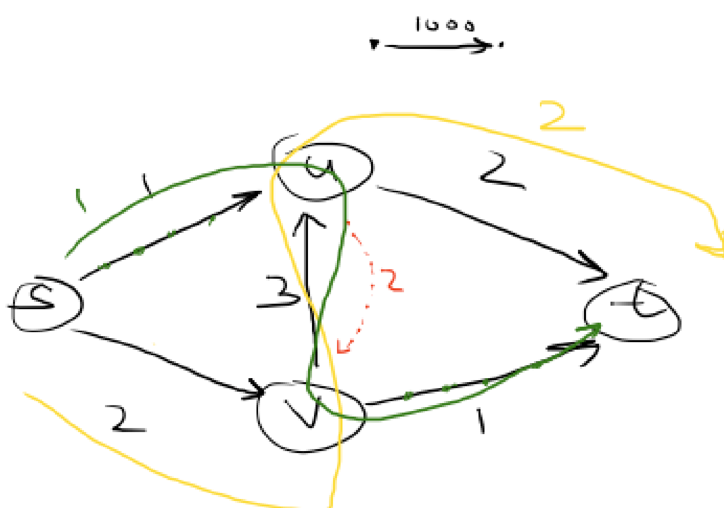
CSCI270 Week10

Jacob Ma

November 8, 2022

1 Proving the Max-Flow/Min-Cut Theorem



1.1 Basic Greedy Algorithmic Idea

Start with the empty flow $f = 0$.
In each iteration, find a path P from s to t on which we can still add flow. Send as much flow on P as possible.
Terminate when no more path exists.

This may terminate with a flow that's not a max-flow - see example above.

To fix this problem, we want to be able to "undo" flow.
To make that formal, we define **backward edges** and **residual graphs**.

1.2 Backward Edges & Residual Graphs

Definition 1.1: Backward Edges & Residual Graphs

Given an input graph G with capacities c and an $s-t$ flow f, the **residual graph** $G(f)$ with respect to the flow f is defined as follows:

- Whenever $f(e) < c(e)$, it contains the **forward edges** e with

$$c'(e) = c(e) - f(e)$$

- Whenever $e = (u, v)$ has flow $f(e) > 0$, $G(f)$ contains the **backward edge** (v, u) with $c'(v, u) = f(e)$.

1.3 Ford-Fulkerson Algorithm

- Start with $f(e) = 0$ for all edges e.
- while the residual graph $G(f)$ still contains an $s-t$ path
  - Let P be an $s-t$ path in $G(f)$
  - Augment f along P (add as much flow as possible along P)

Augmentation of f along P:

- Let $P = (e(1), e(2), \dots, e(k))$
- Let $\epsilon = \min_{\text{all edges } e(i) \text{ in } P} c'(e(i))$
- For each edge $e(i)$ in P:
  - If $e(i)$ is a forward edge, then set $f'(e(i)) = f(e(i)) + \epsilon$
  - Otherwise, $e(i) = (v, u)$ is a backwards edge. Set $f'(u, v) = f(u, v) - \epsilon$
- Return f'

Lemma 1.1.1: Augment is correct.

If f (input) was a flow, then the output f' of Augment is a flow. If f was integral, and all capacities are integers, then f' will be integral. Also, $v(f') > v(f)$.

Proof. To prove that f' is a flow:

(1) Conservation: consider a node v on the path P, with an incoming edge (u, v) and outgoing edge (v, w). Four cases based on whether $(u, v), (v, w)$ are forward/ backwards.

(a) (u, v) and (v, w) are both forward.
Then $f'(u, v) = f(u, v) + \epsilon$ and $f'(v, w) = f(v, w) + \epsilon$. Because f was a flow, in-flow to v was equal to out-flow of v before, and we added ϵ to both, so they are the same.

(b) (u, v) is forward, and (v, w) is backwards.

Then $f'(u,v) = f(u,v) + \epsilon$, $f'(w,v) = f(w,v) - \epsilon$.

$\implies$ total incoming flow into v stays the same. Total outgoing flow also stays the same, because we didn't use outgoing edges.

Because conservation held before, it still holds.

(c) (u,v) and (v,w) are both backward. Same as (1).

(d) (u,v) is backwards, (v,w) is forwards. Same as (2).

(2) Non-negativity:

For forward edges, they were non-negative before and we add something positive.

For backwards edges (u,v): We subtract $\epsilon = \min c'(e) \leqslant c'(u,v) = f(v,u)$. So cannot become negative.

(3) Trivial for backwards edges.

For forward edges (u,v): We add $\epsilon = \min c'(e) \leqslant c'(u,v) = c(u,v) - f(u,v)$, so adding epsilon cannot exceed capacity.

Why does f' stay integral if f was integral?
All residual capacities are integers (because f was integer, and capacities $c(e)$ are integer). So ϵ is an integer.
So all the f will stay integers.
The first edge of P is out of s, and no edge goes into s and $\epsilon > 0$. So flow out of s increases. □

We can now prove that in each iteration, f is a flow, and if all $c(e)$ are integers, it is all integer.
We use induction on the number of iterations.
Base case: $f = 0$ is a flow and integer.
Induction step: Augment Lemma. (Can be applied because by Induction Hypothesis, after $k-1$ iterations, f is valid flow.)

We now want to prove that the output is a **maximum** $s-t$ flow.

Lemma 1.1.2: Cuts are bottlenecks

For an $s-t$ flow f and all $s-t$ cuts $\left(S,\overline{S}\right)$:

$$v(f) \leqslant c\left(S,\overline{S}\right) = \sum_{\substack{e=(u,v)\\ u\in S\\ v\in \overline{S}}} c(u,v)$$

Lemma 1.1.3

For every $s-t$ flow f and $s-t$ cut $\left(S,\overline{S}\right)$:

$$v(f) = \sum_{e \text{ from } S \text{ to } \overline{S}} f(e) - \sum_{e \text{ from } \overline{S} \text{ to } S} f(e)$$

Proof. □