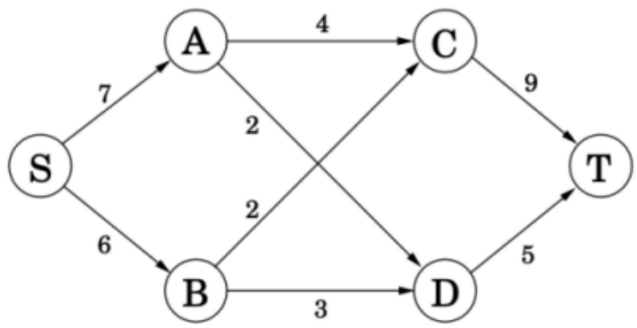
CSC4120 Spring 2024 - Written Homework 10

Yohandi 120040025
Andrew Nathanael 120040007

April 24, 2024

Problem 1.



Consider the above network (the numbers are edge capacities).

(a) Find the maximum flow f and a minimum cut.

(b) Draw the residual graph G_f (along with its edge capacities). In this residual network, mark the vertices reachable from S and the vertices from which T is reachable.

(c) An edge of a network is called a *bottleneck edge* if increasing its capacity results in an increase in the maximum flow. List all bottleneck edges in the above network.

(d) Give a very simple example (containing at most four nodes) of a network which has no bottleneck edges.

(e) Give an efficient algorithm to identify all bottleneck edges in a network. (*Hint: Start by running the usual network flow algorithm, and then examine the residual graph.*)

(a) The maximum flow f is found by using the paths:

- Path $S \to A \to C \to T$, sending 4 units.
- Path $S \to B \to C \to T$, sending 2 units.
- Path $S \to B \to D \to T$, sending 3 units.
- Path $S \to A \to D \to T$, sending 2 units.

A total of $4 + 2 + 3 + 2 = 11$ units are sent by f.

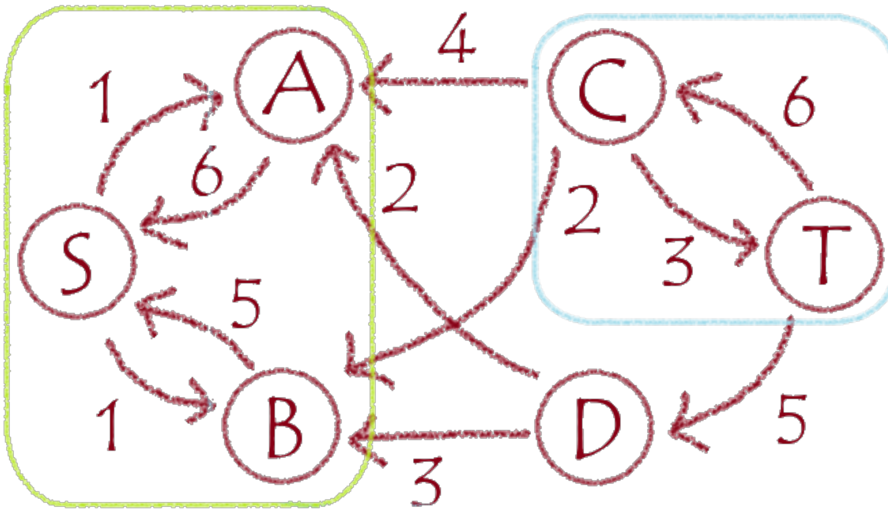
The minimum cut is found by cutting the edges:

- Edge $A \to C$ with the cost of 4 units.
- Edge $B \to C$ with the cost of 2 units.

- Edge $B \to D$ with the cost of 3 units.
- Edge $A \to D$ with the cost of 2 units (alternatively, both edges $B \to D$ and $A \to D$ can be retained, and the edge $D \to T$ with a cost of 5 units can be cut instead).

A total of $4 + 2 + 3 + 2 = 11$ units of cost are incurred as the minimum cut.

(b) The residual graph G_j is drawn as follows:



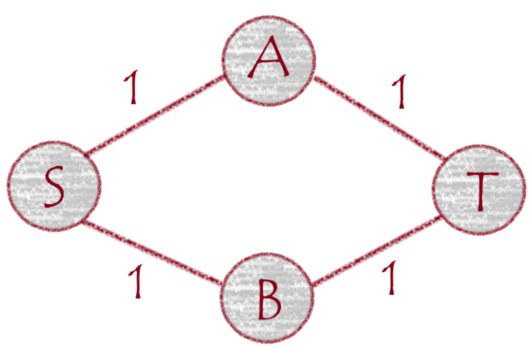
The vertices reachable from S and the vertices from which T is reachable are inside the green and blue rounded rectangles, respectively.

(c) The bottleneck edges in the network are:

- $A \to C$
- $B \to C$

Increasing the capacity of any of those bottleneck edges would increase the maximum flow.

(d) Consider the following network:



Increasing the capacity of any of those edges does not increase the maximum flow; hence, none of them are bottleneck edges.

(e)

Algorithm 1 Bottleneck Edges in a Network

procedure BOTTLENECKEDGES(G, `source`, `sink`)
    $R \leftarrow$ ResidualGraph(G, `source`, `sink`)
    $E_{\text{bottleneck}} \leftarrow$ empty list

    for each edge (u, v) in G's edges **do**
        $\tilde{R} \leftarrow R$
        if $(u, v) \notin R$ **then**
            create a corresponding edge (u, v) with 0 weight in $\tilde{R}$
        end if
        Increase (u, v) in $\tilde{R}$ by 1
        Perform Breadth/Depth-First Search to check if T is reachable from S
        if T is reachable from S **then**
            Append (u, v) to $E_{\text{bottleneck}}$
        end if
    end for
    return $E_{\text{bottleneck}}$
end procedure

Problem 2.

A cohort of k spies resident in a certain country needs escape routes in case of an emergency. They will be traveling using the railway system which we can think of as a directed graph $G = (V, E)$ with V being the cities. Each spy i has a starting point $s_i \in V$, all s_i's are distinct. Every spy needs to reach the consulate of a friendly nation; these consulates are in a known set of cities $T \subseteq V$. In order to move undetected, the spies agree that at most c of them should ever pass through any one city. Our goal is to find a set of paths, one of each of the spies (or detect that the requirements cannot be met).

Model this problem as a flow network. Specify the vertices, edges and capacities, and show that a maximum flow in your network can be transformed into an optimal solution for the original problem. You do not need to explain how to solve the max-flow instance itself.

Construct a new flow network:

- For each vertex v, create two vertices: v_{in} and v_{out}.
- For each vertex v in V, create an edge $(v_{\text{in}}, v_{\text{out}})$ with c capacity.
- For each edge (u, v) in E, create an edge $(u_{\text{out}}, v_{\text{in}})$ with ∞ capacity.

- Create a super-source node S and a super-sink node T.
- For each $s_{i\text{in}}$, create an edge $(S, s_{i\text{in}})$ with 1 capacity.
- For each vertex $t \in T$, create an edge (t_{out}, T) with c capacity.

The maximum flow from S to T in this network represents the optimal routing of spies under the given constraints. If the maximum flow equals the number of spies k, all spies can reach a consulate. Otherwise, it is impossible to route all spies without exceeding the maximum city capacity c.