# Algorithms and Data Structures 2020/21
# Week 7 tutorial sheet

Below are a list of *suggested* exercises. You should also see the tutorial as a resource to get answers to questions you have, don't feel compelled to stick to the sheet.

1. Given a flow network $\mathcal{N} = (\mathcal{G} = (V, E), c, s, t)$, let $f_1$ and $f_2$ be two flows in $\mathcal{N}$ (so $f_1, f_2$ are functions from $V \times V$ to $\mathbb{R}$ satisfying the three flow properties). The *flow sum* $f_1 + f_2$ is the function from $V \times V$ to $\mathbb{R}$ defined by:

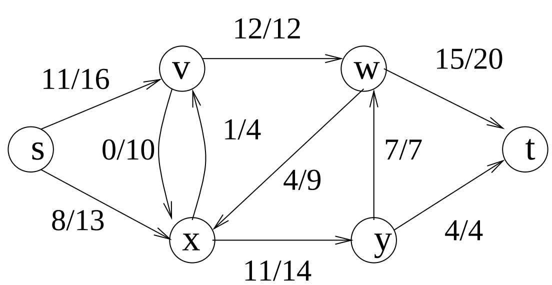
   $$(f_1 + f_2)(u, v) \;=\; f_1(u, v) + f_2(u, v)$$

   for all $u, v \in V$.

   With this definition, which of the three flow properties must the flow sum $f_1 + f_2$ satisfy, and which might it violate?

   *This is Ex. 26.1-6 of [CLRS] (ed 2).*

2. Consider the flow network below (for each arc of the network, the first number is the current flow along that arc, and the second number is the capacity of the arc). We write $f$ for the flow function.




   Two questions:

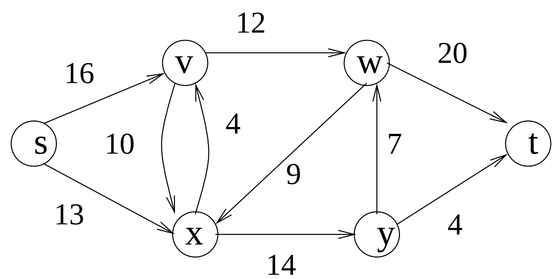
   (a) Find a pair of subsets $X, Y \subseteq V$ such that $f(X, Y) = -f(V - X, Y)$.

   (b) Find a different pair of subsets $X, Y \subseteq V$ such that $f(X, Y) \neq -f(V - X, Y)$.

   **Hint:** This does not have as much to do with this particular network as you might initially think.

   *This is Ex. 26.1-5 of [CLRS] (ed 2).*

3. Execute the Ford-Fulkerson algorithm (*using the Edmonds-Karp heuristic*) on the Network below:




(notice that this is the same network as in question 1, except in this case, there is no flow constructed yet).

*This is Ex. 26.2-2 [CLRS] (ed 2). Similar to Ex 26.2-1 of (ed 3).*

4. A well-known problem in graph theory is the problem of computing a *maximum matching* in a *bipartite graph* $\mathcal{G}$. Give an algorithm which shows how to solve this problem in terms of the network flow problem.

   Definitions: A (undirected) graph $\mathcal{G} = (V, E)$ is *bipartite* if we have $V = L \cup R$ for two disjoint sets $L, R$, such that for every edge $e = (u, v)$ exactly one of the vertices $u, v$ lies in $L$, and the other in $R$. A *matching* in an (undirected) graph $G$ is a collection $M$ of edges, $M \subseteq E$, such that for every vertex $v \in V$, $v$ belongs to *at most one* edge of $M$. A *maximum matching* is a matching of maximum cardinality (for a particular graph).