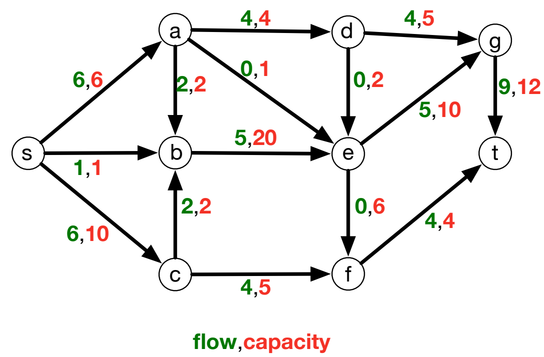
Solutions to Flow Network Practice Problems

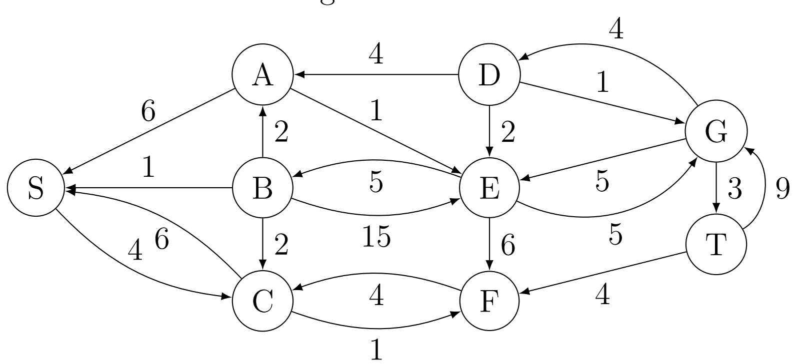
**Practice problems:**

## 1. [DPV] Problem 7.10 (max-flow = min-cut example)

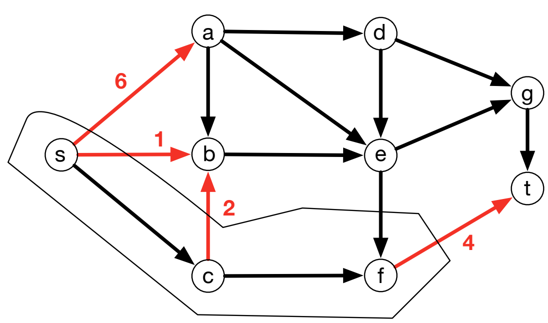
Here is a max flow in the given flow network:




The residual network $G^f$ is the following:



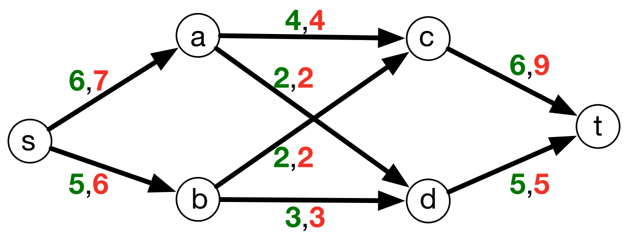

Looking at the residual network $G^f$, the set $L$ of vertices reachable from $s$ in $G^f$ is $L = \{s, c, f\}$. This set $L$ has capacity $13 = 6 + 1 + 2 + 4$. Note the capacity of the cut is determined by the original capacities, it does not depend on the flow. The capacity of this st-cut matches the size of the flow $f$ and hence $f$ is a max-flow and $L$ defines a min-st-cut. Here is an illustration of this min-st-cut:



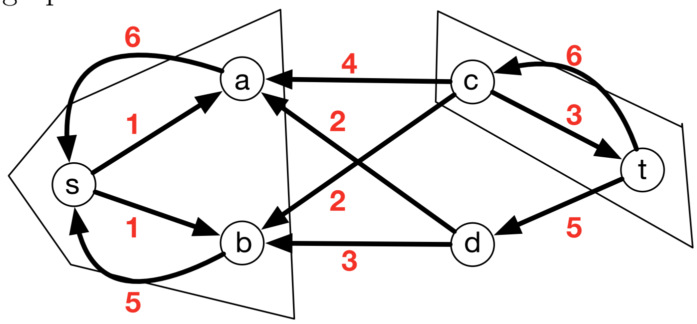
## 2. [DPV] Problem 7.17 (bottleneck edges)

### Parts (a) and (b):

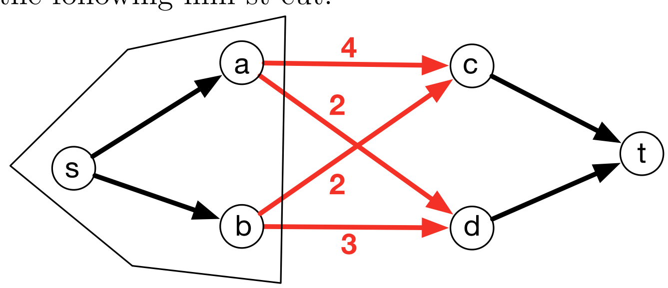
Here is the max flow in the given flow network:




Here is the residual graph $G^f$ for the above flow:




In $G^f$ the set of vertices reachable from $s$ is $\{s, a, b\}$ and the set of vertices that can reach $t$ is $\{c, t\}$. This gives the following min-st-cut:




Notice that the set $\{s, a, b\}$ has capacity $11 = 4 + 2 + 2 + 3$ which matches the size of the above flow.

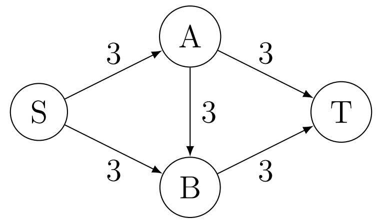
### Part (c):

An edge $\overrightarrow{uv}$ in the original flow network $G$ is a *bottleneck edge* if increasing its capacity results in an increase in the size of the maximum flow.

There are two bottleneck edges in the above network, they are the edges $\overrightarrow{ac}$ and $\overrightarrow{bc}$.

### Part (d):

Here is an example of a flow network with 4 vertices and no bottleneck edges:




Alternatively, in the flow network from question 7.17, if the capacity of the edge $\overrightarrow{ct}$ was reduced from 9 to 6 then there will be no bottleneck edges in this flow network.

**Part (e):**

Here is the general algorithm for finding all of the bottleneck edges in the flow network $G$.

We start by finding a maximum flow $f$ for the flow network $G$. Consider an edge $\overrightarrow{vw}$ in the flow network $G$. Increasing the capacity of $\overrightarrow{vw}$ results in an increase in maximum flow value if and only if there exists a path from $s$ to $v$ and a path from $w$ to $t$ in $G^f$. This is because if there exists these two paths then more flow can be sent from $s$ to $u$, then along the edge $\overrightarrow{vw}$, and finally from $v$ to $t$.

Therefore, our algorithm for finding bottleneck edges is as follows:

(1) Find a maximum flow $f$ on $G$.

(2) Run Explore from $s$ in $G^f$. Let $S$ be the set of vertices reachable from $s$ in $G^f$.

(3) Run Explore from $t$ in the reverse graph of $G^f$. Let $T$ be the set of vertices reachable from $t$ in the reverse graph of $G^f$; note the set $T$ are those vertices which can reach $t$ in $G^f$.

(4) For each $\overrightarrow{vw} \in E(G)$, output $\overrightarrow{vw}$ as a bottleneck edge if $v \in S$ and $w \in T$.

Since steps 2, 3, and 4 take $O(|V|+|E|)$ time, the running time is dominated by the running time of the maximum flow algorithm in step 1.

Note that this algorithm looks for a path $s \to v$ and $w \to t$. What if these two paths share one or more edges? Then, the joined path will have one or more cycles. So, we can drop that cycle (or cycles) and get a shorter path from $s \to t$, but will this path still go through $(v, w)$? If one of the cycles contains edge $e = (v, w)$, then we have an augmenting path in $G^f$ not using $e$, which would mean $f$ is not a max flow. Hence, $e$ cannot be in any of the cycles, so our algorithm works.

### 4. [DPV] Problem 7.19 (verifying max-flow)

Given a flow network $G = (V, E)$ and a flow $f$, we need to verify if $f$ is a **valid max-flow**.

First, we check whether $f$ is valid. We check if flow $f$ violates edge capacities, i.e., $0 \leq f_e \leq c_e$ for all $e \in E$. We also check if flow $f$ is conserved, i.e., for any node $u \in V \backslash \{s, t\}$, $\sum_{(w,u) \in E} f_{wu} = \sum_{(u,z) \in E} f_{uz}$.

Next, if $f$ is valid (otherwise we return false), we check if $f$ is maximum. Note that $f$ is a maximum flow if and only if there is no augmenting path from $s$ to $t$ in the residual graph. Hence to verify that $f$ is of maximum size, we first construct the residual graph $G^f$. We then run Explore from $s$ on $G^f$ to check if there is a path from $s$ to $t$. If $t$ is reachable from $s$ then there is an augmenting path and hence $f$ is not of maximum size. On the other hand if $t$ is not reachable from $s$ in $G^f$ then we know that $f$ is of maximum size.

Validating the flow takes $O(n+m)$ to check edge capacities and if flow is conserved. A single round of DFS on the flow network is also $O(n + m)$, thus the above-mentioned algorithm runs in linear time.