NPTEL MOOC,JAN-FEB 2015
Week 8, Module 4

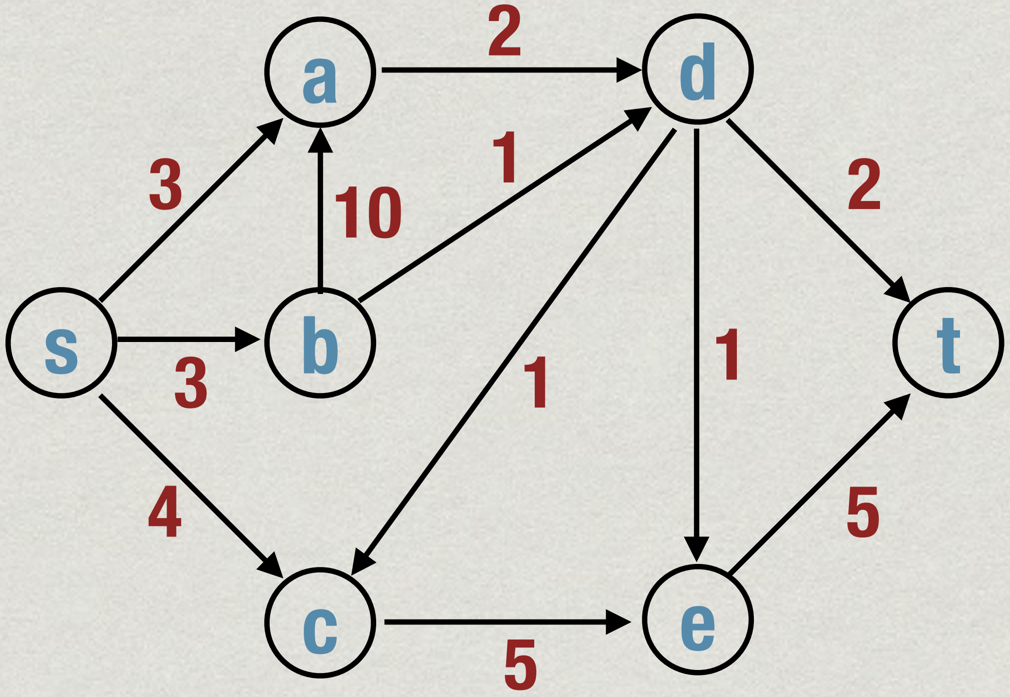
DESIGN AND ANALYSIS OF ALGORITHMS

Network Flows

MADHAVAN MUKUND, CHENNAI MATHEMATICAL INSTITUTE
http://www.cmi.ac.in/~madhavan

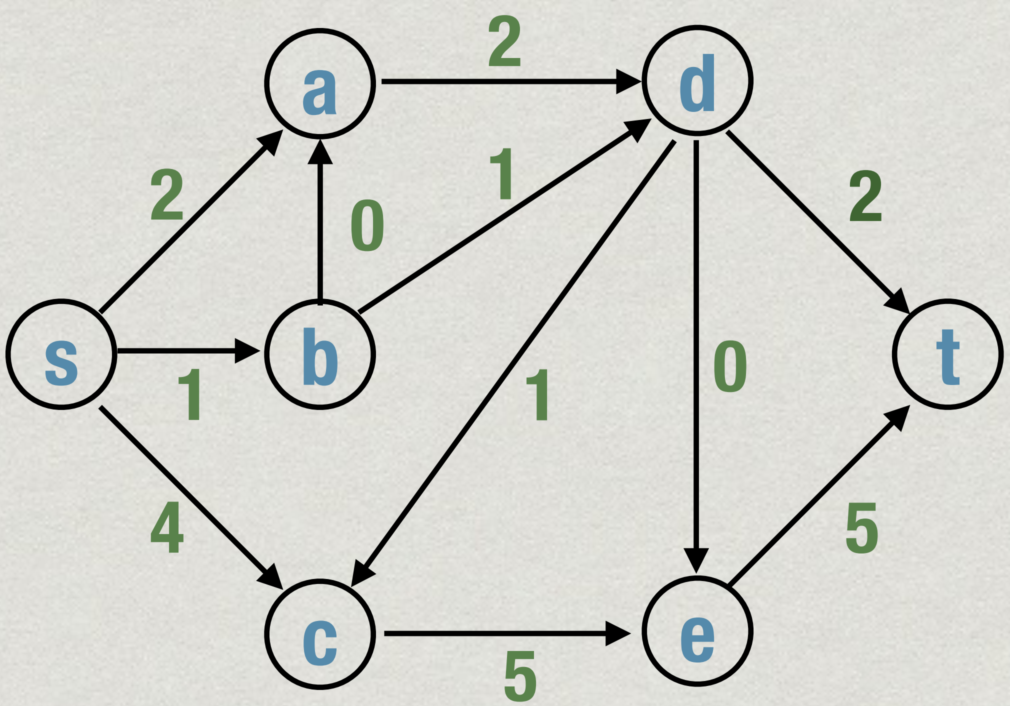
Oil network

- Network of pipelines
- Ship as much oil as possible from s to t
- No storage on the way
- A flow of 7 is possible
- Is this the maximum?



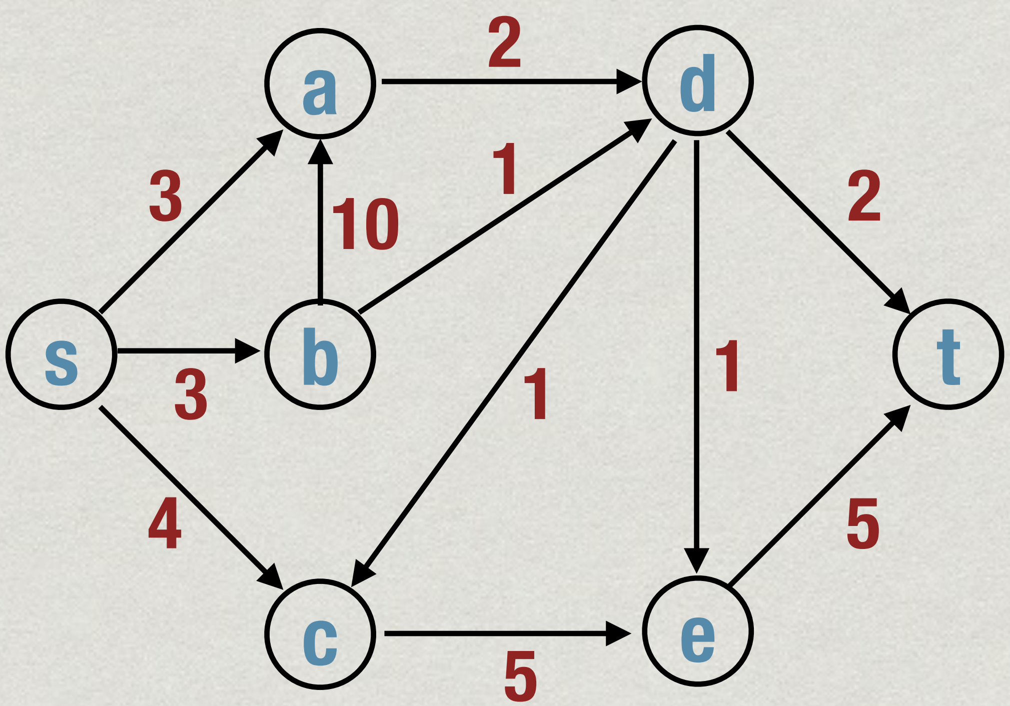
Oil network

- Network of pipelines
- Ship as much oil as possible from s to t
- No storage on the way
- A flow of 7 is possible
- Is this the maximum?



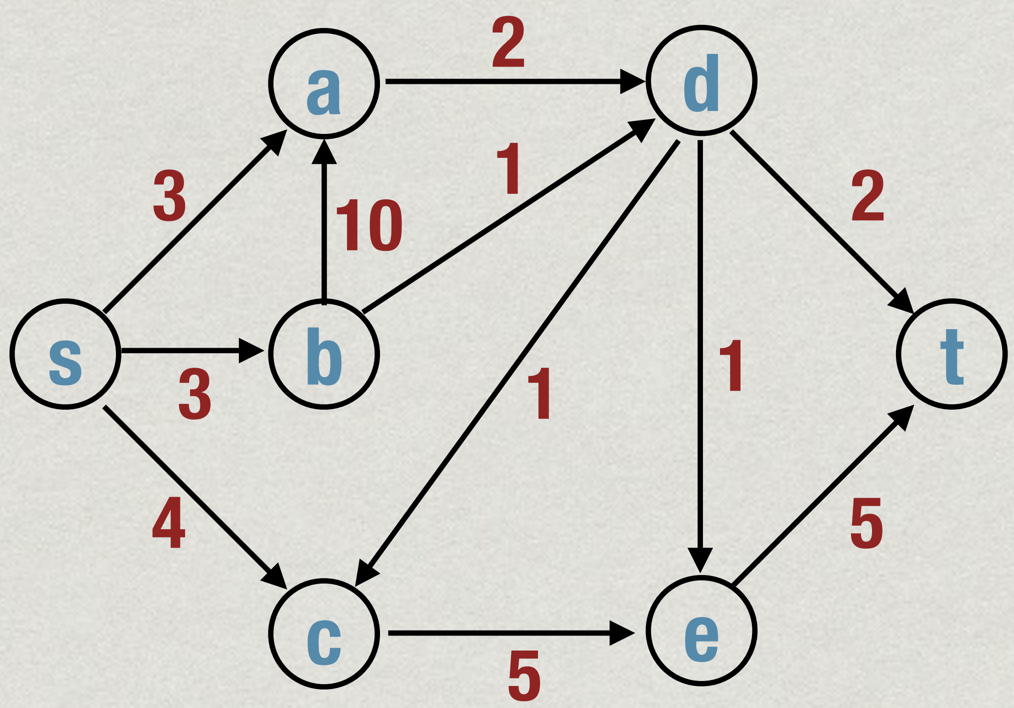
Oil network

- Network: graph $G = (V,E)$
- Special nodes: s (source), t (sink)
- Each edge e has capacity c_e
- Flow: f_e for each edge e
  - $f_e \leq c_e$
  - At each node, except s and t, sum of incoming flows equal sum of outgoing flows
- Total volume of flow is sum of outgoing flow from s



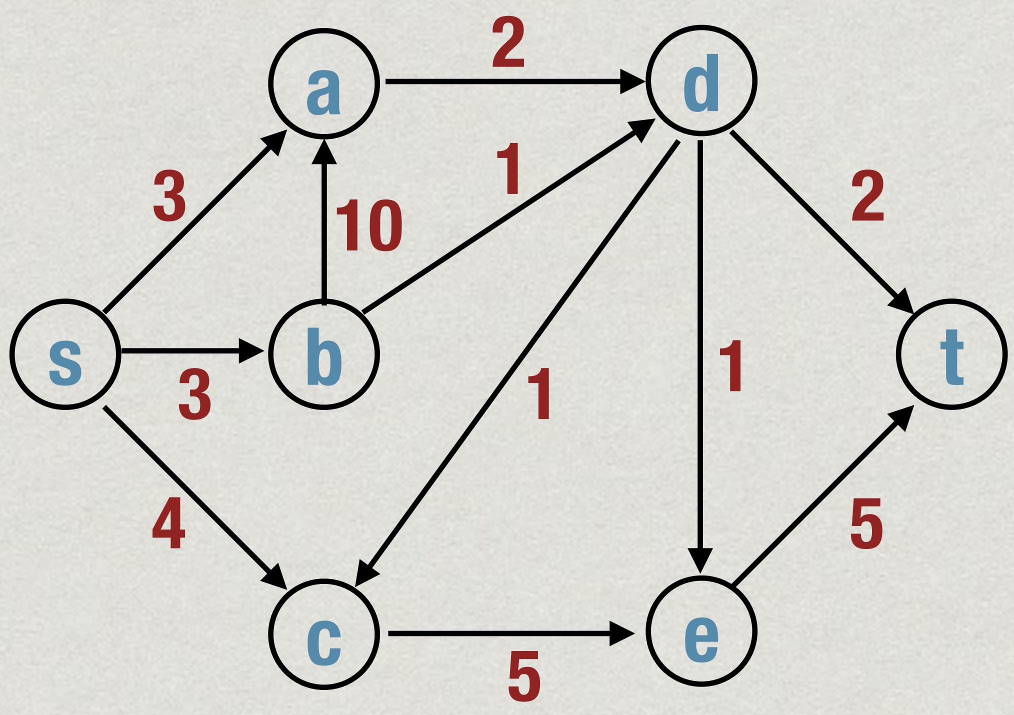
LP formulation

- Variable f_e for each edge e
  - f_{sa}, f_{bd}, f_{ce}, …
- Capacity constraints per edge
  - $f_{ba} \leq 10$, …
- Conservation of flow at each internal node
  - $f_{ad} + f_{bd} = f_{dc} + f_{de} + f_{dt}$, …
- Objective: maximize volume
  - maximize $f_{sa} + f_{sb} + f_{sc}$



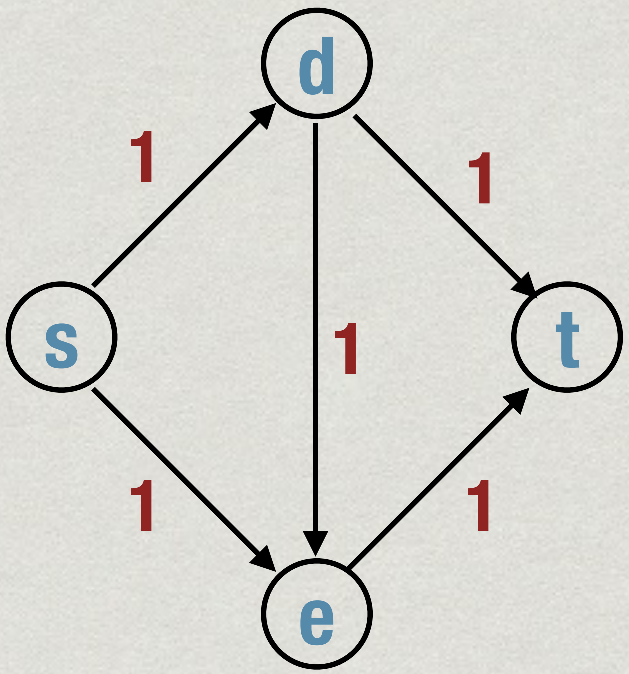
LP formulation

- Simplex solves LP, provides maximum flow, by exploring vertices of feasible region
- Moving from vertex to vertex actually corresponds to a more direct algorithm to find the maximum flow



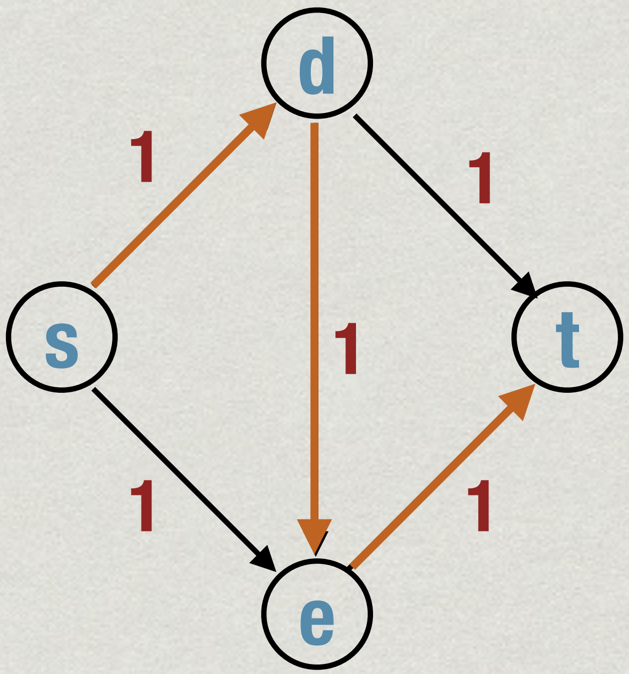
Ford-Fulkerson algorithm

- Start with zero flow
- Choose a path from s to t that is not saturated and augment the flow as much as possible
- Network on the right has max flow 2
- What if one chooses a bad flow to begin with?



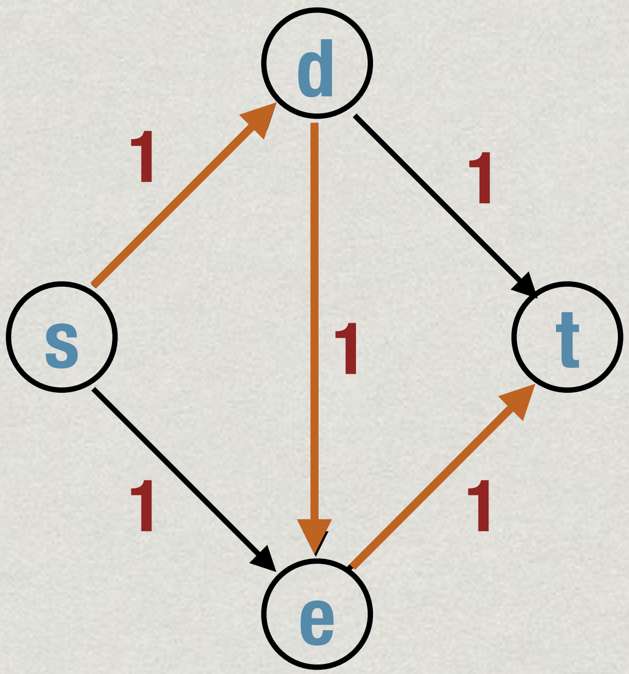
Ford-Fulkerson algorithm

- Start with zero flow
- Choose a path from s to t that is not saturated and augment the flow as much as possible
- Network on the right has max flow 2
- What if one chooses a bad flow to begin with?



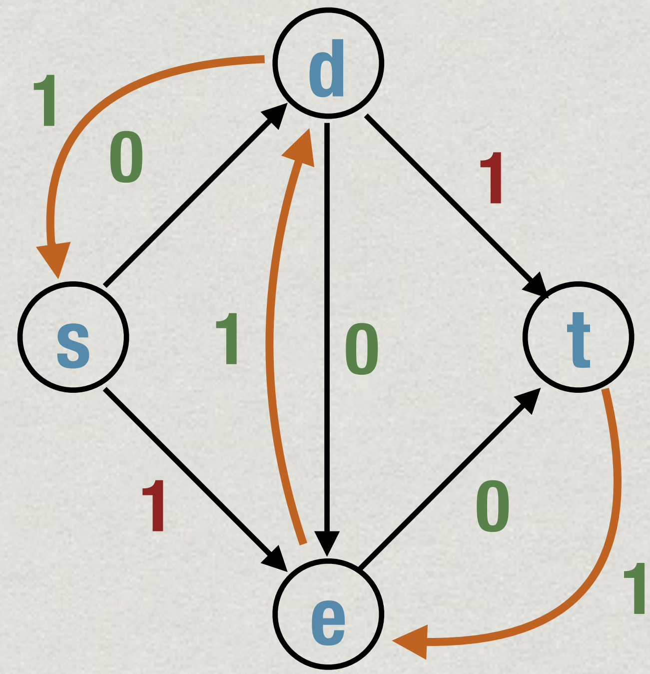
Ford-Fulkerson algorithm

- Add reverse edges to undo flow from previous steps
- Residual graph: for each edge e with capacity c_e and current flow f_e
  - Reduce capacity to $c_e - f_e$
  - Add reverse edge with capacity f_e



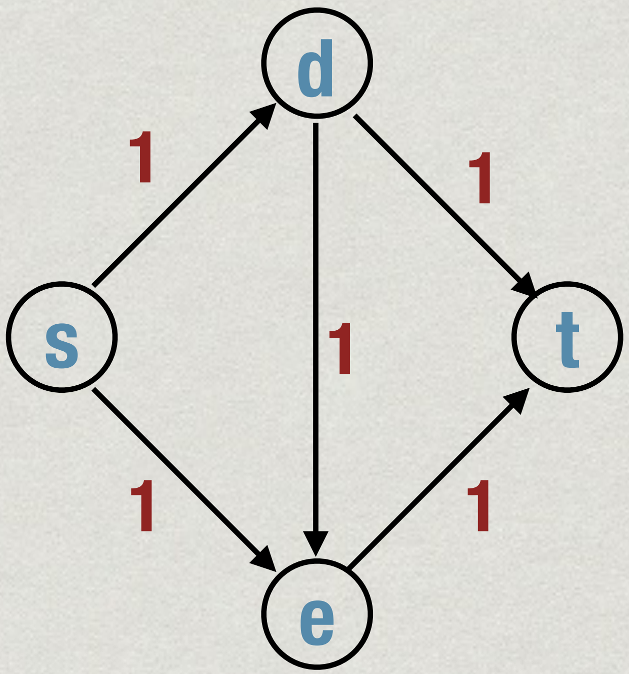
Ford-Fulkerson algorithm

- Add reverse edges to undo flow from previous steps
- Residual graph: for each edge e with capacity c_e and current flow f_e
  - Reduce capacity to $c_e - f_e$
  - Add reverse edge with capacity f_e



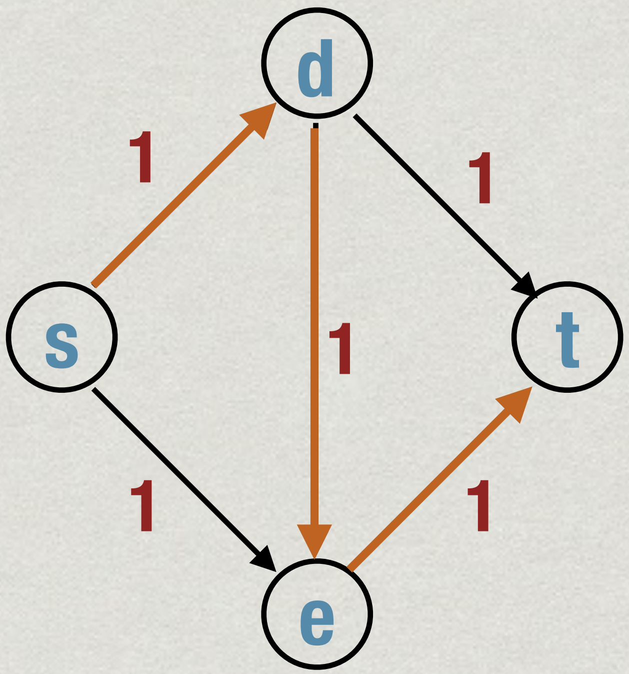
Ford-Fulkerson algorithm

- Start with zero flow
- Choose a path from s to t that is not saturated and augment the flow as much as possible
- Build residual graph
- Repeat previous two steps till there is no feasible flow from s to t



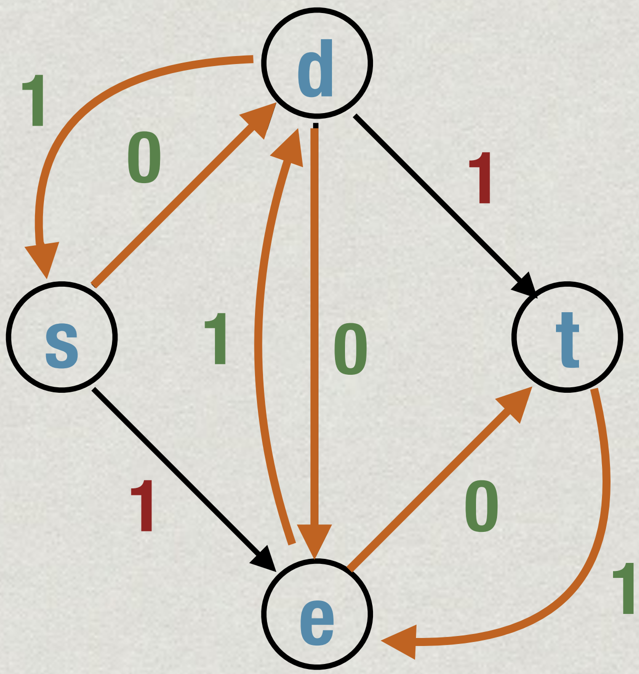
Ford-Fulkerson algorithm

- Start with zero flow
- Choose a path from s to t that is not saturated and augment the flow as much as possible
- Build residual graph
- Repeat previous two steps till there is no feasible flow from s to t

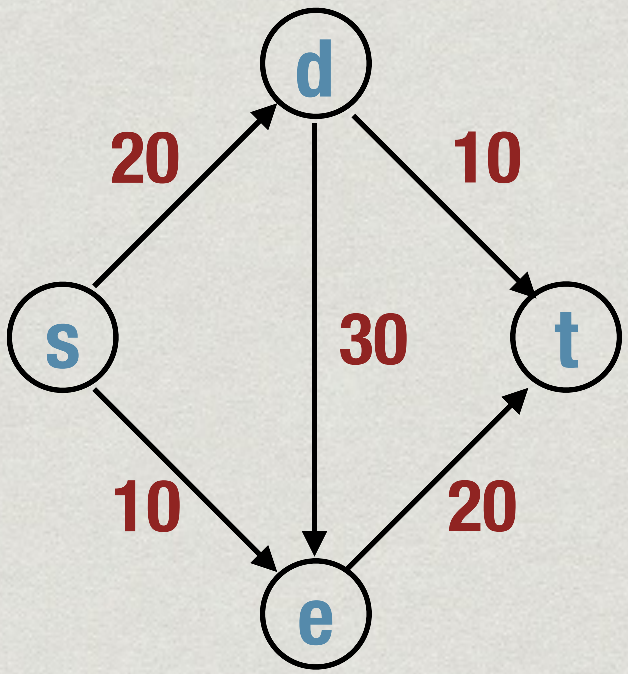


Ford-Fulkerson algorithm

* Start with zero flow
* Choose a path from s to t that is not saturated and augment the flow as much as possible
* Build residual graph
* Repeat previous two steps till there is no feasible flow from s to t

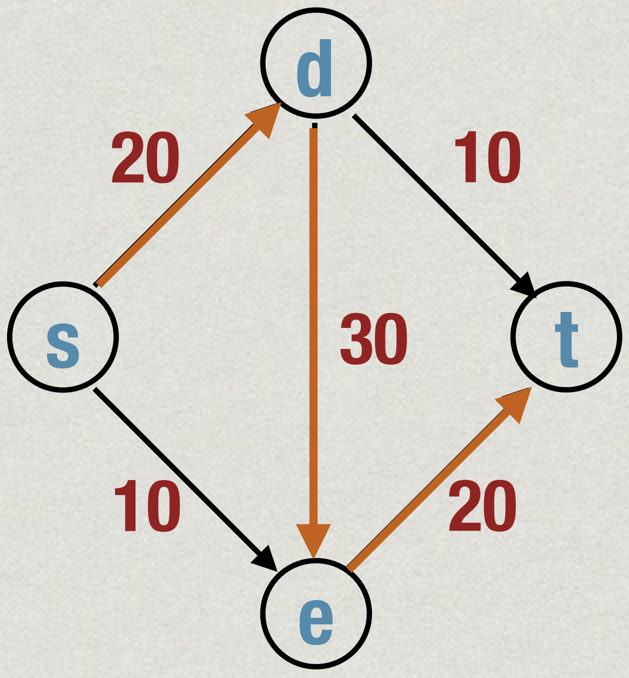


Ford-Fulkerson algorithm



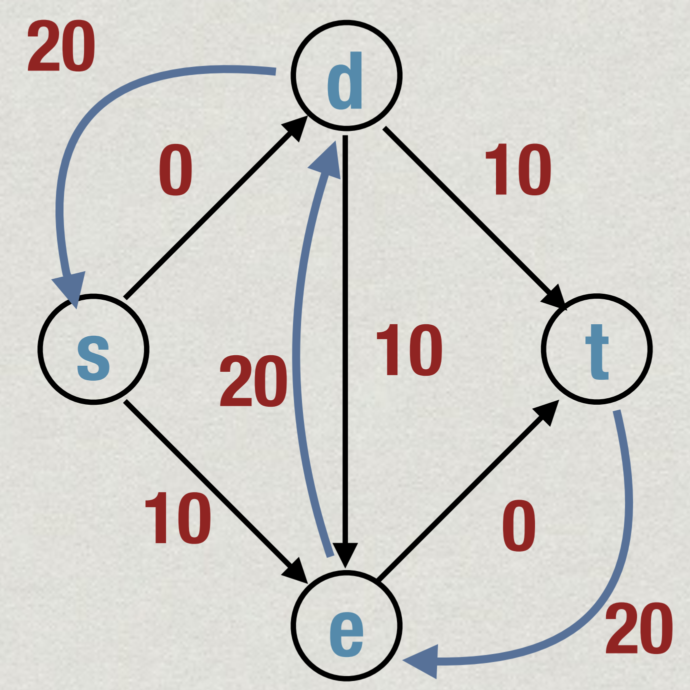
Ford-Fulkerson algorithm

- Start with flow 20, s-d-e-t



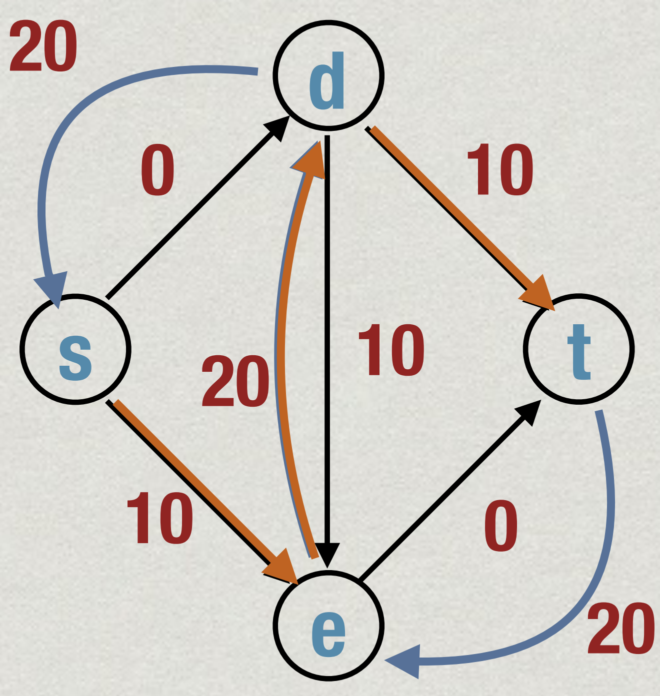
Ford-Fulkerson algorithm

- Start with flow 20, s-d-e-t
- Build residual graph



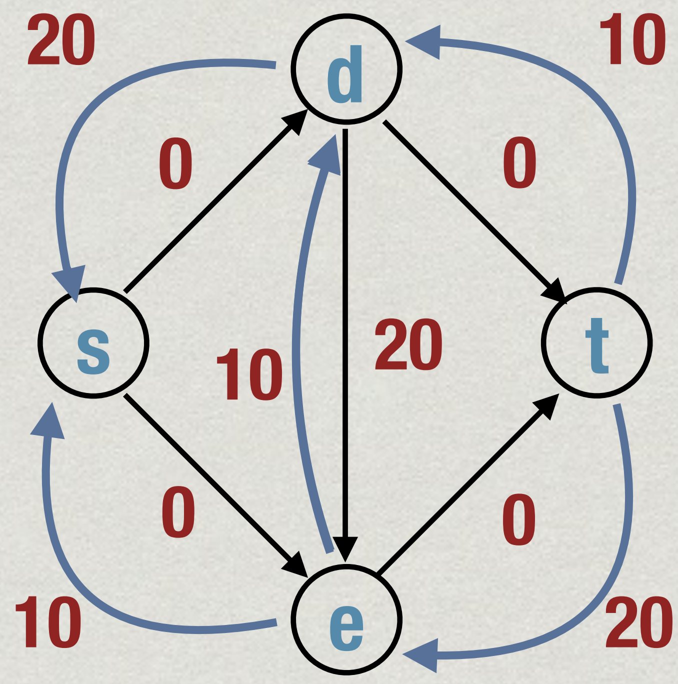
Ford-Fulkerson algorithm

- Start with flow 20, s-d-e-t
- Build residual graph
- Add flow 10, s-e-d-t



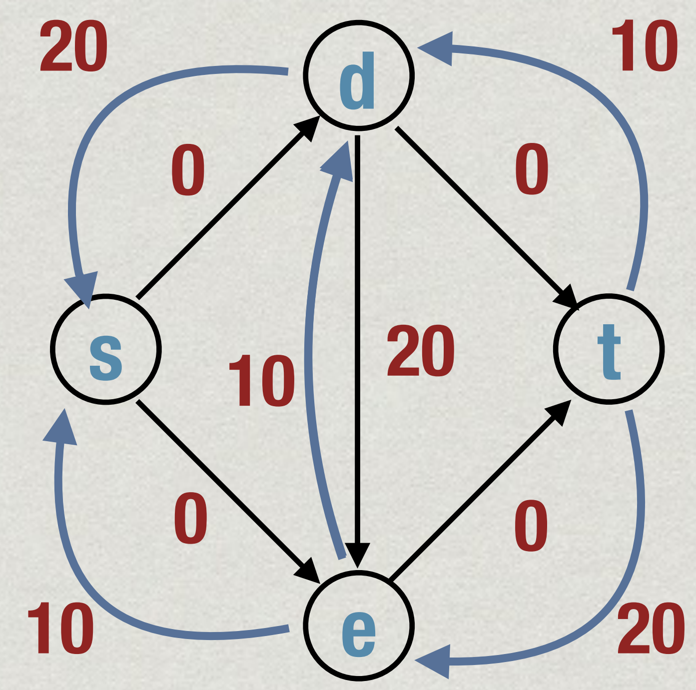
Ford-Fulkerson algorithm

- Start with flow 20, s-d-e-t
- Build residual graph
- Add flow 10, s-e-d-t
- Build residual graph



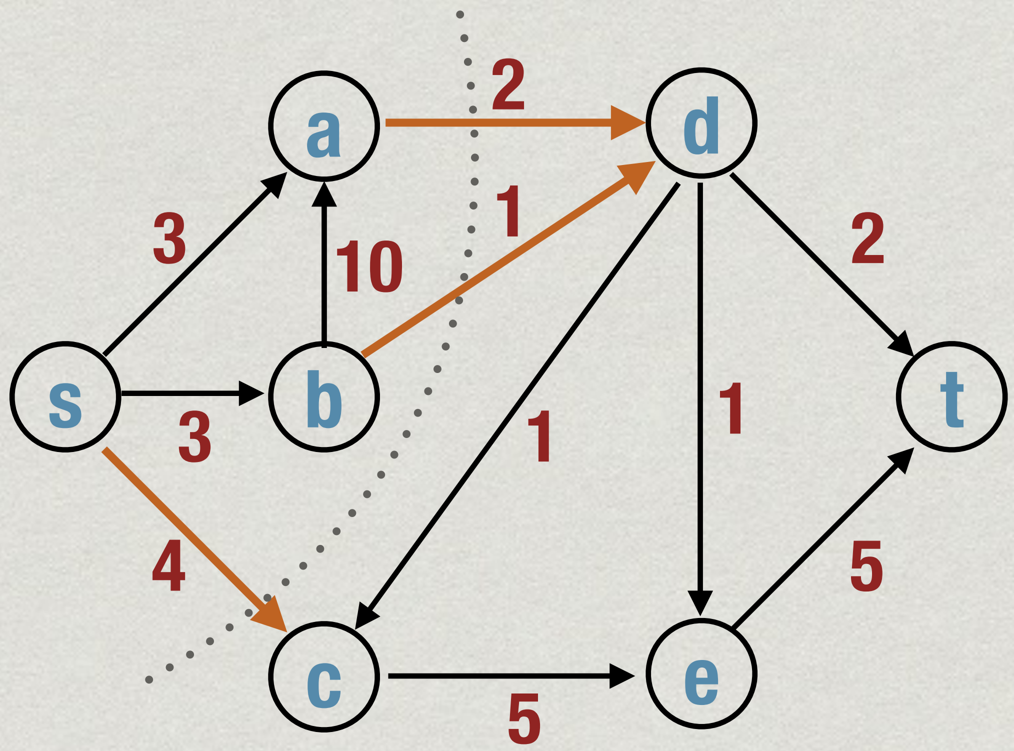
Ford-Fulkerson algorithm

- Start with flow 20, s-d-e-t
- Build residual graph
- Add flow 10, s-e-d-t
- Build residual graph
- No more feasible paths from s to t



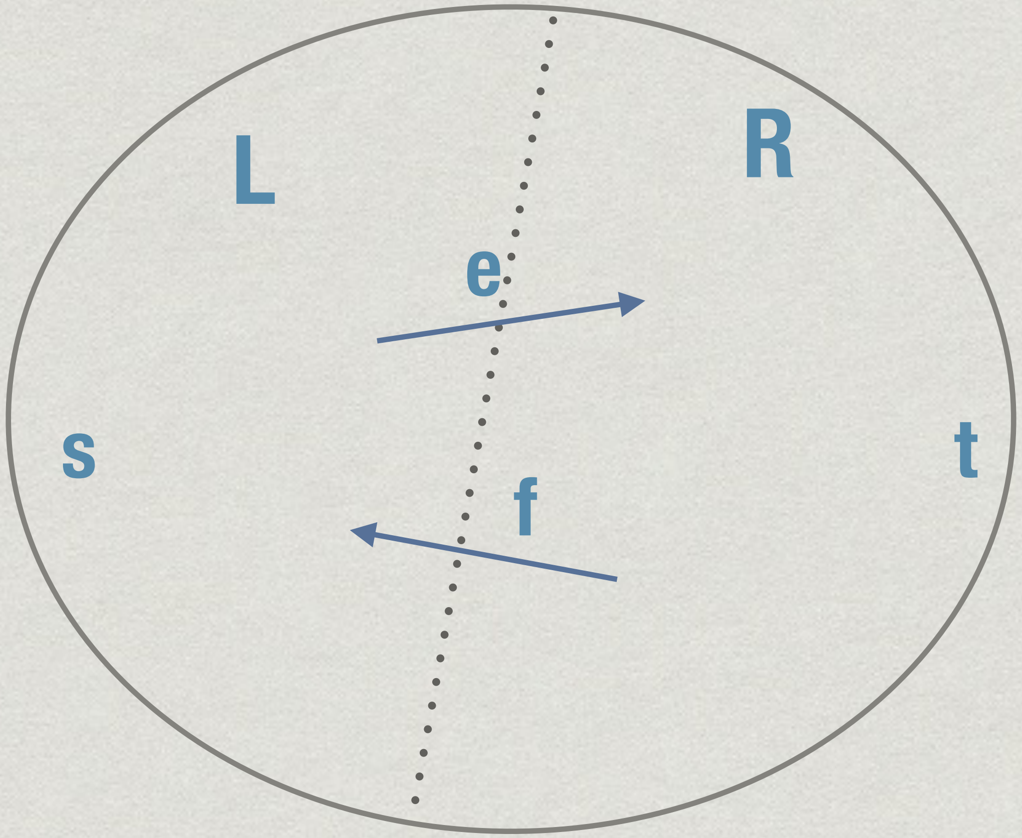
Certificate of optimality

- Edges {ad,bd,sc} disconnect s and t — (s,t)-cut
- Flow from s to t must go through this cut
  - Cannot exceed cut capacity = 7
- In general, max flow cannot exceed min cut capacity



Max flow-min cut theorem

- In fact, max flow is always equal to min cut!
- At max flow, no path from s to t in residual graph
  - s can reach L, R can reach t
  - Any edge e from L to R must be at full capacity
  - Any edge f from R to L must be at zero capacity



Ford-Fulkerson algorithm

- Choose augmenting paths wisely
- If we keep going through the middle edge, 200 iterations to find the max flow
  - FF can take time proportional to max capacity
- Use BFS to find augmenting path with fewest edges — iterations bounded by |V|x|E|, regardless of capacities