UBINET, Master 2 IFI Algorithms for telecommunications

Final Exam, November 2012

3 hours

Course and manuscript notes allowed. Not computers, cellphones, books.

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 exercises are independent.** *The points are indicated so you may adapt your effort.*

1 Algorithmics on graphs

Exercise 1 (4 points, 20 minutes) Consider the elementary network flow N depicted in Figure 1 (left) and the initial flow f from s to t in Figure 1 (right).

- What must be checked to show that f is a flow? What is the value of the flow f?
- Apply the Ford-Fulkerson Algorithm to N starting from the flow f. The first two steps (in particular, the auxiliary digraphs) of the execution of the algorithm must be detailed.
- Give the flow and the cut obtained. Conclusion?

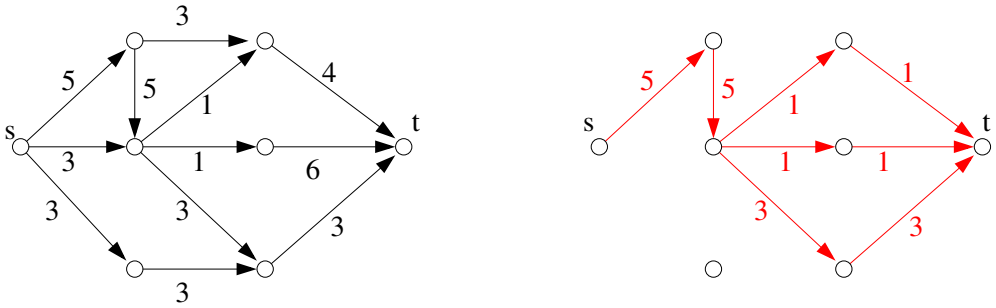


Figure 1: (left) Elementary network flow with arcs' capacity in black. (right) A flow f from s to t: a number close to an arc indicates the amount of flow along it. Arcs that are not represented have no flow.

Exercise 2 (4 points, 20 minutes) Let $N = (G = (V, A), (s, t \in V), c : A \to \mathbb{N})$ be an elementary flow network such that, for all arc $e \in A$, $c(e)$ is an even integer.

1. Prove that the maximum value of a flow is an even integer.
2. (*) Show that there is a maximum flow f such that, for all arc e, f (e) is an even integer.

   hint: start with the null flow and apply the Ford-Fulkerson Algorithm. Show that the amount of flow that is pushed at each step is even

Exercise 3 (6 points, 30 minutes) We recall that a *matching* in a graph $G = (V, E)$ is a set $M \subseteq E$ such that no two edges in M share a node. A matching M in G is *maximal* if, for any edge $e \in E \setminus M$, $M \cup \{e\}$ is not a matching. A matching M in G is *maximum* if, for any matching M' in G, $|M'| \leq |M|$.

A *vertex cover* in G is a set $K \subseteq V$ such that, for any edge $e \in E$, e has at least one end-point that belongs to K (i.e., for any edge $e = \{u, v\} \in E$, either $u \in K$ or $v \in K$). A vertex cover K in G is *minimum* if, for any vertex cover K' in G, $|K| \leq |K'|$.

Let G be the graph depicted in Figure 2. Give

1. a matching in G that is maximal but not maximum
2. a maximum matching in G. Justify.
3. a vertex cover in G of size at most 5.

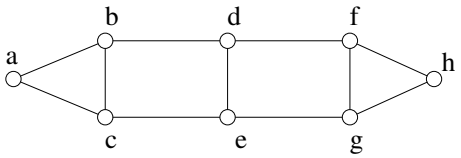


Figure 2: A graph $G = (V, E)$.

Now, let $G = (V, E)$ be any graph.

4. Show that a maximum matching in G is maximal.
5. Assume that M is a maximal matching of size k. Show that there is a vertex cover of size $2k$ in G.
6. Let K be a vertex cover of size k in G. Show that any maximal matching in G has size at least k.

hint: by contradiction

The problem to compute a minimum vertex cover is NP-complete. In particular, this means that no "fast" (polynomial-time) algorithm is known for this problem. On the other hand, Jack Edmonds has proposed a "fast" algorithm to compute a maximum matching in a graph.

7. Give an algorithm that takes a graph G and $k \geq 1$ as inputs and computes a vertex cover of size at most k if the minimum vertex cover of G has size at most k.
8. Give a polynomial-time algorithm that takes a graph G and $k \geq 1$ as inputs and computes a vertex cover of size at most $2k$ if the minimum vertex cover of G has size at most k. Prove its correctness.

Exercise 4 (3 points, 15 minutes) Apply the Dijkstra Algorithm to the graph depicted in Figure 3 to compute a shortest-path tree rooted in r and the distance between any vertex and vertex r. The first three steps of the algorithm must be detailed (**at most three or four lines per steps**). Moreover, indicate the order in which vertices are considered during the execution of the algorithm.

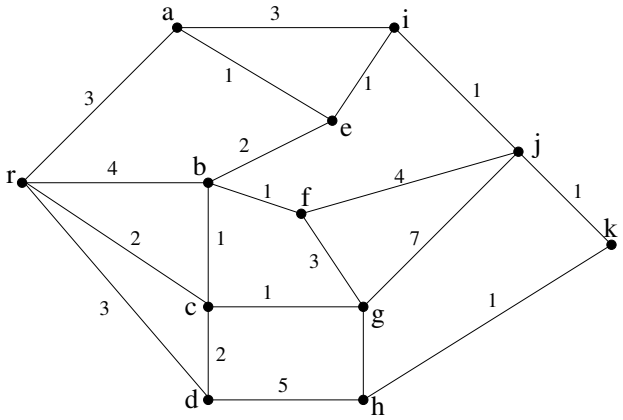


Figure 3: A graph with 12 vertices. A number indicates the length of the edge it is close to.

2 Linear Programming

Exercise 5 (4 points, 20 minutes) *We consider the following linear program.*

$$
\begin{array}{lrcrcrcrcl}
\textit{Minimize} & 8x_1 & + & 6x_2 & + & 10x_3 & + & 2x_4 & & \\
\textit{Subject to:} & & & & & & & & & \\
& 2x_1 & + & 1x_2 & + & 3x_3 & + & 1x_4 & \geq & 6 \\
& 6x_1 & + & 4x_2 & + & 8x_3 & + & 2x_4 & \geq & 3 \\
& 10x_1 & - & 4x_2 & + & 8x_3 & + & 4x_4 & \geq & 2 \\
& -4x_1 & + & 2x_2 & - & 4x_3 & - & 2x_4 & \geq & -3 \\
& & & & & & & x_1, x_2, x_3, x_4, x_5 & \geq & 0.
\end{array}
$$

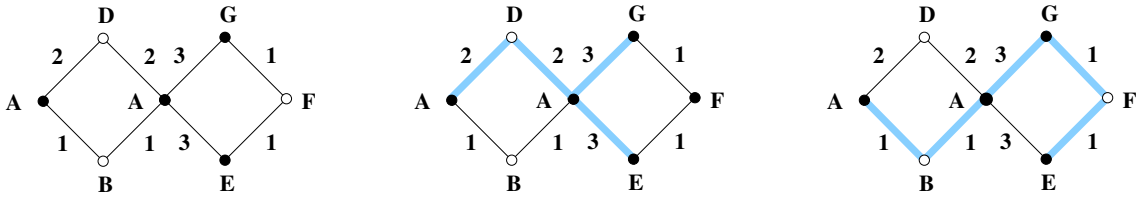


Figure 4: *Left:* A Steiner tree problem instance. $G = (V, E)$ with $V = \{A, B, C, D, E, F, G\}$. The set of terminals is $S = \{A, C, E, G\}$, the black vertices. The white vertices are non-terminal vertices. *Middle:* A Steiner tree of cost 10 (thick lines). *Right:* The Steiner tree of minimum cost 7 (thick lines).

1) *Write the dual of this linear program.*

2) *Is the solution* $y_1^* = 4, y_2^* = 0, y_3^* = 0, y_4^* = \frac{1}{2}$ *optimal for the dual? Use the method presented during the course and explain every step of the resolution.*

Exercise 6 (3 points, 10 minutes) The company N&N's produces chocolate drops. Each chocolate drop is made of a core in chocolate covered with colored sugar. The drops are sold in bags of 100 grams (g). To produce 1000g of drops, 750g of chocolate and 250g of sugar are needed. Four colors are available to color the drops: green, yellow, red and brown. Each bag has to contain at least 20% of drops of each color and the quantity of red and yellow drops should not be lower that the quantity of green and brown ones. We suppose that each bag has the same proportions of drops of each color. For the next month, the company has at its disposal:

- C tons (1 ton=1000Kg) of chocolate
- S tons of sugar
- brown coloring for any number of drops
- red coloring for at most R tons of drops
- yellow coloring for at most Y tons of drops
- green coloring for at most G tons of drops

Formulate the problem of determining the maximum number of bags that the company N&N's can produce during next month as a liner program.

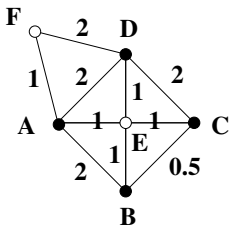
3 Problem: Steiner Trees

Exercise 7 (12 points, 60 minutes) *All questions are independent.*

Take a graph $G = (V, E)$, a set a subset of vertices $S \subseteq V$ called terminals and a cost function w defined on the edge set E. We here consider the case in which all the costs are strictly positive, that is $w : E \to \mathbb{R}_+^*$. The *Steiner tree problem* is to find the interconnection of minimum cost between all terminals.

Formally, a *Steiner tree* is a tree spanning S, meaning that all vertices of S are included in the set of vertices of the tree. Note that vertices that are not terminals (in $V \setminus S$) can also belong to the tree. The *Steiner tree problem* is the problem of finding a Steiner tree of minimum cost, where the cost of a tree is the sum of the costs of its edges. See Figure 4 for an example.

1. Let $G = (V, E)$ be the graph depicted below, $V = \{A, B, C, D, E, F\}$. Give the Steiner tree of minimum cost along with its costs for the set of terminals $R = \{A, B, C, D\}$.



2. (a) Write a linear formulation of the Steiner tree problem.
   Hint: Remark that the Steiner tree can be seen as a minimal subgraph having a path between each pair of terminals. In fact, we can even restrict our attention to pairs of terminals containing a specified vertex $r \in S$. This vertex r plays the role of the root of a Steiner tree.

   (b) Would this formulation work if the costs could be negative? Explain why and give an example.

3. Our goal now is to build a 2-approximation algorithm of the *Steiner problem*, that is an algorithm returning a Steiner tree T' of cost $c(T') \leq 2c(T^*)$ with $c(T^*)$ the cost of a minimum Steiner tree.

   To help us, we introduce the following definition. The Steiner problem is said *metric* if $G = (V, E)$ is the complete graph and if the cost function w satisfies the *triangular inequality*, that is

   $$\forall u, v, w \in V, c(uv) \leq c(uw) + c(wv).$$

   Consider the minimum *spanning tree* of S in the graph $G[S]$ (not considering the nodes of $V \setminus S$). We note it $T^{\#}$ and we note its cost $c(T^{\#})$.

   (a) Let K_n be the complete graph with n vertices and with cost 2 on all edges. We build a graph G_n from K_n by adding a vertex w connected to each vertex of K_n with an edge of cost 1.
   Give the cost of a minimum spanning tree of K_n. Give the cost of a Steiner tree of minimum cost of G_n when the set of terminals are the vertices of K_n. What happens when n goes to infinity? Conclude.

   (b) Consider now any instance of the *metric* Steiner tree problem. Show that $c(T^{\#}) \leq 2c(T^*)$, where $c(T^*)$ is the cost of the Steiner tree of minimum cost T for the *metric* problem.
   Hint: Take a DFS (Depth-first search) of the minimum Steiner tree T^* and consider the order in which vertices are visited, during the discovery and the backtracking phases of the algorithm. It gives a sequence $s_1 = x_0, x_1, ...x_k$ with $x_k = x_0$, see Figure 5. Let now $s_2 = y_0, y_1, ...y_{|R|}$ be the sequence obtained s_1 by removing the vertices in S and keeping only the first occurence of each vertex in S. Give the costs of the s_1 and s_2. Conclude.

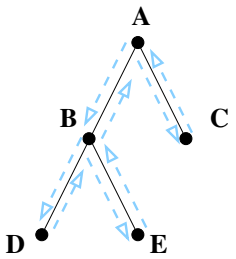


Figure 5: The sequence given by the DFS is ABDBEBACA

4. We come back to the general Steiner tree problem (*general* means that we are not necessarily in the metric case). Give an example showing that the minimum spanning tree of S can give a solution as bad as 1000 times the cost of the minimum Steiner tree.

5. Nevertheless, we can use the approximation for the metric Steiner tree problem to get a 2-approximation of the general Steiner problem. Explain how.
   Hint: compute the cost of the shortest path between each pair of nodes to build a new graph.