cs 5800 - hw12

Yijing Xiao

December 2021

1 Exercise 26.1-3

Let $G = (V, E)$ be a flow network with a capacity function c. Let s be the source of the network, and let t be the sink. A flow in G is a real-valued function $f : V \times V \to R$ that satisfies the following two properties:

Capacity constraint: For all $u, v \in V$, we require $0 \leq f(u, v) \leq c(u, v)$

Flow conservation: For all $u \in V - \{s, t\}$, we require

$$\sum_{v \in V} f(v, u) = \sum_{v \in V} f(u, v)$$

when $(u, v) \notin E$, there can be no flow from u to v, and $f(u, v) = 0$. We call the nonnegative quantity $f(u, v)$ the flow from vertex u to vertex v. The value $|f|$ of a flow f is defined as

$$|f| = \sum_{v \in V} f(s, v) - \sum_{v \in V} f(v, s)$$

that is, the total flow out of the source minus the flow into the source. Typically, a flow network will not have any edges into the source, and the flow into the source, given by the summation $\sum_{v \in V} f(v, s)$ will be 0. Since u is a vertex for which there is no path $s \leadsto u \leadsto t$, $f(s, u) = f(u, t) = 0$ which means there is no flow in or flow out for vertex u. Since there is no flow passing through u, $f(u, v) = f(v, u) = 0$ for all vertices $v \in V$.

2 Exercise 26.1-4

A flow in G is a real-valued function $f : V \times V \to R$ that satisfies the following two properties:

Capacity constraint: For all $u, v \in V$, we require $0 \leq f(u, v) \leq c(u, v)$

Flow conservation: For all $u \in V - \{s, t\}$, we require

$$\sum_{v \in V} f(v, u) = \sum_{v \in V} f(u, v)$$

Since f_1 and f_2 are flow, we have $0 \le f_1(u,v) \le c(u,v)$ and $0 \le f_2(u,v) \le c(u,v)$. Assume for all α in the range $0 \le \alpha \le 1$, we have:

$$\begin{array}{c|c} \mathbf{0}{*}\alpha \le \alpha f_1(u,v) \le \alpha c(u,v) & \mathbf{0}{*}(\mathbf{1}\text{-}\alpha) \le (1-\alpha) f_2(u,v) \le (1-\alpha)c(u,v) \\ \mathbf{0}\le \alpha f_1(u,v) \le \alpha c(u,v) & \mathbf{0}\le (1-\alpha) f_2(u,v) \le (1-\alpha)c(u,v) \end{array}$$

Let's make summation for two inequality:

$$0 + 0 \le \alpha f_1(u,v) + (1-\alpha) f_2(u,v) \le \alpha c(u,v) + (1-\alpha)c(u,v)$$

$$0 \le \alpha f_1(u,v) + (1-\alpha) f_2(u,v) \le c(u,v)$$

A flow also satisfies Flow conservation: For all $u \in V - \{s,t\}$, we have:

$$\begin{array}{c|c} \sum_{v\in V} f_1(v,u) = \sum_{v\in V} f_1(u,v) & \sum_{v\in V} f_2(v,u) = \sum_{v\in V} f_2(u,v) \\ \alpha \sum_{v\in V} f_1(v,u) = \alpha \sum_{v\in V} f_1(u,v) & (\mathbf{1}\text{-}\alpha) \sum_{v\in V} f_2(v,u) = (1-\alpha) \sum_{v\in V} f_2(u,v) \end{array}$$

After summation, we have:

$$\alpha \sum_{v\in V} f_1(v,u) + (1-\alpha) \sum_{v\in V} f_2(v,u) = \alpha \sum_{v\in V} f_1(u,v) + (1-\alpha) \sum_{v\in V} f_2(u,v)$$

$$\sum_{v\in V} \alpha f_1(v,u) + (1-\alpha) f_2(v,u) = \sum_{v\in V} \alpha f_1(u,v) + (1-\alpha) f_2(u,v)$$

Since $\alpha f_1 + (1-\alpha) f_2$ satisfies the two properties which are capacity constraint and flow conservation, then the flows in a network from a convex set.

3 Exercise 26.2-2

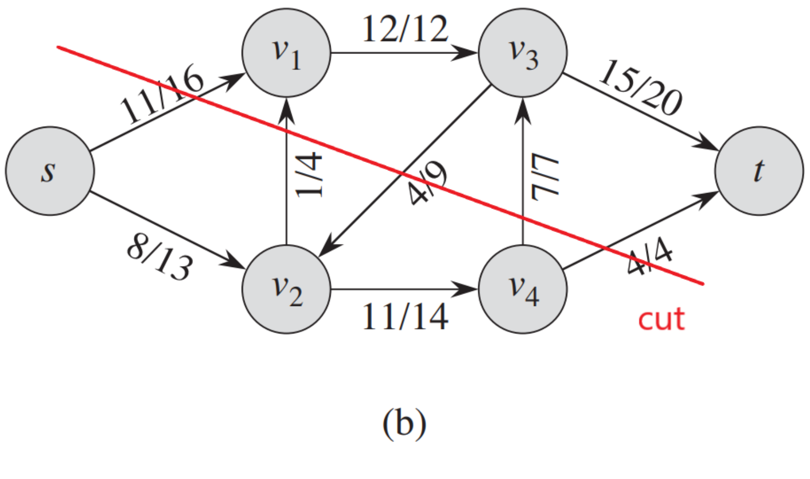

Figure 1: Figure 26.1 (b)

Flow: 11+1+7+4-4 = 19, Capacity: 16+4+7+4 = 31

4 Implement Push-Relabel for finding maximum flow

```
class Node:
    def __init__(self):
        self.h = 0
        self.e = 0
    def relabel(self):
        self.h += 1
    def push(self, flow):
        self.e += flow
    def pour(self, flow):
        self.e -= 1

class Graph:
    def __init__(self, node_size):
        self.size = node_size
        self.node = dict()
        self.edges = [Node() for i in range(node_size+1)]
```

5 Explain in a brief paragraph

Each vertex has an outflow pipe leading to an arbitrarily large reservoir that can accumulate fluid, and all its pipe connections sit on a platform whose height increases as the algorithm progresses. Vertex heights determine how flow is pushed: we push flow only downhill, that is, from a higher vertex to a lower vertex. The flow from a lower vertex to a higher vertex may be positive, but operations that push flow push it only downhill. We fix the height of the source at $|V|$ and the height of the sink at 0. All other vertex heights start at 0 and increase with time. The algorithm first sends as much flow as possible downhill from the source toward the sink. The amount it sends is exactly enough to fill each outgoing pipe from the source to capacity; that is, it sends the capacity of the cut $(s, V - \{s\})$. When flow first enters an intermediate vertex, it collects in the vertex's reservoir. From there, we eventually push it downhill. Since only pipes that leave a vertex u and are not already saturated with flow connect to vertices that are on the same level as u or are uphill from u, in this case, we must increase its height—an operation called "relabeling" vertex u to rid an overflowing vertex u of its excess flow.