# Homework: The Max-Flow Problem

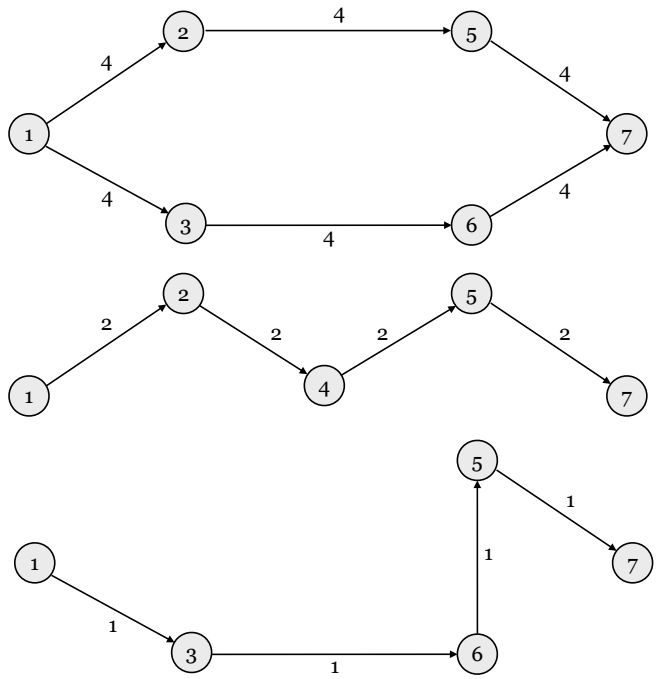
## Solutions

**1.** **(a)** A feasible flow vector $x$, is a flow vector from source $s$ to sink $t$ that is unblocked (i.e., it fulfills the capacity constraints $b_{ij} \leq x_{ij} \leq c_{ij}$, for all $(i, j) \in \mathcal{A}$).

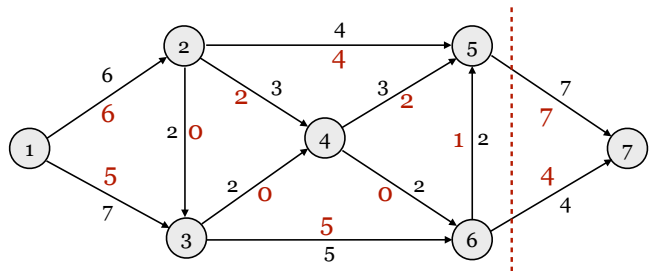
**(b)** Yes, if the lower bound is zero for every arc, the max-flow problem has at least one feasible solution, namely the zero flow vector.

**(c)** It depends; if the paths from source to sink include disjoint feasible flow ranges, the max-flow problem has no feasible solution. If this is not the case, and the capacity constraints involve lower bounds other than zero, the flow $x_{ij}$ can be replaced by a variable $z_{ij} = x_{ij} - b_{ij}$.

**2.** The flows can be summarized as follows:




The final result is



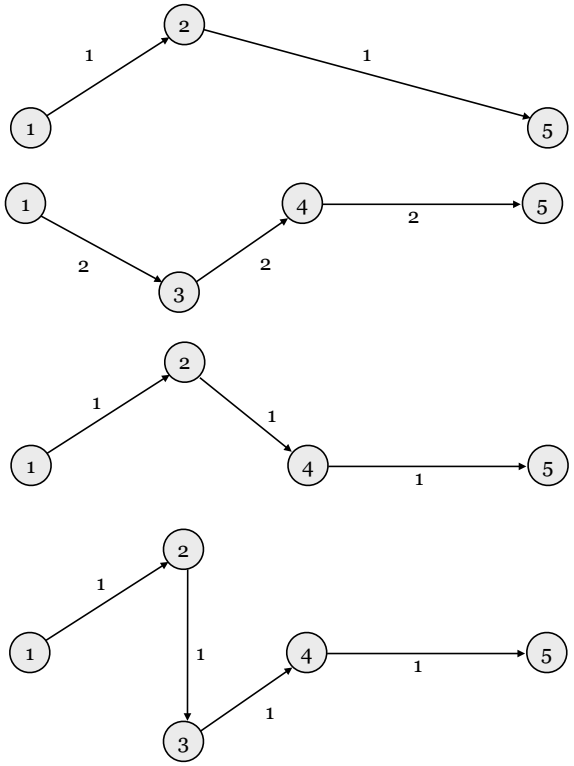

Note that the simplification works in this simple example where the flows are all forward and do not block separate paths. It is not advised to use this approach (intuition), but it helps to understand some concepts.

**3.** **(a)** The cuts are:

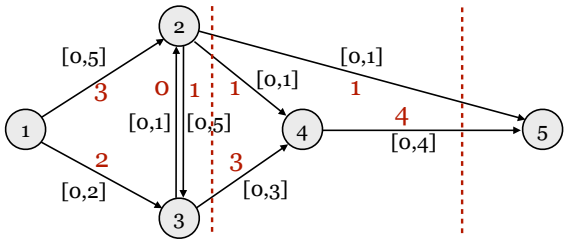
$$\mathcal{S}_1 = \{1\} \qquad C_1 = 5+2 = 7$$
$$\mathcal{S}_2 = \{1,2\} \qquad C_2 = 5+1+1-0+2 = 9$$
$$\mathcal{S}_3 = \{1,3\} \qquad C_3 = 5+1-0+3 = 9$$
$$\mathcal{S}_4 = \{1,4\} \qquad C_4 = 5-0+4-0+2 = 11$$
$$\mathcal{S}_5 = \{1,2,3\} \qquad C_5 = 3+1+1 = 5$$
$$\mathcal{S}_6 = \{1,2,4\} \qquad C_6 = 2-1+5-3+4+1 = 8$$
$$\mathcal{S}_7 = \{1,3,4\} \qquad C_7 = 5+1-0-0+4 = 10$$
$$\mathcal{S}_8 = \{1,2,3,4\} \qquad C_8 = 4+1 = 5$$

**(b)** Minimal saturated cut $Q_5 = [\mathcal{S}_5, \mathcal{N}-\mathcal{S}_5]$; Maximal saturated cut $Q_8 = [\mathcal{S}_8, \mathcal{N}-\mathcal{S}_8]$.

**(c)** The flows can be summarized as follows:




The final result is



**4.** **(a)** The set $\{T_0, T_1, \ldots, T_{k+1}\}$ can be constructed as a BFS tree, where $T_i$ is the set of nodes at the $i$-th level of the tree. The $k+1$-th level contains the node $j$ and the link is labelled either as $(n_k, j)$ or $(j, n_k)$ where $n_k$ is a node at the $k$-th level. Therefore, by starting from the destination node, $j$, and tracing back to the source node, $i$, on either forward or backward links that form a path we can construct the path.

**(b)** The tree provides the shortest simple path from $i$ to node $j$:

$$P = \{i, n_1, n_2, \ldots, n_k, j\} .$$

**(c)** Let $T_0 = \{i\}$. For $k = 0, 1, 2, \ldots$, let

$$T_{k+1} = \{n \notin \cup_{p=0}^{k} T_p | \text{for some node } m \in T_k, (m, n)\},$$

and mark each node $n \in T_{k+1}$ with a label "$(m, n)$" where $m$ is a node of $T_k$ such that $(m, n)$ is an arc. The algorithm terminates if either (1) $T_{k+1}$ is empty or (2) $j \in T_{k+1}$.