CS340 - Analysis of Algorithms

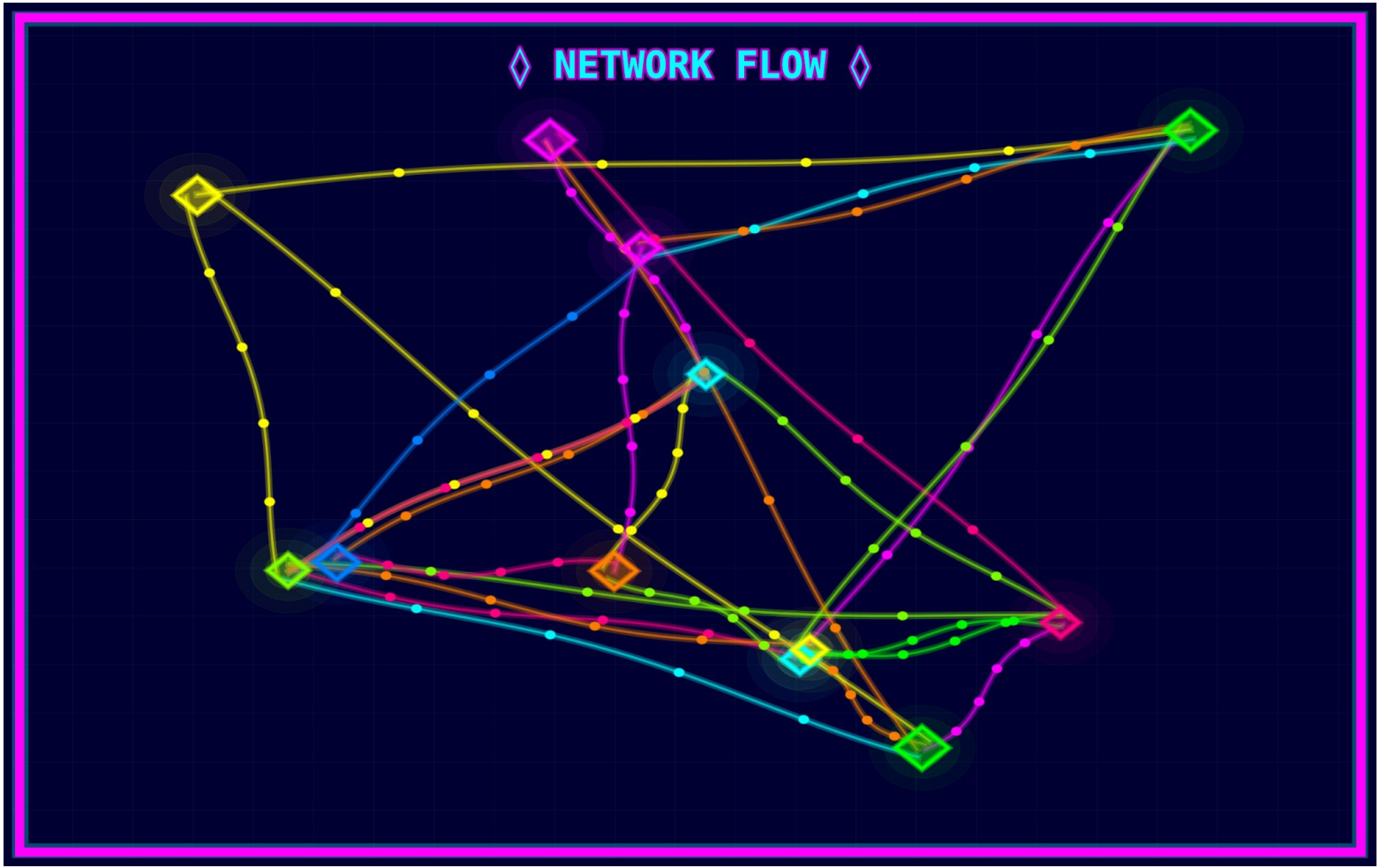
Network Flow

Announcements:

HW6 Due Today

HW7 Released - Due Monday November 17

Project Feedback will be sent out today


◊ NETWORK FLOW ◊

Flow Network

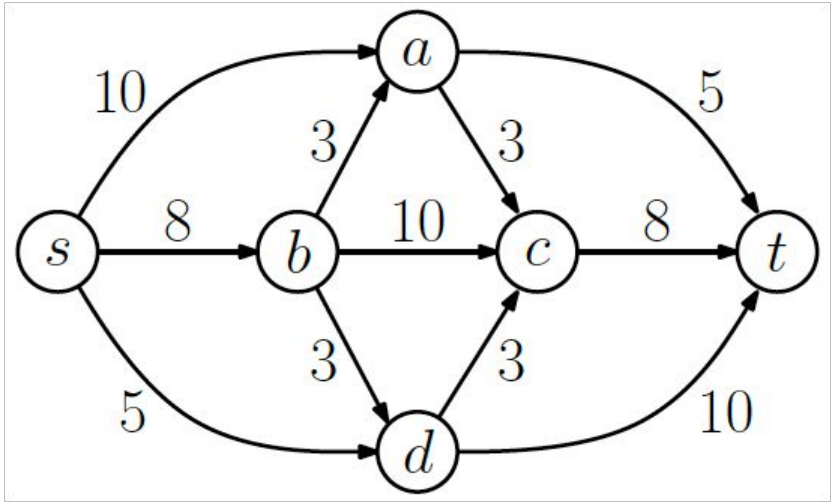
- Directed graph with non-negative weights
- Edges: directional pipes
  - Fluid, traffic, currency, etc
  - Weights are the capacity of the edge
- Goal: maximize the flow from some source node to some sink

Flow Network

- Directed graph G = (V,E)
- Each edge (u,v) ∈ E has a non-negative *capacity* c(u,v) ≥ 0
- If (u,v) ∉ E, c(u,v) = 0

- Source node: s
- Sink node: t

- Called an “s-t network”



Flows

- Given an s-t network, a flow is a function f that maps each edge to a nonnegative real number
- $f: E \rightarrow R^+$
- Represents the amount of flow carried by an edge e
- A flow must satisfy the following properties:
  a. Capacity: The flow on an edge cannot exceed the capacity of an edge
  b. Conservation: The amount of flow entering a node (other than source or sink) must equal the amount of flow going out

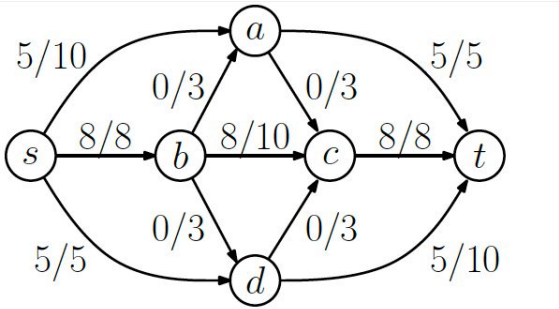
Flows

- A flow must satisfy the following properties:
  a. Capacity: The flow on an edge cannot exceed the capacity of an edge
     - $\forall (u, v) \in E, f(u, v) \leq c(u, v)$
  b. Conservation: The amount of flow entering a node (other than source or sink) must equal the amount of flow going out
     - $\forall v \in V \backslash \{s, t\}, f^{in}(v) = f^{out}(v)$
     - $f^{in}(v) = \sum_{(u,v) \in E} f(u, v)$
     - $f^{out}(v) = \sum_{(v,w) \in E} f(v, w)$

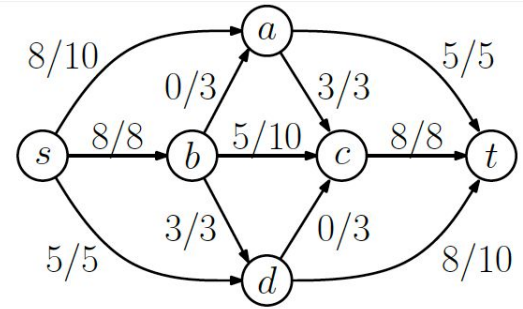
Maximum Flow Problem

- Given a flow network, arrange the traffic to make as efficient use as possible of the available capacity
- Total flow of a network:

$$|f| = f^{out}(s) = \sum_{w \in V} f(s, w)$$



A valid flow ($|f| = 18$)



A maximum flow ($|f| = 21$)

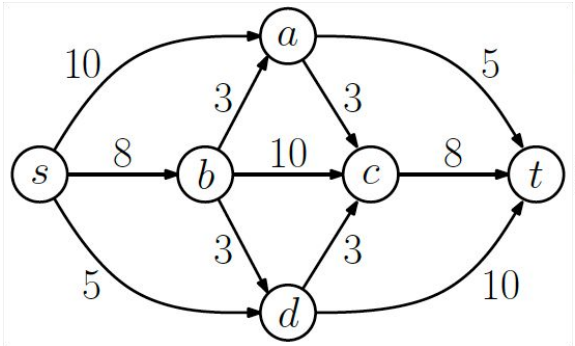
Maximum Flow - A Greedy Attempt

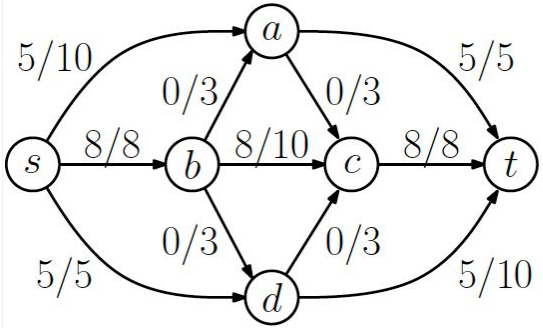
- Suppose we start with zero flow: f(e) = 0 for all e
- Find any s-t path P through the flow network
- Increase the value of f by “pushing” flow along the path up to the limits imposed by edge capacities (greedy!)
- Let C_{min} be the minimum capacity of any edge on this path
- Set the flow to be C_{min}

Path: {(s,a),(a,t)}
- Increase f(s,a) and f(a,t) to 5

Path: {(s,b),(b,c),(c,t)}
- f(s,b), f(b,c), f(c,t) to 8

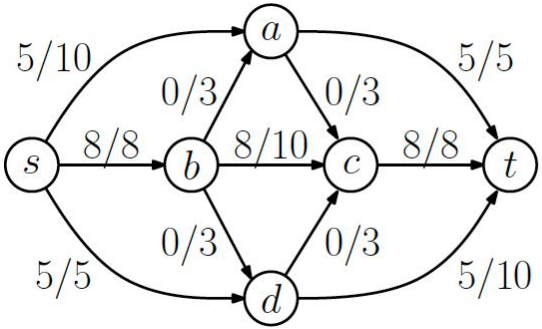
Path: {(s,d),(d,t)}
- f(s,d) and f(d,t) to 5

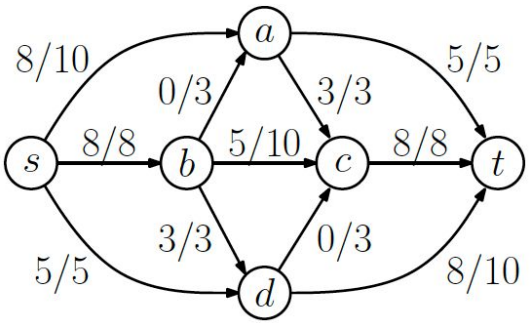




Maximum Flow - A Greedy Attempt

- To get a max flow, we'd like to push 5 more units of flow along (s,a)
- We can put 3 units of the overflow on (a,c)
- We still have 2 units too many coming out of a
- We need to "undo" 2 units of flow on (s,a)



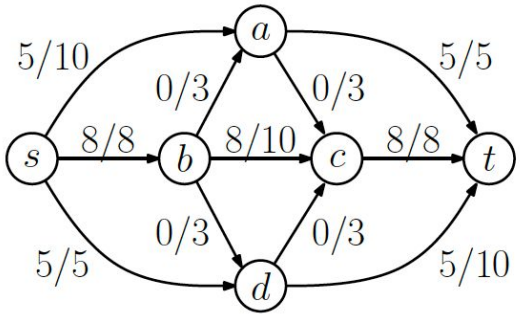


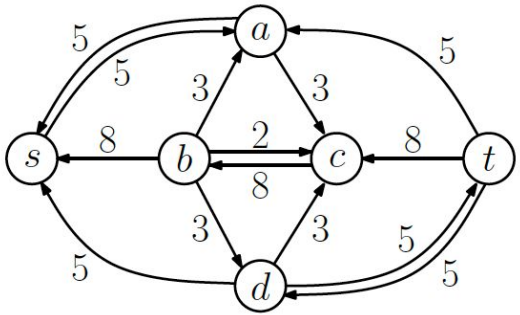
A maximum flow ($|f| = 21$)

- Push forward on edges with leftover capacity
- Push backward on edges that are already carrying flow to divert it into a different direction

Residual Network (G_f)

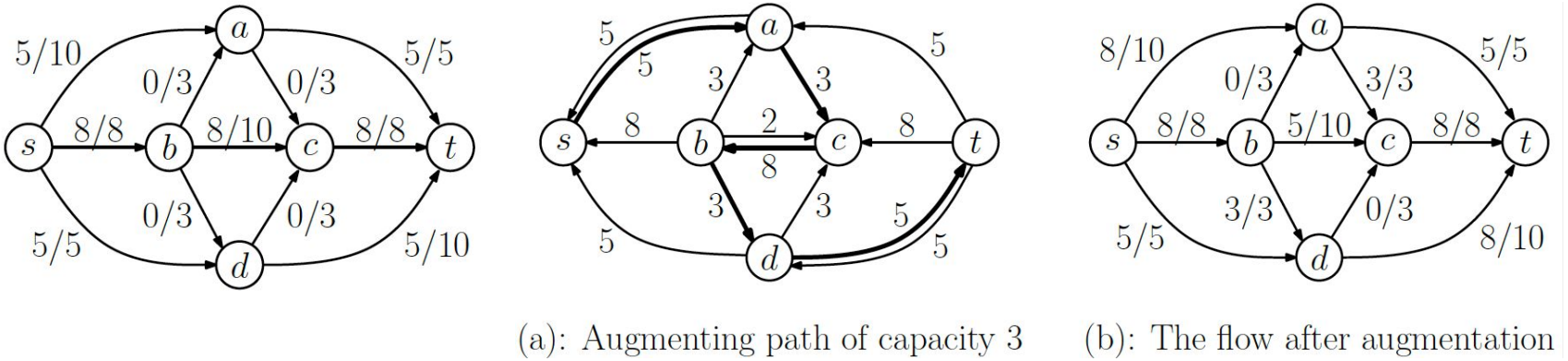
- Systematic way to search for forward-backward operations
- Given a flow network G and a flow f, the *residual network* G_f is a network with the same vertex set and the following edges:
  - forward: for each edge (u,v) on which f(u,v) < c(u,v), there are c(u,v) - f(u,v) leftover units of capacity on which we could try pushing flow forward
  - backward: for each edge (u,v) on which f(u,v) > 0, there are f(u,v) units of flow that we can "undo" if we want to, by pushing flow backward





Augmenting Paths

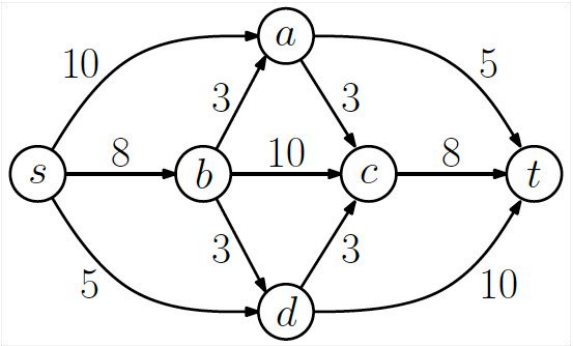
- An *augmenting path* is an s-t path in G_f
- The *residual capacity* of an augmenting path P is the minimum capacity of any edge on the path, denoted $c_f(P)$



(a): Augmenting path of capacity 3

(b): The flow after augmentation

Ford-Fulkerson




```
ff(G=(V, E, s, t)) {
  f = 0 (all edges carry 0 flow)
  while (true) {
    G' = residual-network of G for f
    if (G' has no s-t augmenting path) break
    P = any-augmenting-path of G'
    c = min capacity of P
    augment by adding c to every edge of P in G
    update f
  }
  return f
}
```

Runtime Analysis

```
ff(G=(V, E, s, t)) {
  f = 0 (all edges carry 0 flow)
  while (true) {
    G' = residual-network of G for f
    if (G' has no s-t augmenting path) break
    P = any-augmenting-path of G'
    c = min capacity of P
    augment by adding c to every edge of P in G
    update f
  }
  return f
}
```

We will do full runtime complexity next class

Runtime Complexity of Augmentation

1. Compute Residual Network (G_f)
   a. O(n+m)
2. Find augmenting path P in G_f
   a. DFS or BFS on G_f starting at S
   b. O(n+m)
3. Compute min-cost edge along P
   a. O(m)
4. Increase flow for every edge in P
   a. O(m)

For network flow the graph must be connected.

$m > n-1$

Each augmentation is O(m)

Proof Of Correctness

How do we know that the flow returned by Ford-Fulkerson has the maximum possible value of any flow in G?

The structure of the flow network places upper bounds on the maximum value of an s-t flow.
Let's use this to devise an upper bound

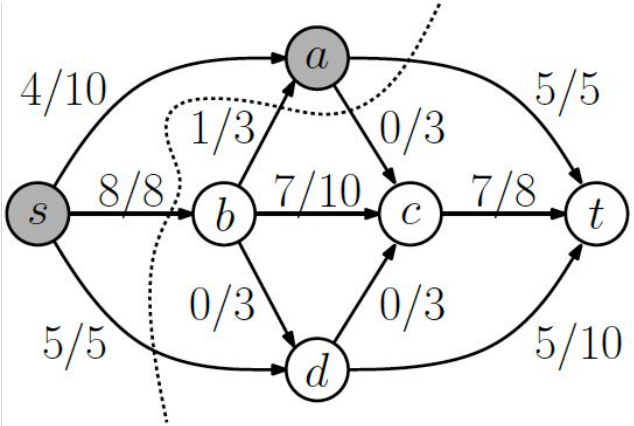
Suppose we divide V into two sets A and B so that $s \in A$ and $t \in B$. We will call this the "cut"

Any flow that goes from s to t must cross from A to B at some point.

We will use this to show that the maximum-flow value is equal to the capacity of the minimum cut

The Cut

- Given a network G, a s-t *cut* is a partition of the vertices into two disjoint subsets $X \subseteq V$ and $Y \subseteq V \backslash \mathrm{X}$, where $s \in X$ and $t \in \mathbf{Y}$



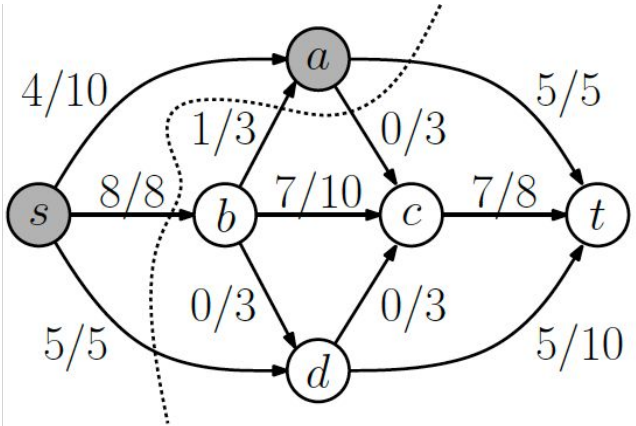
$$X = \{s, a\}$$

$$Y = \{b, c, d, t\}$$

The Cut

We can define capacity or *net flow* across a cut as the sum of flows from X to Y minus the sum of flows from Y to X

$$f(X,Y) = \sum_{x \in X} \sum_{y \in Y} f(x,y) - \sum_{y \in Y} \sum_{x \in X} f(y,x)$$



$$X = \{s, a\}$$

$$Y = \{b, c, d, t\}$$

$$f(X,Y) = 5 + 0 - 1 + 8 + 5 = 17$$

Lemma: |f| = f(X,Y)

$$f(X,Y) = \sum_{x \in X} \sum_{y \in Y} f(x,y) - \sum_{y \in Y} \sum_{x \in X} f(y,x)$$

Let (X,Y) be any s-t cut in a network. Given any flow f, the value of f is equal to the net flow across the cut.

- Proof
  - $|f| = f^{out}(s) - f^{in}(s) = f^{out}(s)$
  - all other nodes of X satisfy flow conservation

$$|f| = \sum_{x \in X} (f^{out}(x) - f^{in}(x)) = f^{out}(X) - f^{in}(X)$$

  - only edges that cross the cut contribute to net flow $|f| = f(X,Y)$

Cut Capacity

- The capacity $c(X, Y)$ of a cut (X, Y) is the sum of the capacities of the edges leading from X to Y

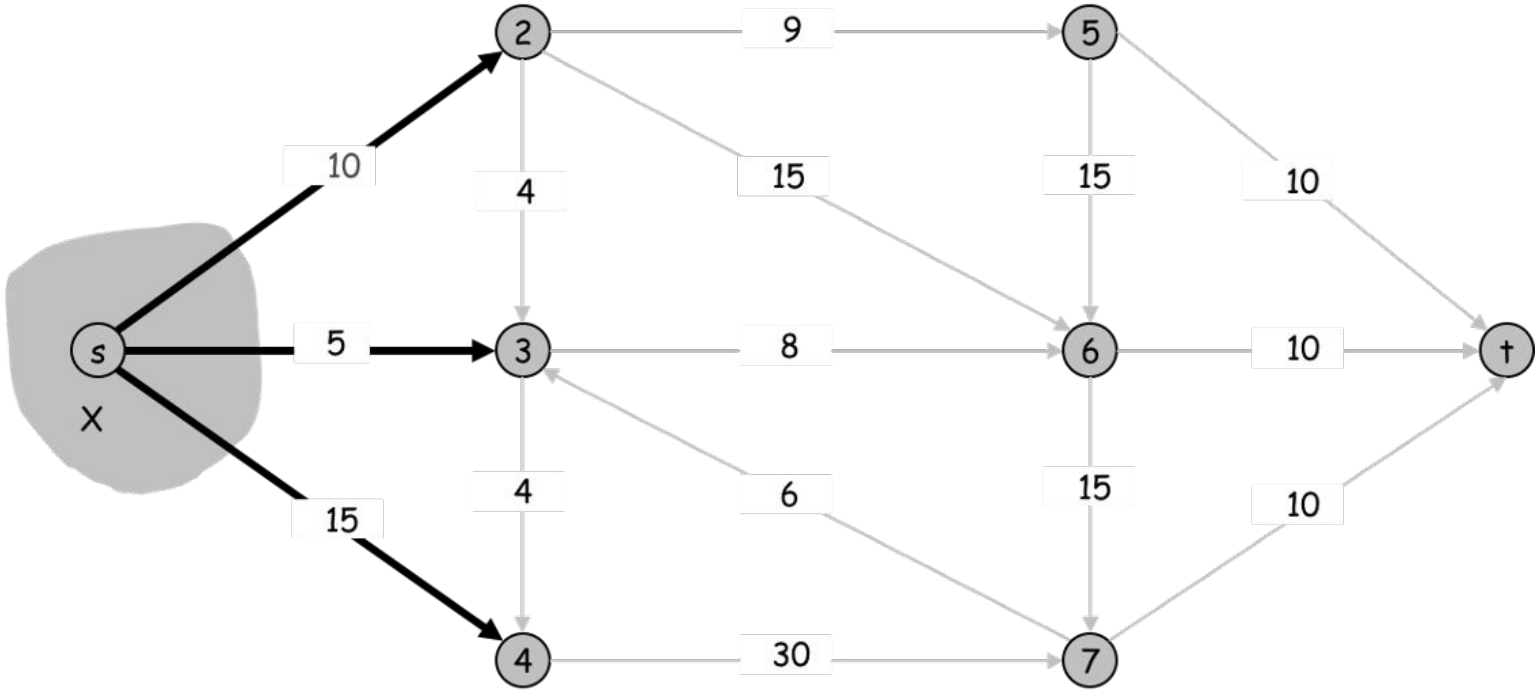
$$c(X, Y) = \sum_{x \in X} \sum_{y \in Y} c(x, y)$$

- Edges from Y to X are not included
- Given any s-t cut (X, Y) and any flow f, $|f| \leq c(X, Y)$

Cut Capacity

$c(X,Y) =$

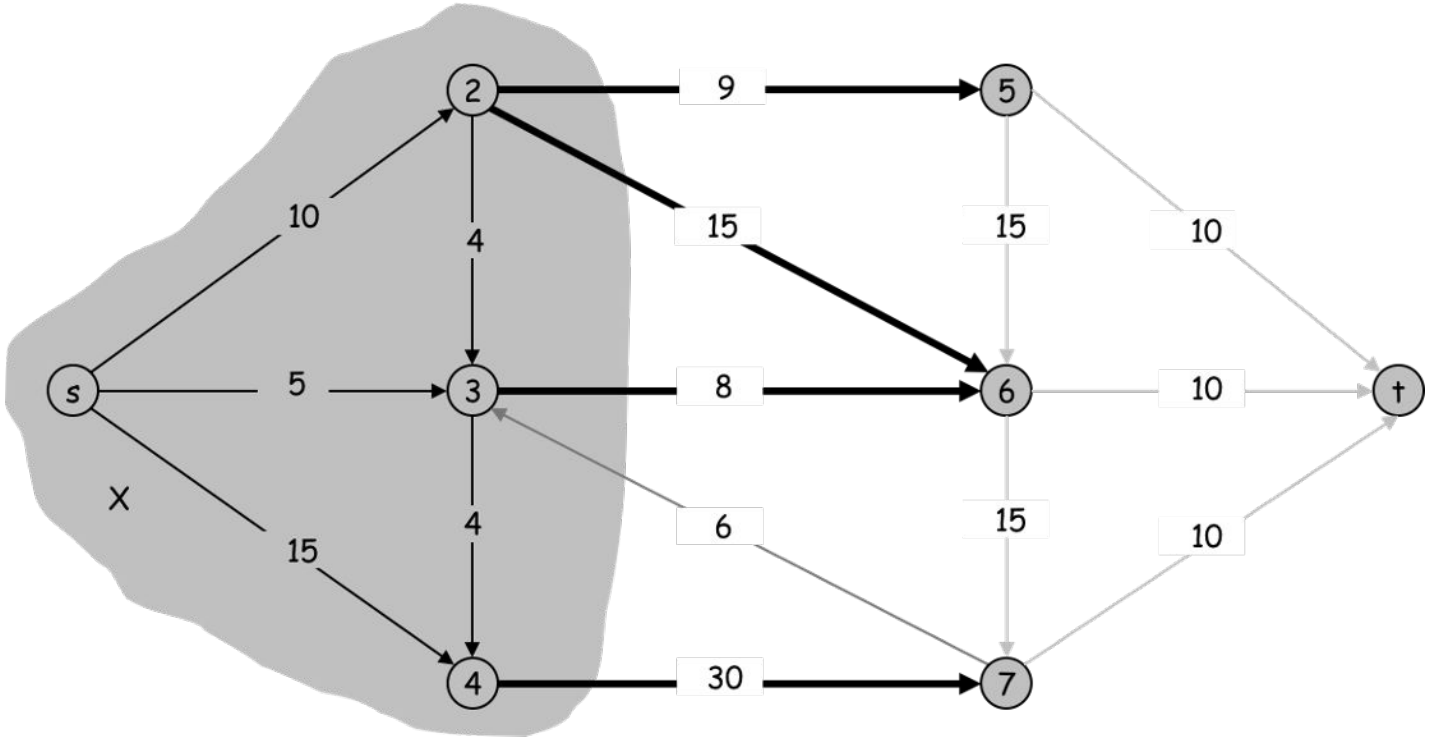
$15 + 5 + 10 = 30$



Cut Capacity

$c(X,Y) =$

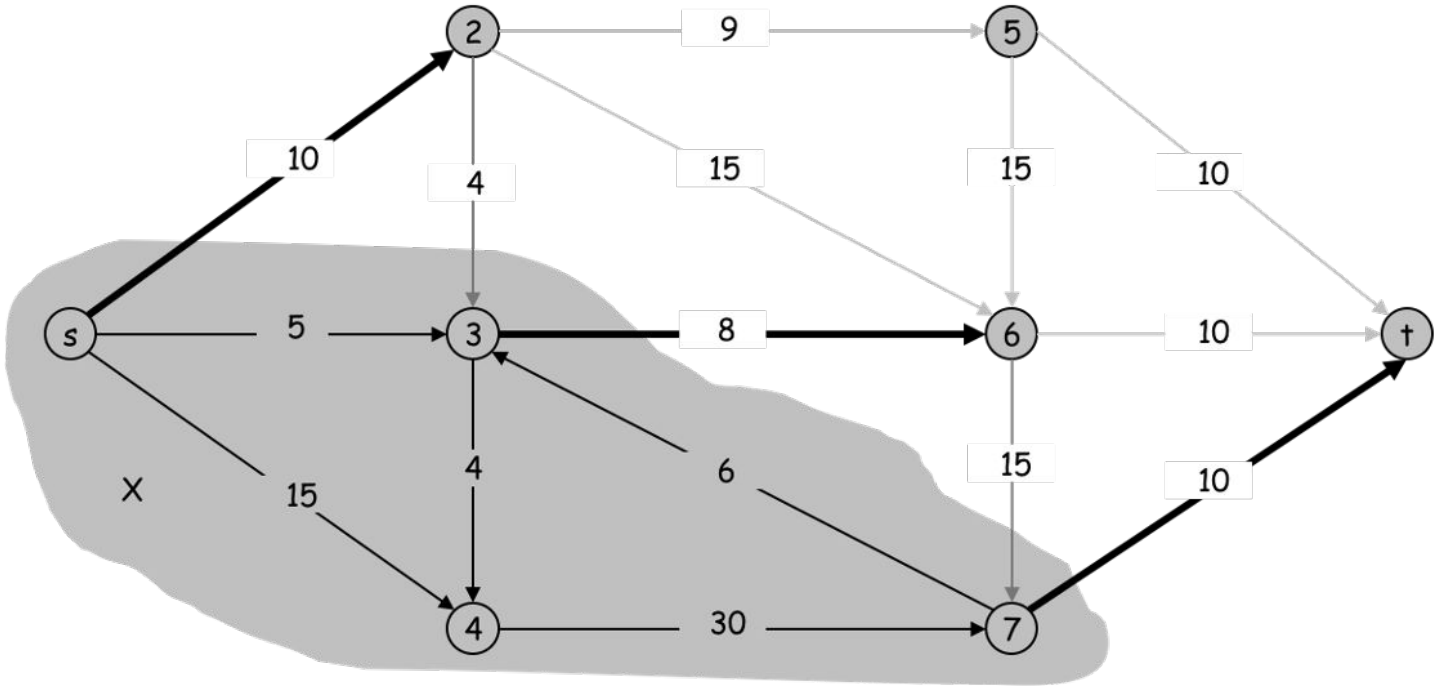
$9 + 15 + 8 + 30 = 62$



Cut Capacity

$c(X,Y) =$

$10 + 8 + 10 = 28$



Proof Of Correctness

maximum-flow value is equal to the capacity of the minimum-cut

To prove this, we will show that the following conditions are equivalent:

1. f is a maximum flow in G
2. The residual graph G_f contains no augmenting paths
3. |f| = c(X,Y) for some cut (X,Y) of G

Proof Of Correctness

1 => 2 by contradiction:

We will prove that if f is a maximum flow in G, the residual graph G_f contains no augmenting paths.

Suppose there is an augmenting path P in G_f with residual capacity $c_f(P) = c$.

We can augment f along P by c units to create a new flow f' with value $|f'| = |f| + c$

This contradicts our claim that f is a maximum flow

Proof Of Correctness

2 => 3:

We will prove that if the residual graph G_f contains no augmenting paths, then |f| = c(X,Y) for some cut (X,Y) of G.

If there are no augmenting paths left then s and t are not connected in G_f.

Let X be vertices reachable from s in G_f and Y be the rest. (X,Y) forms a cut in G.

In G_f, no forward edges cross (X,Y)

In G, each forward edge crossing the cut must be saturated and the backward edge must be 0. (by definition of residual graph)

Because each forward edge is saturated, it follows that the flow across the cut equals the capacity of the cut. Thus |f| = C(X,Y)

Proof Of Correctness

3 => 1 by contradiction:

We will show that if |f| = c(X,Y) for some cut (X,Y) of G, f is a maximum flow in G.

Suppose there is some f’ such that |f’| > |f|.

But, |f| = c(X,Y) which is the capacity across the cut.

|f’| > c(X,Y) but any additional units of flow cannot cross the cut, so an f’ cannot exist.

Therefore f must be a maximal flow.

Proof Of Correctness

```
ff(G=(V, E, s, t)) {
  f = 0 (all edges carry 0 flow)
  while (true) {
    G' = residual-network of G for f
    if (G' has no s-t augmenting path) break
    P = any-augmenting-path of G'
    c = min capacity of P
    augment by adding c to every edge of P in G
    update f
  }
  return f
}
```

Given that Ford-Fulkerson terminates when there is no augmenting path in the residual graph, and we proved that 2=>1, it is correct

Termination of Ford-Fulkerson

```
ff(G=(V, E, s, t)) {
  f = 0 (all edges carry 0 flow)
  while (true) {
    G' = residual-network of G for f
    if (G' has no s-t augmenting path) break
    P = any-augmenting-path of G'
    c = min capacity of P
    augment by adding c to every edge of P in G
    update f
  }
  return f
}
```

Not obvious! We will do this next class.

Summary

- Ford-Fulkerson:
  - Start with zero flow: f(e) = 0 for all edges
  - While an augmenting path exists in the residual graph, find the capacity c
  - Augment the graph with flow c

- Correctness:
  - The following are equivalent:
    - f is a maximum flow
    - G_f contains no augmenting path
    - |f| = c(X,Y) for some cut (X,Y)

- Hw7 due next Monday