CS583 Lecture 10

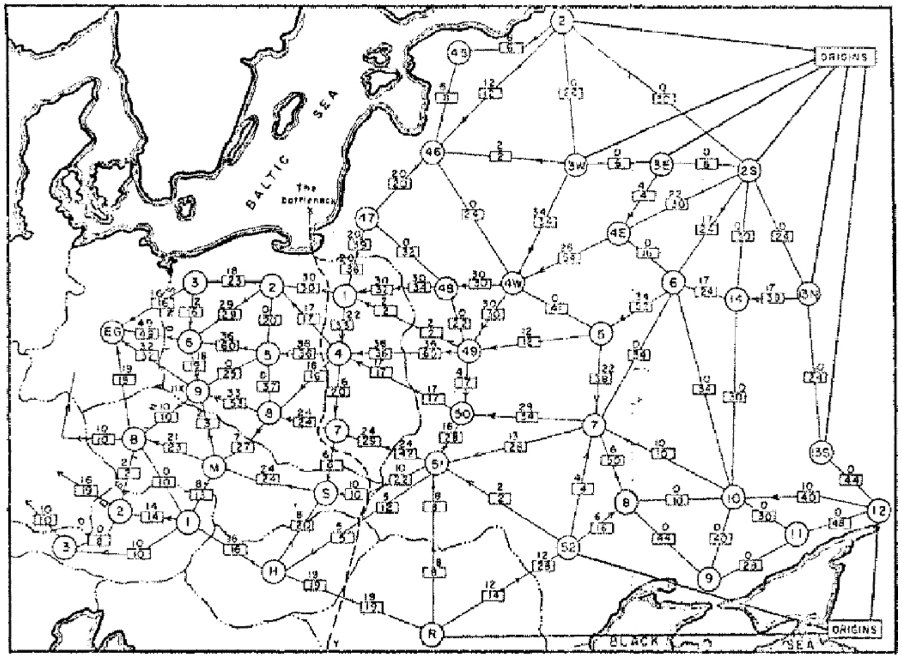
Max Flow & Min Cut

Jyh-Ming Lien

some materials here are based on Prof. Kevin Wayne, Prof. Shehu, and Prof. Wang's past lecture notes

Problem Description

- Soviet Rail Network, 1955 [A. Schrijver 02]



- What is the maximum goods can be sent from city A to city B?

Max flow vs. Min Cut

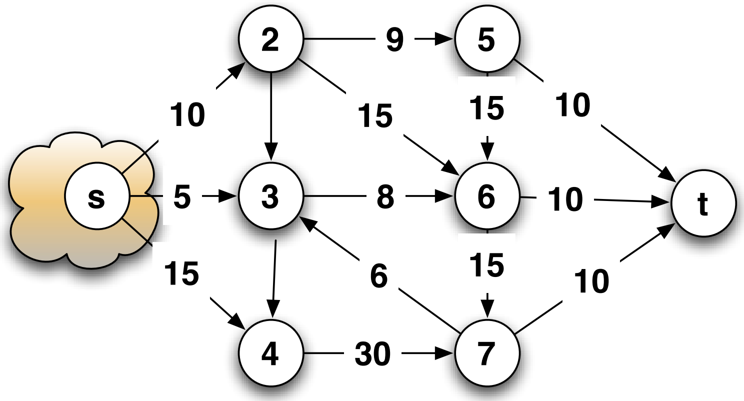
- Max flow and Min cut
  - two very rich algorithmic problems
  - cornerstone problems in combinatorial optimization
  - duality (as in linear programming)

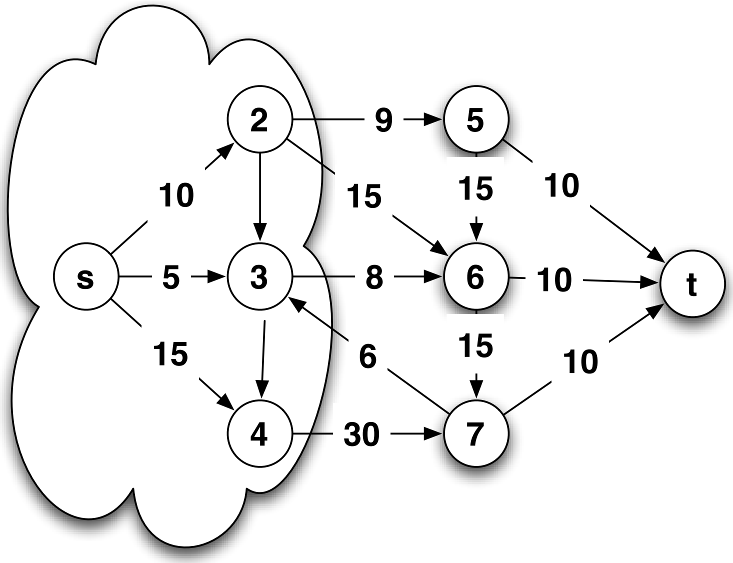
Min Cut Problem

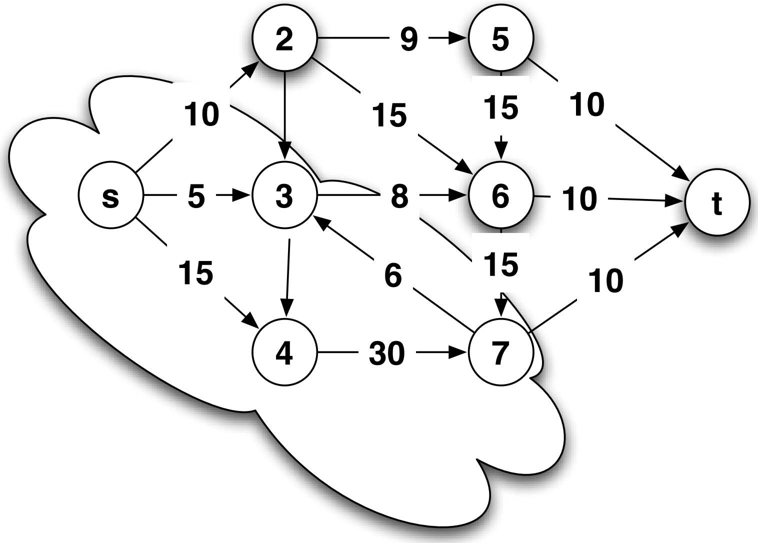
- Input: G, s (source), t (target), c (capacity)
- **Cut**: a set of edges whose removal separate s and t into two connected components
- **Cut capacity**: sum of weights of edges **leaving** s
- **Min cut problem**: Given G, find a cut with minimum cut capacity

Min Cut Problem

- Example:





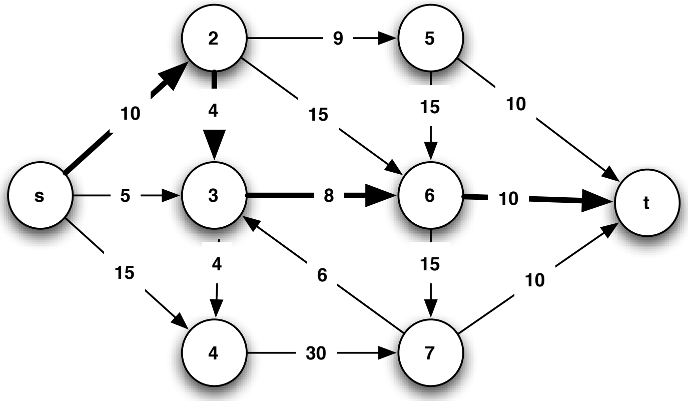


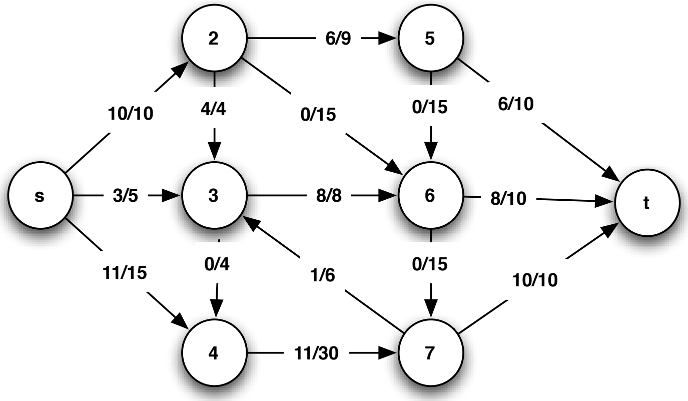
Max Flow

- Input: G, s (source), t (target), c (capacity)
- **Flow**
  - conservation: inflow=outflow for each vertex (except s and t)
  - flow cannot exceed edge capacity
- **Max Flow problem**: Given G, find max flow sent from s to t

Max Flow

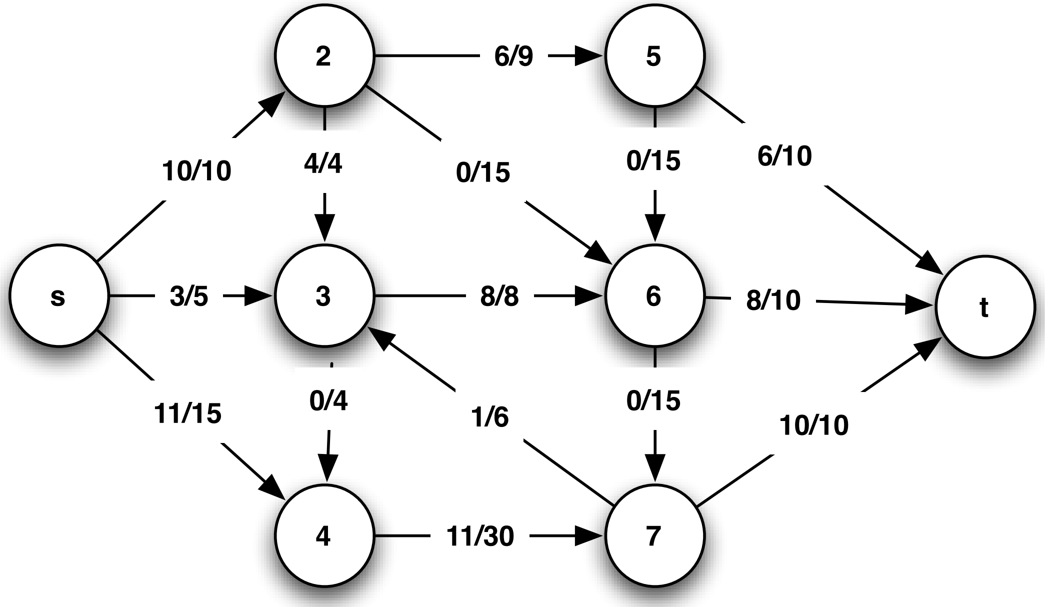
- Examples





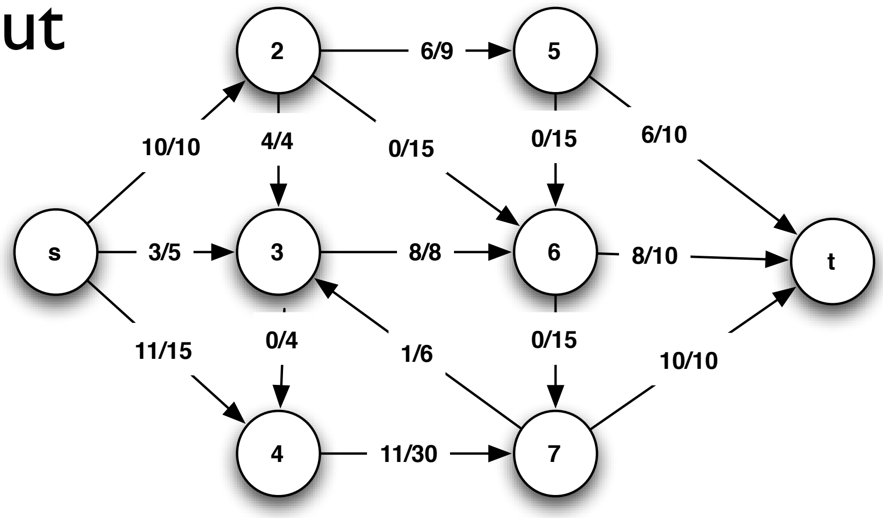
Flow and Cut

- Property 1: The flow across a cut is equal to the amount reaching t (target)



Flow and Cut

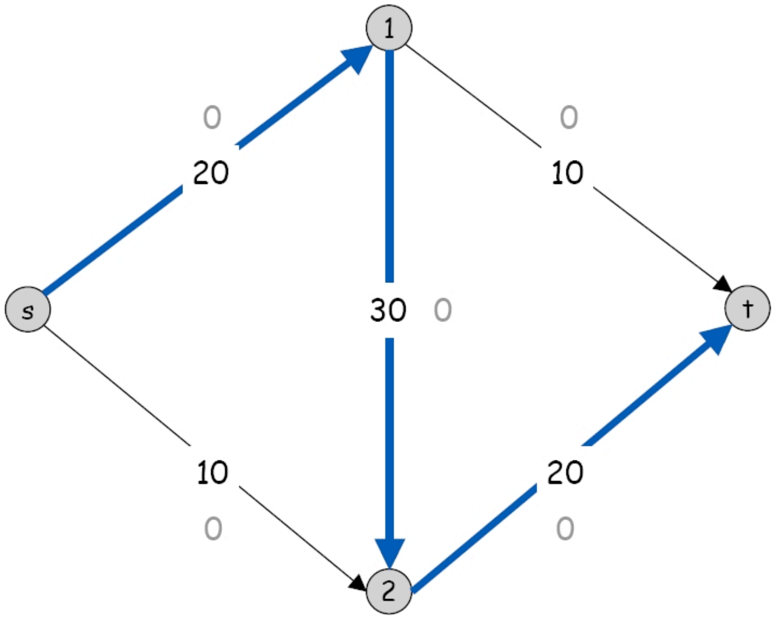
- Property 2: The flow from s to t is at most the capacity of a cut



- Property 3: If the capacity of a cut equals a flow. Then the flow is a max flow and the cut is a min cut.

First Attempt

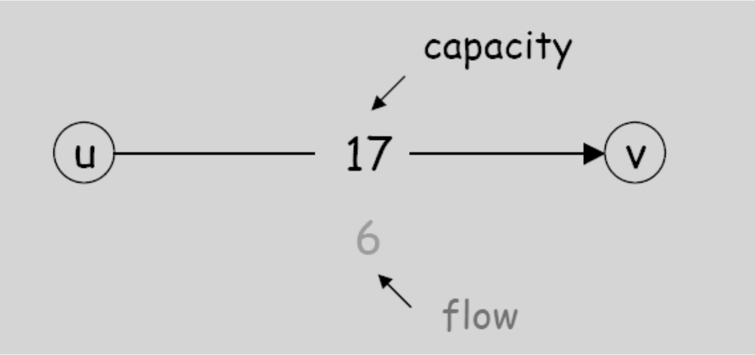
- Greedy algorithm
  - find a path and send a flow
  - decrease the capacity along the path
  - repeat until no flow can be sent

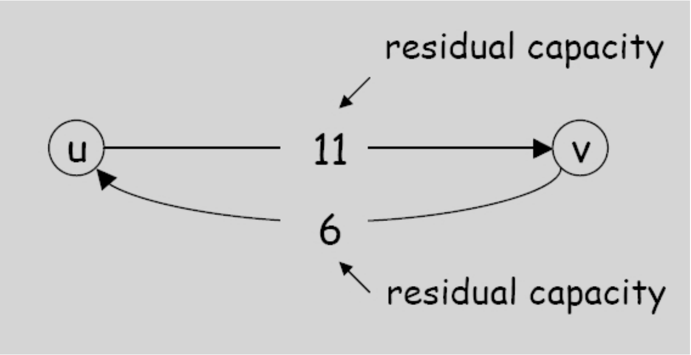


Flow value = 0

First Attempt

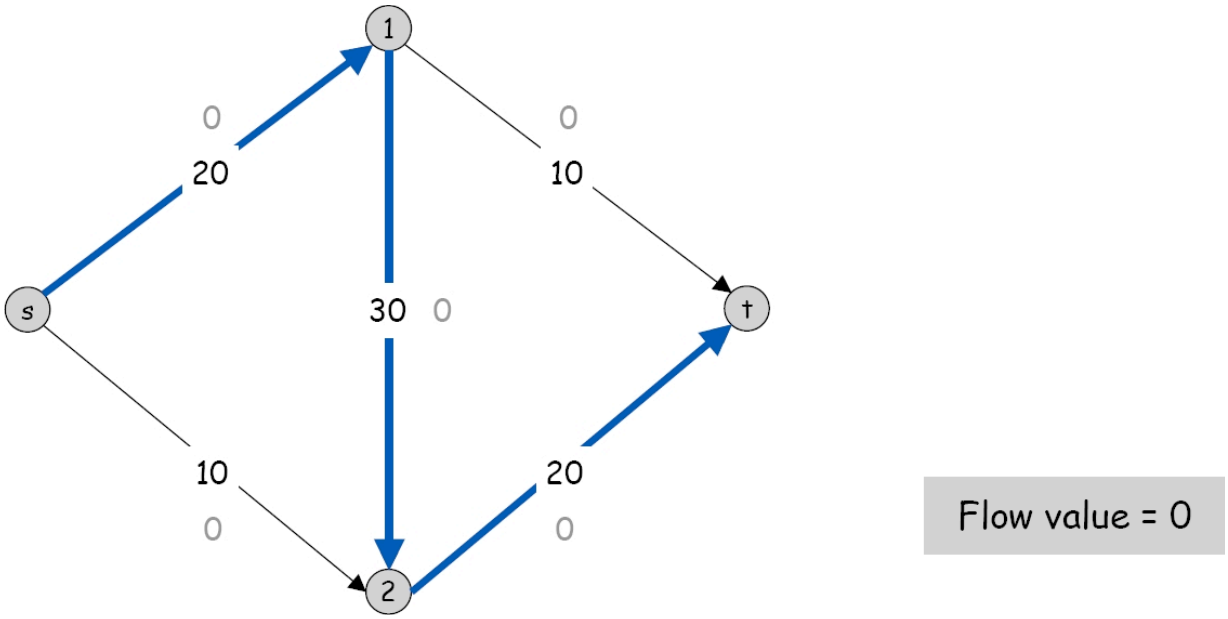
- What is wrong with the greedy approach?
- Can we do something to fix the greedy algorithm?
- Idea: Residual graph





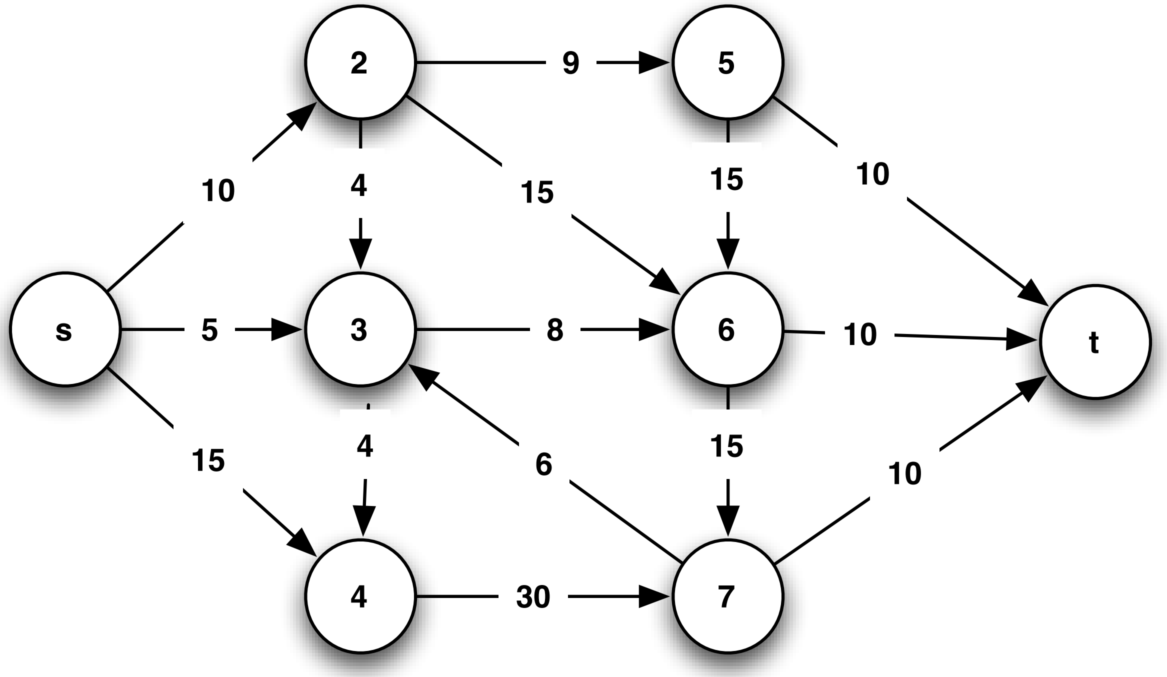
Ford-Fulkerson

- Ford-Fulkerson's algorithm
  - find a path and send a flow
  - **augmenting flow along the path**
  - repeat until no flow can be sent



Ford-Fulkerson

- Example



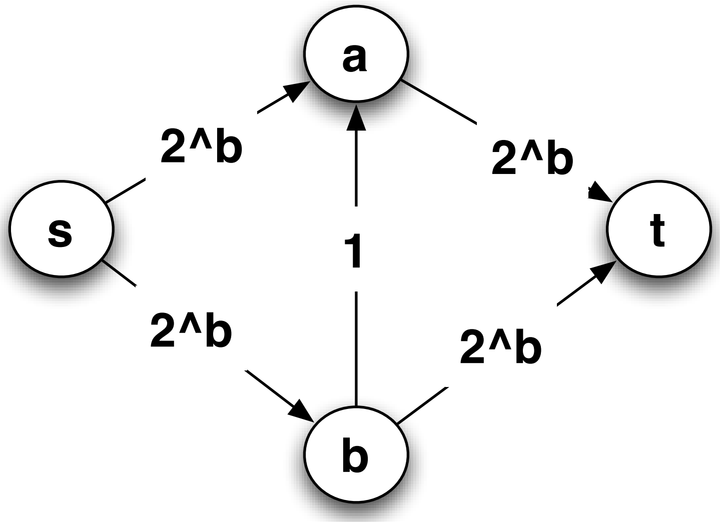
Ford-Fulkerson

- Prove the correctness

- time complexity:

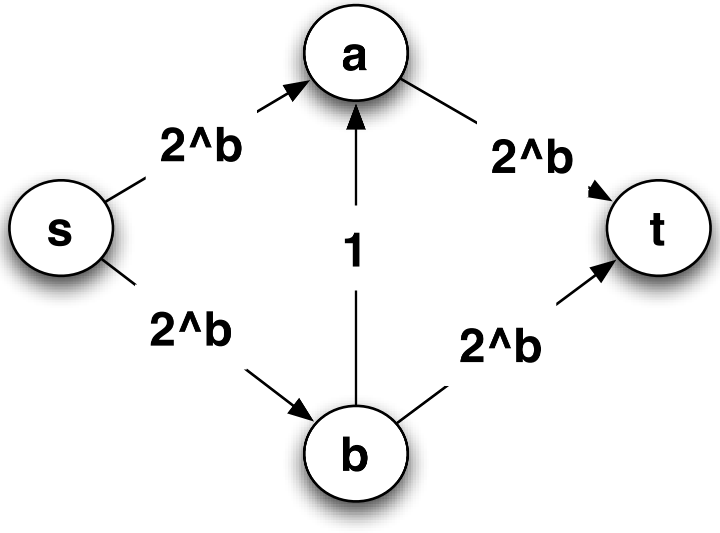
Ford-Fulkerson

- issue: How to select a good augmenting path?



Edmonds-Karp

- Ideas of choosing good augmenting paths
  - shortest path
  - fattest path

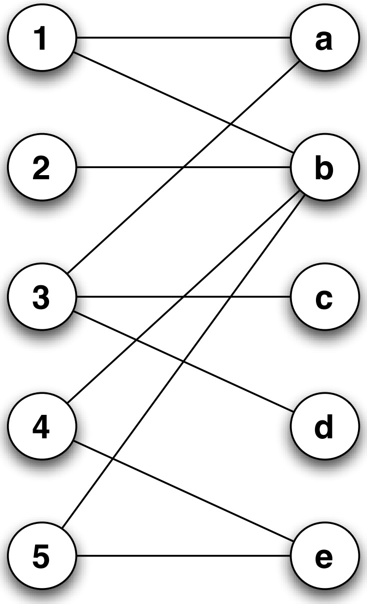


Edmonds-Karp

- Time complexity

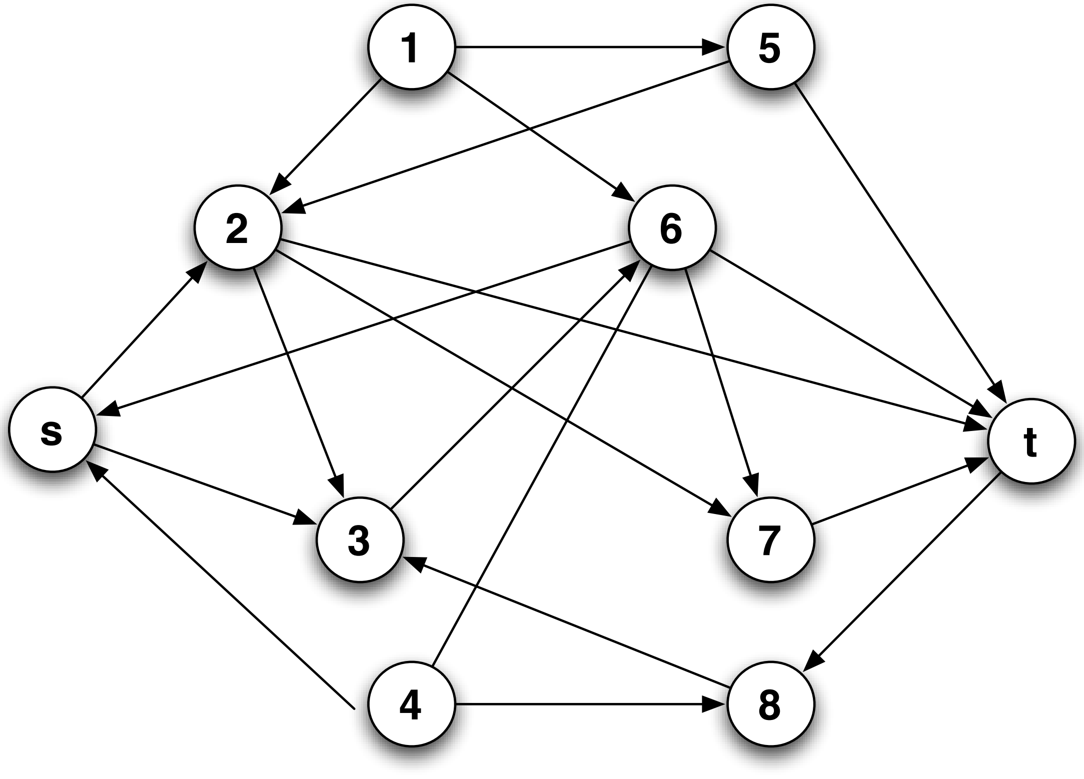
Applications

- Bipartite matching
  - input
  - output



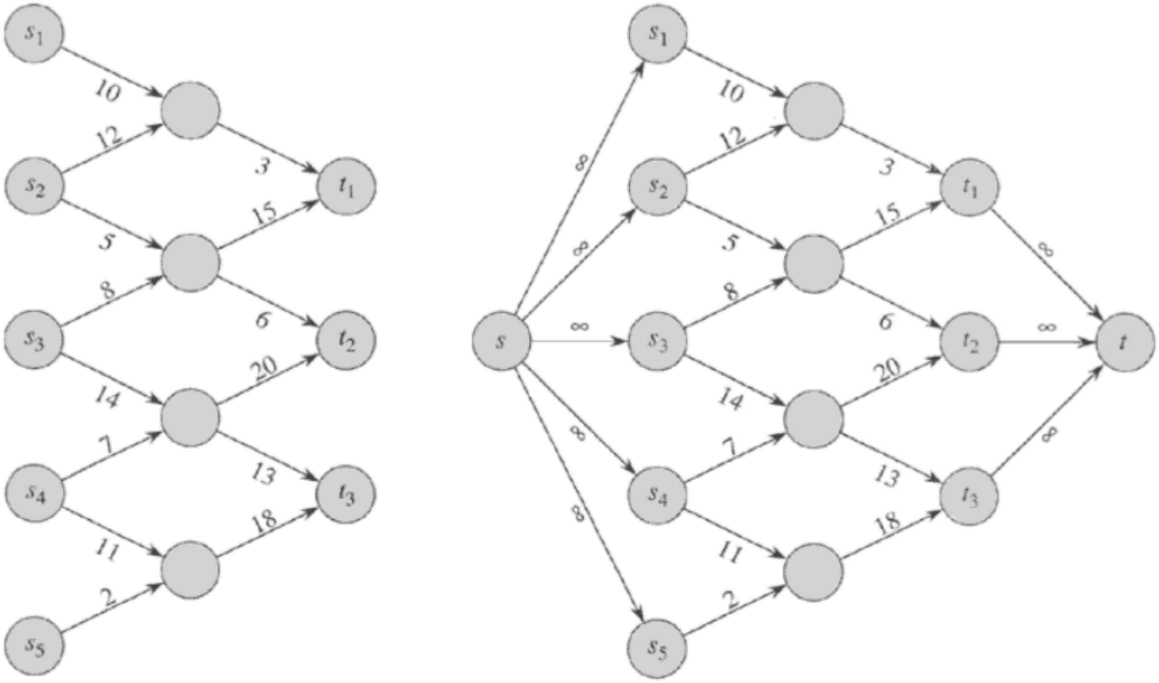
Application

- Edge disjoint paths in a graph
  - input:
  - output



Applications

- multi-sources multi-targets max flow problem



(a)

(b)