Master 2 Graphs
Final Exam, February 2021
3 hours
No documents are allowed. No computers, cellphones. No covid.

Instruction and comments: the points awarded for your answer will be based on the correctness of your answer as well as the clarity of the main steps in your reasoning. All proposed solutions must be proved. **All the sections are independent.**

1 Dijkstra. (3 points, 15 minutes)

Consider the following graph H with weighted edges depicted in Figure 1. Give the definition

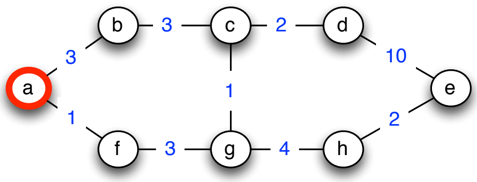

Figure 1: A weighted graph H. Blue integers denote the length of corresponding edges.

of a shortest path tree rooted in a in the graph H.

Applying the Dijkstra's algorithm on H, compute a shortest path tree rooted in a. Use the representation with a table (as seen during the lecture) to describe the execution of the algorithm.

In particular, indicate the order in which vertices are considered during the execution of the algorithm, and give the final solution.

2 Flow. (4 points, 25 minutes)

Consider the elementary network flow depicted in Figure 2 (left) and the initial flow f from s to t in Figure 2 (right).

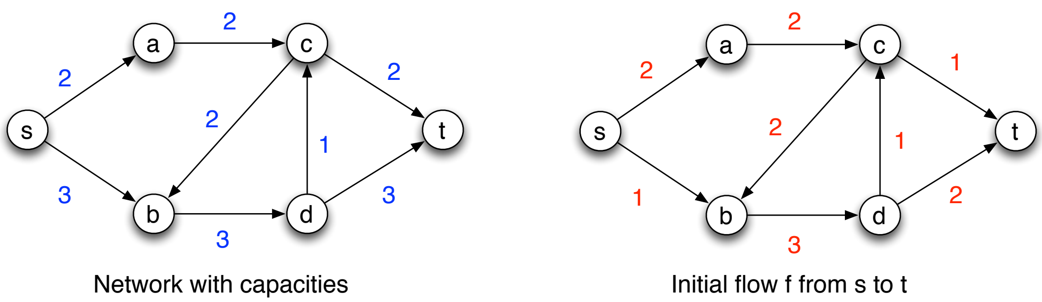

Figure 2: Flow.

1. What is the value of the inital flow?

2. Apply the Ford-Fulkerson Algorithm to N starting from the flow f. The auxiliary digraphs built during the execution of the algorithm must be given.

3. Give the flow and the cut obtained. Conclusion (state the min cut-max flow theorem)?

3 Modeling as a linear program. (4 points, 35 minutes)

A company builds two products and wants to increase the level of production to maximize the benefit.

Each unit of product 1 brings in a profit of 120 euros, when one unit of product 2 brings in 500 euros. Due to limitations in the production line, one cannot produce more than 200 products 1 and 300 products 2. Furthermore, one cannot produce more that 400 products in total because of the limited workforce.

1. Write a linear program which maximize the profit of the company.

2. Solve graphically the problem.

4 Trees. (3 points, 20 minutes)

1. What is the definition of a tree? of a planar graph?

2. By induction on the number of vertices, show that every tree T is planar and that every planar embedding (planar drawing) of T has a single face.

3. Deduce the relation between the number of vertices of any tree and its number of edges (state the result on planar graphs, seen during the lecture, that you should use here).

5 Outerplanar graphs. (5 points, 60 minutes)

This section is devoted to study some aspects of a subclass of planar graphs. We consider only simple graphs, i.e., without loops nor parallel edges.

A graph is *outerplanar* if it admits a planar embedding such that every vertex lies on the outerface (i.e., the unbounded face). To simplify, let us restrict further our study to 2-connected outerplanar graphs for which we give a simpler definition below.

1. What is the definition of a connected graph?

A connected graph $G = (V, E)$ is 2-*connected* if, for every $v \in V$, removing the vertex v from G leaves the graph connected.

A 2-connected graph $G = (V = \{v_0, \cdots, v_{n-1}\}, E)$ is *outerplanar* if it admits a Hamiltonian cycle $C = (v_0, \cdots, v_{n-1})$ (i.e., a cycle passing through all vertices, i.e., for every $1 \leq i \leq n$, $\{v_i, v_{i+1 \mod n}\} \in E$) and a planar embedding defined as follows. First draw the cycle C (in a planar way), then all remaining edges (those of $F = E \setminus E(C)$) can be drawn in a planar way (i.e., without crossing) "inside" C (i.e., in the bounded disk delimited by the drawing of C). Note that, following this drawing, every vertex lies in the outerface and so the graph is outerplanar. See Figure 3 for an example. **If this definition is not clear enough, ask me !**

2. Among the three graphs depicted in Figure 4, which ones are planar? are outerplanar? (prove your answers)

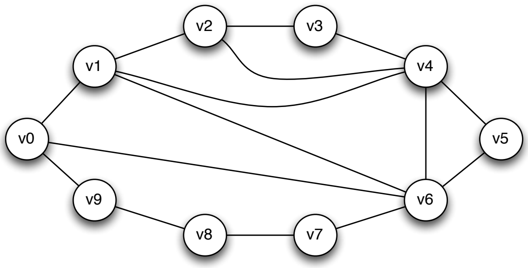

Figure 3: Example of a 2-connected outerplanar graph.

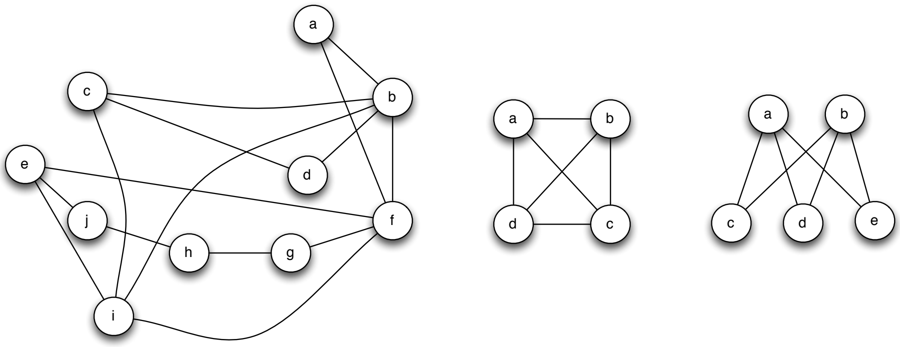

Figure 4: Some graphs from question 2.

Let $G = (V, E)$ be a 2-connected outerplanar graph with $C = (v_0, \cdots, v_{n-1})$ being the Hamiltonian cycle described above and $F = E \setminus E(C)$. If $F \neq \emptyset$, let $e_0 = \{x_0, y_0\} \in F$ be the edge of F minimizing the distance in C between its endpoints. That is, for every edge $\{u, v\} \in F$, the distance between u and v in C is a least the distance between x_0 and y_0 in C.

3. Show that the distance between x_0 and y_0 is at least 2 in C (recall that G is simple).

Without lost of generality (up to symmetry), let us assume that $x_0 = v_0$, $y_0 = v_i$ for $1 < i \leq \lceil n/2 \rceil$ and so the distance between x_0 and y_0 in C equals i.

4. Show that all vertices in $X = \{v_1, \cdots, v_{i-1}\}$ have degree 2 in G.
5. Show that the graph G' obtained from G by removing the vertices in X is a 2-connected outerplanar graph.
6. What is the chromatic number $\chi(G)$ of a graph?
7. Prove that, for every 2-connected outerplanar graph G, $\chi(G) \leq 3$.

   hint: use above question and induction on the number of vertices.
8. Assume that for every $v \in V(G')$, the distances $dist_{G'}(x_0, v)$ (i.e., the distance between x_0 and v in G') and $dist_{G'}(y_0, v)$ are known. For every $v \in V(G)$ and every $w \in \{x_0 = v_0, v_1, \cdots, y_0 = v_i\}$, gives the distance $dist_G(v, w)$ between v and w in G.

9. Give an algorithm that takes a 2-connected outerplanar graph $G = (V, E)$ and a vertex $a \in V$ and computes, for every vertex $v \in V$, the distance $dist_G(a, v)$ between v and a in G. What is the time complexity of the algorithm you have proposed?

6 Distributed algorithm. (3 points, 25 minutes)

Consider a mobile agent (a robot) lost in a graph. The goal of this section is to design an algorithm that the robot must follow to ensure it visits all vertices of the graph.

The robot has only a local view of the graph. That is, when it stands at a vertex v, it can only see the degree of this vertex and the port numbers of the edges leaving this vertex. That is, each vertex with degree d has d incident edges that are labelled from 1 to d (and these numbers are visible). Moreover, when the robot arrives in a new node v, it knows the port number of the edge by which it reached v.

As an example (but for a different objective), the famous *right hand rule* allows to find the exit of a maze by simply executing the following algorithm. When you enter a new room, just leave it by the corridor to the right from the one you entered.

1. Give a distributed algorithm that allows the robot to visit all nodes of the graph if the graph is a tree. (Note that the robot does not need to know when it has achieved its task). Prove that your algorithm is correct.
2. Give an example of graph where your algorithm is not successful (and explain why).