

MAXIMUM FLOW

- How to do it...
- Why you want it...
- Where you find it...

The Tao of Flow:

“Let your body go with the flow.”
-Madonna, *Vogue*

“Maximum flow...because you’re worth it.”
-L’oreal

“Go with the flow, Joe.”
-Paul Simon, *50 ways to leave your lover*

“Life is flow; flow is life.”
-Ford & Fulkerson, *Ford & Fulkerson Algorithm*

“Use the flow, Luke!”
-Obi-wan Kenobi, *Star Wars*

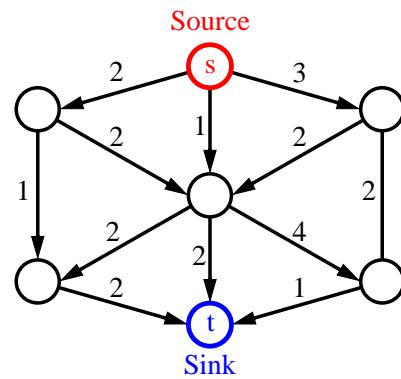
“Learn flow, or flunk the course”
-CS16 Despot, as played by Roberto Tamassia

Maximum Flow

1

Flow Networks

- Flow Network:
 - digraph
 - weights, called **capacities** on edges
 - two distinguishes vertices, namely
 - Source, “s”:
Vertex with no incoming edges.
 - Sink, “t”:
Vertex with no outgoing edges.

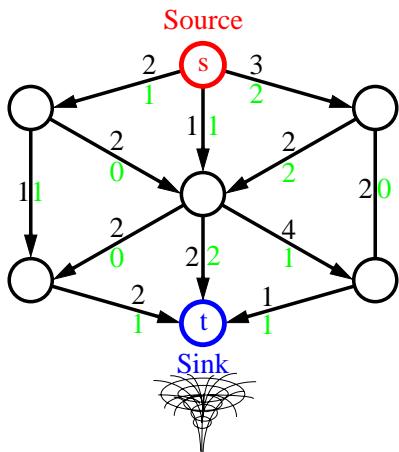


Maximum Flow

2

Capacity and Flow

- Edge Capacities:
Nonnegative weights on network edges
- Flow:
 - Function on network edges:
 $0 \leq \text{flow} \leq \text{capacity}$
flow into vertex = flow out of vertex
value: combined flow into the sink



Maximum Flow

3

The Logic of Flow

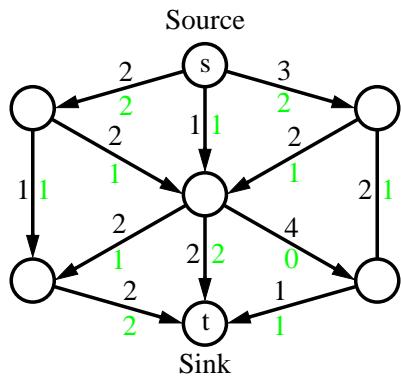
- Flow:
 - flow(u,v) \forall edge(u,v)**
 - Capacity rule:** \forall edge (u,v)
 $0 \leq \text{flow}(u,v) \leq \text{capacity}(u,v)$
 - Conservation rule:** \forall vertex $v \neq s, t$
 $\sum_{u \in \text{in}(v)} \text{flow}(u,v) = \sum_{w \in \text{out}(v)} \text{flow}(v,w)$
 - Value of flow:**
 $|f| = \sum_{w \in \text{out}(s)} \text{flow}(s,w) = \sum_{u \in \text{in}(t)} \text{flow}(u,t)$
- Note:
 - \forall means “for all”
 - $\text{in}(v)$ is the set of vertices u such that there is an edge from u to v
 - $\text{out}(v)$ is the set of vertices w such that there is an edge from v to w

Maximum Flow

4

Maximum Flow Problem

- “Given a network N , find a flow f of maximum value.”
- Applications:
 - Traffic movement
 - Hydraulic systems
 - Electrical circuits
 - Layout

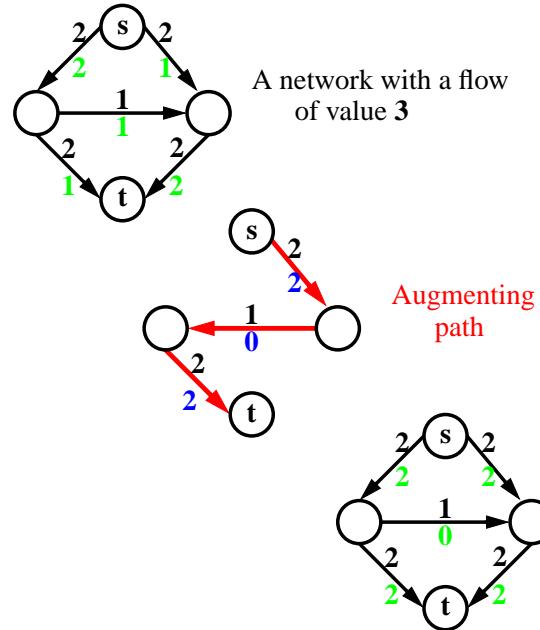


Example of Maximum Flow

Maximum Flow

5

Augmenting Flow



- Voila! We have increased the flow value to 4!
But wait! What's an augmenting path?!!? →

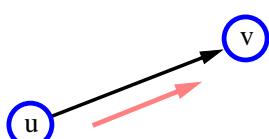
Maximum Flow

Maximum Flow

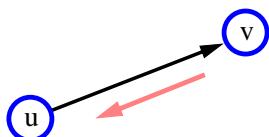
6

Augmenting Path

- Forward Edges
 $\text{flow}(u,v) < \text{capacity}(u,v)$
flow can be increased!



- Backward Edges
 $\text{flow}(u,v) > 0$
flow can be decreased!



Maximum Flow

7

Maximum Flow Theorem

A flow has maximum value if and only if it has no augmenting path.

Proof:

Flow is maximum \Rightarrow No augmenting path
(The *only-if* part is easy to prove.)

No augmenting path \Rightarrow Flow is maximum
(Proving the *if* part is more difficult.)

Maximum Flow

Maximum Flow

8

Ford & Fulkerson Algorithm

- One day, Ford phoned his buddy Fulkerson and said, “Hey Fulk! Let’s formulate an algorithm to determine maximum flow.” Fulk responded in kind by saying, “Great idea, Ford! Let’s just do it!” And so, after several days of abstract computation, they came up with the Ford Fulkerson Algorithm, affectionately known as the “Ford & Fulkerson Algorithm.”

initialize network with null flow;

Method FindFlow

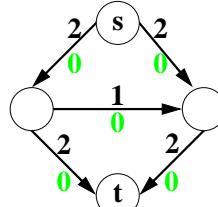
```
if augmenting paths exist then
    find augmenting path;
    increase flow;
    recursive call to FindFlow;
```

- And now, for some algorithmic animation...

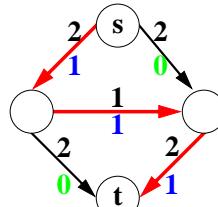
Maximum Flow

9

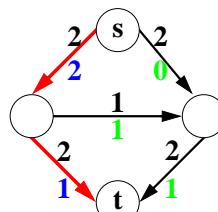
Finding the Maximum Flow



Initialize the network with a null flow. Note the **capacities above the edges**, and the **flow below the edges**.



Send one unit of flow through the network. Note the **path of the flow unit traced in red**. The **incremented flow values are in blue**.

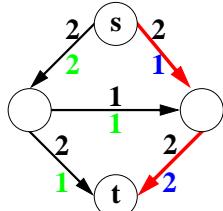


Send another unit of flow through the network.

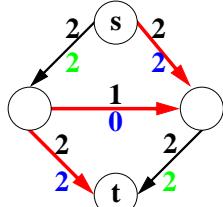
Maximum Flow

10

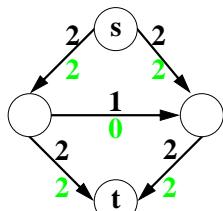
Finding the Maximum Flow



Send another unit of flow through the network. Note that there still exists an augmenting path, that can proceed *backward* against the directed central edge.



Send one unit of flow through the augmenting path. Note that there are no more augmenting paths. That means...



With the help of both Ford & Fulkerson, we have achieved this network’s **maximum flow**.

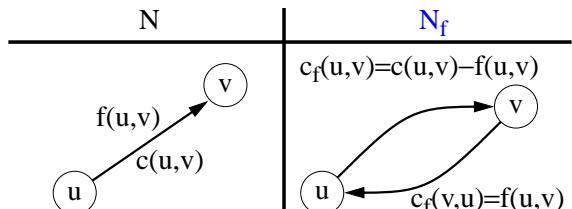
Is this power, or what?!?

Maximum Flow

11

Residual Network

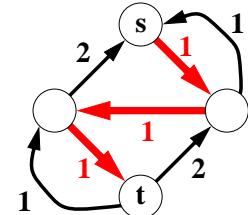
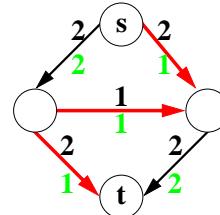
- Residual Network $N_f = (V, E_f, c_f, s, t)$



- In the residual network N_f , all edges (w, z) with capacity $c_f(w, z) = 0$ are removed.

Augmenting path in network N

↔ Directed path in the residual network N_f



Augmenting paths can be found performing a depth-first search on the residual network N_f .

Maximum Flow

12

Maximum Flow Algorithm

Part I: Setup

Start with null flow:

$$f(u,v) = 0 \quad \forall (u,v) \in E;$$

Initialize residual network:

$$N_f = N;$$

Part II: Loop

repeat

```
    search for directed path p in  $N_f$  from s to t  
    if (path p found)  
         $D_f = \min \{c_f(u,v), (u,v) \in p\};$   
        for (each  $(u,v) \in p$ ) do  
            if (forward  $(u,v)$ )  
                 $f(u,v) = f(u,v) + D_f;$   
            if (backward  $(u,v)$ )  
                 $f(u,v) = f(u,v) - D_f;$   
            update  $N_f$ ;  
    until (no augmenting path);
```

Maximum Flow: Time Complexity

- And now, the moment you've all been waiting for...the time complexity of Ford & Fulkerson's Maximum Flow algorithm. Drum roll, please!
[Pause for dramatic drum roll music]

$$O(F(n + m))$$

where F is the maximum flow value, n is the number of vertices, and m is the number of edges

- The problem with this algorithm, however, is that it is strongly dependent on the **maximum flow value F**. For example, if $F=2^n$ the algorithm may take exponential time.
- Then, along came Edmonds & Karp...

Maximum Flow: Improvement

- Theorem: [Edmonds & Karp, 1972]

By using BFS, **Breadth First Search**, a maximum flow can be computed in time...

$$O((n + m)n m) = O(n^5)$$

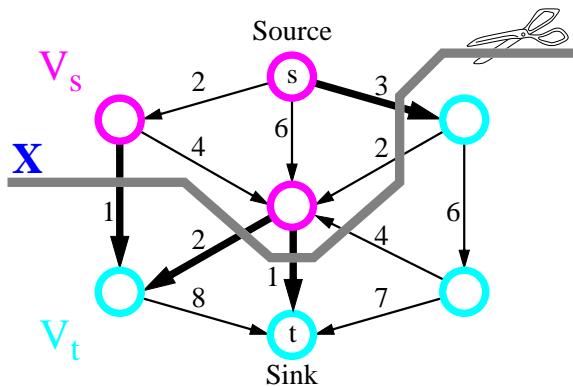
- Note:

- Edmonds & Karp algorithm runs in time $O(n^5)$ regardless of the maximum flow value.
- The worst case usually does not happen in practice.

CUTS

- What is a **cut** ?
 - a) skin-piercing laceration
 - b) sharp lateral movement
 - c) opposite of paste
 - d) getting dropped from the team
 - e) fronties on the lunch line
 - f) common CS16 attendance phenomenon
 - g) line of muscular definition
 - h) Computer Undergraduate Torture (e.g., CS16, Roberto Tamassia, dbx, etc.)
 - i) a partition of the vertices $X=(V_s, V_t)$, with $s \in V_s$ and $t \in V_t$
- The answer is:

What Is A Cut?



- Capacity of a cut $X = (V_s, V_t)$:
 $c(X) = \sum_{v \in V_s, w \in V_t} \text{capacity}(v, w) = (1+2+1+3) = 7$
- The **cut partition** (X in this case) must pass through the entire network, and can not pass through a vertex.

Maximum Flow

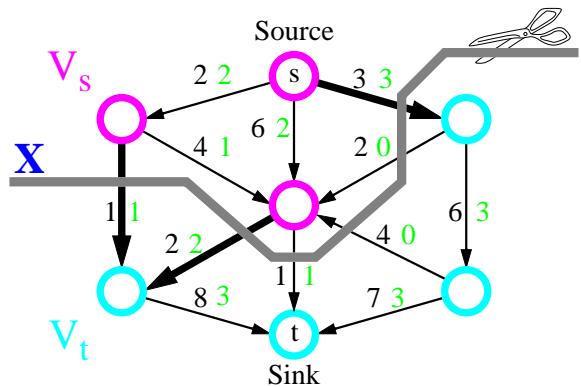
17

Maximum Flow vs. Minimum Cut

(value of maximum flow)

=

(capacity of minimum cut)



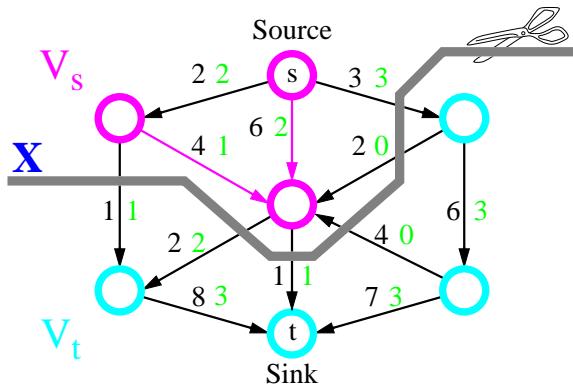
- Value of maximum flow: 7 flow units
- Capacity of minimum cut: 7 flow units

Maximum Flow

18

Finding a Minimum Cut

- Let V_s be the set of vertices reached by augmenting paths from the source s , V_t the set of remaining vertices, and place the cut partition X accordingly.



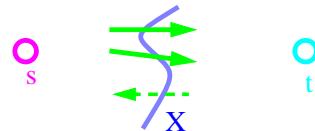
- Hence, a minor modification of the Ford & Fulkerson algorithm gives a **minimum cut**
- **Value of maximum flow: 7 flow units**
- **Capacity of minimum cut: 7 flow units**

Maximum Flow

19

Why is that a Minimum Cut?

- Let f be a flow of value $|f|$ and X a cut of capacity $|X|$. Then, $|f| \leq |X|$.



- Hence, if we find a flow f^* of value $|f^*|$ and a cut X^* of capacity $|X^*| = |f^*|$, then f^* must be the maximum flow and X^* must be the minimum cut.
- We have seen that from the flow obtained by the Ford and Fulkerson algorithm we can construct a cut with capacity equal to the flow value. Therefore,
 - we have given an alternative proof that the Ford and Fulkerson algorithm yields a maximum flow
 - we have shown how to construct a minimum cut

Maximum Flow

20