Exercise 7 - Problem Set 7

- **GitHub** https://github.com/JanPastorek/1-AIN-413-22-Graphs

Today's exercises are based on:

- Stanoyevitch, A. (2011). *Discrete structures with contemporary applications*. CRC Press, Taylor & Francis Group.
- West, D. B. (2001). *Introduction to graph theory* (2nd ed). Prentice Hall.
- https://brilliant.org/wiki/ford-fulkerson-algorithm/
- https://brilliant.org/wiki/edmonds-karp-algorithm/Ford-Fulkerson
- https://www.youtube.com/watch?v=RppuJYwlcI8&ab_channel=WilliamFiset
- Grimaldi, R. P. (2006). *Discrete and combinatorial mathematics*

transport (flow) network (Grimaldi 13.1)

Let $N = (V, E)$ be a loop-free connected directed graph. Then N is called a *network*, or *transport network*, if the following conditions are satisfied:

a) There exists a unique vertex $a \in V$ with $id(a)$, the in degree of a, equal to 0. This vertex a is called the *source*.

b) There is a unique vertex $z \in V$, called the *sink*, where $od(z)$, the out degree of z, equals 0.

c) The graph N is weighted, so there is a function from E to the set of nonnegative integers that assigns to each edge $e = (v, w) \in E$ a *capacity*, denoted by $c(e) = c(v, w)$.

Flow (Grimaldi 13.2)

Definition 13.2 If $N = (V, E)$ is a transport network, a function f from E to the nonnegative integers is called a *flow* for N if

a) $f(e) \leq c(e)$ for each edge $e \in E$; and

b) for each $v \in V$, other than the source a or the sink z, $\sum_{w \in V} f(w, v) = \sum_{w \in V} f(v, w)$. (If there is no edge (v, w), then $f(v, w) = 0$.)

- what goes in → goes out (conservation of flow)

(flow) saturation of edges

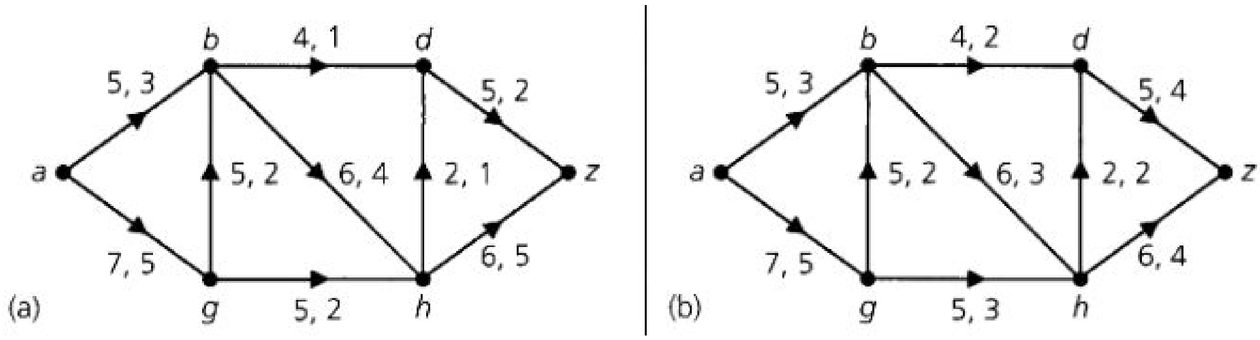
Definition 13.3 Let f be a flow for a transport network $N = (V, E)$.

a) An edge e of the network is called *saturated* if $f(e) = c(e)$. When $f(e) < c(e)$, the edge is called *unsaturated*.

b) If a is the source of N, then $\text{val}(f) = \sum_{v \in V} f(a, v)$ is called the *value of the flow*.

1. Decide whether these are flows:
2. Which edges are saturated?

3. Calculate *value of the flow*.



$MAX - FLOW(G) \rightarrow$ flow that achieves the greatest possible value

Definition 13.4 If $N = (V, E)$ is a transport network and C is a cut-set for the undirected graph associated with N, then C is called a *cut*, or an *a-z cut*, if the removal of the edges in C from the network results in the separation of a and z.

The *capacity of a cut*, denoted $c(P, \overline{P})$, is defined by

$$c(P, \overline{P}) = \sum_{\substack{v \in P \\ w \in \overline{P}}} c(v, w),$$

Calculate capacity of cut C_1 and C_2

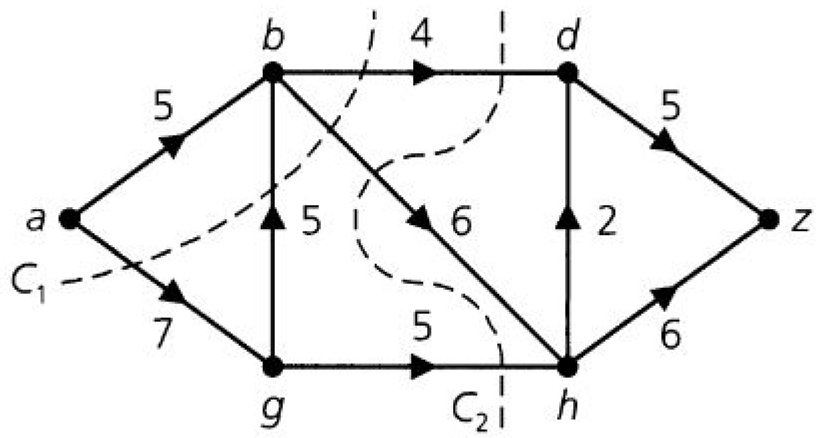
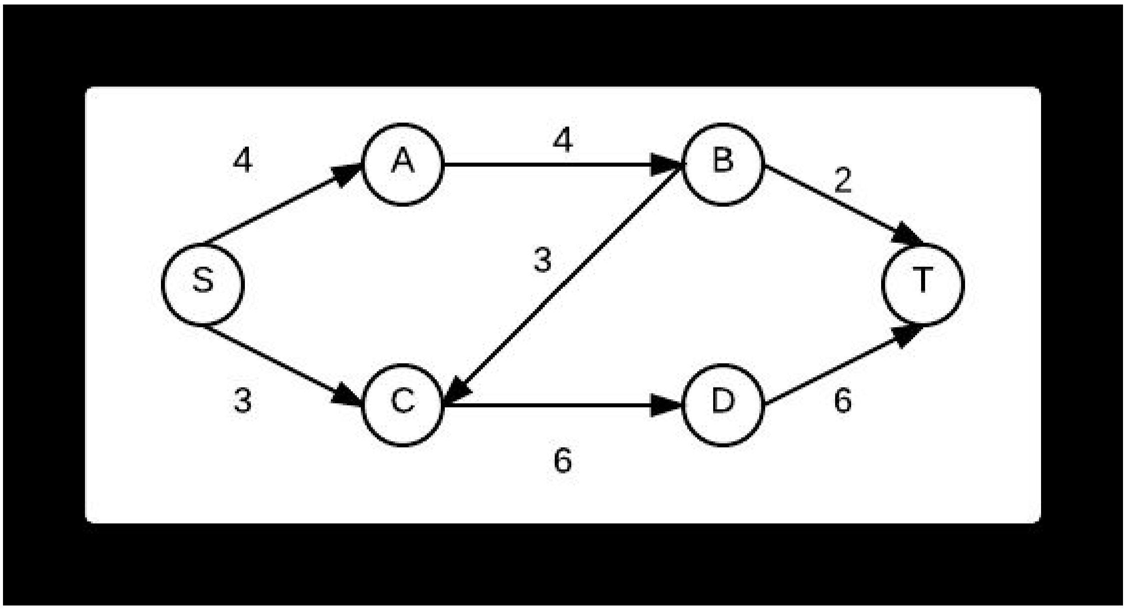


Figure 13.11



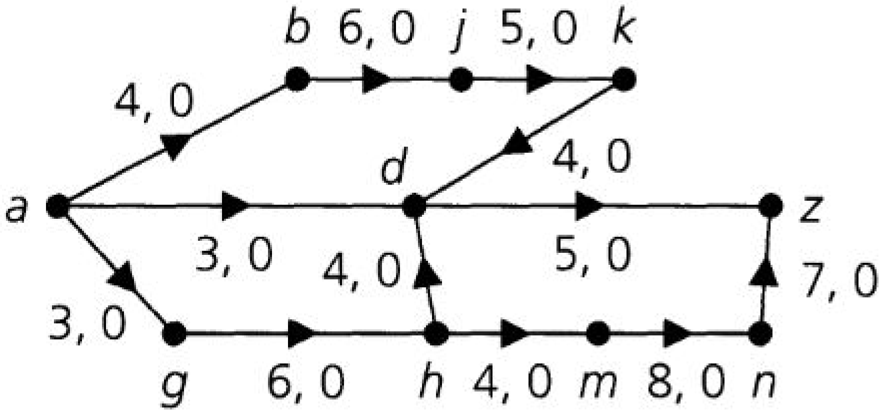
Ford-Fulkerson & Edmonds-Karp

The intuition goes like this:

- as long as there is a path from the source to the sink that can take some flow the entire way, we send it. This path is called an *augmenting path*.
- We keep doing this until there are no more augmenting paths.

```
initialize flow to 0
path = findAugmentingPath(G, s, t)  # this is not specified how → DFS / BFS (EdmondsKarp)
while path exists:
    augment flow along path                 #This is purposefully ambiguous for now
    G_f = createResidualGraph()
    path = findAugmentingPath(G_f, s, t)
return flow
```

```
flow = 0
for each edge (u, v) in G:
    flow(u, v) = 0
while there is a path, p, from s -> t in residual network G_f: # use DFS / BFS to find p: s -> t
    # residual network is reconstructed from original graph $c_f(u,v)=c(u,v)-f(u,v)$.
    residual_capacity(p) = min(residual_capacity(u, v) : for (u, v) in p)   # the min is the
bottleneck value
    flow = flow + residual_capacity(p)
    for each edge (u, v) in p:
        if (u, v) is a forward edge:
            flow(u, v) = flow(u, v) + residual_capacity(p)
        else:
            flow(u, v) = flow(u, v) - residual_capacity(p)
return flow
```



(i) val(f) = 0

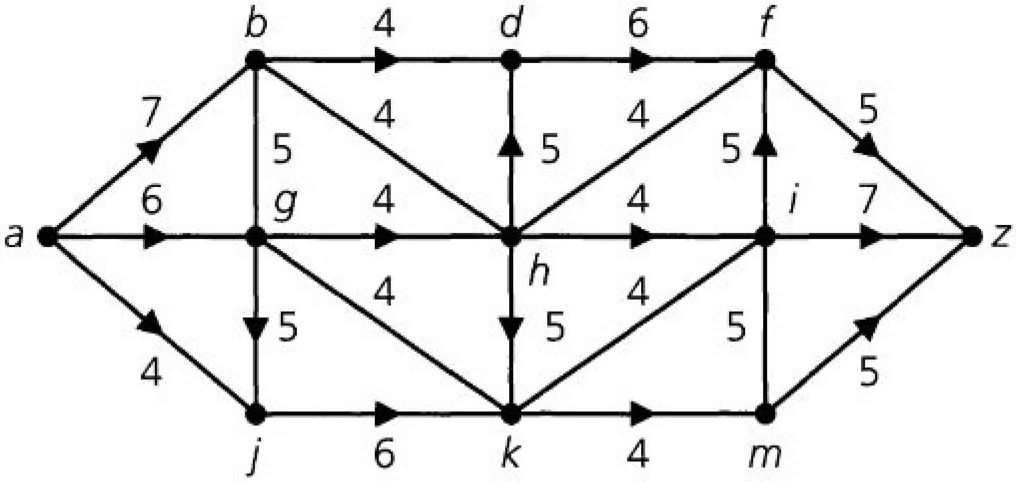


Figure 13.23