CentraleSupélec

Algorithmics and complexity

TD 3/7 – Flow graphs

The goal of this TD is to practice solving graph-based problems. In particular, the problem of maximum flow.

Exercise 1 : Water pump

A water company wants to supply water to villages. It has 3 water pumps A, B and C to supply 4 villages D, E, F and G. Table 1 gives the pumps capacities and Table 2 gives the villages demand (in m^3/s).

Pump	Capacity
A	45
B	25
C	30

Table 1: pumps capacities

Village	Demand
D	30
E	10
F	20
G	30

Table 2: villages demand

The geographical situation and the network configuration reduces the capacity of supply. The water network is composed of direct channels between pumps and villages. A channel connects a pump to a village with a maximum debit not to exceed. Table 3 gives the maximum debits of the existing channels

Pompe / Village	D	E	F	G
A	10	15	10	25
B	20		15	5
C		10	5	10

Table 3: Supplying capacity of the network

The problem here is twofold. Can we meet the demand ? How to choose the rates that each pump must deliver in each channel ?

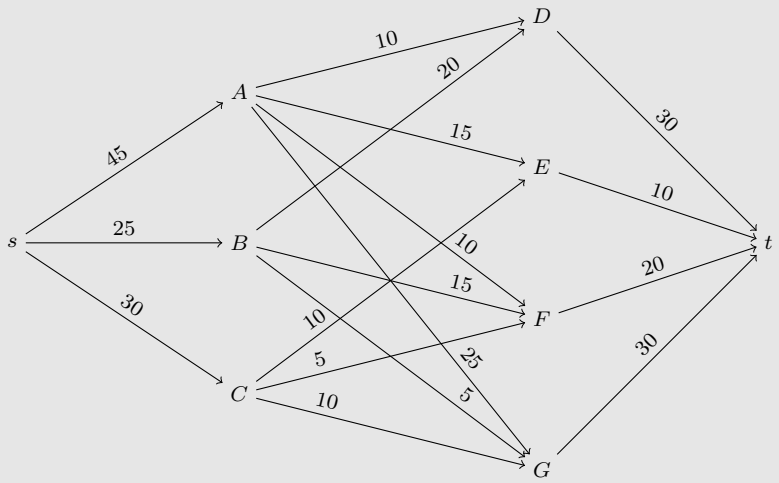
Question 1

Model the problem as a flow problem.

Solution elements :

To do this, we create a graph with a source vertex s, a terminal vertex t, 3 vertices representing the pumps (A, B, C) and finally 4 vertices to represent the villages (D, E, F G).

The arcs of the graph are: from the source to the pump vertices, the capacity of these arcs is the production capacity of the pumps. The pump vertices are connected to the village vertices with the capacity associated to the routing network. Finally each village vertex is connected to the terminal vertex with the demand capacity.



Question 2

Apply the Ford-Fulkerson algorithm.

Solution elements :

1. we can apply a **DFS** at each iteration and find the following augmenting chains:
   - $s \to A \to D \to t : 10$
   - $s \to A \to E \to t : 10$
   - $s \to A \to F \to t : 10$
   - $s \to A \to G \to t : 15$
   - $s \to B \to D \to t : 20$
   - $s \to B \to F \to t : 5$
   - $s \to C \to E \leftarrow A \to G \to t : 10$ (here we unload the flow between E and A we take the edge in the opposite direction)
   - $s \to C \to F \to t : 5$
   - $s \to C \to G \to t : 5$
2. the algorithm **stops** and finds in the set of visited nodes the cut of minimum capacity that is the closest to the source (here it finds the cut : $\{s, A, C, E, G\} \cup \{B, D, F, t\}$).

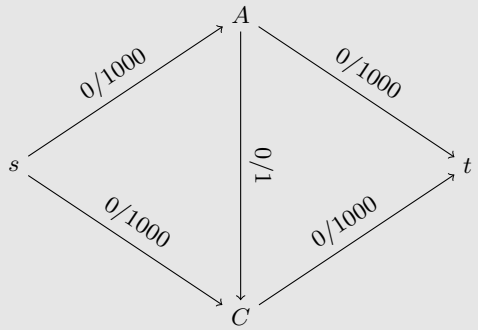
Question 3

What is the complexity of the Ford-Fulkerson algorithm? How could it be reduced?

Solution elements :

The Ford-Fulkerson complexity is $\mathcal{O}((|V| + |E|) \times f_{max})$ (if we have fun increasing the flow unit by unit and thus f_{max} times a traversal $\mathcal{O}(|V| + |E|)$)

The Ford-Fulkerson algorithm searches for any augmenting chain without imposing a search algorithm. The following example can require 2000 steps (1000 augmenting chains $s \to A \to C \to t$ + 1000 chains $s \to C \to A \to t$) or only 2 steps ($s \to A \to t$ and $s \to C \to t$)!



Edmonds-Karp algorithm is a variant that uses a BFS. The augmenting chain found this way is the shortest path in number of edges that has available flow. This solves the previous example in 2 steps. The complexity of the Edmonds-Karp algo is surprisingly $\mathcal{O}(|V| \times |E|^2)$. This algo does not depend on the quantity of the flow and has a polynomial complexity.

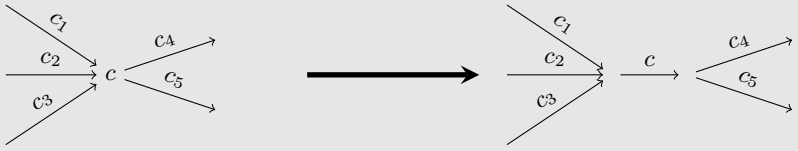
Exercise 2 : Escape problem

Question 1

Consider a variant of the maximum flow problem where the edges and vertices have capacities not to exceed. Suggest a transformation to reduce this problem to the ordinary maximum flow problem.

Solution elements :

We proceed as follows:

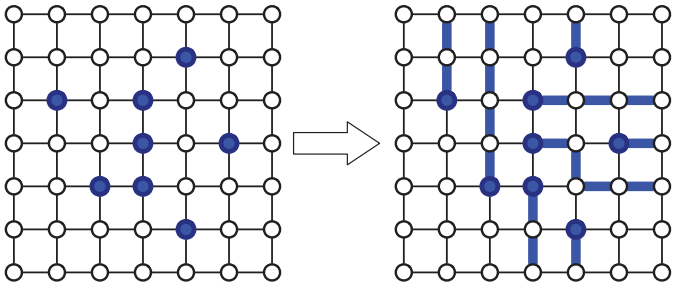


Formally, from a graph $G = (V, E)$ with capacities on the vertices $C_V : V \to \mathbb{R}$ and the edges $C_E : E \to \mathbb{R}$, we create a graph $G' = (V', E')$ with one capacity function $C : E' \to \mathbb{R}$:

- $V' = \{v_{in} \mid v \in V\} \cup \{v_{out} \mid v \in V\}$
- $E' = \{(u_{out}, v_{in}) \mid (u, v) \in E\} \cup \{(v_{in}, v_{out}) \mid v \in V\}$
- $\forall (u, v) \in E, \quad C(u_{out}, v_{in}) = C_E(u, v)$
- $\forall v \in V, \quad C(v_{in}, v_{out}) = C_V(v)$

Question 2

Consider a square grid of $n \times n$ points. There are m dangerous prisoners on m distinct points of the grid ($m < n^2$) . The escape problem is whether there are m disjoint paths (with no common vertices) leading the prisoners outside the grid (the prison) without crossing. The figure below illustrates such escape plan :



Give a solution to the escape problem.

Solution elements :

It can be transformed to a flow problem (to its variant) as follows:

1. Add a source vertex s and connect it to the m starting positions.
2. Add a terminal vertex t and connect it to the $4n - 4$ output points.
3. Each non-oriented u, v edge of the grid is transformed into two oriented (u, v) and (v, u) edges.
4. Each edge and each vertex has a capacity equal to 1.
5. Compute the max flow and check whether it is m?