Eidgenössische
Technische Hochschule
Zürich

Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Federal Institute of Technology at Zurich

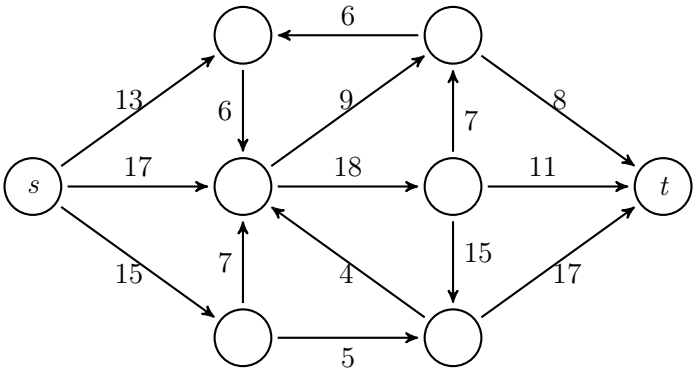
Institut für Theoretische Informatik
Peter Widmayer
Thomas Tschager
Antonis Thomas

18th May 2016

Datenstrukturen & Algorithmen Exercise Sheet 12 FS 16

Exercise 12.1 *Max-Flow by Hand.*

Apply one of the algorithms presented in the lecture for finding a maximum flow from s and t in the following network. The capacities are given next to the corresponding edges. Provide the resulting maximum flow, minimum cut and residual graph.



Exercise 12.2 *Championship Problem.*

Until 1995, in the swiss soccer national league A the so-called *2 point rule* was used: A win gives 2 points, a tie gives 1 point and a loss gives 0 points. If two teams have the same number of points, then the better position in the table is assigned to the team with the better *goal difference*, i.e. the difference of scored goals minus the number of conceded goals. Suppose that there were two more games and the current table looked like this:

	Team	**Points**	**Next opponents**
1)	FC St. Gallen (FCSG)	37	FCB, FCW
2)	BSC Young Boys (YB)	36	FCW, FCB
3)	FC Basel (FCB)	35	FCSG, YB
4)	FC Luzern (FCL)	33	FCZ, GCZ
5)	FC Winterthur (FCW)	31	YB, FCSG

At first glance, FC Luzern may win the championship if both next games are won and FC St. Gallen loses both next games (in this case, both had 37 points, and FC Luzern would win in case of a the better goal difference).

Model the situation above as a flow problem and specify the value that a maximum flow needs to have such that FC Luzern can win the championship (with a better goal difference). Use that in every game exactly two points are distributed among the opponents. Use the max-flow min-cut theorem to conclude that FC Luzern cannot win the championship in any case.

Note: A comparable real situation existed in the german soccer national league 1964/65.

Exercise 12.3 *Edmonds-Karp algorithm (Programming exercise).*

In this exercise we are going to implement the Edmonds-Karp algorithm for the *maximum flow problem*. We define a flow as seen in the lecture: Let $G = (V, E)$ be a directed graph with the edge capacity function $c : E \to \mathbb{Q}^+$, $s \in V$ the source and $t \in V$ the target vertex. A *flow* is a function $f : E \to \mathbb{Q}^+$ such that

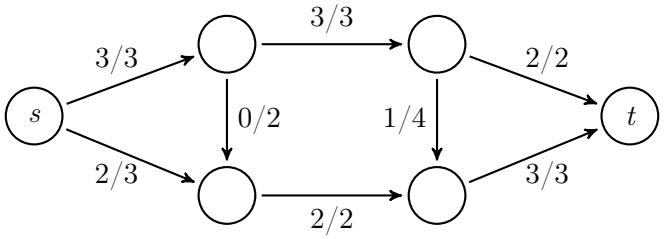
- the flow remains below the capacities, i.e. $\forall e \in E : f(e) \leq c(e)$,
- *Kirchhoff's current law* is satisfied for every vertex except for the source and the sink, i.e.

$$\forall v \in V \backslash \{s, t\} : \sum_{e=(\cdot,v)\in E} f(e) = \sum_{e=(v,\cdot)\in E} f(e).$$

The *value* of the flow f is the overall amount of flow leaving the vertex s minus the amount of flow reaching s, i.e.,

$$\mathsf{val}(f) := \left(\sum_{e=(s,\cdot)\in E} f(e) - \sum_{e=(\cdot,s)\in E} f(e) \right).$$

In the maximum flow problem, the goal is to compute a flow of maximum value. An example of a graph and a maximum flow is shown in the following picture. Next to each edge, there are two numbers: the first one is the flow value while the second one is the capacity of that edge. The overall value of a maximum flow is 5.



Input The first line contains only the number T of test instances. After that, we have exactly one line per test instance containing the description of the input graph $G = (V, E)$ in the form $n, m, u_1, v_1, c_1, ..., u_m, v_m, c_m$. We have $1 \leq n, m \leq 100$ with $V = \{1, ..., n\}$ and $|E| = m$. For every i, $1 \leq i \leq m$, the numbers $u_i, v_i \in \{1, ..., n\}$ define the edge $(u_i, v_i) \in E$ having a capacity of c_i, $1 \leq c_i \leq 10^7$. The source is the first ($s = 1$) and the target the last ($t = n$) vertex.

Output For every test instance we output only one line. This line contains the value of a maximum flow from vertex $s = 1$ to vertex $t = n$.

Example

Input:

```
2
2 1 1 2 1
6 8 1 2 3 1 3 3 2 3 2 2 4 3 3 5 2 4 5 4 4 6 2 5 6 3
```

Output:

```
1
5
```

Directions We provide you with a template. It contains the necessary code to read the input. The graph is then represented by its adjacency list in `ArrayList<Integer>[] graph`, which is an array (of length n) of ArrayLists of integers. Each Arraylist represents the adjacency list of the corresponding vertex. In addition, there are two $n \times n$ arrays: `capacity` holds the capacity of the edge between two vertices and `flow` holds the flow that goes through the corresponding edge. We expect you to implement the Edmonds-Karp algorithm.

There is only one testset for 100 points in this exercise.

Hand-in: Wednesday, 25th May 2016 in your exercise group.