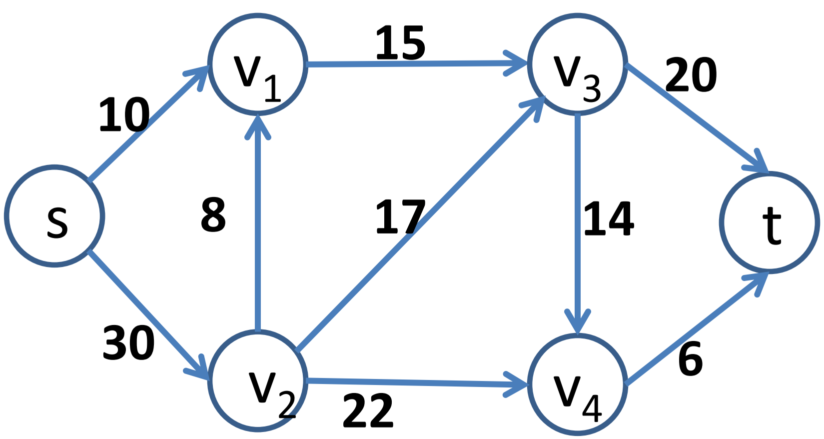
Flow Networks and Bipartite Matching

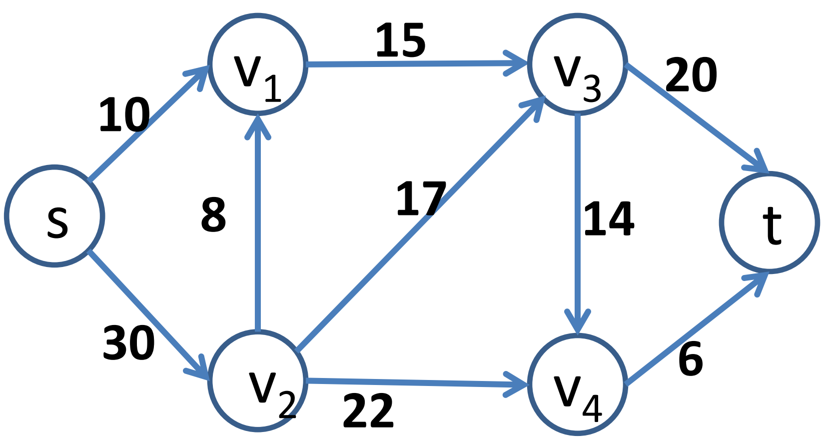
Alexandra Stefan

Flow Network

- A *flow network* is a **directed** graph G = (V,E) in which each edge, (u,v) has a **non-negative capacity, c(u,v) ≥ 0**, and for any pair of vertices (u,v) it has only one edge (it does not have edges in both directions: both (u,v) and (v,u)).
  - 2 special vertices: *source, s,* and *sink*, *t*.
- Applications: Shipping network, Internet network
- A flow in G is a function f:VxV -> R, s.t.:
  - Capacity constraint: for any two vertices u,v, 0 ≤ f(u,v) ≤ c(u,v)
  - Flow conservation: for each $u \in V - \{s,t\}$: $\sum_{v \in V} f(v,u) = \sum_{v \in V} f(u,v)$
- Goal: find a maximum flow through G.



Give maximum flow example:



Ford-Fulkerson Method

Ford-Fulkerson-Method(*G,s,t*)

1. Initialize flow *f* to 0
2. While there exists an augmenting path , *p*, in the residual network G_f, augment flow *f* along *p*
3. Return *f*

- Residual graph G_f:

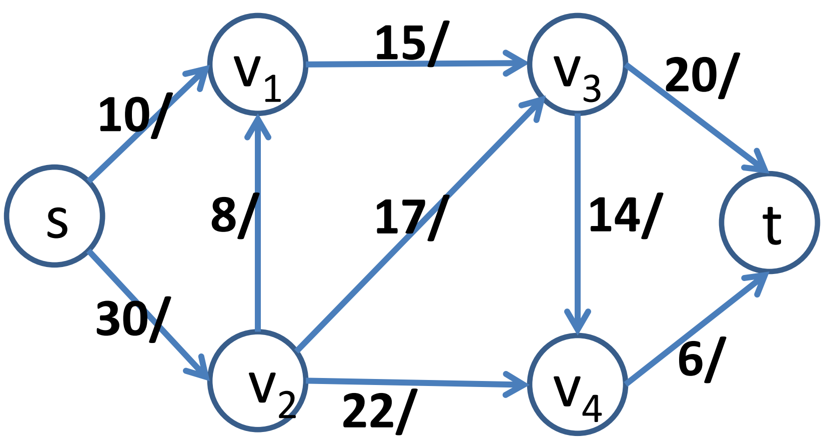
$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & \text{if } (u,v) \in E \ , \\ f(v,u) & \text{if } (v,u) \in E \ , \\ 0 & \text{otherwise} \ . \end{cases}$$

- Augmenting path: a path in G_f from *s* to *t*.

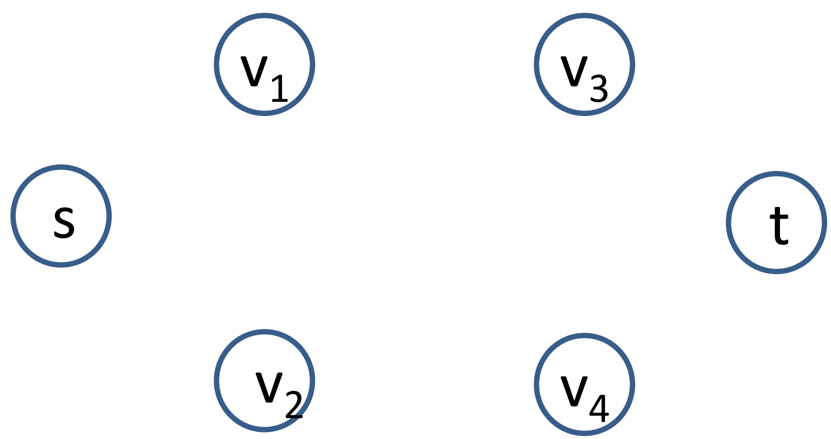
Ford-Fulkerson Method

Work sheet

Graph, G, and current flow, *f*.



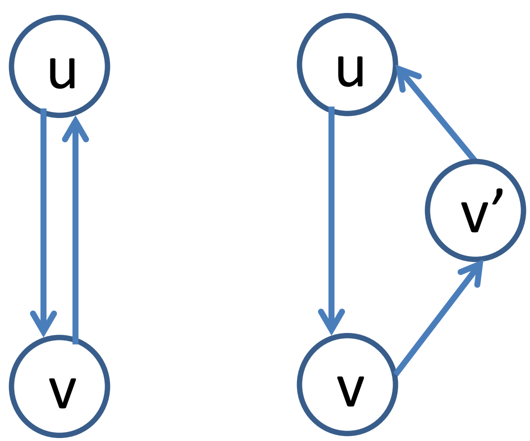
Residual graph, G_f



- Additional properties for a flow network:
  - No self loops.
  - Every vertex, v, is on a path from s to t => connected and $|E| \geq |V|-1$
- Antiparallel edges
  - Edges in both directions: (u,v) and (v,u)

Variations

- Multiple source and multiple sink nodes:
  – Add one extra source and one extra sink
- Antiparallel edges exist:
  – If both (u, v) and (v, u):
    - Add vertex v’,
    - Replace edge (v,u) with edges (v, v’) and (v’, u).



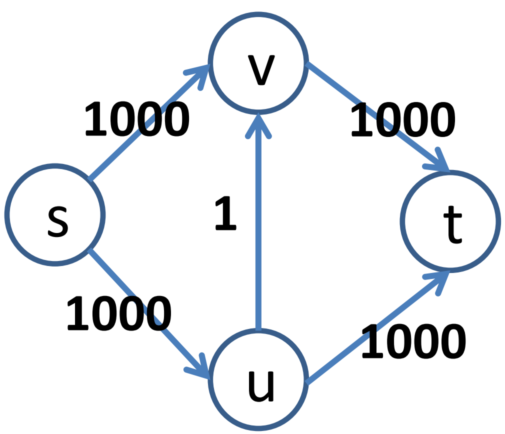
Max-Flow Min-Cut Theorem

- If f is a flow in a flow network G = (V,E) with source s and sink t, then the following conditions are equivalent:
  1. f is a maximum flow in G.
  2. The residual network G_f contains no augmenting paths.
  3. $|f|$ = c(S,T) for some cut (S,T) of G.
     1. c(S,T) is the sum of flows on edges from S to T minus the sum of flows on edges from T to S.

Time Complexity Analysis

- If the flow has real values, the algorithm may never terminate.

- Worst case: $O(E\ |f^*|)$ (f* - maximum flow)
  - In G_f, pick paths that use the small-capacity edges: (u,v) and, when available (v,u).



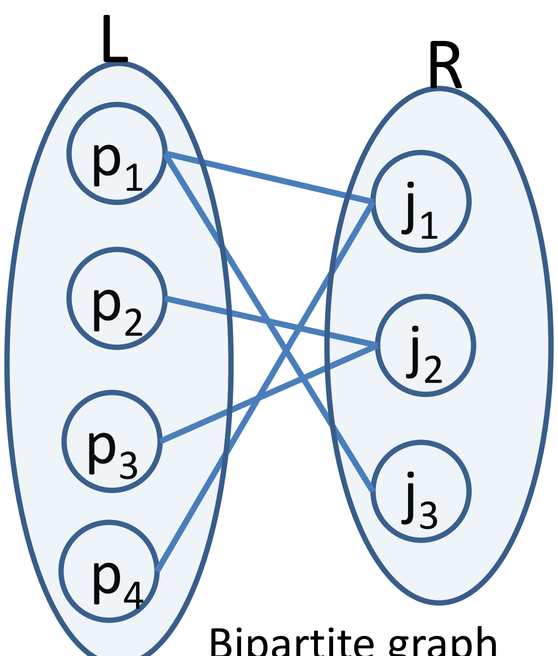
Edmonds-Karp Algorithm

- In the residual graph, pick as augmenting path the shortest path from s to t (given by breadth-first search when all edges have weight 1).

- Time complexity of Edmonds-Karp algorithm: $O(VE^2)$
  - Intuition:
    - Finding the augmenting path takes O(E) (due to BFS)
    - Each edge can become critical at most O(V) times.
    - There are O(E) edges in the residual graph.

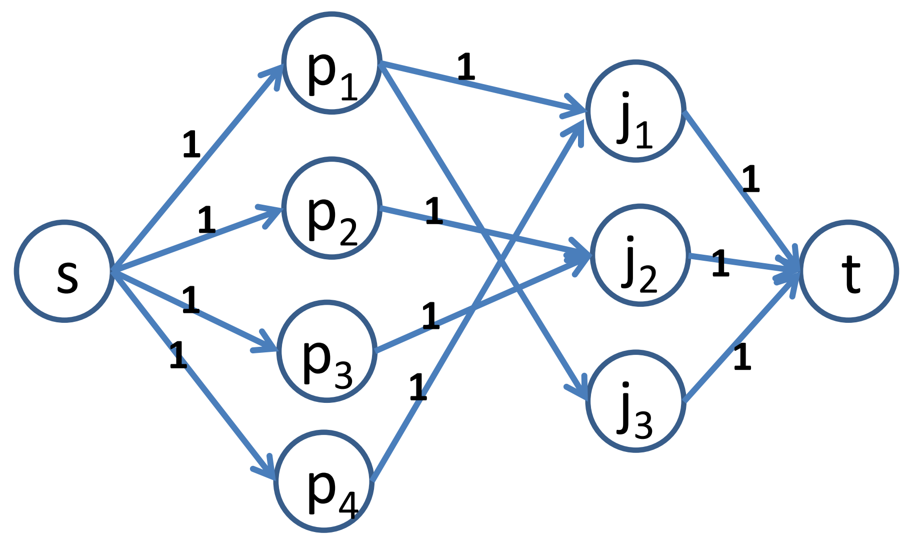
Bipartite Matching

- *Bipartite undirected* graph, G = (V,E):
  - V = L ∪ R
  - All edges are between L and R.
- Model dependencies:
  - Employees and Jobs
  - Resources and processes
- Goal: maximize pairing vertices
  - E.g. assign employees s.t. maximum number of jobs is done.
- Solved with maximum-flow
  - Add extra source and sink nodes
  - Put capacity of 1 on all edges
  - Solve for maximum flow.



Bipartite graph

Corresponding flow network
(with capacity 1 for each edge)



Work sheet