# CS 31: Algorithms (Spring 2019): Lecture 16

Date: 16th May, 2019

Topic: Graph Algorithms 6: The Ford-Fulkerson Algorithm

*Disclaimer: These notes have not gone through scrutiny and in all probability contain errors. Please notify errors on Piazza/by email to deeparnab@dartmouth.edu.*

---

## 1 The Ford Fulkerson Algorithm

First, we define augmentation along a path in a residual network given the previous lecture's intuition.

**procedure** AUGMENT($G_f, s, t, p$):
▷ *Augment along path $p$ in the residual network $G_f$.*
▷ *Modifies $f(e)$ for every $e \in G$; modifies $u_f(e)$ for every edge $e \in E_f$.*
$\delta := \min_{e \in p} u_f(e)$.
For every edge $e = (x, y) \in p$:

- If $(x, y) \in E$:
  - $f(x, y) \leftarrow f(x, y) + \delta$;
  - $u_f(x, y) \leftarrow u_f(x, y) - \delta$;
  - $u_f(y, x) \leftarrow u_f(y, x) + \delta$;
- If $(x, y) \in E_{\text{rev}}$:
  - $f(y, x) \leftarrow f(y, x) - \delta$; ▷ Note: $(y, x) \in E$
  - $u_f(y, x) \leftarrow u_f(y, x) + \delta$;
  - $u_f(x, y) \leftarrow u_f(x, y) - \delta$;

The following invariants should be checked from the pseudocode above.

**Claim 1** (Invariants of Augmentation)**.**

I1. For every edge $e \in E \cup E_{\text{rev}}$, $u_f(e) \geq 0$
I2. For every edge $(x, y) \in E$, $f(x, y) + u_f(x, y) = u(x, y)$
I3. For every $(x, y) \in E_{\text{rev}}$, $f(y, x) = u_f(x, y)$.

✍

**Exercise:** *Formally prove the above claim.*

**Claim 2.** If $f$ satisfied the capacity constraints before AUGMENT, then it does so after AUGMENT too.

*Proof.* This follows from the Invariants: For any edge $(x, y) \in E$, we have $f(x, y) = u(x, y) - u_f(x, y) \leq u(x, y)$ (from I2 and I1, respectively). Similarly, I1 implies $u_f(y, x) \geq 0$ which in turn implies $f(x, y) \geq 0$. □

**Claim 3.** If $f$ is an $s,t$ flow in $G$ which satisfies flow conservation constraints at every vertex $v \neq s,t$, then the flow after AUGMENT step also satisfies flow conservation constraints at every vertex $v \neq s,t$.

*Proof.* If $v \notin p$, then there is nothing to discuss. So assume $v \in p$. Since $v \notin \{s,t\}$ it is an internal node in $p$ and let $(w,v)$ and $(v,x)$ be the two edges of $p$ incident on it. There are four cases to consider.

- Case 1: $(w,v) \in E, (v,x) \in E$. In this case, both $f(w,v)$ and $f(v,x)$ go up by $\delta$, implying the increase in excess is $0$.
- Case 2: $(w,v) \in E, (v,x) \in E_{\mathsf{rev}}$. In this case, $f(w,v)$ goes up by $\delta$ and $f(x,v)$ goes down by $\delta$, implying the increase in excess is $0$.
- Case 3: $(w,v) \in E_{\mathsf{rev}}, (v,x) \in E$. In this case, $f(v,w)$ goes down by $\delta$ and $f(v,x)$ goes up by $\delta$, implying the increase in excess is $0$.
- Case 4: $(w,v) \in E_{\mathsf{rev}}, (v,x) \in E_{\mathsf{rev}}$. In this case, both $f(v,w)$ and $f(x,v)$ go down by $\delta$, implying the increase in excess is $0$.

□

**Claim 4.** After AUGMENT, the $\mathsf{excess}_f(t)$ goes up by $\delta$.

*Proof.* Let $(v,t) \in p$ be the edge incident on $t$. If $(v,t) \in E$, then $f(v,t)$ increases by $\delta$ and the flow on no other edge incident on $t$ changes, implying $\mathsf{excess}_f(t)$ goes by $\delta$. If $(v,t) \in E_{\mathsf{rev}}$, then $f(t,v)$ decreases by $\delta$ and the flow on no other edge incident on $t$ changes, implying $\mathsf{excess}_f(t)$ goes by $\delta$. □

Now we are ready to describe the maximum flow algorithm. We call it the FORDFULKERSON algorithm after the discoverers Lester Ford and Ray Fulkerson.

```
1: procedure FordFulkerson(G, s, t, u):
2:     Initialize f ≡ 0 and u_f ≡ u and G_f ≡ G.
3:     while true do:
4:         Check if there is an s,t path p in G_f with all u_f(e) = 0 edges removed.
5:         If not, break.
6:         Else, Augment(G_f, s, t, p).
7:     return (f, G_f).
```

**Lemma 1.** If $u(e)$s are integer valued, then FORDFULKERSON returns an integer valued valid $f$ in $O(nmU)$ time, where $U := \max_{e \in E} u(e)$.

*Proof.* Since the $0$-flow is valid, and the Augmentation Claims imply AUGMENT maintains validity, we get that the final flow is valid.

We claim that the Line 4 in AUGMENT will set $\delta$ to a positive integer valued. To see this, we need to prove $u_f$ is integer valued. But this is true in the beginning (when $u_f \equiv u$),

and since subsequently $f$ is augmented in $\delta$-installments, the $f$ is always integral which in turn leads to $u_f$ being integral.

Furthermore, each time $\text{excess}_f(t)$ grows by $\delta \geq 1$. Since the final flow is valid, the total value of this flow $\text{excess}_f(t) \leq nU$ since there can be at most $n$ edges of the form $(v, t)$ and each has capacity at most $U$. Thus, the algorithm terminates in $O(nU)$ rounds. Finally, note each round takes $O(n + m)$ time. Why? □

✍

**Exercise:** *Show an example of a network with $U := \max_{e \in E} u(e)$ where* FORDFULKERSON *takes $\Omega(U)$ iterations. In your example you may feed the algorithm any path $p$ from $s$ to $t$ in $G_f$ with $\min_{e \in p} u(e) > 0$.*

Now comes the main crux.

**Lemma 2.** The flow $f$ returned by FORDFULKERSON when it terminates is a maximum valued flow.

*Proof.* We prove the lemma by invoking Corollary from previous lecture. We describe a cut induced by a subset $S$ which satisfied the properties of the corollary. In fact

$$S = \{v : v \text{ is reachable from } s \text{ in } G_f \text{ with all } u_f(e) = 0 \text{ edges removed.}\}$$

Clearly, $s \in S$. Since the algorithm terminates, $t \notin S$.

Now fix an $(x, y) \in \partial^+(S)$. Since $y$ is not reachable from $s$ using positive residual capacity edges, we get $u_f(x, y) = 0$. By I2, this implies

$$\text{For } (x, y) \in \partial^+(S), \quad f(x, y) = u(x, y)$$

Now consider an $(x, y) \in \partial^-(S)$. Since $x$ is not reachable from $s$ using positive residual capacity edges, we get $u_f(y, x) = 0$ for $(y, x) \in E_{\text{rev}}$. That is,

$$\text{For } (x, y) \in \partial^-(S), \quad f(x, y) = 0$$

But these are precisely the conditions of the corollary. Thus, $f$ is a maximum $s, t$ flow and $S$ is a minimum $s, t$ cut. □

Putting everything together,

**Theorem 1.** Given a flow network $(G, s, t, u)$ where $u(e)$ is a positive integer for every edge $e \in E(G)$, the FORDFULKERSON algorithm finds a maximum $s, t$ flow which is integral, and a minimum $s, t$ cut in $O(nmU)$ time.

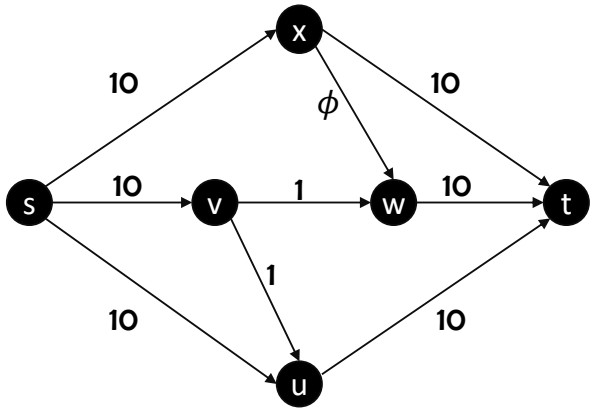
The following important theorem follows as a corollary.

**Theorem 2** (The Max-Flow-Min-Cut Theorem.)**.** The maximum $s, t$ flow in any network equals the minimum $s, t$ cut.

## 1.1 The issue with FORDFULKERSON and real capacities

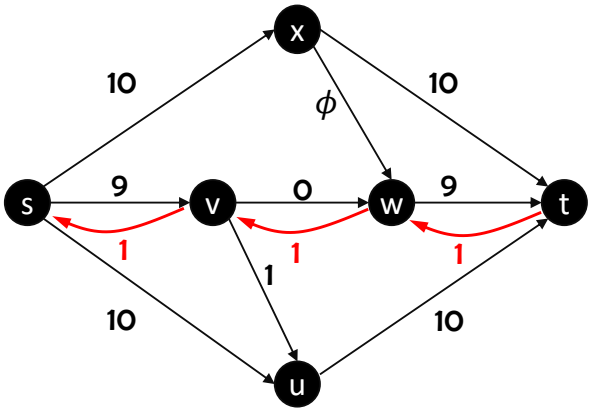
*This subsection was not covered in class.*

Lemma 2 says that the flow returned by FORDFULKERSON *when* it terminates is a maximum flow. Lemma 1 proves that the algorithm terminates when $u(e)$'s are integer valued. The exercise right after the Lemma asks you to find an example Interestingly, the algorithm *may not even terminate* if the $u(e)$'s are irrational numbers! As the example below shows, in each iteration the value of $\min_{e \in p} u(e)$ keeps decreasing geometrically leading to a situation which never ends. More frustratingly, the maximum flow value it converges to is also *far* from the maximum flow. The example is the following network in Figure 1.1. The value $\phi := \frac{\sqrt{5}-1}{2}$ is (one of) the irrational numbers satisfying $\phi^2 + \phi - 1 = 0$.

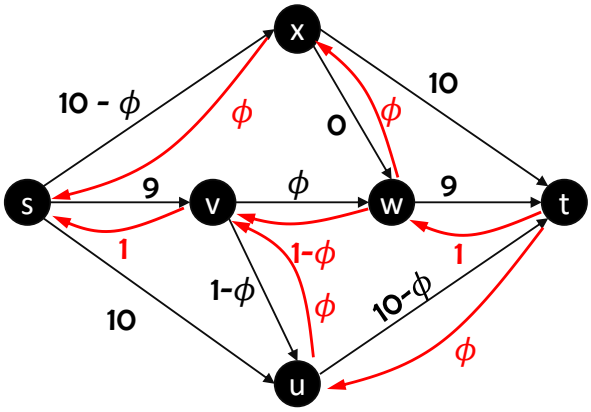



Before we describe the "bad paths" for FORDFULKERSON let's just note that the maximum flow value for the above network is $21$. We can see this as follows: send $10$ units of flow on the path $(s, x, t)$; send $10$ units of flow on the path $(s, u, t)$, and then send $1$ unit of flow along the path $(s, v, w, t)$. To see this is the maximum flow, consider the cut $S = \{s, v, u\}$; note that $\partial^+ S := \{(s, x), (v, w), (u, t)\}$ of capacity $21$. So, if we had indeed chosen the above three paths to augment on, FORDFULKERSON would've terminated to the correct answer in $3$ rounds. Instead consider what happens next. We use $F$ to maintain the value of the flow.

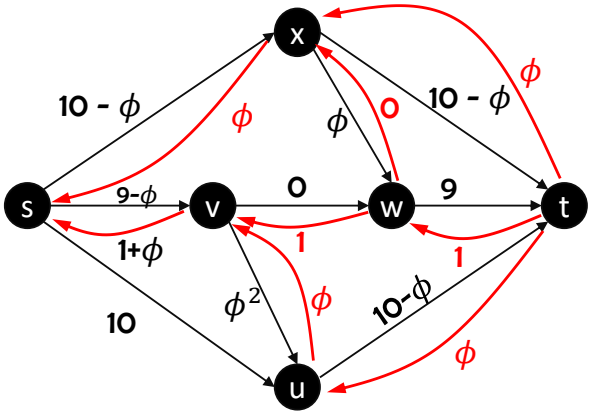
In Iteration $1$, we provide the path $(s, v, w, t)$ to the algorithm. We get $\delta_1 := 1$ and $F = 1$. After sending this flow, the residual network is this.



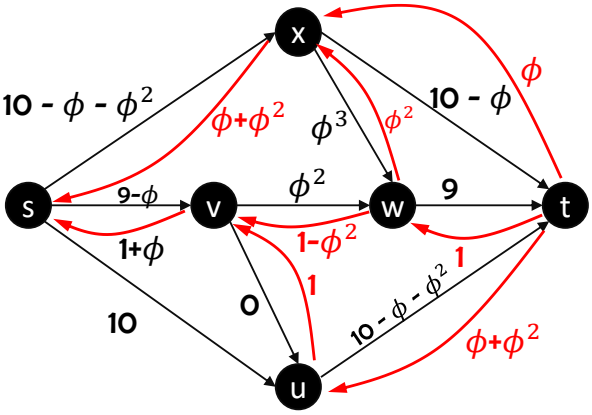

In Iteration 2, we provide the path $(s, x, w, v, u, t)$. How much flow can we send on this path? Since $\phi < 1$, we see that $\delta_2 := \phi$. This makes the current value $F = 1 + \phi$. After sending this flow, the residual network is this (actually, draw it yourself and then check).




In Iteration 3, we provide the path $(s, v, w, x, t)$. Note that $\delta_3 = \phi$ again, and the current flow becomes $F = 1 + 2\phi$. After sending this, the residual network is (you are drawing first, right, and then checking?)

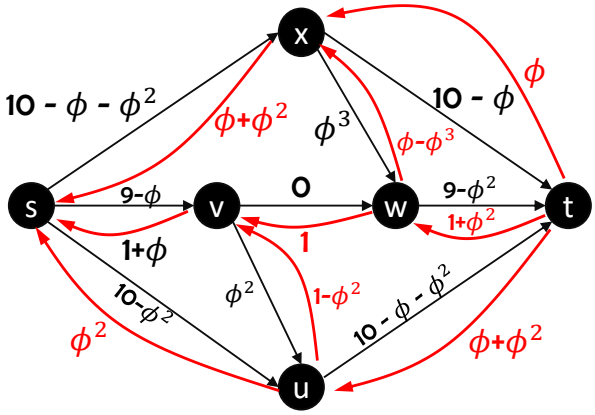



Now, in Iteration 4, we provide the path $(s, x, w, v, u, t)$ again. The minimum $u_f(e)$ edge is $(v, u)$ with residual capacity $\delta_4 := 1 - \phi = \phi^2$. The value of $F = 1 + 2\phi + \phi^2$. After augmenting $\phi^2$ flow on this path, we get the following residual network.




Note that the residual capacity of $(x, w)$ is $\phi - \phi^2 = \phi(1 - \phi) = \phi^3$.

In Iteration 5, we provide the path path $(s, u, v, w, t)$. Note on this path $\delta_5 = \phi^2$. After sending this flow the value becomes $F = 1 + 2\phi + 2\phi^2$, and the residual network becomes




Next, we repeat the paths sent in the previous $5$ iterations again in this order. That is, we send $\phi^3$ flow on $(s, x, w, v, u, t)$. This makes the residual capacity of $(v, u)$ equal to $\phi^2 - \phi^3 = \phi^4$. Follow up with a flow of $\phi^3$ on $(s, v, w, x, t)$ making the residual capacity of $(x, w)$ equal to $\phi^3$. The total flow is now $F = 1 + 2\phi + 2\phi^2 + 2\phi^3$. Next send $\phi^4$ flow on $(s, x, w, v, u, t)$ following with a flow of $\phi^4$ on $(s, u, v, w, t)$. This will make the residual capacity of $(x, w)$ equal to $\phi^5$ and the residual capacity of $(v, u)$ equal to $\phi^4$, and the total flow $F = -1 + 2(1 + \phi + \phi^2 + \phi^3 + \phi^4)$. And so and so forth – you get the drift.

Note that the value of $F$ converges to (but never achieves) the value $F = -1+2/(1-\phi)$ which is nowhere close to the maximum flow value of $21$.