CS 410/584, Algorithm Design & Analysis, Lecture Notes 5

Ford-Fulkerson Method

Flow maximization in a network (graph) with capacities

Basic idea:

- Find a path from source to target that still has flow capacity (*augmenting path*)
- Add the maximum flow allowed along this path
- Repeat until

Issues

1. How do we account for flow by

2. Does adding an augmenting path lead to a legal flow?

3. Will this process converge?

4. If so, will it lead to

Problem Formulation

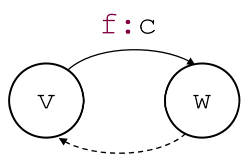
Directed graph G = (N, E)

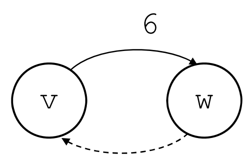
- Two special nodes: s t
- Assume for any node v ∈ N, there are paths

Capacity c(v,w) ≥ 0

- If (v, w) not an edge, then

Flow f(v, w) can be

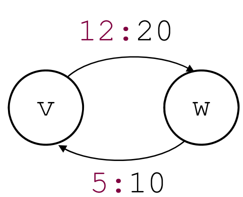


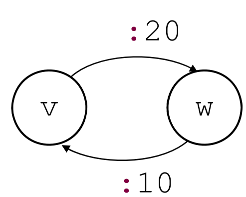


Detail

Assume no "useless" flows between nodes

- Positive flow in only one direction





Legal Flow f

1. f(v, w) ≤ c(v, w)

2. f(v, w) = -f(w, v)

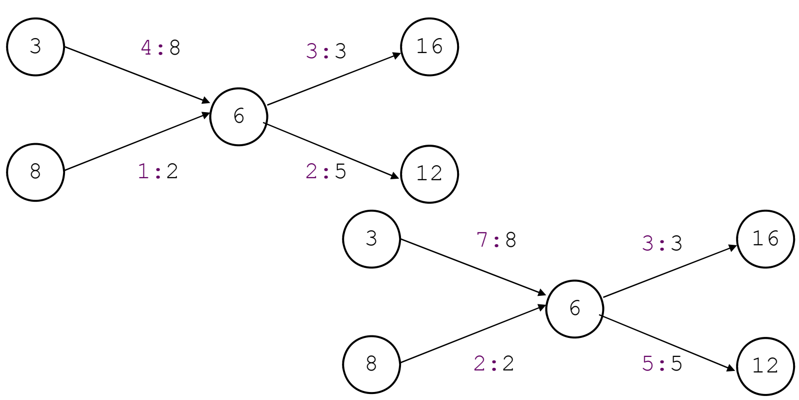
3. For any v ≠ s, t

   $\sum f(v, w) = 0$

Inputs = Outputs

Consider node 6

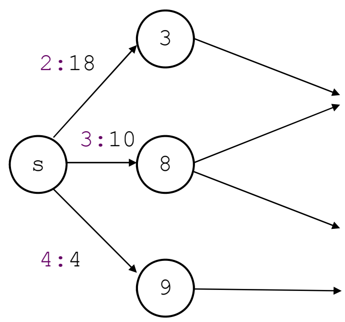
f(6,16) + f(6,12) + f(6,8) + f(6,3)



Value of Flow f

Total flow out of source

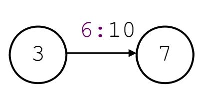
$|f| = \sum f(s, w)$



Residual Capacity for a Flow

Residual Capacity between nodes v and w:

$r(u,v) = c(u,v) - f(u,v)$



$r(3,7) = c(3,7) - f(3,7)$

$r(7,3) = c(7,3) - f(7,3)$

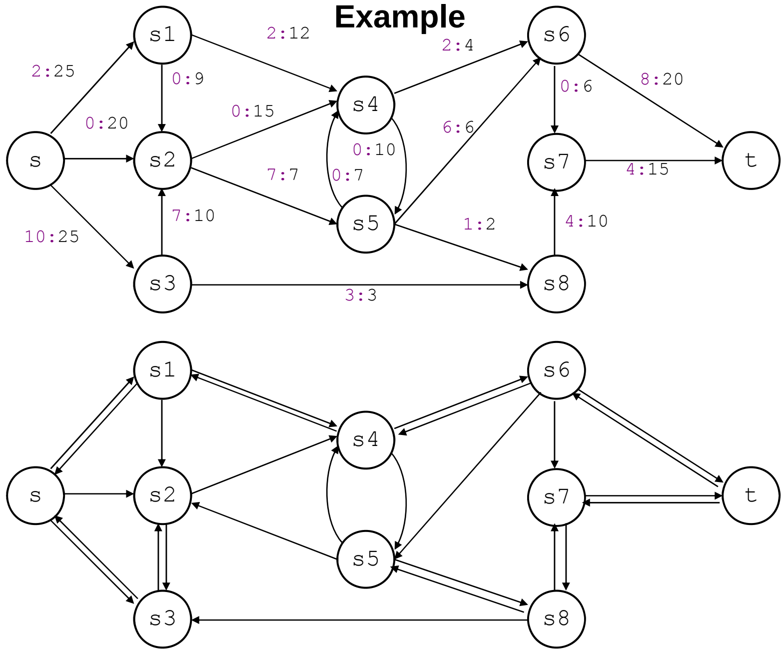
Residual Graph for Flow f

R = (N, E′)

E′ = { (v,w) | r(v,w)   }

Capacities are residual capacities for f

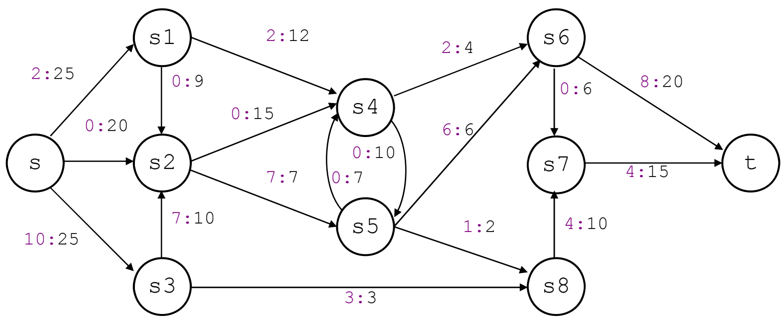
Example



Capacity of a Path

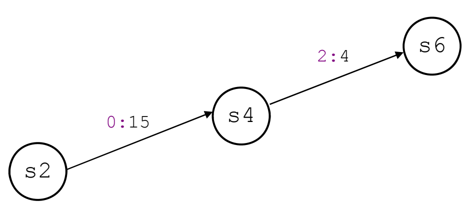
Minimum capacity edge

Add a flow of   along the path

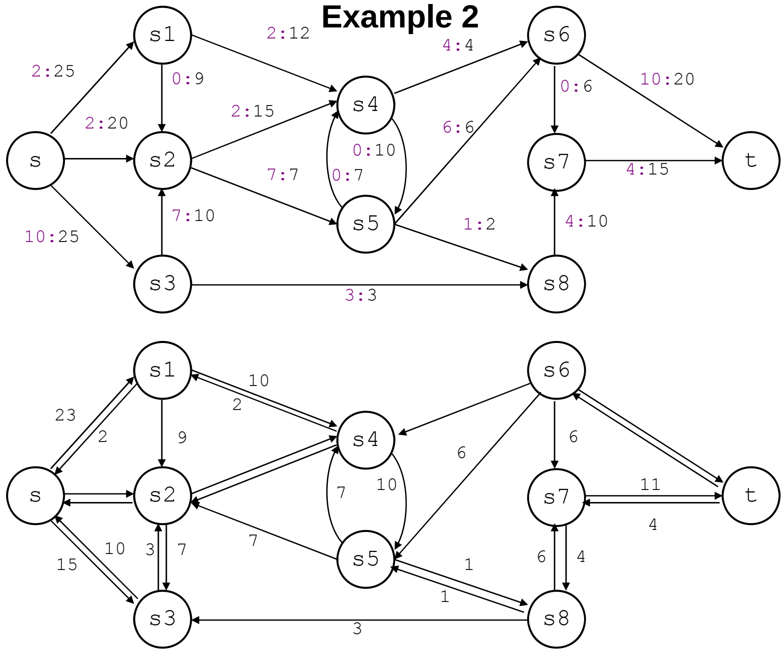


Is the Result a Legal Flow?

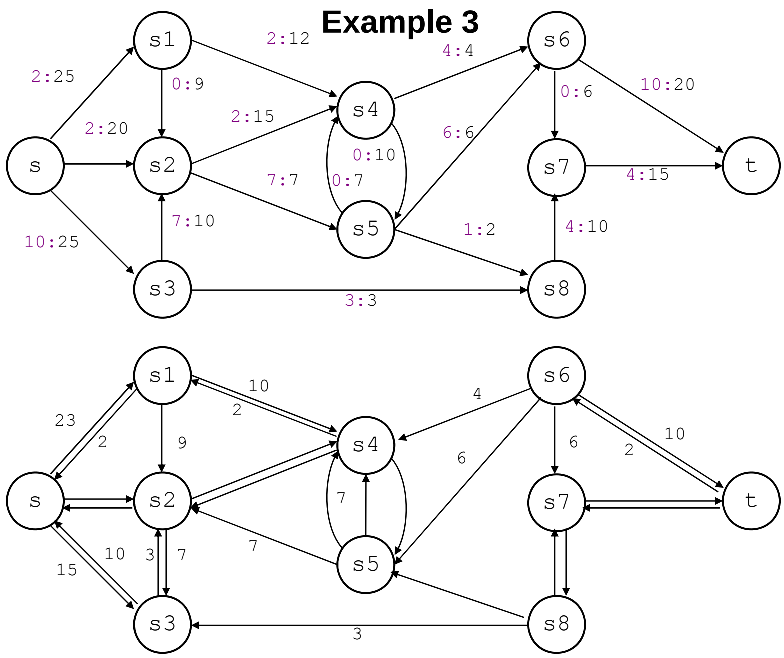
- Capacity?
- Skew symmetry?
- Conservation?



Example 2



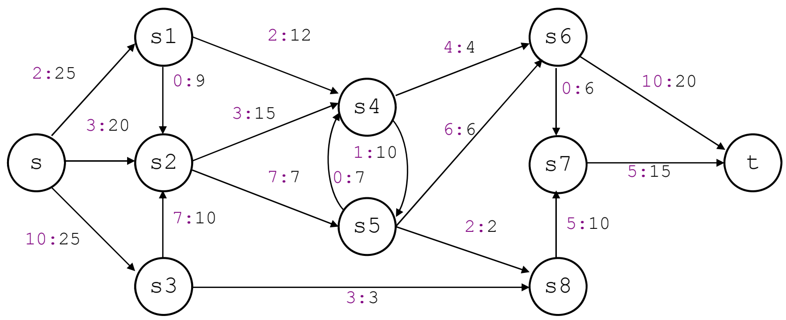
Example 3



Is it a Maximum Flow?

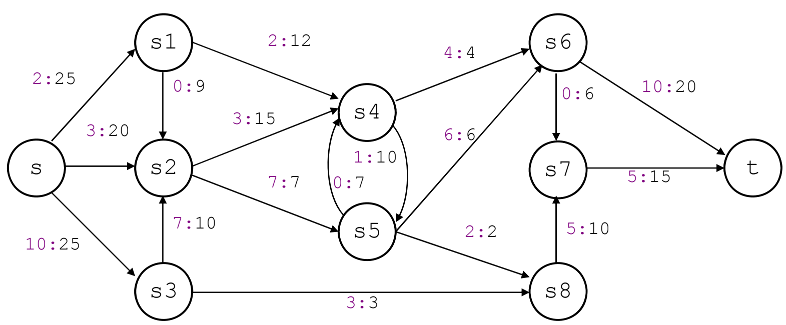
Would seem so:

- Have a group of edges that divides `s` from `t` and



Cut of a Graph

Divide nodes of N into two groups S, T



Net flow across cut Σ `f(v, w)`

Capacity across cut Σ `c(v, w)`

Results

- Net flow across any cut
- |f| is bounded above by
- Max-flow/min-cut theorem
  1. f is maximum flow
  2. residual graph has no
  3. |f| is capacity of

$2 \Rightarrow 3$

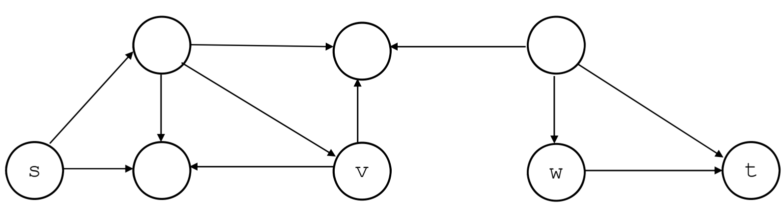
Consider residual graph R with no augmenting path

S = {v| } T = N - S

Must have

Claim that for (v,w) with $v \in S$, $t \in T$, must have

Suppose not. Then r(v,w) > 0. Then R has



Basic Implementation

Start with 0 flow

Repeat

- Add flow along an augmenting path

Does it always converge?

- Yes, if capacities are integers. Flow grows by at least

How long does it take?

- If you pick augmenting paths arbitrarily
- O(|E|·|f*|)

Edmonds-Karp Algorithm

Start with 0 flow

Repeat

- Add flow along an augmenting path with

Time complexity no longer depends on value of maximum flow

- O(|N|·|E|^2) time
- Intuition: Length of shortest path to a node v in residual graph
  - Each addition of flow increases distance to one node
  - Distance to a node v can be increased at most

Edmonds-Karp Example