Max-flow, Min-cut

Network flow

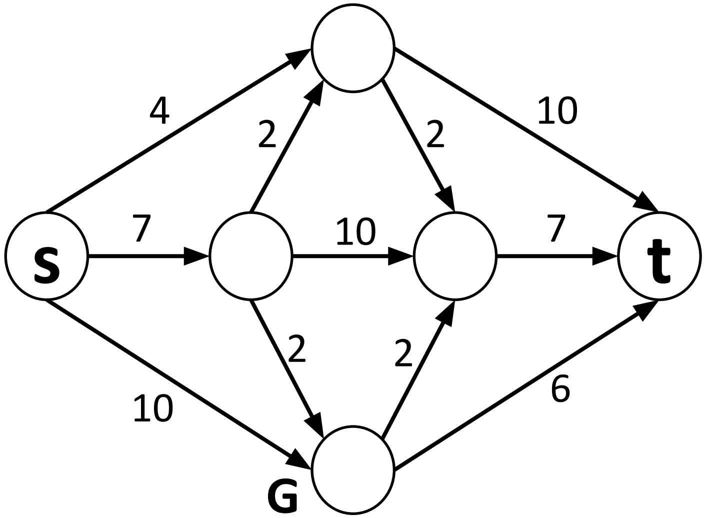
Max-flow

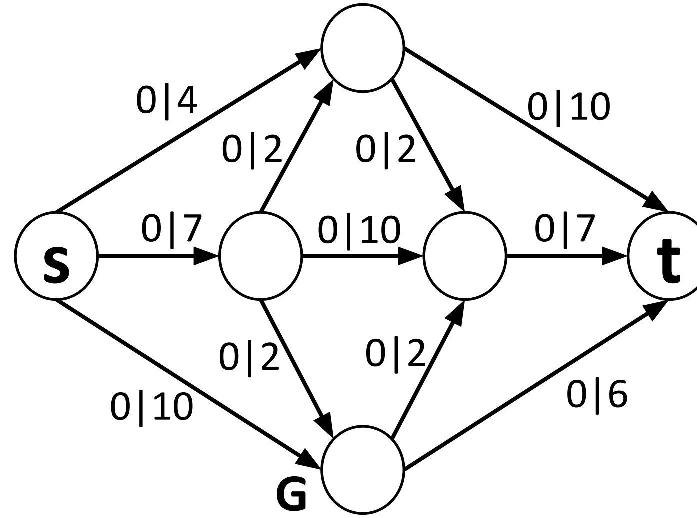
- Maximize the total amount of flow from s to t subject to two constraints
  - Flow on edge e doesn't exceed c(e)
  - For every node v ≠ {s, t}, incoming flow is equal to outgoing flow

Max-flow: Ford-Fulkerson

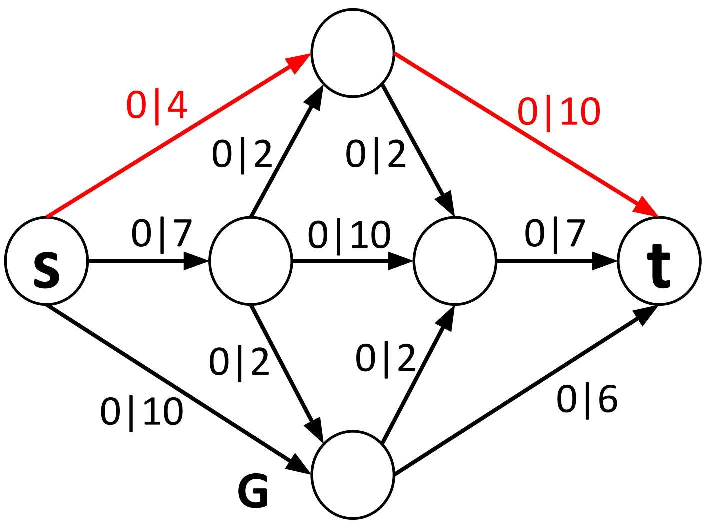
- Find paths from s to t using depth first search
- Find paths using the residual graph G’

Ford-Fulkerson: example

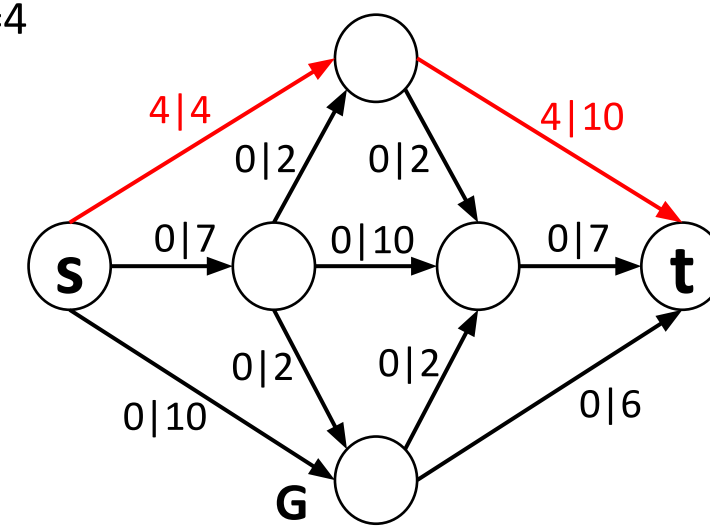




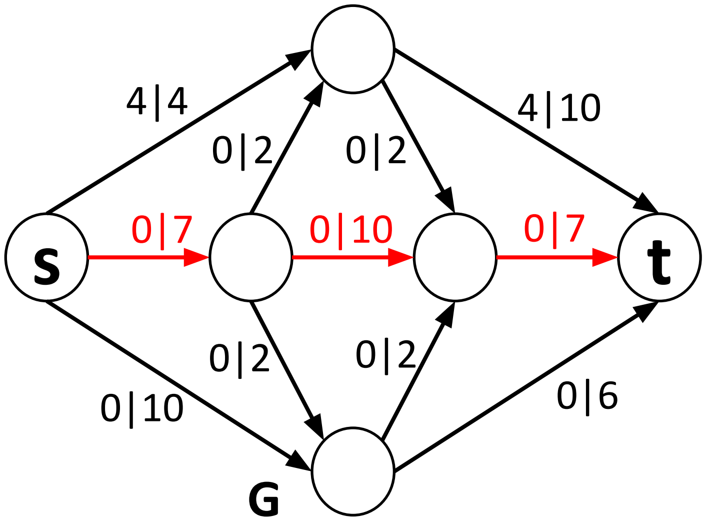
Depth first search



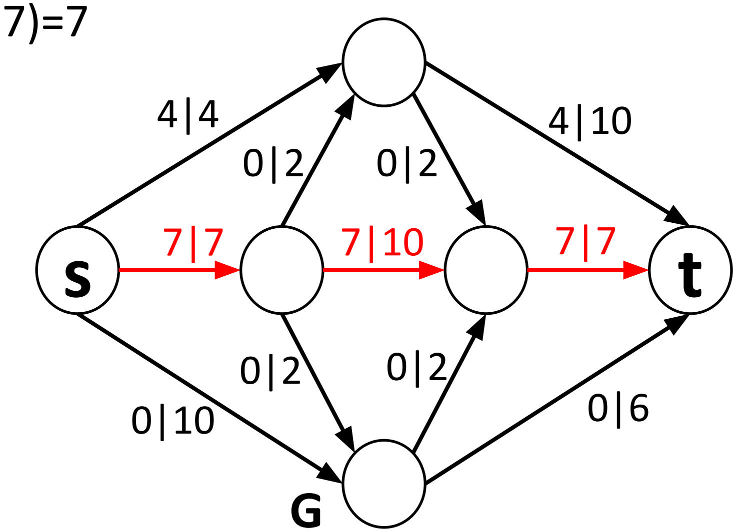
f = min(4,10)=4

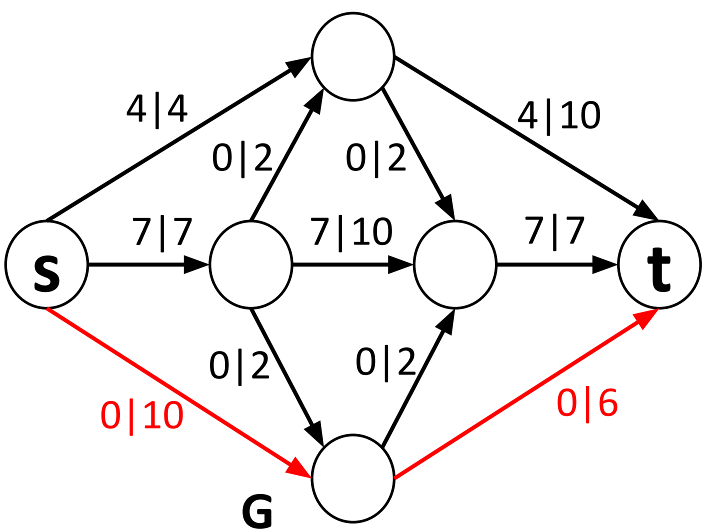


Ford-Fulkerson: DFS

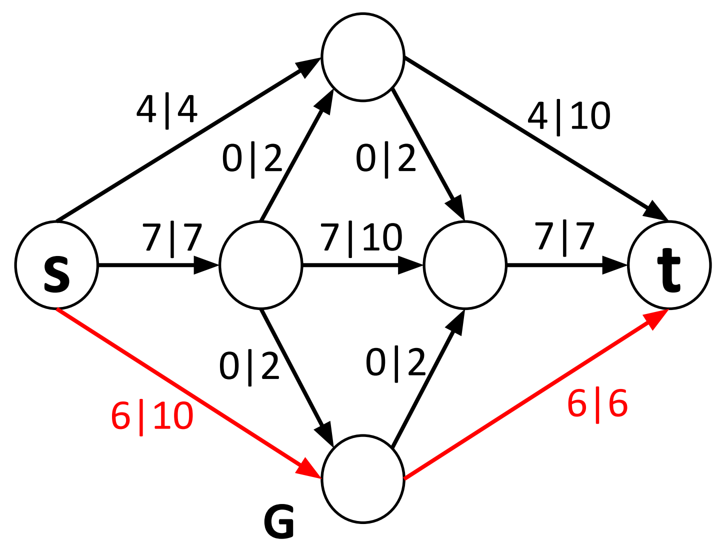


f = min(7,10,7)=7

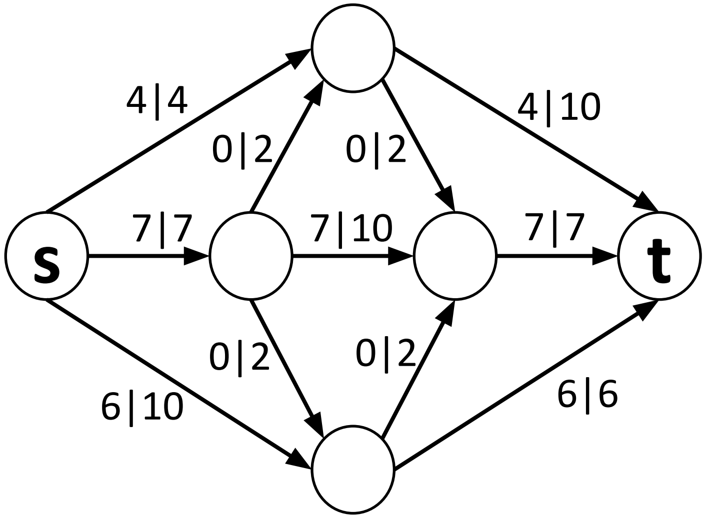




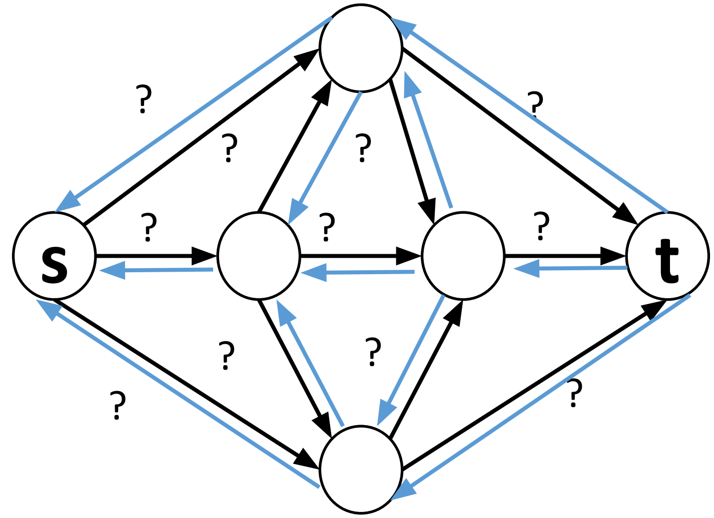
f = min(10,6)=6



Ford-Fulkerson: Residual graph

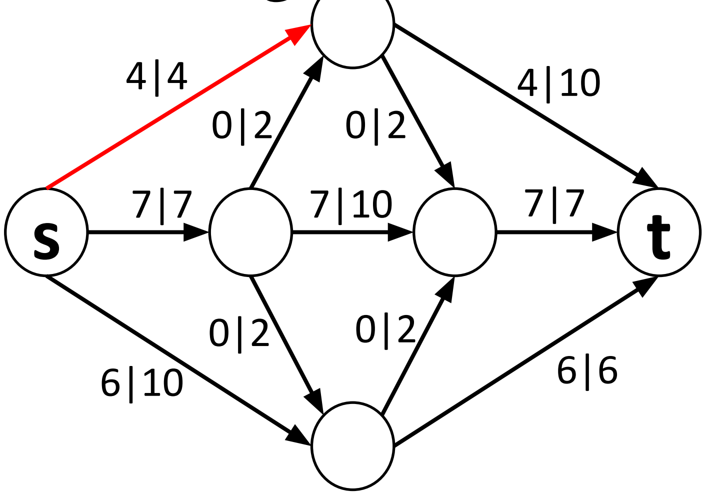


G

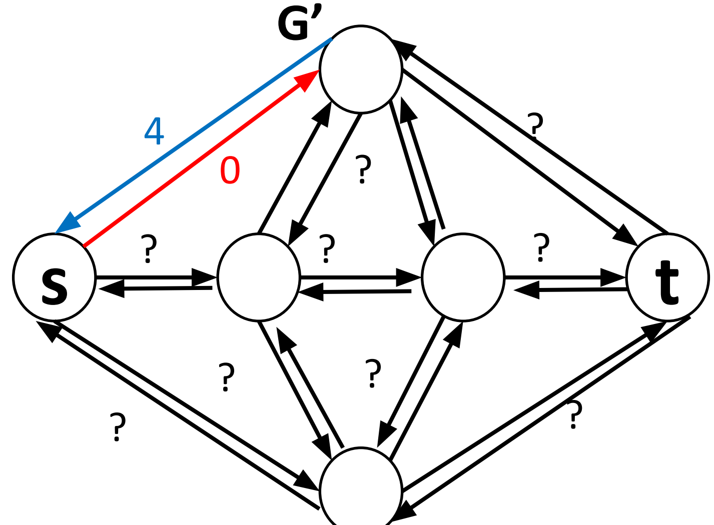


G'

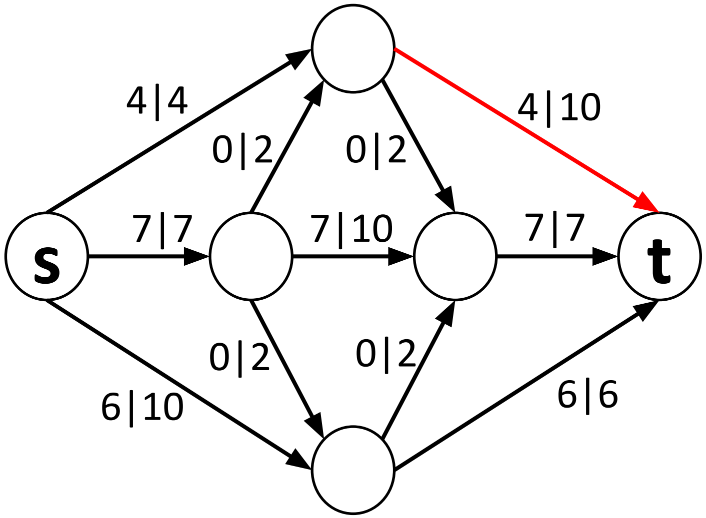
G



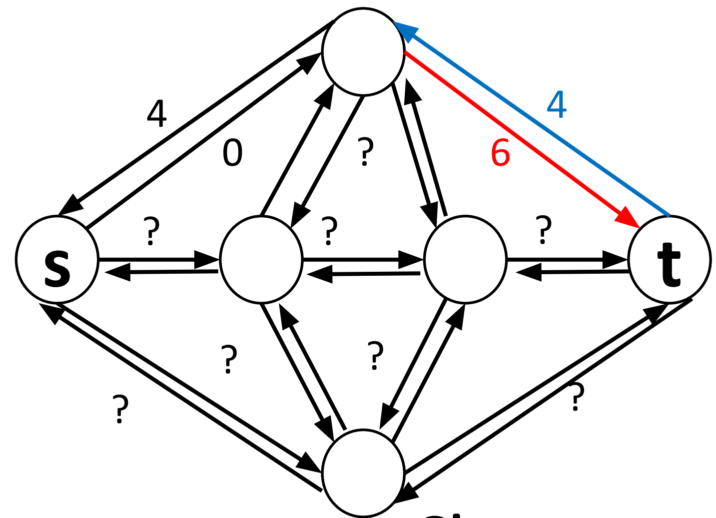
G'



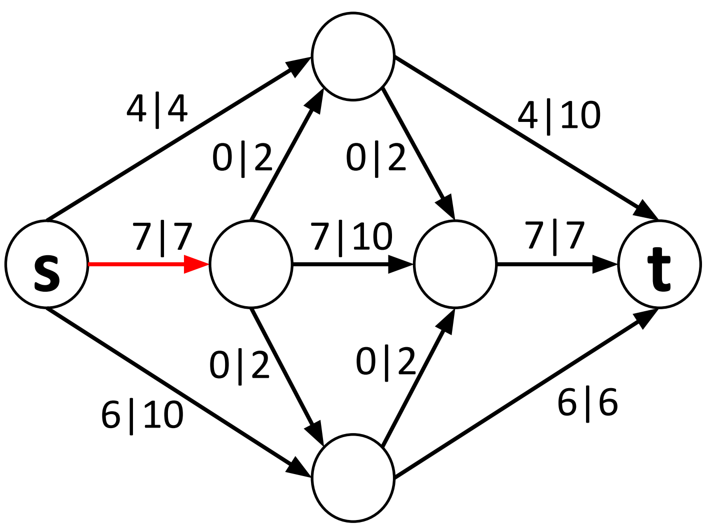
Ford-Fulkerson: Residual graph



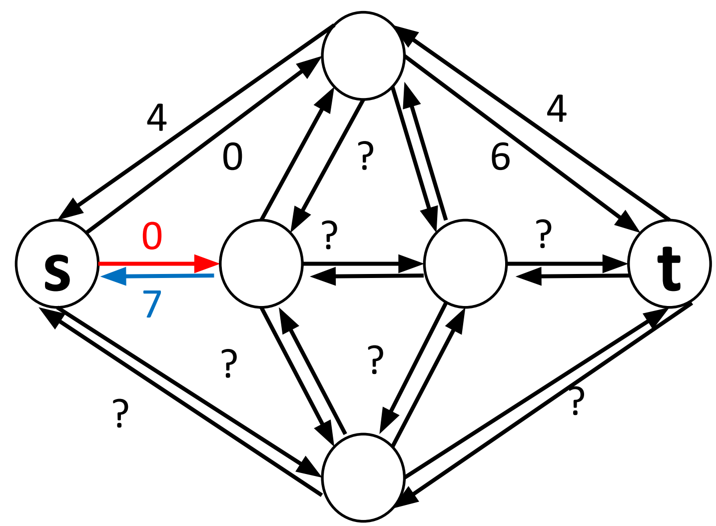
G



G’

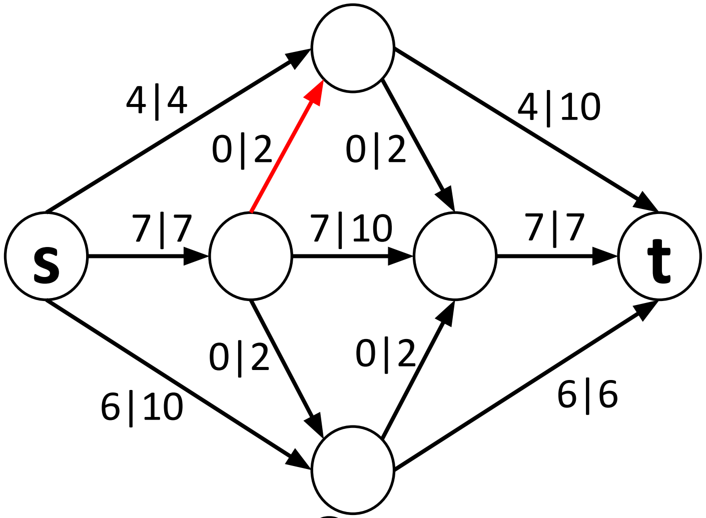


G

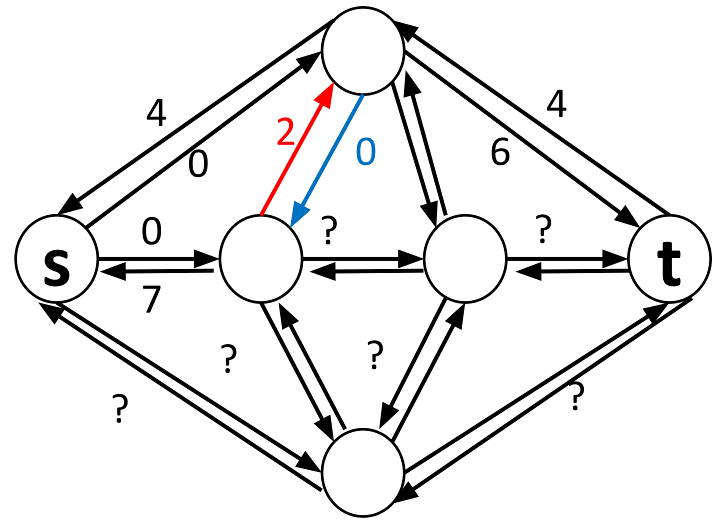


G’

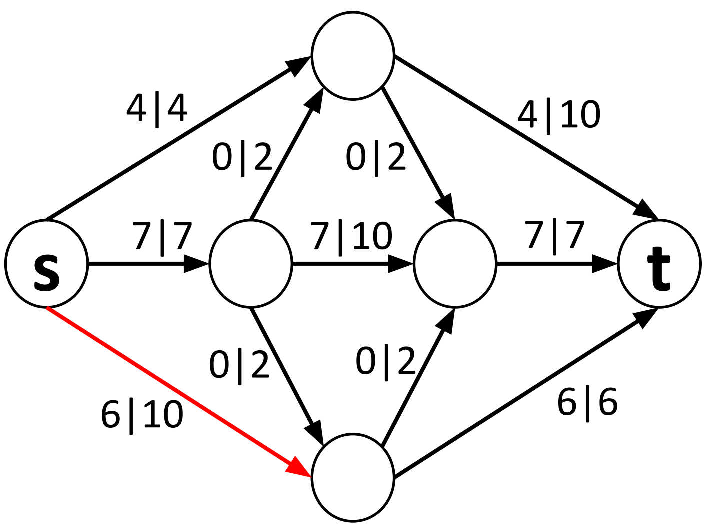
Ford-Fulkerson: Residual graph



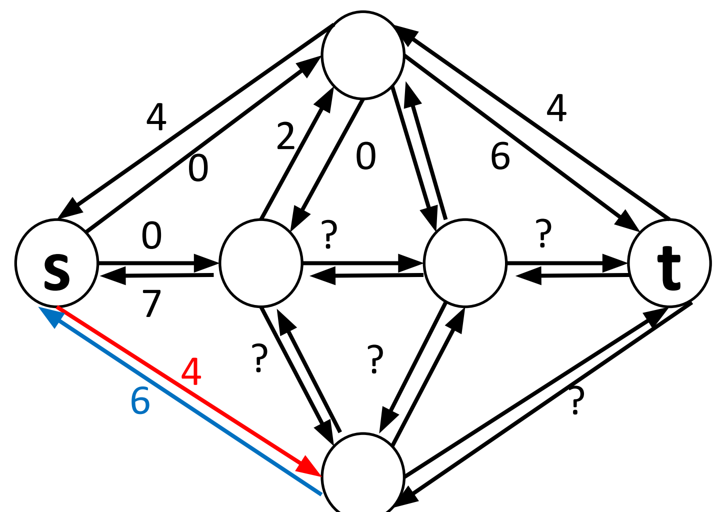
G



G’



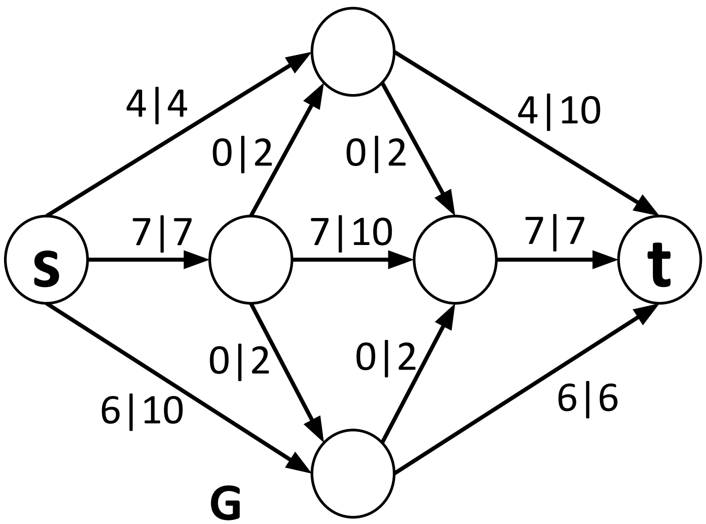
G

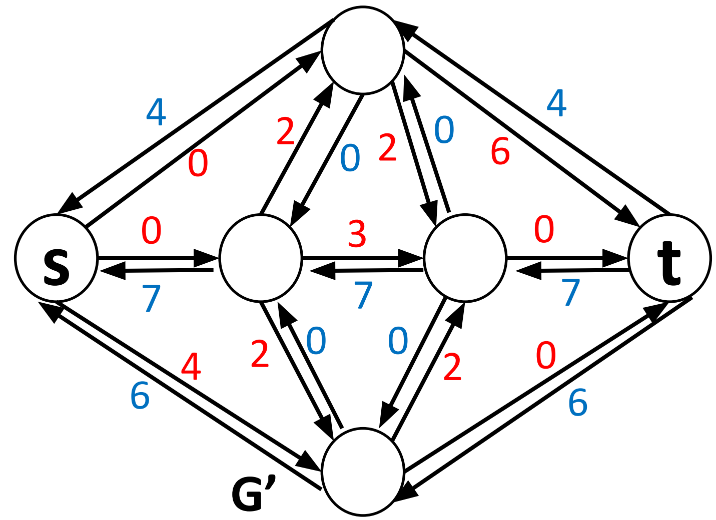


G’

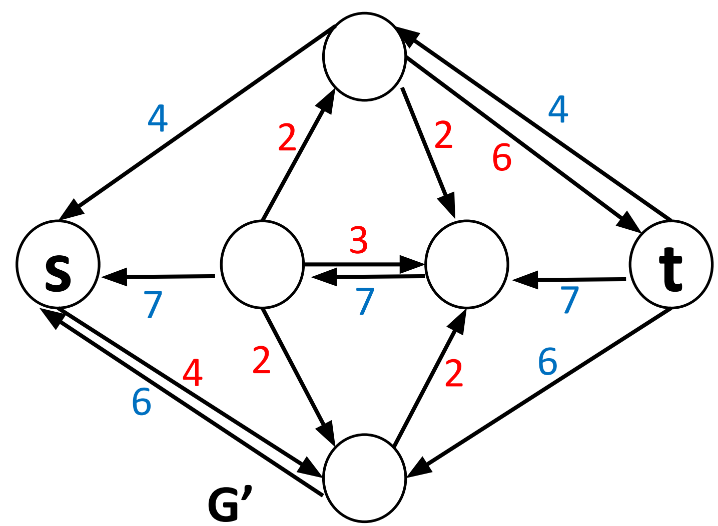
Ford-Fulkerson: Residual graph

Keep work on for the rest of edges



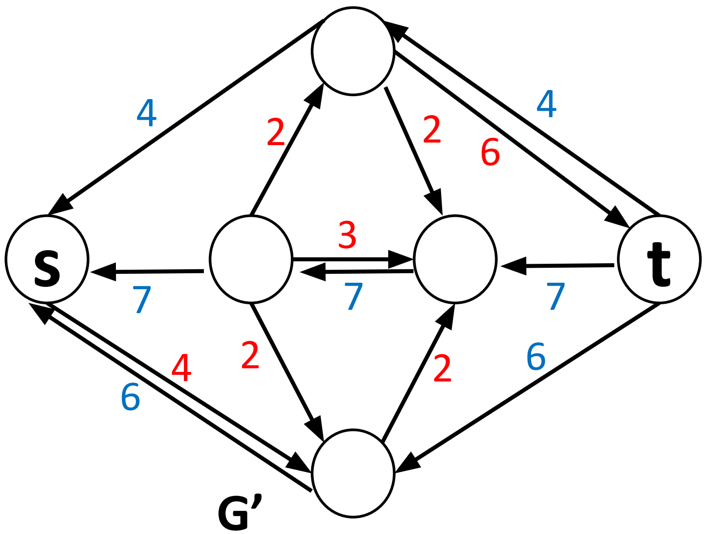


Remove "0" edge (optional)

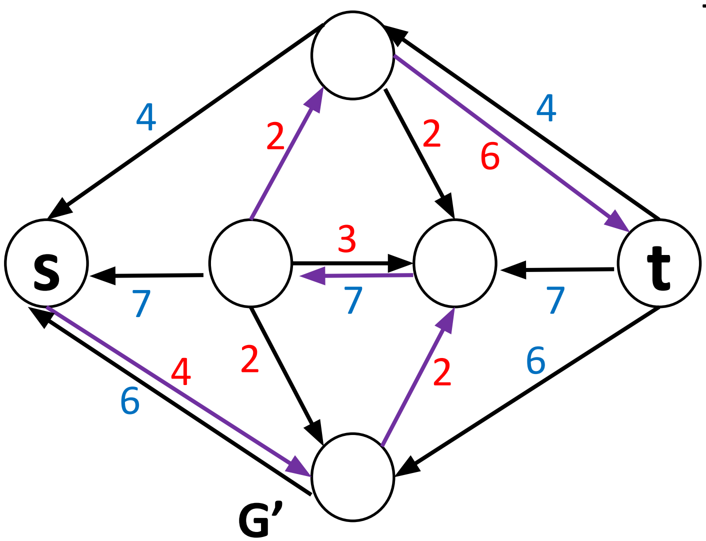


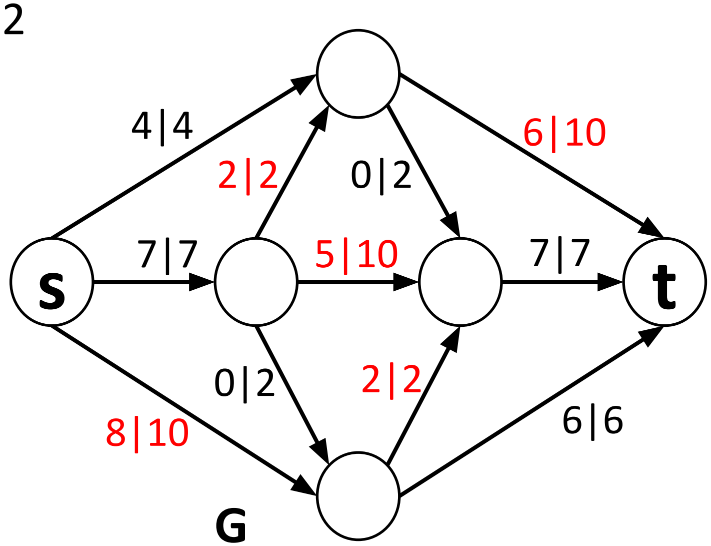
Ford-Fulkerson: Residual graph

Any more paths?

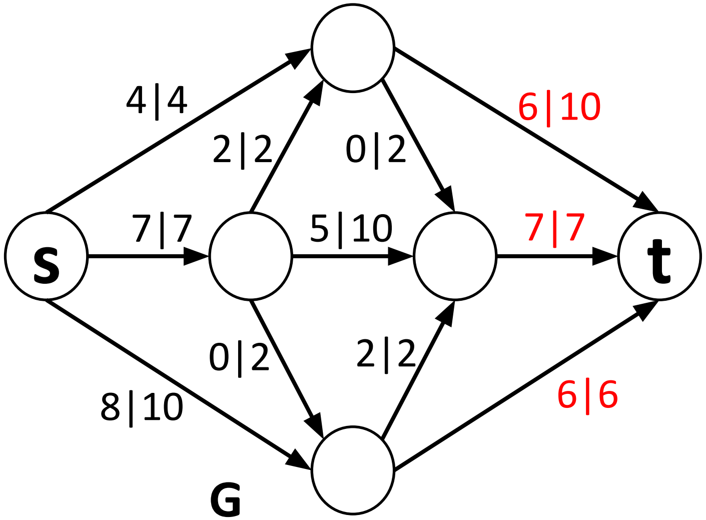


f = min(6,2,3,2,4)=2





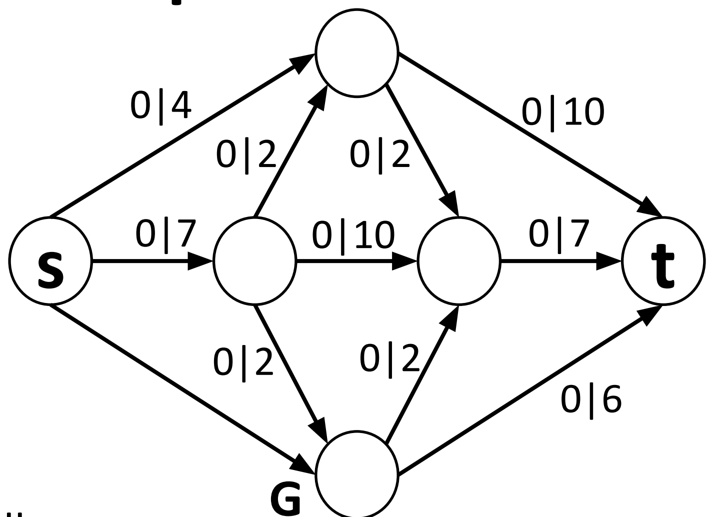
Max flow = 6 + 7 +6 = 19



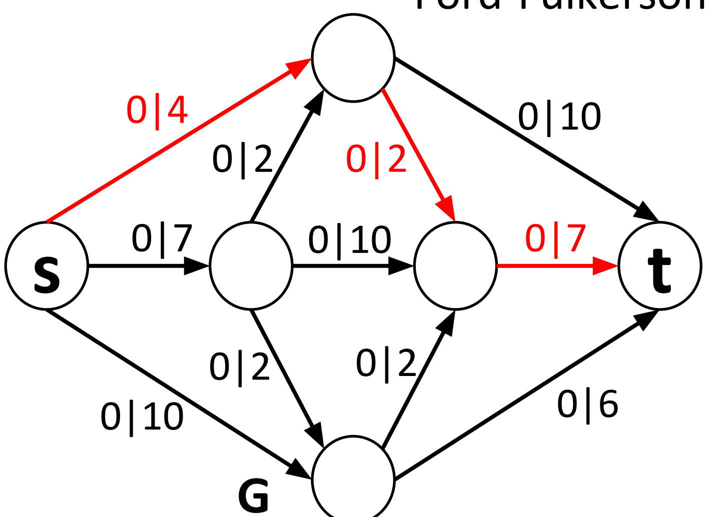
Edmonds-Karp

- Edmonds-Karp = Ford-Fulkerson + "Choose the augmenting path with the smallest number of edges" or "Choose the augmenting path with the largest bottle neck value"

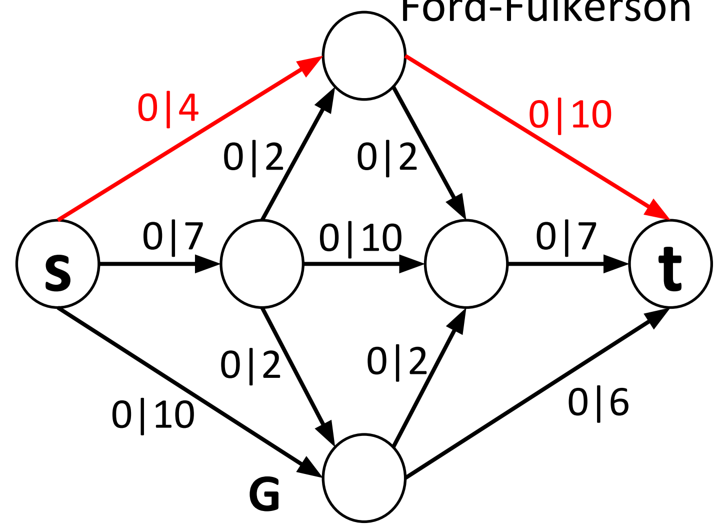
Edmonds-Karp vs Ford-Fulkerson



Ford-Fulkerson



Edmonds-Karp
Ford-Fulkerson



Which one is the valid first choice of Edmonds-Karp?
Which one is the valid first choice of Ford-Fulkerson?

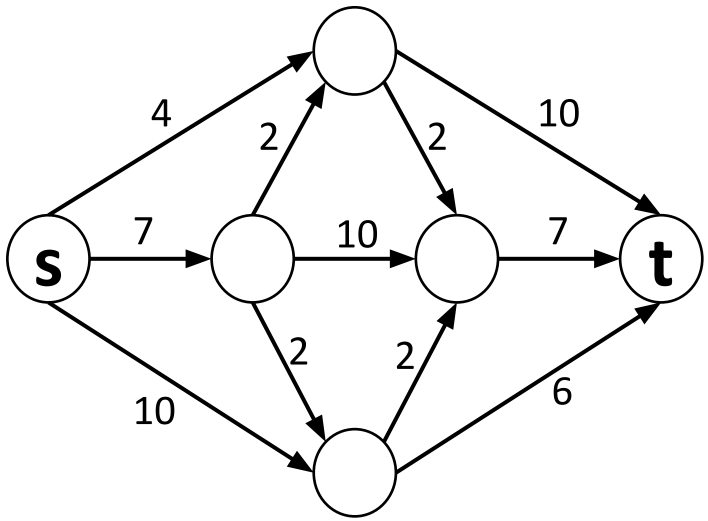
Min cut

- We want to remove some edges from the graph such that after removing the edges, there is no path from s to t
- The cost of removing e is equal to its capacity c(e)
- The minimum cut problem is to find a cut with minimum total cost

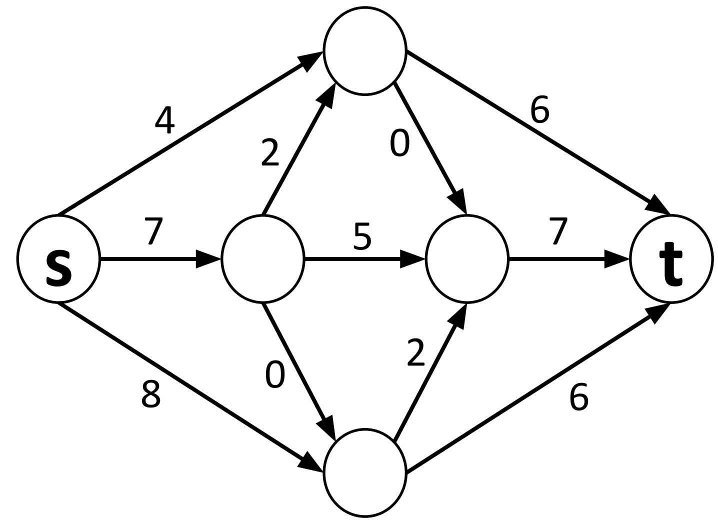
Min cut: approach

- "Subtract" the max-flow from the original graph
- Mark all nodes reachable from s. Call the set of reachable nodes A
- Now separate these nodes from the others
- Cut edges going from A to $V - A$

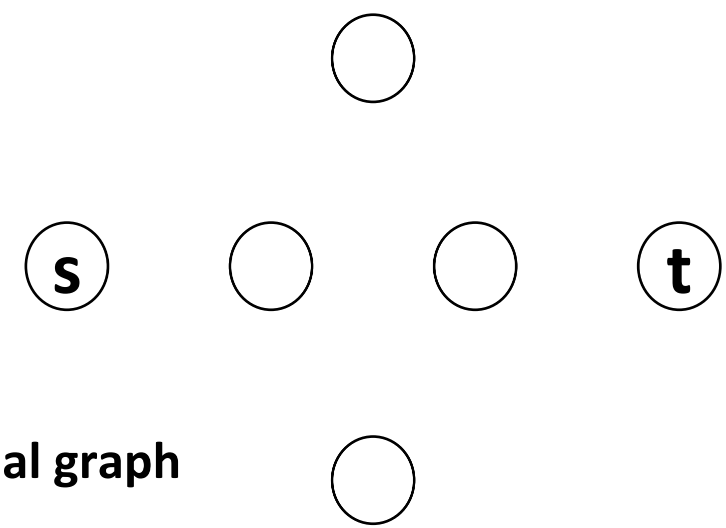
Min cut: example



G

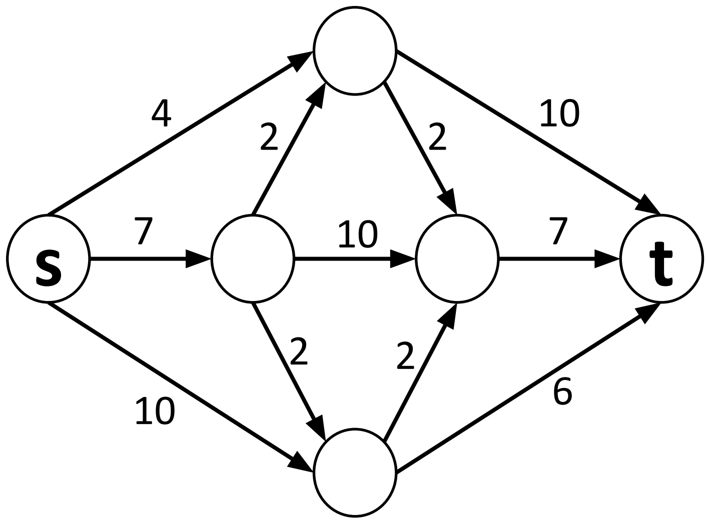


G max flow

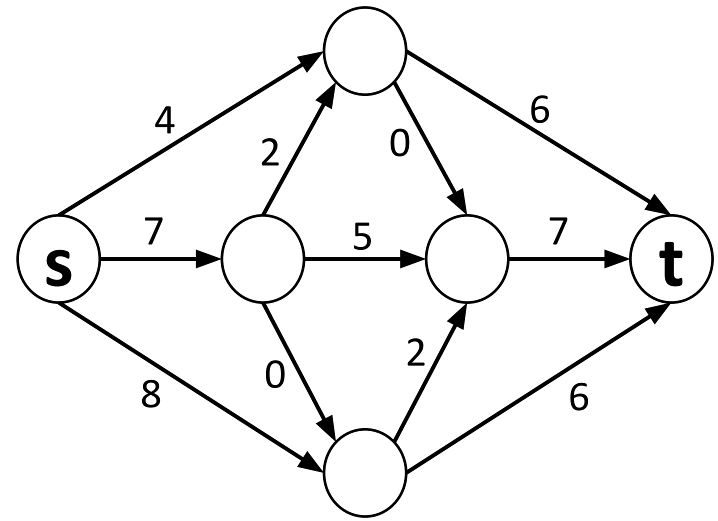


G – G max flow = residual graph

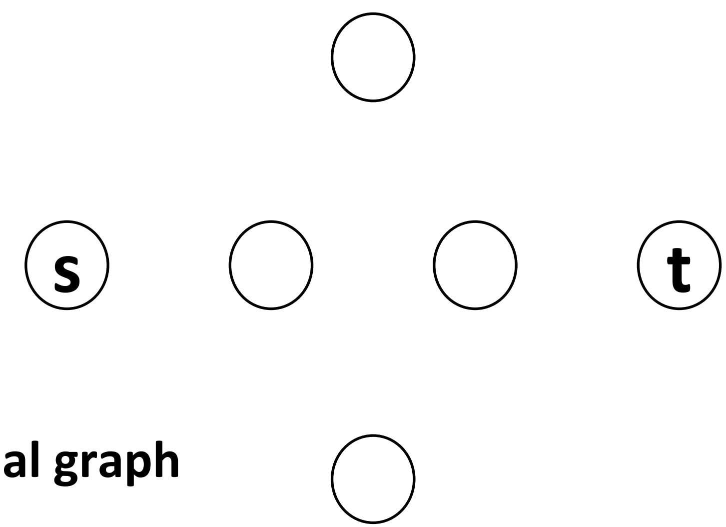
Min cut: example



G

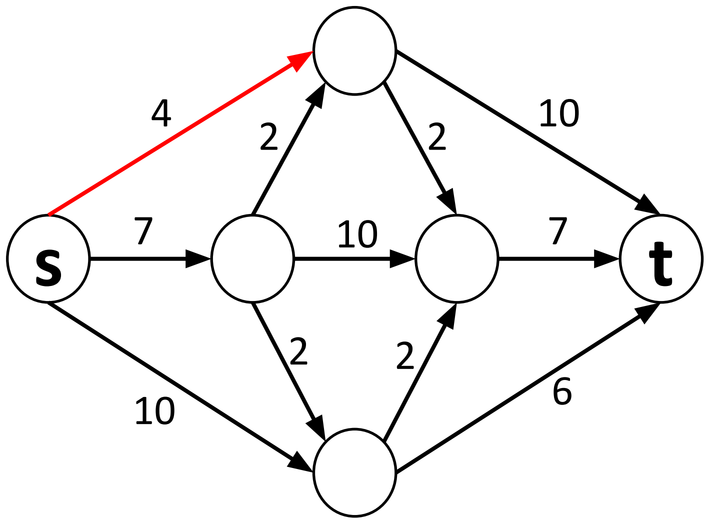


G max flow

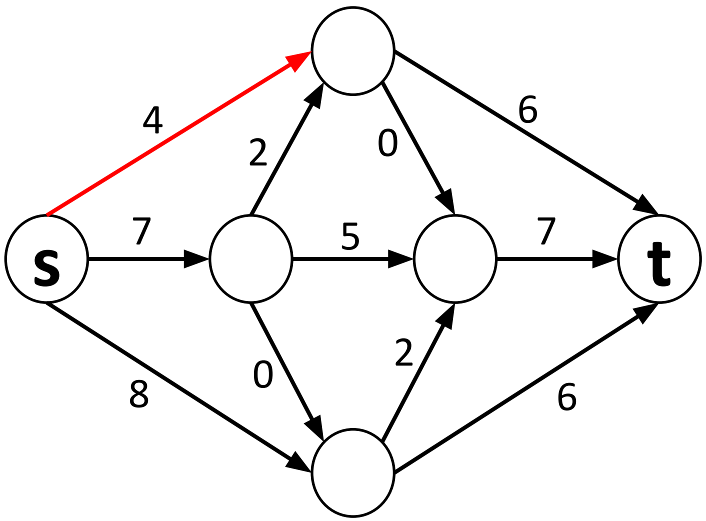


G – G max flow = residual graph

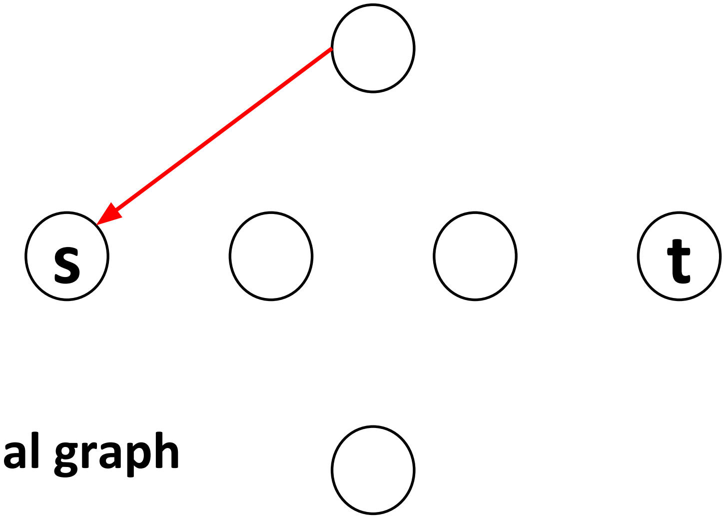
Min cut: example



G

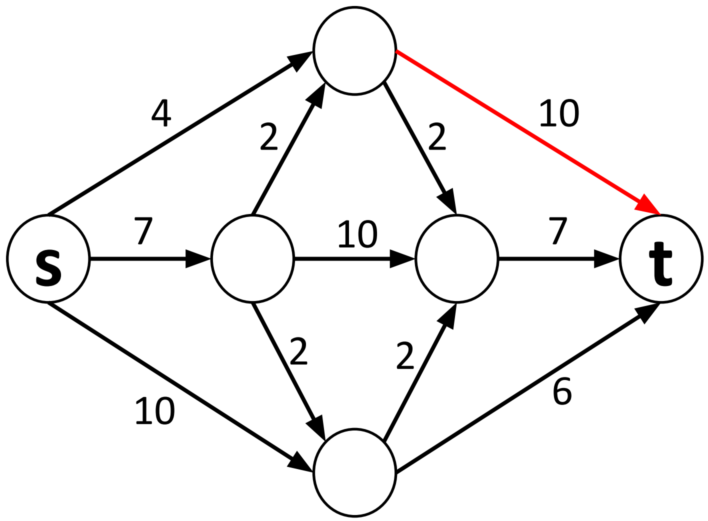


G max flow

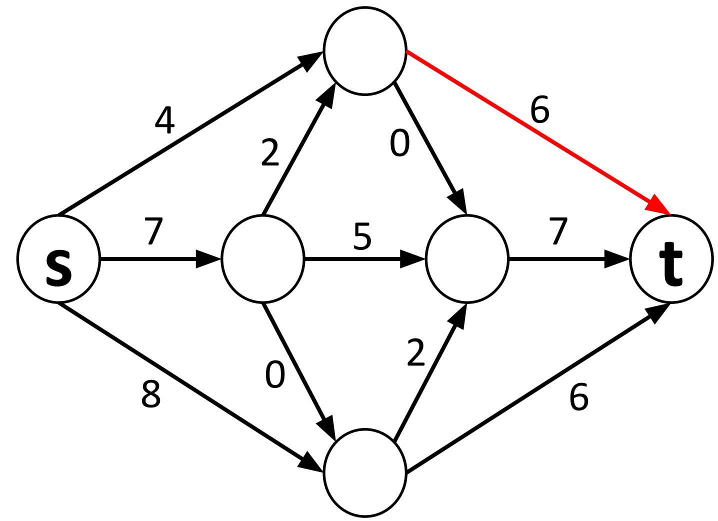


G – G max flow = residual graph

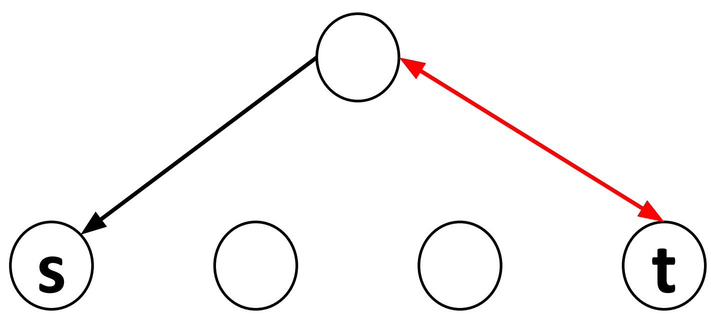
Min cut: example



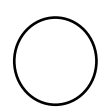
G



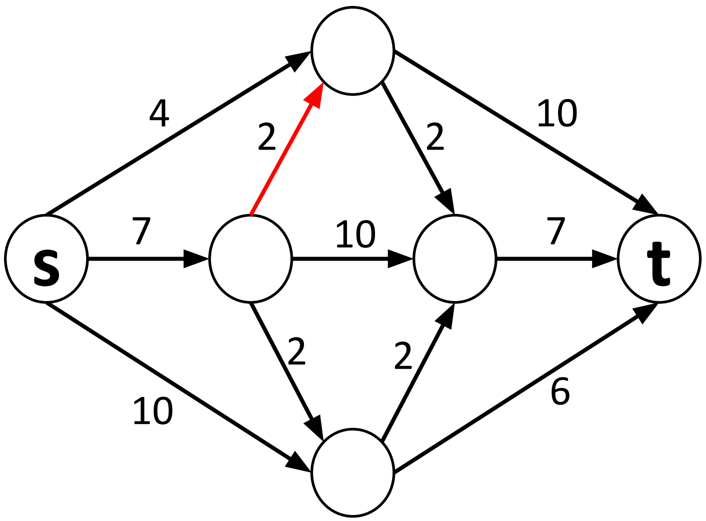
G max flow



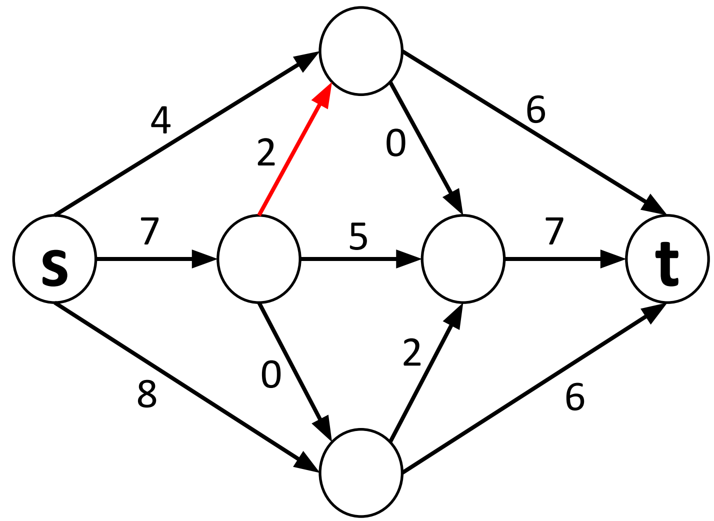
G – G max flow = residual graph



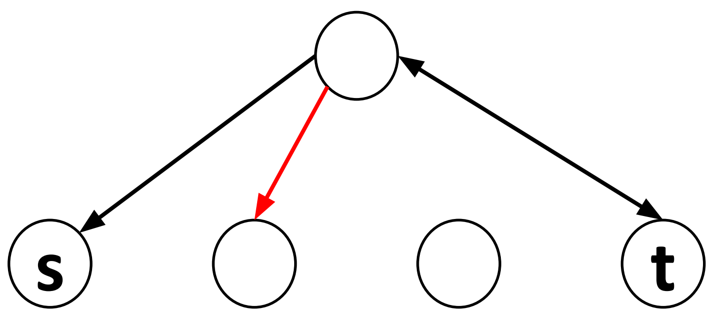
Min cut: example



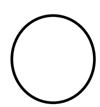
G



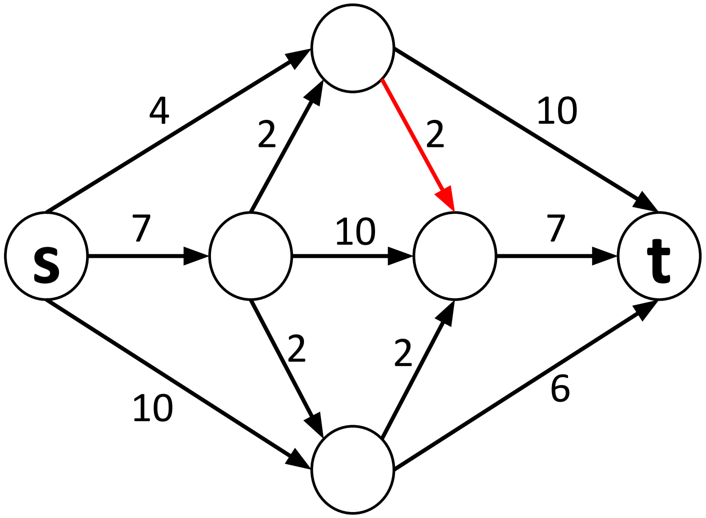
G max flow



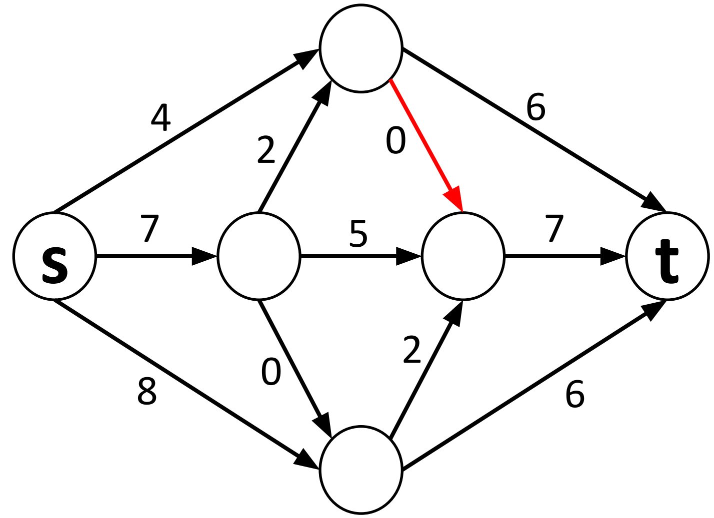
G – G max flow = residual graph



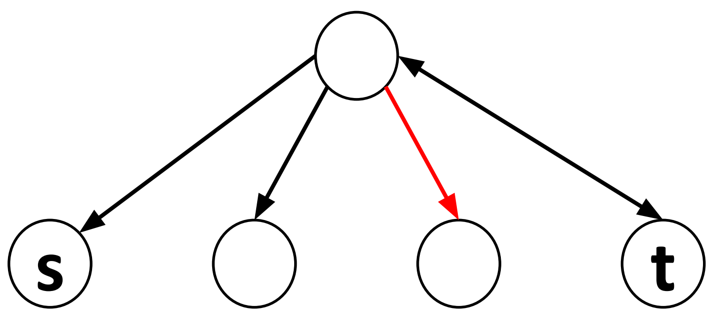
Min cut: example



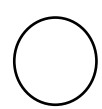
G



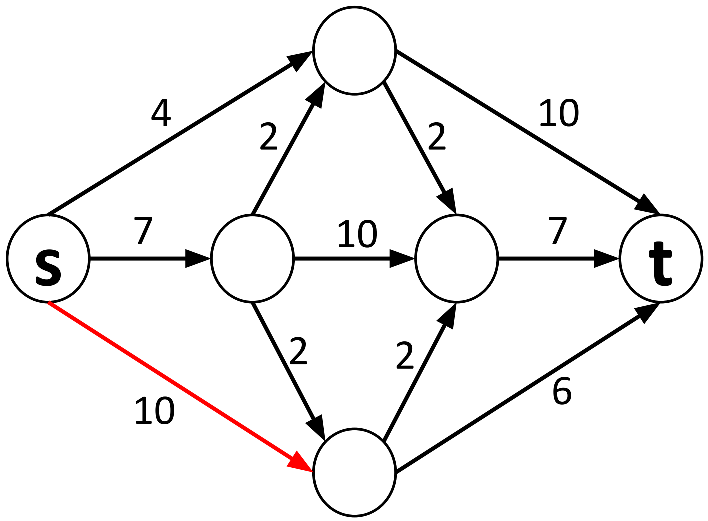
G max flow



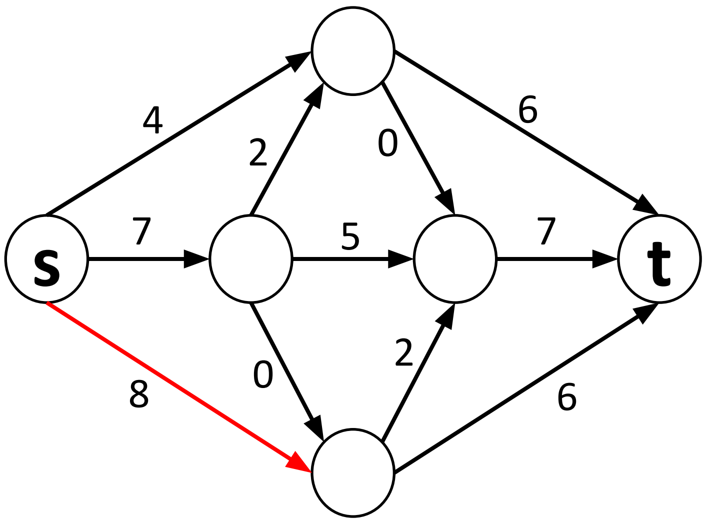
G – G max flow = residual graph



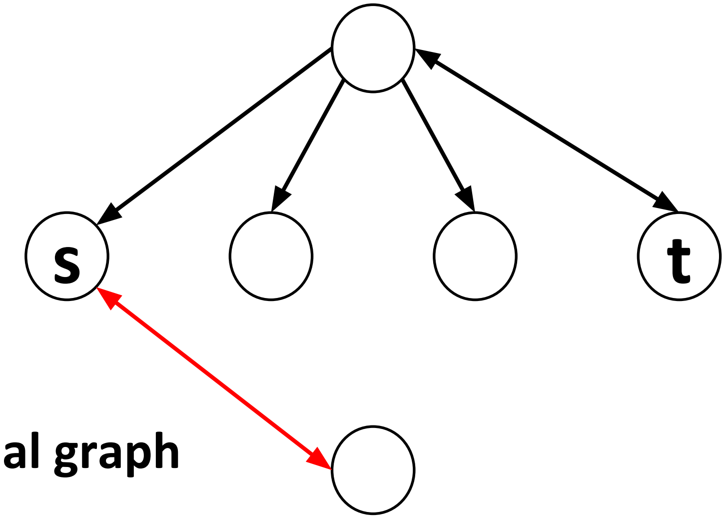
Min cut: example



G

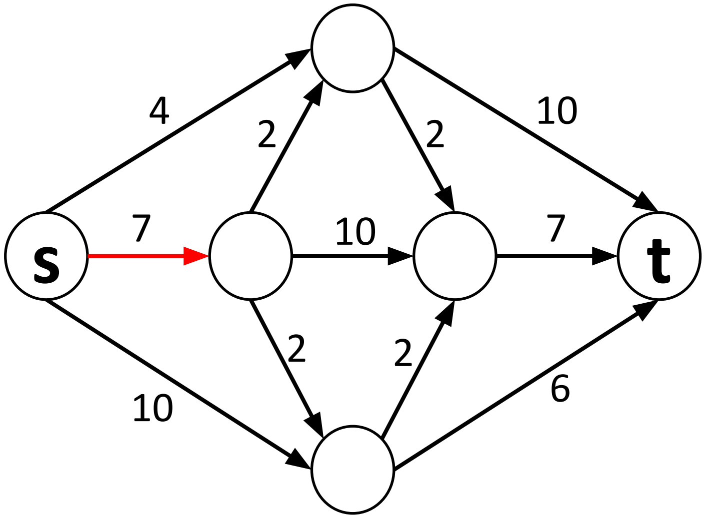


G max flow

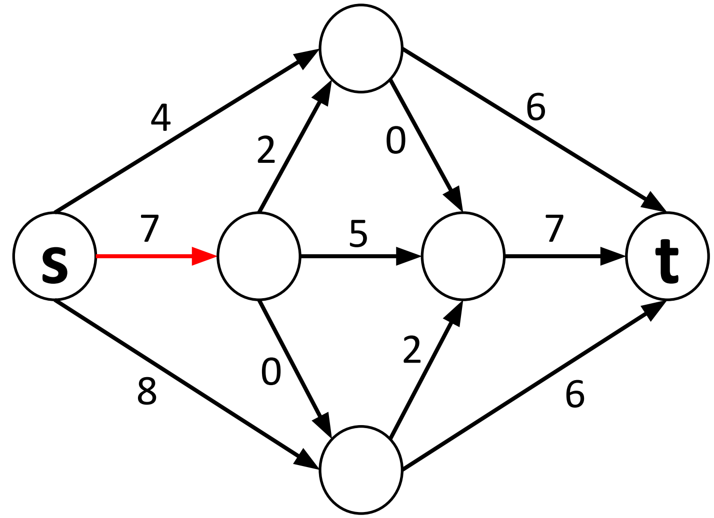


G – G max flow = residual graph

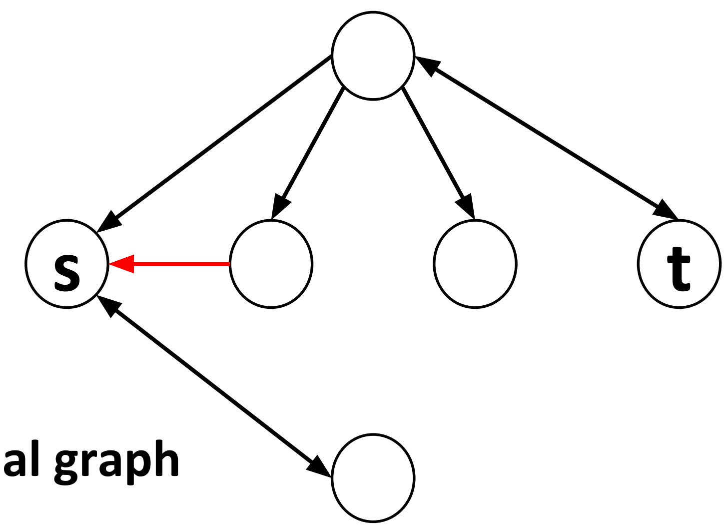
Min cut: example



G

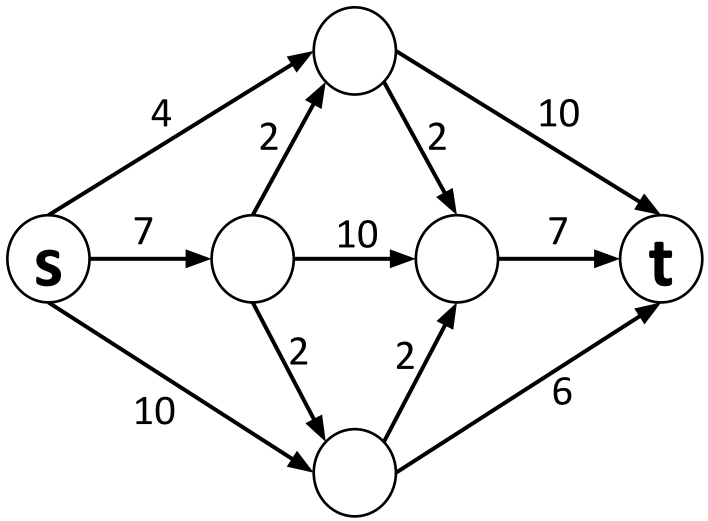


G max flow

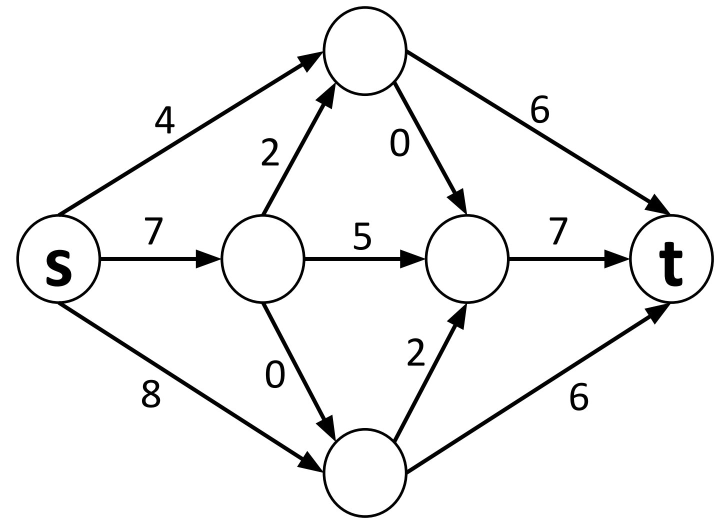


G – G max flow = residual graph

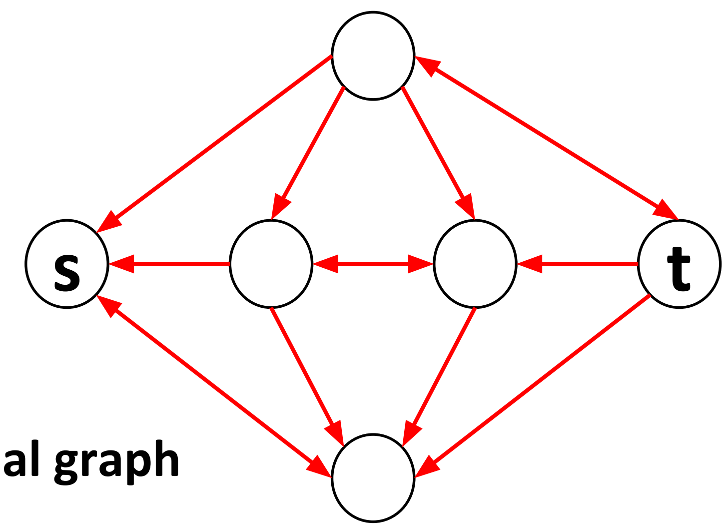
Min cut: example



G

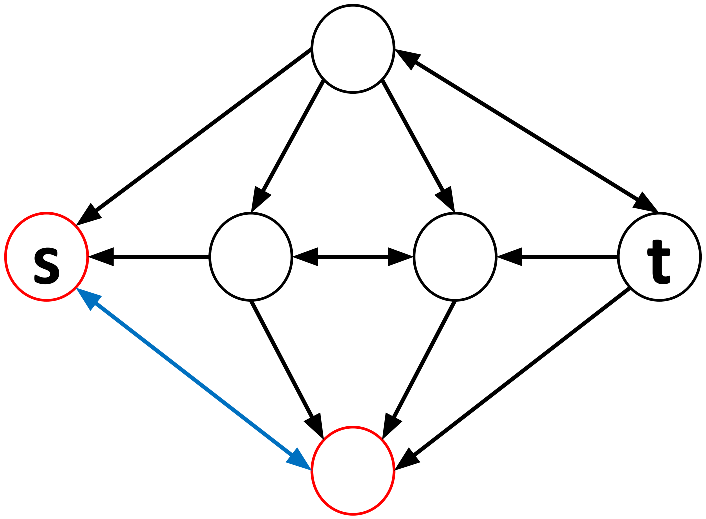


G max flow

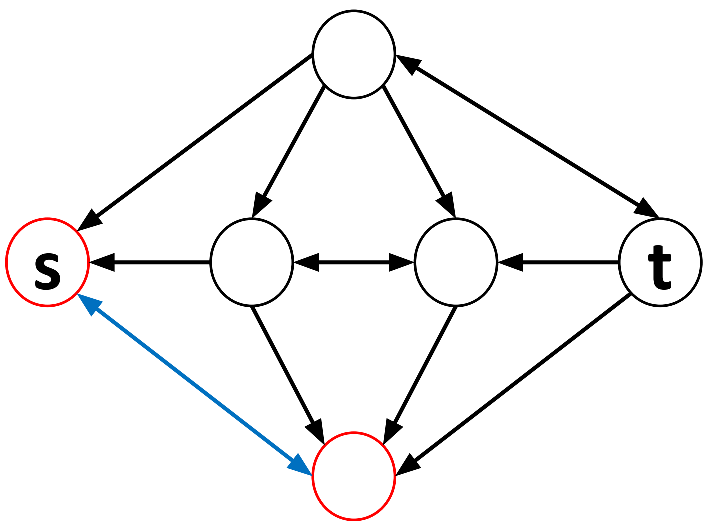


G – G max flow = residual graph

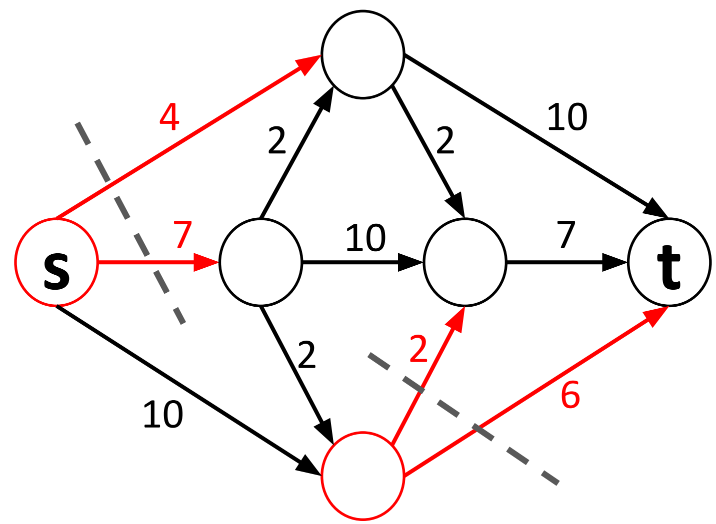
Nodes reachable from s (A)



Cut edges come from V - A



G – G max flow = residual graph



G

Cost of min cut = 4 + 7 + 2 + 6 = 19 = max flow value