CS 341: ALGORITHMS

Lecture 17: max flow

Readings: CLRS 26.2

Trevor Brown
https://student.cs.uwaterloo.ca/~cs341
trevor.brown@uwaterloo.ca

QUICK REVIEW OF LAST TIME

RECALL: MAX-FLOW MIN-CUT THEOREM

- **Theorem 3:** every max s-t flow has value equal to the capacity of a min s-t cut
- We give an **algorithmic proof** of this theorem
  - (showing that one algorithm solves both max-flow and min-cut at the same time)

FORD-FULKERSON METHOD

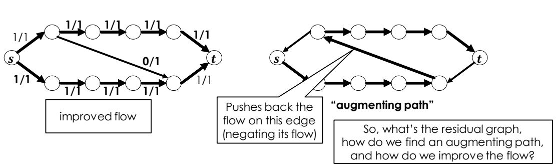
Algorithm development

(mixed together with proof of max-flow min-cut theorem)

FORD-FULKERSON METHOD

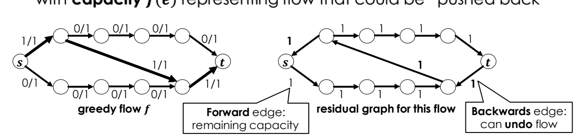
Same Ford as in Bellman-Ford :)

- Can **undo** previous decisions to improve the flow
  - Can effectively "push back" some flow using an **augmenting path** through a **residual graph**



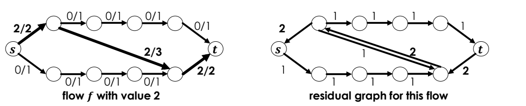
RESIDUAL GRAPH

- A **residual graph** R_f is defined for a **given flow** f and **graph** G
- R_f has the same vertices as G
- For each edge $e = uv$ in G,
  - If $f(e) < c(e)$, then R_f contains a **forward** edge (u, v) with the **remaining capacity** $c(e) - f(e)$
  - If $f(e) > 0$, then R_f contains a **backwards** edge (v, u) with **capacity** $f(e)$ representing flow that could be "pushed back"



ANOTHER EXAMPLE RESIDUAL GRAPH

- Recall: for each edge $e = uv$ in G,
  - If $f(\mathrm{e}) < c(e)$, then R_f contains a **forward** edge (u, v) with the **remaining capacity** $\boldsymbol{c(e) - f(e)}$
  - If $f(e) > 0$, then R_f contains a **backwards** edge $(\boldsymbol{v}, \boldsymbol{u})$ with **capacity** $\boldsymbol{f(e)}$ representing flow that could be "pushed back"



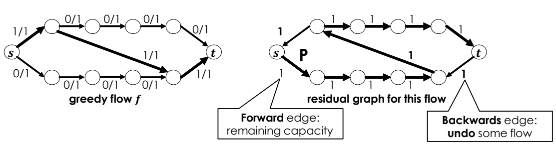
7

CONTINUING WITH NEW MATERIAL

8

FORD-FULKERSON METHOD

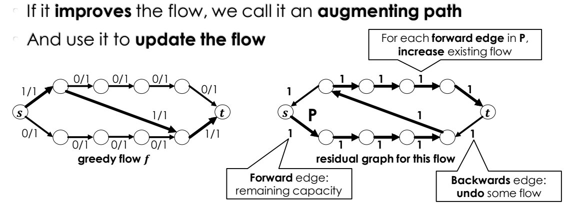
- Find a **shortest path P** from s to t in the **residual graph**
  - If it **improves** the flow, we call it an **augmenting path**
  - And use it to **update the flow**



9

FORD-FULKERSON METHOD

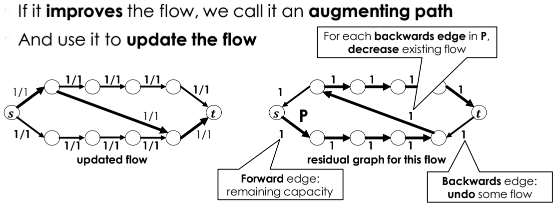
- Find a **shortest path P** from s to t in the **residual graph**
  - If it **improves** the flow, we call it an **augmenting path**
  - And use it to **update the flow**



10

FORD-FULKERSON METHOD

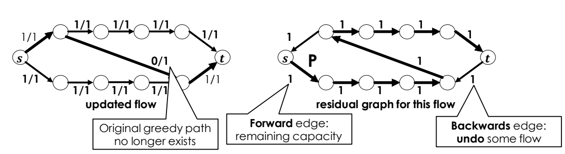
- Find a **shortest path P** from s to t in the **residual graph**
  - If it **improves** the flow, we call it an **augmenting path**
  - And use it to **update the flow**



11

FORD-FULKERSON METHOD

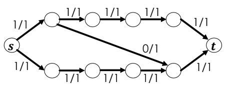
- Find a **shortest path P** from s to t in the **residual graph**
  - If it **improves** the flow, we call it an **augmenting path**
  - And use it to **update the flow**



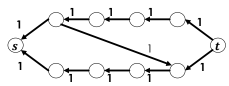
12

FORD-FULKERSON METHOD

- Find a **shortest path P** from s to t in the **residual graph**
  - If it **improves** the flow, we call it an **augmenting path**
  - And use it to **update the flow**



updated flow



new residual graph for this flow

No path from s to t in residual graph. Done!

13

IMPROVING A FLOW f
GIVEN AN AUGMENTING PATH P

no cycles!

- An augmenting path w.r.t a flow f is a **simple** s-t path in R_f
- Let P be an augmenting path w.r.t f
- Let $\text{bottleneck}(f, P)$ be the minimum capacity of an edge in P
- We show this subroutine $\text{augment}(f, P)$ always improves the value of flow f

```
augment(f, P)
    let b = bottleneck(f, P)
    for each edge e = (u,v) in P
        if e is a forward edge
            f(e) = f(e) + b
        else if e is a backwards edge
            let e' = (v,u)
            f(e') = f(e') - b
```

14

LEMMA 4: AUGMENT() IMPROVES FLOW f

- Let f be a flow in G with $f^{in}(s) = 0$, and P be an augmenting path w.r.t f
- Let f' be the resulting flow after running $\text{augment}(f, P)$
- Then **f' is a flow with $\mathbf{value}(f') = \mathbf{value}(f) + \mathbf{bottleneck}(f, P)$**
- That is, $\mathbf{augment}(f, P)$ **increases the flow by $\mathbf{bottleneck}(f, P)$**

15

PROOF

- Claim: $\text{augment}(f, P)$ increases the flow by $\text{bottleneck}(f, P)$
- First check f' is a flow
  - Prove capacity and conservation constraints, and $f'^{in}(s) = 0$
- **Are capacity constraints satisfied?**
  - We add/subtract $\text{bottleneck}(f, P)$ to/from each edge
  - And $\text{bottleneck}(f, P)$ is the minimum of the smallest remaining capacity, and the current flow
  - So capacity constraints are satisfied

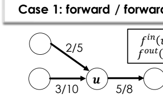
16

PROOF

- Claim: $\text{augment}(f, P)$ increases the flow by $\text{bottleneck}(f, P)$
- **How about conservation of flow?**
  - Consider how the flow into and out of each vertex $u \notin \{s, t\}$ changes as a result of running $\text{augment}(f, P)$
  - We show the change in $f^{in}(u)$ is the same as the change in $f^{out}(u)$
  - There are 4 cases, depending on whether the edges entering/leaving u are **forward** or **backward** edges

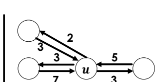
17

Case 1: forward / forward

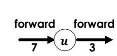


flow $\boldsymbol{f}$

$f^{in}(u) = 5$
$f^{out}(u) = 5$

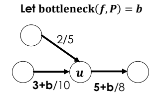


residual graph $\boldsymbol{R_f}$



augmenting path P in R_f

Let $\mathbf{bottleneck}(f, P) = b$



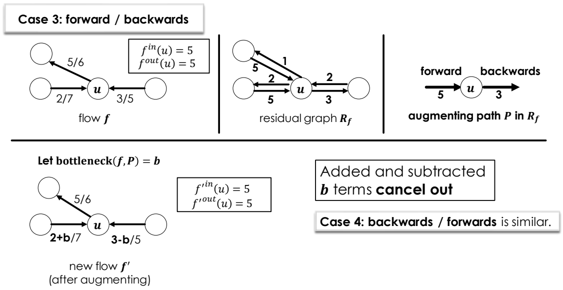
new flow $\boldsymbol{f'}$
(after augmenting)

$f'^{in}(u) = 5 + b$
$f'^{out}(u) = 5 + b$

Both $f^{in}(u)$ and $f^{out}(u)$ are increased by $\text{bottleneck}(f, P)$

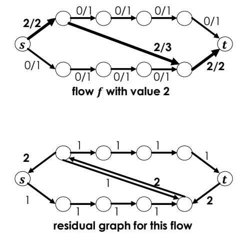
Case 2: backwards / backwards is similar. Both $f^{in}(u)$ and $f^{out}(u)$ are decreased by $\boldsymbol{b}$

18



SHOWING $f'^{in}(s) = 0$

- Last step in showing f' is a flow
  - Prove: s still has no flow into it
- Since f is a flow, $f^{in}(s) = 0$
- To get $f'^{in}(s) > 0$, an augmenting path must include an edge **into s**
- But then an augmenting path starts at s, then returns to s, forming a cycle -- contradiction!



FINISHING LEMMA 4: AUGMENT() IMPROVES FLOW

- Finally we argue $\text{value}(f') = \text{value}(f) + \text{bottleneck}(f, P)$
- f and f' are flows, so $\text{value}(f') = f'^{out}(s)$ and $value(f) = f^{out}(s)$
- We thus show $f'^{out}(s) = f^{out}(s) + \text{bottleneck}(f, P)$
- The augmenting path P is a **simple** path (leaving s exactly once)
- And there is no flow into s, so the edge $e \in P$ leaving s is a **forward edge**
- This means $\text{augment}(f, P)$ **adds** $\text{bottleneck}(f, P)$ to $f(e)$
- So $f'^{out}(s) = f^{out}(s) + \text{bottleneck}(f, P)$

FORD-FULKERSON METHOD

- By Lemma 4, starting from any flow f, if we can **find an augmenting path** P w.r.t f in R_f, then we can use $\text{augment}(f, P)$ to **improve our flow**
- Ford-Fulkerson does this repeatedly **starting** from an **empty flow**

```
FordFulkerson(G=(V,E))
    for e in E
        f(e) = 0

    while there is a simple s-t path P in Rf do
        augment(f, P)
        and update the residual graph Rf
```


What we have proved so far: **augmenting improves flow.**

We **don't know yet** if

1. **we can actually obtain the max flow**, or
2. **whether max-flow = min-cut.**

MAX-FLOW MIN-CUT THEOREM PROOF

PROOF STRATEGY

- Claim: when there is **no augmenting path**, there is a **cut with capacity equal to** the value of **the current flow**.
- Proving this will simultaneously
  - prove the max-flow min-cut theorem,
  - prove correctness of the Ford-Fulkerson method,
  - solve the max flow problem, and
  - solve the min cut problem

PROVING MAX FLOW = MIN CUT

Two directions:
max flow ≤ min cut and **max flow ≥ min cut**

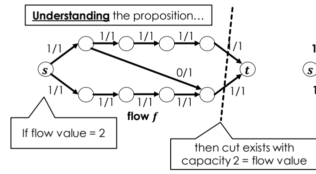
We actually proved the ≤ direction already (**Lemma 2 last time**) when discussing upper bounds for max flow

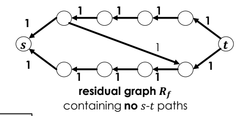
It remains to prove the ≥ direction.

25

PROVING MAX FLOW ≥ MIN CUT

- Proposition: if f is an s-t flow such that there is no s-t path in the residual graph R_f, then there is an s-t cut S s.t. $\text{value}(f) = c^{out}(S)$

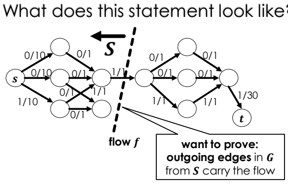


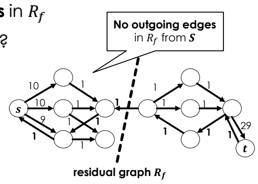


26

PROVING THE PROPOSITION

- Since there is no s-t path in R_f, there is a subset S of vertices with $s \in S$, $t \notin S$ such that S has **no outgoing edges** in R_f
- What does this statement look like?

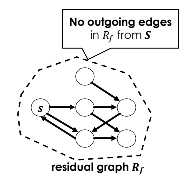


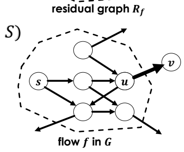


27

PROVING THE PROPOSITION

- Since there is no s-t path in R_f, there is a subset S of vertices with $s \in S$, $t \notin S$ such that S has **no outgoing edges** in R_f
- Claim: $\boldsymbol{c^{out}(S)} = \text{value}(f)$
- Consider two types of edges. Type 1:
  - uv **exiting S in G** ($uv \in \delta^{out}(S)$ in G, $u \in S, v \notin S$)
  - Since S has no outgoing edge in R_f, we know $uv \notin R_f$
  - This implies $f(uv) = c(uv)$, as otherwise uv would be a forward edge in R_f

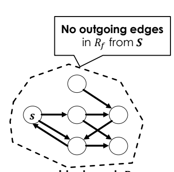


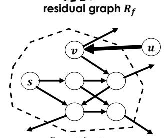


28

PROVING THE PROPOSITION

- Claim: $\boldsymbol{c^{out}(S)} = \text{value}(f)$
- Consider two types of edges. Type 2:
  - uv **entering S in G** ($uv \in \delta^{in}(S)$ in G, $u \notin S, v \in S$)
  - Since S has no outgoing edge in R_f, we know there is no edge $vu \notin R_f$ (note vu would be directed out of S)
  - This implies $f(uv) = 0$, as otherwise vu would be a backwards edge in R_f

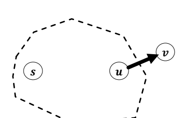


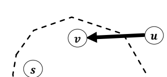


29

PROVING THE PROPOSITION

- We just showed
  - For edge uv directed out of S, $f(uv) = c(uv)$
  - For edge uv directed into S, $f(uv) = 0$
- So $f^{out}(S) - f^{in}(S) = c^{out}(S) - 0 = c^{out}(S)$
- This proves the proposition. I.e., given flow f, if there are no s-t paths in R_f, then **there is a cut matching the flow**





Note this was the last thing remaining to prove the min-cut max-flow theorem, and the correctness of Ford-Fulkerson

30

TIME COMPLEXITY

of the Ford-Fulkerson method

RUNTIME OF FORD-FULKERSON

- Depends on the implementation

```
FordFulkerson(G=(V,E))
    for e in E
        f(e) = 0

    while there is a simple s-t path P in Rf do
        augment(f, P)
        and update the residual graph Rf
```

- How do we find an augmenting path?
- How many times do we need to augment before we terminate?

RUNTIME OF FORD-FULKERSON

- Assume we find any arbitrary augmenting path P, using any technique, in $O(n+m)$ time
- Then every time $\text{augment}(f, P)$ is run, we know only that the flow **increases**
- If capacities are **integers**, the increase is at least 1
- In this case, **if max flow is k then runtime is $O(k(n+m))$**
  - For max flow we assume a connected graph, so this is $\boldsymbol{O(km)}$
  - **Very bad if k is large**

If capacities are reals (and in particular some are irrational), this may **never** terminate!

WORST CASE FOR THIS APPROACH

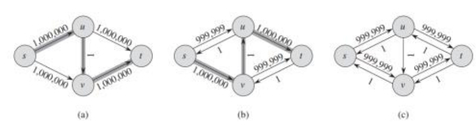


Figure 26.7 **(a)** A flow network for which FORD-FULKERSON can take $\Theta(E\,|f^*|)$ time, where f^* is a maximum flow, shown here with $|f^*| = 2{,}000{,}000$. The shaded path is an augmenting path with residual capacity 1. **(b)** The resulting residual network, with another augmenting path whose residual capacity is 1. **(c)** The resulting residual network.

EDMONDS-KARP APPROACH

- **Use BFS** to find a shortest path (in terms of number of edges) and use that as an augmenting path
- It turns out this always **terminates after $\boldsymbol{O(nm)}$ augmenting paths**
  - (even with real capacities)
- BFS takes $O(n+m)$ time; $\boldsymbol{O(m)}$ since the graph is connected
- **So total runtime is $\boldsymbol{O(nm^2)}$**

There are more sophisticated algorithms with $O(V^2E)$ and even $O(V^3)$ runtimes (**optional**: CLRS 26.4, 26.5)

In 2022, researchers found *an almost linear time algorithm*, which leverages techniques from convex optimization and sophisticated data structures