Assignment 7 solutions

Responsible TA: Claudi Lleyda Moltó

Delivery instructions

To deliver an assignment you have to make sure you deliver both parts. You can deliver as many times as you want before the deadline on Blackboard. Delivery should be done in two separate files according to the following specifications.

The theoretical question answers must be uploaded as PDF, how you answer the questions (handwritten scans, Word, LaTeX, etc.) doesn't matter, as long as they are delivered as a single PDF file and the answers are readable. The filename should contain your NTNU username and the assignment number, for instance: `karinor-07.pdf`.

The notebook should be delivered in full (as `.ipynb`). That way the TAs can easily run your code and verify the output. The filename should contain your NTNU username and the assignment number, for instance: `karinor-07.ipynb`.

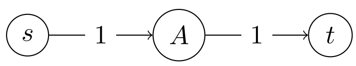
Theory

Task 1 - Maximum flow and minimum capacity cut

Prove or disprove the following statements.

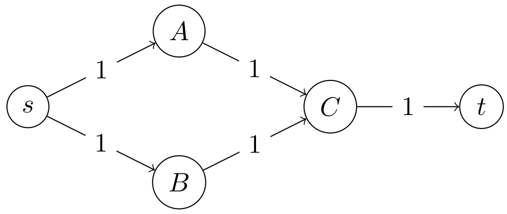
1. Each maximum flow defines a unique minimum capacity cut.
2. Each minimum capacity cut defines a unique maximum flow.

Solution. The first statement is false. Consider the following counter-example.



Here the maximum flow is 1, but there exist two minimum capacity cuts.

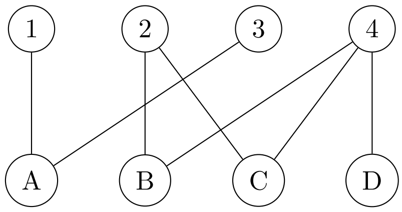
The second statement is also false. Consider the following counter-example.



Here the minimum capacity cut has capacity 1, but there exist two different maximum flows. $\diamondsuit$

Task 2 - Bipartite graph matching

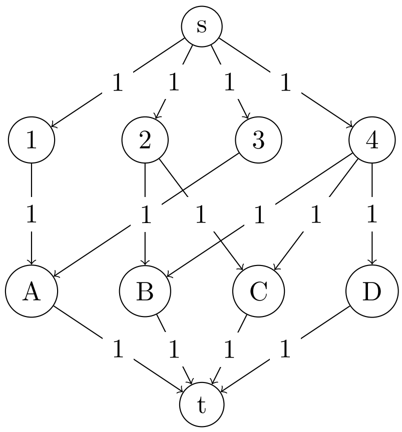
Consider the following bipartite graph.



1. Use Hall's Theorem to determine whether a perfect matching exists.
2. Use the Ford–Fulkerson algorithm to find a maximal matching.

Solution. If we take $S = \{1, 3\}$ we have $N(S) = \{A\}$, and Hall's Theorem implies that there is no perfect matching for this graph.

To find a maximal matching using the Ford–Fulkerson algorithm we first add two nodes s and t, and an edge from s to every node in the left set of the graph, and an edge from every node in the right set of the graph to t. With all the edge weights set to 1 we get the following graph



$\diamondsuit$

Running the Ford–Fulkerson algorithm we can find several different maximal matchings, such as $\{(1, A), (2, B), (4, C)\}$.

Task 3 - Antenna connections

Suppose we have n devices that want to connect to k cell towers. A device can only connect to a cell tower if it is close enough, and each cell tower can handle a maximum of N simultaneous connections.

Find a network flow algorithm that determines whether it is possible that every device connects to a cell tower with the given constraints.

Solution. Consider a graph with nodes s, t, a node for each of the n devices, and a node for each of the k cell towers.

We add edges with capacity 1 from s to each of the n device nodes, and edges with capacity N from each of the k cell tower nodes to t.

Additionally, for every device and cell tower in range of each other, we add an edge with capacity 1 between their associated nodes, going from the device to the cell tower.

Solving the stated problem is equivalent to finding whether the maximum flow from s to t is equal to n in this graph. $\Diamond$

Programming

This part should be solved in the corresponding Jupyter Notebook. Refer to the notebook for further details and instructions regarding the programming tasks.

Assignment submission

The assignments will be given out on Blackboard where each assignment has a written theoretical part and a programming part in Python using Jupyter Notebook. The notebook task can be solved locally (for instance in VSCode), or you can uploaded the notebook to JupyterHub and run the code there.

The notebooks will guide you though a, more or less, specific solution to a problem. You are however encouraged to attempt to implement your own/other solutions, as long as

- you don't use any additional libraries or packages,
- you don't alter the input values or structures, and
- your solution solves the problem described in the assignment.