Advanced Algorithms and Data Structures
Assignment 1

Isabella Odorico	Caroline Kierkegaard	Mikkel Willén	Fie Hammer
jkd427	qlj556	bmq419	gsr530

November 27, 2023

26.1-1

Supposing that flow network G contains edge (u, v) and we create a new network G' by creating vertex x and replacing (u, v) by new edges (u, x) and (x, v) with $c(u, x) = c(x, v) = c(u, v)$, we show that a maxflow in G' is the same as a maxflow in G.

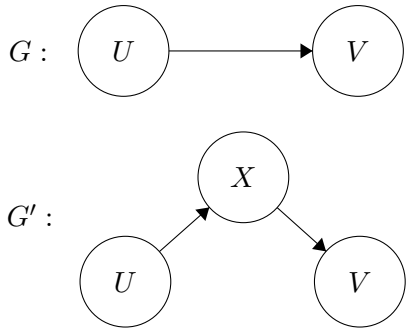

Figure 1: G and G'

Because of flow conservation, the flow entering and leaving u will be the same. In the network G the flow $f(u, v)$ goes from u to v. In the new network G' the exact same flow amount from u enters x instead of u. So $f(u, v) = f(u, x)$. Again because of flow conservation and beacuse we no other edges enters or leaves x, the same amount from u will enter x and leave and enter x. So,

$$f(u, v) = f(u, x) = f(x, v)$$

Since $f(u, v) \leq c(u, v)$ and $c(u, v) = c(u, x) = c(x, v)$, the flow capacity contraint holds.

$$f(u, x) = f(u, v) \leq c(u, v) = c(u, x)$$
$$f(x, v) = f(u, v) \leq c(u, v) = c(x, v)$$

Since nothing else in the network has changed and the flow of the new edges equals the old, the maxflow in G' will be the same value as the max flow in G.

26.1-4

We have that the new flow f' is

$$f' = \alpha f_1 + (1 - \alpha) f_2, \quad \alpha \in [0, 1]$$

For this to be a flow, we have to prove flow conservation and the capacity constraints.

Proof of flow conservation:
Following the definition for flow conservation

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

if we multiply both sides with α and multiply through the sums, we get

$$\sum_{(u,v)\in E} \alpha f(u, v) = \sum_{(v,w)\in E} \alpha f(v, w)$$

which shows flow conservation for each vertex in f_1. We can do the same for f_2 which is:

$$\sum_{(u,v)\in E} (1 - \alpha) f(u, v) = \sum_{(v,w)\in E} (1 - \alpha) f(v, w) \tag{1}$$

showing flow conservation for f_2.

Proof of the capacity constraints:
The capacity of f' is

$$c' = c_1 + c_2$$

and since

$$c_1 \geq f_1 \geq \alpha f_1 \text{ and } c_2 \geq f_2 \geq (1 - \alpha) f_2$$

we have that

$$c' \geq f'$$

26.1-7

For each vertex, we add one outgoing edge (with the capacity of the vertex capacity) to a new vertex. From this vertex we add the outgoing edges of the previous vertex. This way we have inserted the vertex capacity as an edge capacity instead.
So, for each vertex we add an edge to the amount of edges and for each vertex we add a vertex to the amount of vertices.

$$E' = E + V$$
$$V' = 2V$$

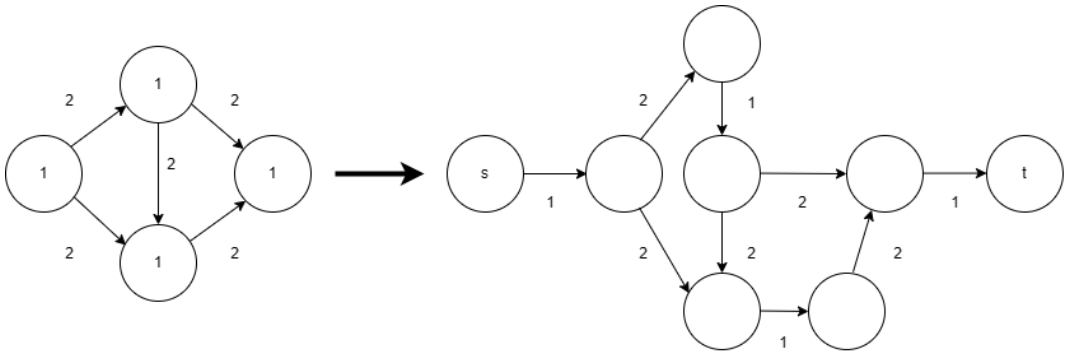

Figure 2: G and G'

26.2-2

We are given the cut $S = \{s, v_2, v_4\}$ and $T = \{v_1, v_3, t\}$ from Figure 26.1(b). The flow $f(S,T)$ across the cut (S,T) is defined to be

$$\begin{aligned} f(S,T) &= \sum_{u \in S} \sum_{v \in T} f(u,v) - \sum_{u \in S} \sum_{v \in T} f(v,u) \\ &= f(s,v_1) + f(v_2,v_1) + f(v_4,v_3) + f(v_4,t) - f(v_3,v_2) \\ &= 11 + 1 + 7 + 4 - 4 \\ &= 19 \end{aligned}$$

The capacity of the cut (S,T) is

$$\begin{aligned} c(S,T) &= \sum_{u \in S} \sum_{v \in T} c(u,v) \\ &= c(s,v_1) + c(v_2,v_1) + c(v_4,v_3) + c(v_4,t) \\ &= 16 + 4 + 7 + 4 \\ &= 31 \end{aligned}$$

26.2-4

In the example of Figure 26.6, the minimum cut corresponding to the maximum flow shown is

$$S = \{s, v_1, v_2, v_4\}, T = \{v_3, t\}$$

A minimum cut of a network is a cut whose capacity is minimum over all cuts of the network and in this case it also has to correspond with the maximum flow, which is 23. The flow $f(S,T)$ is 23 and the capacity $c(S,T)$ is also 23.

Out of the augmenting paths shown in the example, the one that cancels flow is (c). The path cancels the flow from $v_3 \to v_2$ and $v_2 \to v_1$.

26.2-3

Here is an execution of the Edmonds-Karp algorithm on the flow network of Figure 26.1(a).

The first augmented path goes from $s \to v_1 \to v_3 \to t$, where the maximum flow is at the bottleneck, which is 12:

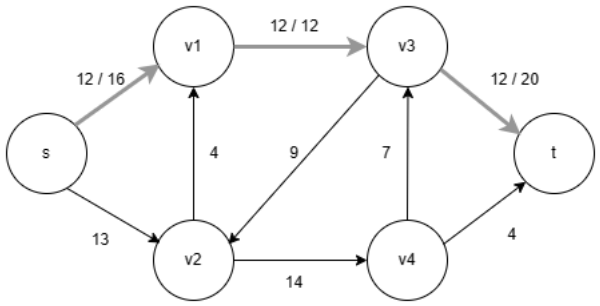

Figure 3: First augmented path

The second augmented path goes from $s \to v_2 \to v_4 \to t$, where the maximum flow is at the bottleneck, which is 4:

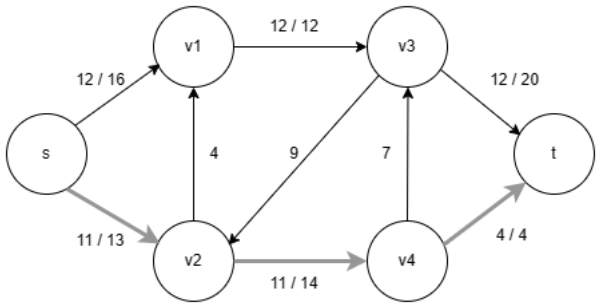

Figure 4: Second augmented path

The last augmented path goes from $s \to v_2 \to v_4 \to v_3 \to t$, where maxflow is the bottleneck, which is 7. This we add to the previous, giving us:

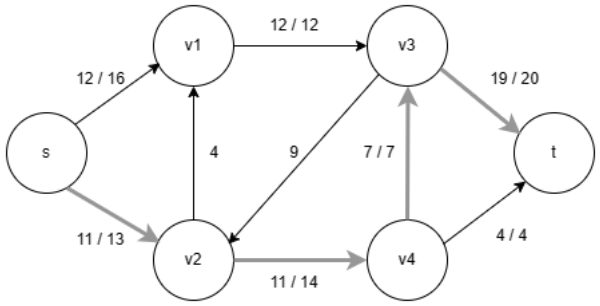

Figure 5: Third augmented path

The minimum cut is defined as: $f(s,t) = c(s,t)$, and thus here, the minimum cut looks like this:

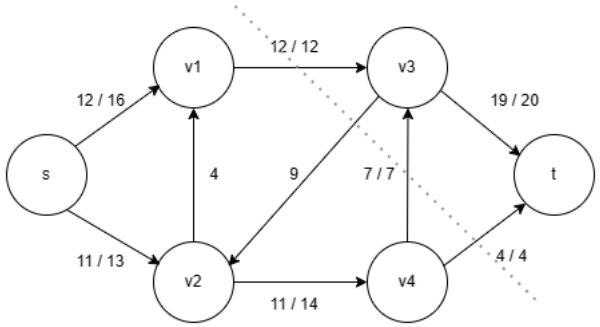

Figure 6: Minimum cut

26.2-7

In order to prove **Lemma 26.2**, we have to prove that f_p is a flow in G_f with the value $|f_p| = c_f(p) > 0$ This we can do in two parts. We first prove that the lemma lives up to flow conversation and that it ensures capacity constraint. These are the two properties a flow f in G needs to satisfy. Then we need to prove that its value will hold true.

1.1: We can see by the lemma, that all $f_p(u,v)$ for all (u,v) values, will be set to $c_f(p)$ as long as the (u,v) value is on p's path. This ensures that it lives up to the flow constraint which states: for all $u \in V - \{s,t\}$, we require: $\sum_{v \in V} f(v,u) = \sum_{v \in V} f(u,v)$. And as every vertex on the path will distribute $c_f(p)$, whether entering or leaving the vertex, we can see that the flow constraint holds true.

1.2: As for the capacity constraint, we have to look at how $f_p(u,v)$ is either $c_f(p)$ or 0. Taking a look at each outcome we see that:

$$f_p(u,v) = c_f(p)$$

$$f_p(u,v) = 0$$

This must mean that we can say that: $f_p(u,v) = c_f(p) \leq c(u,v)$ and $f_p(u,v) = 0 \leq c(u,v)$ and with that, state that: $0 \leq f_p(u,v) \leq c(u,v)$ for any edge $(u,v) \in V$.
This, $0 \leq f_p(u,v) \leq c(u,v)$, is the definition of capacity constraint which is the other property that needs satisfaction to define a flow in G.

2.1 By definition and our lemma, an edge with the capacity of 0 cannot be on the augmented path of p. With this in mind, $c_f(p)$ is defined as the smallest capacity on p. This means that while the smallest capacity will always be chosen, $c_f(p)$ will never be 0. Therefore, $|f_p| = c_f(p) > 0$ must be true.

And with that, we have proven the lemma.

26.2-9

Given a flow f in G and a flow f' in G_f, the augmented flow $f \uparrow f' : V \times V \to \mathbb{R}$ is

$$\left(f \uparrow f'\right)(u,v) := \begin{cases} f(u,v) + f'(u,v) - f'(v,u) & \text{if } (u,v) \in E \\ 0 & \text{otherwise} \end{cases}$$

But for this task, the flow f' is not in G_f, but also in the network G. When we compute $f \uparrow f'$, the computed flow will not necessarily follow the capacity constraint. Here is an example:

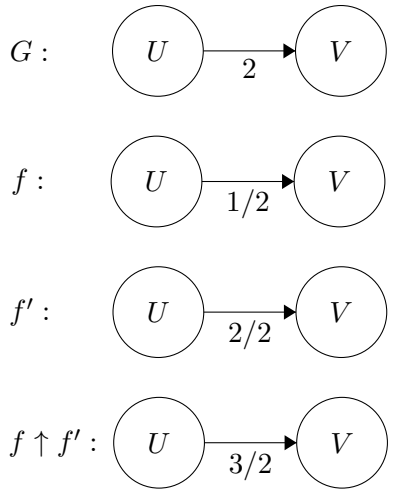

Figure 7: Example of capacity constraint not satisfied

We use the lemma: $f \uparrow f'$ is a flow in G of value $|f \uparrow f'| = |f| + |f'|$.

$$\left|f \uparrow f'\right| = |f| + \left|f'\right| = 1 + 2 = 3$$

The capacity is only 2 and thus the constraint is not satisfied. It makes sense that the constraint is always satisfied, when the flow f' is in G_f, since it would then only be able to use the amount of flow available after the flow f has used its amount.

The augmented flow satisfies the flow conservation property. Because each flow f and f' is a valid flow and thus satisfies the flow conservation property, then we have that for all $u \in V - \{s, t\}$,

$$\begin{aligned}
\sum_{v \in V} \left(f \uparrow f'\right)(u, v) &= \sum_{v \in V} \left(f(u, v) + f'(u, v) - f'(v, u)\right) \\
&= \sum_{v \in V} f(u, v) + \sum_{v \in V} f'(u, v) - \sum_{v \in V} f'(v, u) \\
&= \sum_{v \in V} f(v, u) + \sum_{v \in V} f'(v, u) - \sum_{v \in V} f'(u, v) \\
&= \sum_{v \in V} \left(f(v, u) + f'(v, u) - f'(u, v)\right) \\
&= \sum_{v \in V} \left(f \uparrow f'\right)(v, u),
\end{aligned}$$

The third line follows from the second line by flow conservation, since flow conservation holds for f and f' meaning all flow entering, is equal to all flow leaving in each of the pairs of sums.

26.3-2

The ***Integrality theorem*** states that:

If the capacity function c takes on only integral values, then the maximum flow f produced by the Ford-Fulkerson method has the property that $|f|$ is an integer. Moreover, for all vertices u and v, the value of $f(u, v)$ is an integer.

In order to prove this we will observe the loop invariants within the Ford-Fulkerson method, and use proof by induction on the number of iterations.

Base case (number of iterations is 1):
Within the first `for-loop` the values are set at $(u, v).f = 0$ where all the values must be integers for $(u, v) \in G.E$.
Within the `while-loop` the residual capacity must also be an integer and within the next `for-loop` we either add or subtract this integer to 0, resulting in the flow being an integer.

Induction step:
Like in the base case, we will continue to add or subtract integers from integers, resulting in the flow always being an integer.

Dispositions

Isabella (jkd427)

- Introduction to flows and flow networks
  - Definitions and properties
  - Max flows + short example
- Ford-Fulkerson method
  - Short explanation
  - Residual networks and augmenting paths
  - Definitions of cuts (maxflow and minimal cut)
- Edmonds-Karp algorithm
  - Short explanation
  - Running time + example (if time)

Mikkel (bmq419)

- Introduction
  - Flows and flow networks
  - Maxflow
  - Residual network
  - Augmenting paths
- Edmonds-karp algorithm
  - Short explanation
  - Example
  - Analysis of running time

Fie (gsr530)

- Introduction
  - Properties of a flow
  - Cuts
  - Maxflow
- Ford-Fulkerson
  - Residual networks
  - Augmented paths and edges
  - Example - maxflow and minimal cut (if time)

- Edmonds-Karp
  - Comparison to Ford
  - Running time

Caroline (qlj556)

- Make short example of network
- Introduce Ford-Fulkerson method
- Show max flow on network
- Explain residual networks and augmented paths
- Explain cuts and min cut/max flow theorem
- Introduce Edmonds-Karp algorithm (example and analysis)
- Compare Edmonds-karp with Ford-Fulkerson