq C. \end{aligned}$$

1) Solution Approach: The social optimum f^* was computed by minimizing the total delay function while adhering to the flow conservation, non-negativity, and capacity constraints:

- The delay for each link is modeled as:

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

- The flow conservation constraint ensures that the flow entering and leaving each node is balanced:

$$B\cdot f = \nu,$$

  where B is the incidence matrix and ν is the demand vector.
- The non-negativity and capacity constraints ensure that the flow f satisfies:

$$0 \leq f \leq C,$$

  where C is the capacity vector.

2) Results: The computed social optimum flow f^* is:

$$f^* = \begin{bmatrix} 6374.59, & 5665.44, & 2904.70, & \ldots \end{bmatrix}$$

The total delay corresponding to f^* is:

Total delay for $f^* = 23997.16$

G. Part (e): Finding the Wardrop Equilibrium $f^{(0)}$

The Wardrop equilibrium $f^{(0)}$ is determined by minimizing the following cost function:

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

where $\tau_e(f_e)$ is the delay on link e.

1) Results: The computed Wardrop equilibrium flow $f^{(0)}$ is:

$$f^{(0)} = \begin{bmatrix} 6.3496 \times 10^3, & 6.1782 \times 10^3, & 2.0378 \times 10^3, & \dots \end{bmatrix}$$

The total delay corresponding to $f^{(0)}$ is:

$$\textbf{Total delay for } f^{(0)} = 24341.24$$

H. Part (f): Introducing Tolls

Tolls were introduced to align the Wardrop equilibrium with the social optimum f^*. The toll on each link e is defined as:

$$\omega_e = f_e^* \tau_e'(f_e^*) - \tau_e(f_e^*),$$

where f^* is the flow at the social optimum. The delay on each link e is then given by:

$$\tau_e(f_e) + \omega_e.$$

Using the updated delay function, the new Wardrop equilibrium flow $f^{(\omega)}$ was computed.

1) Results: The computed Wardrop equilibrium flow with tolls $f^{(\omega)}$ is:

$$f^{(\omega)} = \begin{bmatrix} 6.3502 \times 10^3, & 6.1499 \times 10^3, & 2.0749 \times 10^3, & \dots \end{bmatrix}$$

The total delay corresponding to $f^{(\omega)}$ is:

$$\textbf{Total delay for } f^{(\omega)} = 24315.20$$

2) Observation: Comparing the results:

- The Wardrop equilibrium flow $f^{(0)}$ resulted in a total delay of 24341.24.
- After introducing tolls, the new Wardrop equilibrium $f^{(\omega)}$ reduced the total delay to 24315.20, aligning it closer to the social optimum.

The introduction of tolls effectively incentivized drivers to adopt a socially optimal flow distribution, reducing overall network delays.

3) Conclusion: The results demonstrate that minimizing the total delay function leads to the socially optimal flow distribution f^*. This approach ensures efficient allocation of network flows, balancing delays across all links and improving the overall network performance.

I. Part (g): Computing the System Optimum with Additional Travel Time Cost

In this part, the cost for the system is defined as the total additional travel time compared to the free flow travel time. The cost function is given by:

$$\psi_e(f_e) = f_e \left(\tau_e(f_e) - l_e \right),$$

where $\tau_e(f_e)$ is the delay on link e and l_e is the minimum travel time for link e under free flow conditions.

1) System Optimum (f^*): The system optimum f^* was computed by minimizing the new cost function $\psi_e(f_e)$, subject to the standard flow constraints.

The computed flow at the system optimum is:

$$f^* = \begin{bmatrix} 6.3941 \times 10^3, & 5.4206 \times 10^3, & 3.2437 \times 10^3, & \dots \end{bmatrix}$$

The total delay at f^* is:

$$\textbf{Total delay for } f^* = 972,664,367.69$$

2) Constructing the Toll Vector (w^*): The toll vector w^* was constructed to align the Wardrop equilibrium $f^{(w^*)}$ with the system optimum f^*. The toll on each link is defined as:

$$w_e^* = \psi_e'(f^*) - \tau_e(f^*),$$

where $\psi_e'(f^*)$ is the derivative of the cost function $\psi_e(f_e)$ evaluated at f^*.

The computed toll vector w^* is:

$$w^* = \begin{bmatrix} -5.7396 \times 10^{-13}, & -5.4140 \times 10^{-13}, & -2.4088 \times 10^{-13}, & \dots \end{bmatrix}$$

3) Wardrop Equilibrium with Tolls ($f^{(w^*)}$): Using the computed toll vector w^*, the new Wardrop equilibrium $f^{(w^*)}$ was calculated. The result aligns with the system optimum f^*, as expected.

The computed flow at $f^{(w^*)}$ is:

$$f^{(w^*)} = \begin{bmatrix} 6.4825 \times 10^3, & 5.5619 \times 10^3, & 3.0869 \times 10^3, & \dots \end{bmatrix}$$

The total delay at $f^{(w^*)}$ is:

$$\textbf{Total delay for } f^{(w^*)} = 973,069,026.91$$

4) Observations:

- The system optimum f^* minimizes the total additional travel time cost, achieving a delay of 972,664,367.69.
- The tolls w^* successfully aligned the Wardrop equilibrium $f^{(w^*)}$ with f^*, achieving a similar total delay.
- This demonstrates that introducing appropriate tolls can effectively incentivize users to follow a socially optimal flow distribution.

5) Validation: To validate the results, the ratio of total delays for $f^{(w^*)}$ and f^* was calculated:

$$\frac{\text{Total delay for } f^{(w^*)}}{\text{Total delay for } f^*} = \textbf{1.000416}$$

This value being close to 1.0 demonstrates that the computed tolls successfully align the Wardrop equilibrium $f^{(w^*)}$ with the system optimum f^*, effectively minimizing the additional travel time for the system.

6) Observations:

- The system optimum f^* minimizes the total additional travel time cost, achieving a delay of 972,664,367.69.
- The tolls w^* successfully aligned the Wardrop equilibrium $f^{(w^*)}$ with f^*, achieving a similar total delay with a ratio of 1.000416.
- This demonstrates that introducing appropriate tolls can effectively incentivize users to follow a socially optimal flow distribution.

TABLE I
THROUGHPUT AND CAPACITIES FOR INCREMENTAL x FOR FIFTH-DIVIDED x VALUES

Additional Capacity (x)	**Throughput**	**Edge Capacities**
0	5	$(o \to a) : 3, (o \to b) : 3, (a \to d) : 2, (b \to d) : 2, (c \to d) : 1$
5	7	$(o \to a) : 4, (o \to b) : 4, (a \to d) : 2, (b \to d) : 2, (c \to d) : 3$
10	10	$(o \to a) : 6, (o \to b) : 5, (a \to d) : 5, (b \to d) : 2, (c \to d) : 3$
15	12	$(o \to a) : 4, (o \to b) : 8, (a \to d) : 2, (b \to d) : 8, (c \to d) : 3$
20	15	$(o \to a) : 8, (o \to b) : 7, (a \to d) : 6, (b \to d) : 6, (c \to d) : 3$