IE 400: Principles of Engineering Management

Set 7: Network Optimization

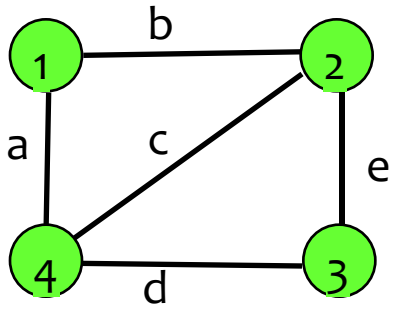
Network Models

- Optimization models that exhibit a special structure
- In some cases, this special structure dramatically reduces computational complexity (running time)
- First widespread application of LP problems of industrial logistics
- Addresses huge number of diverse applications

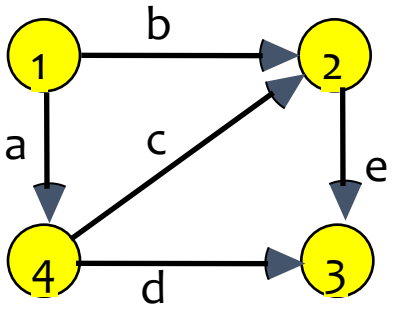
Network Flows and Routing Models

- Phone network: route calls, messages, data
- Power network: transmit power from power plants to consumers
- Highway, street, rail, airline, shipping networks: transport people, vehicles, goods
- Computer networks: transmit info, data, messages
- Pipeline networks: transport crude oil, gasoline
- Satellite networks: worldwide communication system
- Abstract networks: using networks to model relations

Notation and Terminology



An Undirected Graph or
Undirected Network



A Directed Graph or
Directed Network

Network G = (N, A)

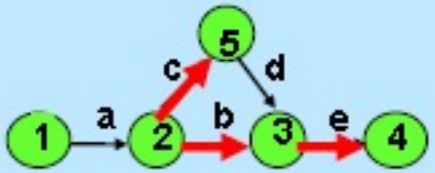
Node set N = {1, 2, 3, 4}

Arc Set A = {(1,2), (1,4), (2,3), (3,4), (2,4)}

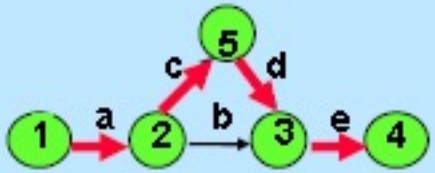
In an undirected graph, (i,j) = (j,i)

Notation and Terminology

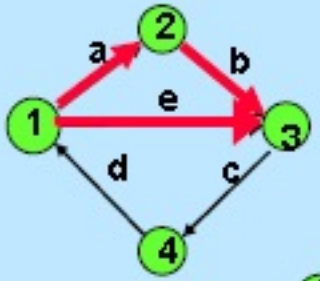
Path: **Example: 5, 2, 3, 4.**
(or 5, c, 2, b, 3, e, 4)
- No node is repeated.
- Directions are ignored.



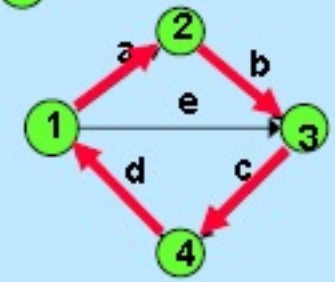
Directed Path **. Example: 1, 2, 5, 3, 4**
(or 1, a, 2, c, 5, d, 3, e, 4)
- No node is repeated.
- Directions are important.



Cycle (or circuit or loop)
1, 2, 3, 1. (or 1, a, 2, b, 3, e)
- A path with 2 or more nodes, except that the first node is the last node.
- Directions are ignored.

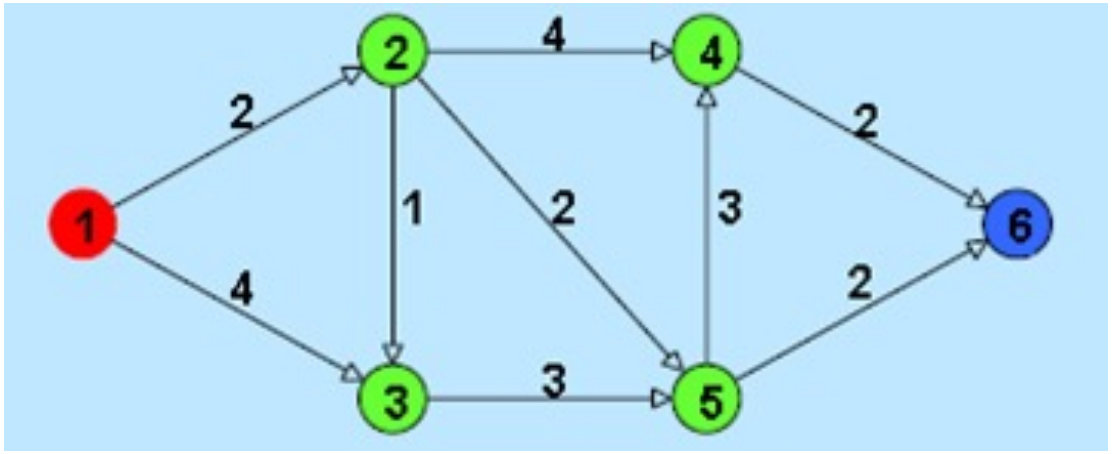


Directed Cycle: (1, 2, 3, 4, 1) or
1, a, 2, b, 3, c, 4, d, 1
- No node is repeated, except that the first node is the last node.
- Directions are important.



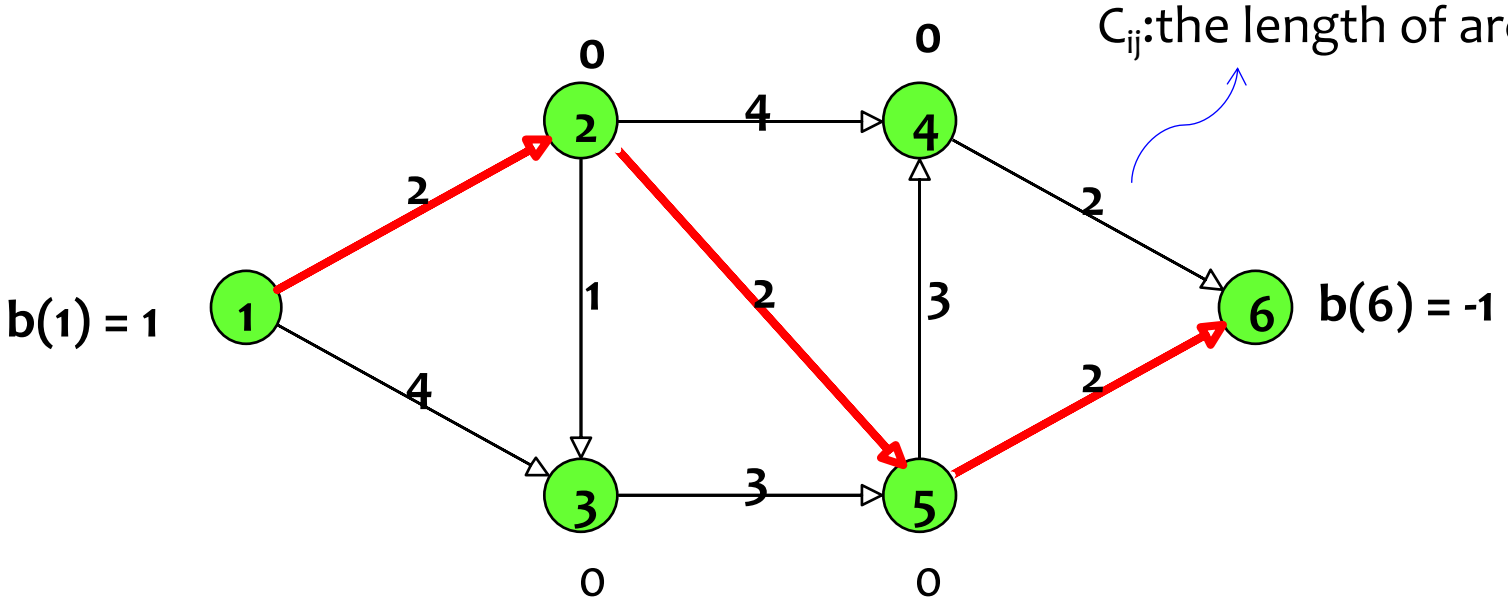
Shortest Paths

- Single Pair Shortest Path Problem
- Single Source Shortest Path Problem
- All-Pairs Shortest Path Problem



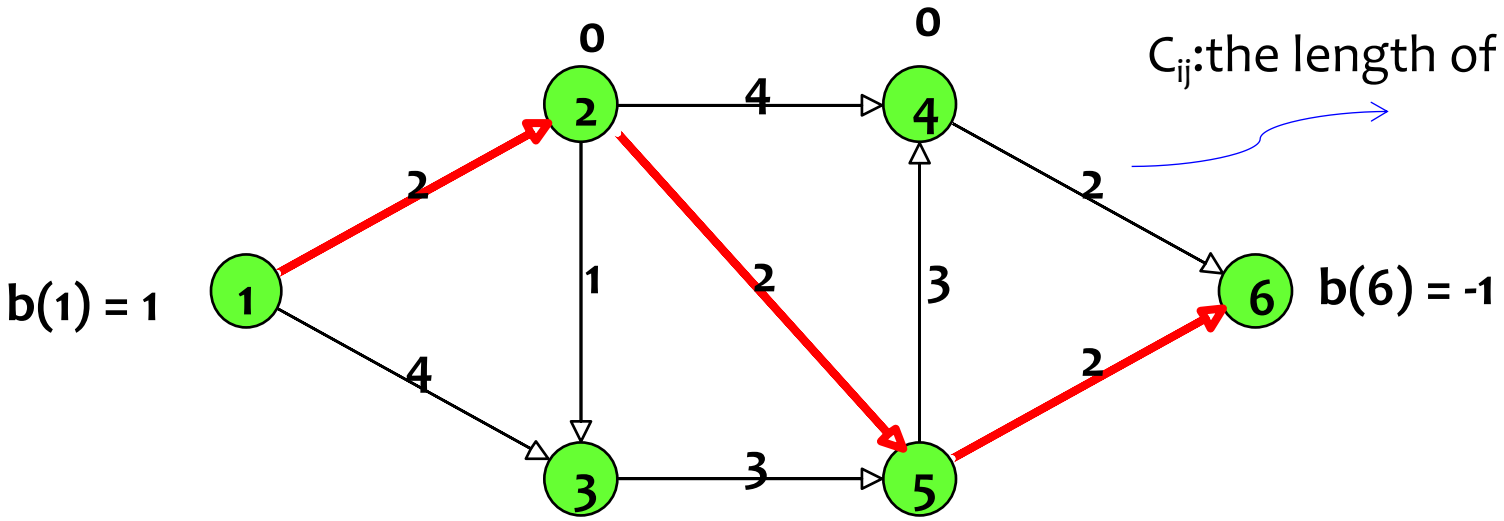
- Length of (directed) path=total cost of arcs on this path

Find the shortest path from source node 1 to sink node 6 (single-pair shortest path problem)



- Think of sending one unit of flow from 1 to 6
- This transformation works so long as there are no negative cost cycles in G.
- What if there are negative cost cycles?

LP Formulation of the Shortest Path Problem



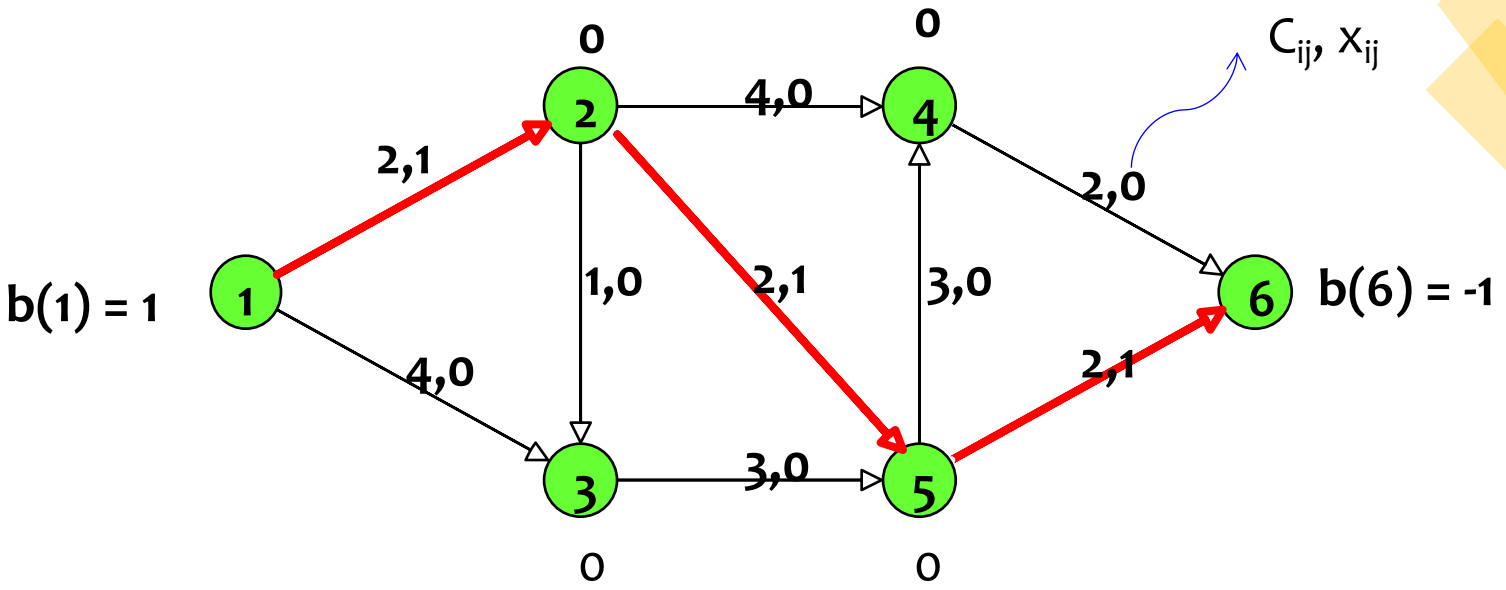
$$Min \sum_{(i,j)\in A} c_{ij} x_{ij}$$

Find the shortest path from node 1 to node 6

$$s.t. \quad \sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ji} = \begin{cases} 1 & i = s \\ 0 & i \neq s,t \\ -1 & i = t \end{cases}$$

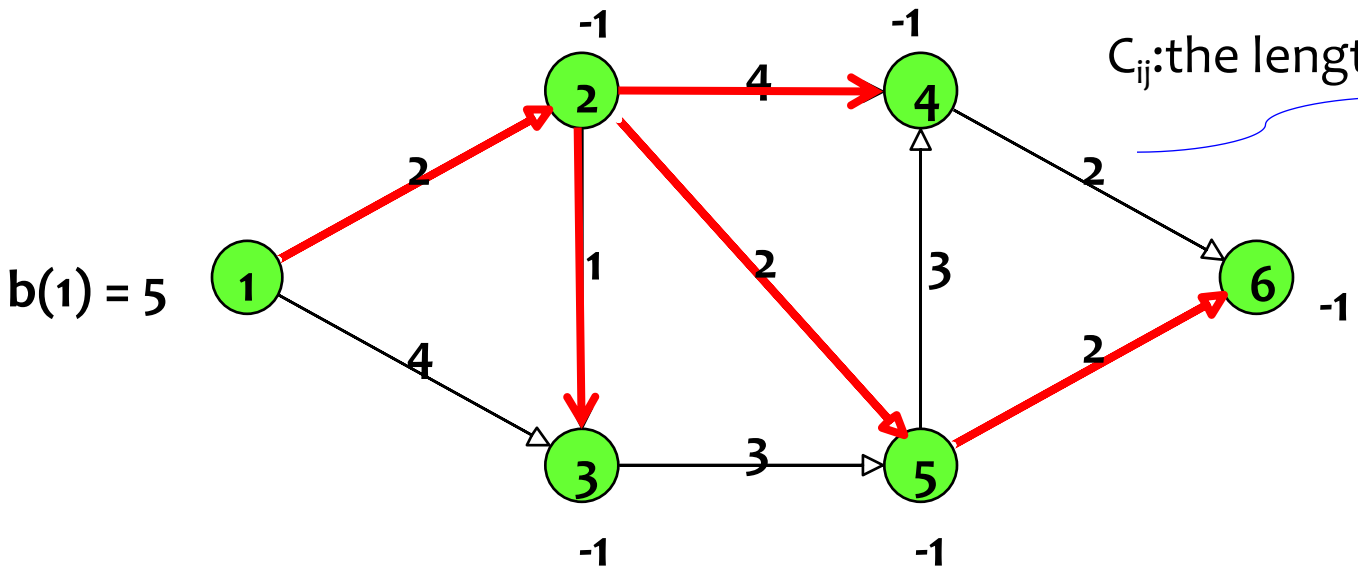
$$x_{ij} \geq 0, \ integer \qquad (i,j) \in A$$

Find the shortest path from source node 1 to sink node 6 (single-pair shortest path problem)



- Optimal flow is to send one unit of flow along 1-2-5-6.
  - $x_{12}=1$, $x_{25}=1$, $x_{56}=1$; $x_{13}=x_{23}=x_{24}=x_{35}=x_{45}=x_{46}=0$
  - Objective function value=2.1+2.1+2.1=6
- C_{ij}: the length of arc (i,j); x_{ij}: the amount of flow on arc (i,j)

Find the shortest path from source node 1 to all other nodes (single-source shortest path problem)



- Each node except the source node is treated as a sink node
- Assume that a source of n-1 units is at node 1 and a demand of 1 unit at all other nodes

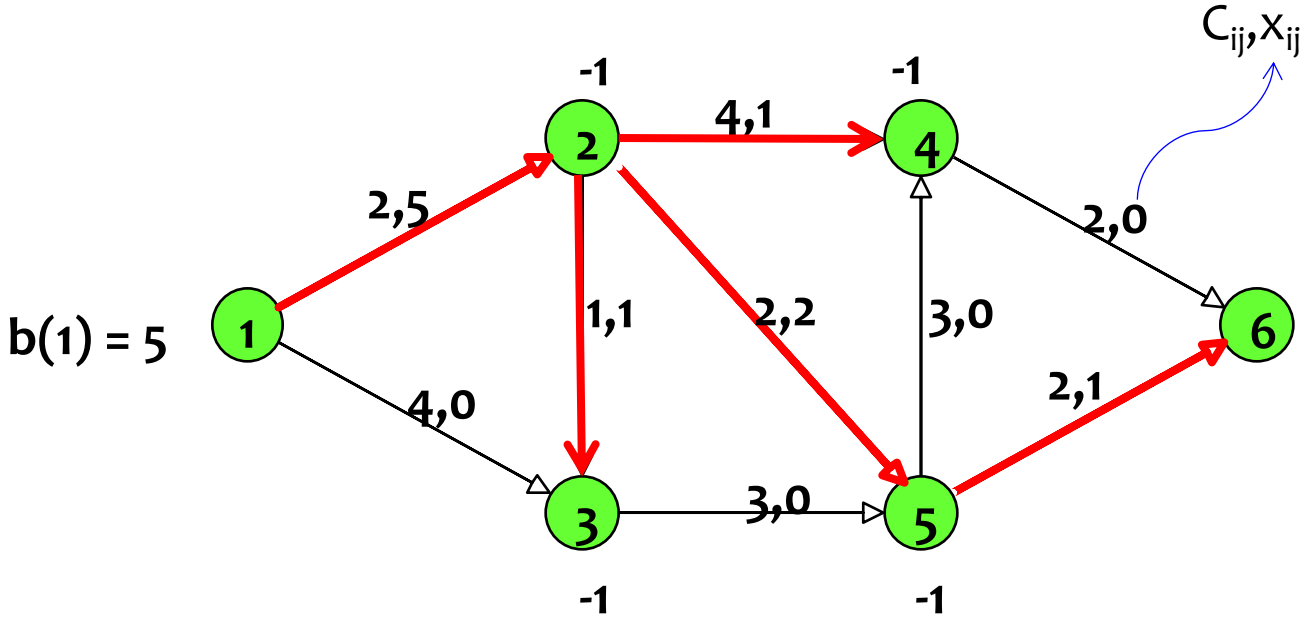
IP Formulation of the single source Shortest Path Problem

$$Min \sum_{(i,j)\in A} c_{ij}x_{ij}$$

$$s.t. \quad \sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ji} = \begin{cases} n-1 & i = s \\ -1 & i \neq s \end{cases}$$

$$x_{ij} \geq 0, integer \qquad (i,j) \in A$$

Find the shortest path from source node 1 to all other nodes (single-source shortest path problem)



Objective function value=2.5+2.2+1.1+4.1+2.1=21

Integrality Property

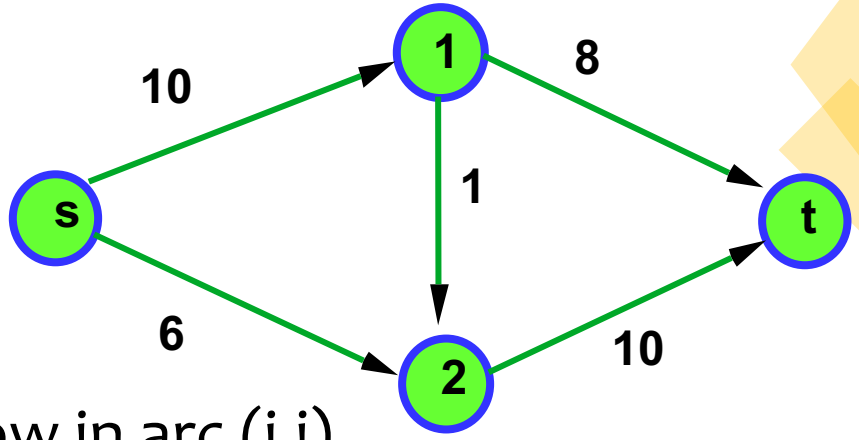
- If we relax the integrality requirement and solve as an LP, we get an integral solution and the arcs with positive x_{ij} values give a shortest path tree rooted at s.
- The constraint matrix has a special property called "total unimodularity". Any extreme point is an integer vector and hence the optimal solution is integral.
- In general, network flow problem have this special "integrality property".

Total Unimodularity

- A matrix A is called *totally unimodular* if each of its subdeterminants equals 0, 1, or -1. (In particular, each entry of A is 0, 1, or -1.)
- If A is totally unimodular and b is integer valued, then the polyhedron $P=\{x|Ax\leq b\}$ is integral.

Maximum Flows

Given:



G = (N,A)

u_{ij} = capacity of flow in arc (i,j)

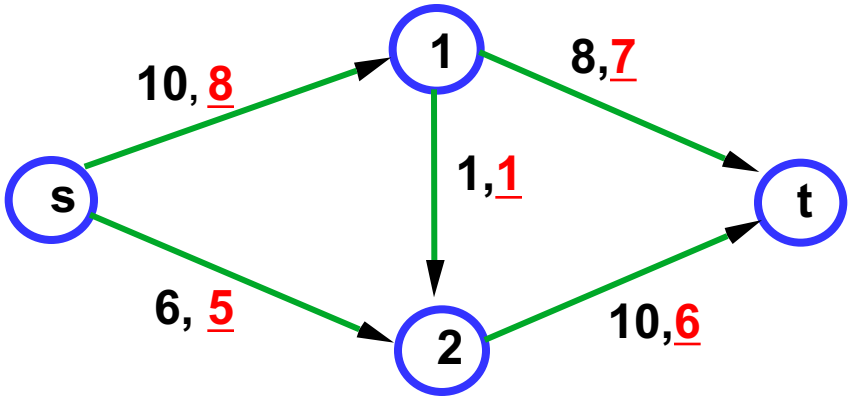
s = source node

t = sink node

- Find the maximum amount of net flow out of node s into node t obeying the capacity limitations
- Intermediate nodes should satisfy flow balance

Maximum Flows

We refer to a flow x as *maximum* if it is feasible and maximizes the net flow out of s. Our objective in the max flow problem is to find a maximum flow.



- A max flow problem.
- Capacities and a non-optimum flow.

IP Model for Maximum Flow Problem

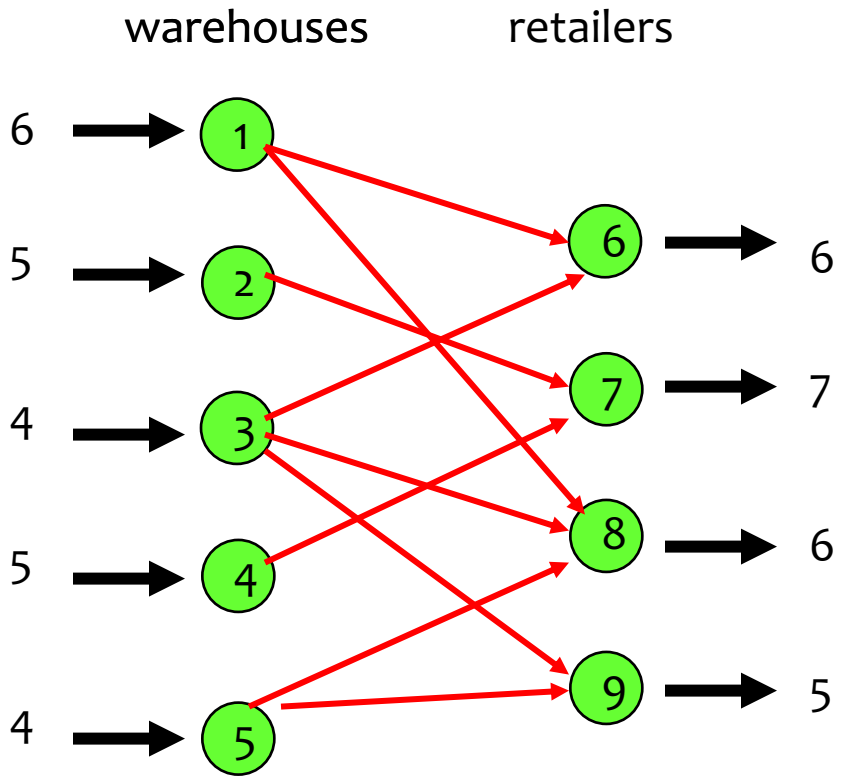
Decision Variables:

x_{ij} = flow on arc (i,j)

v = net flow out of s (into t)

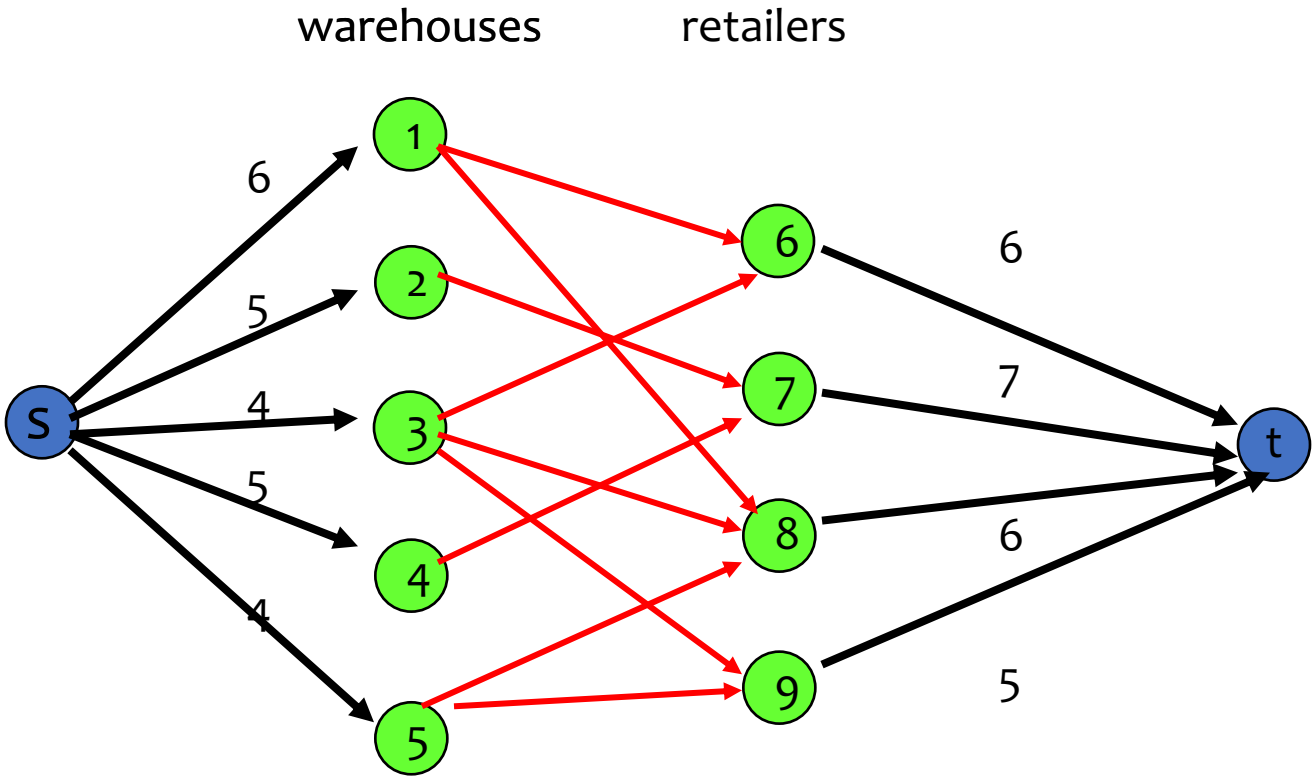
Model:

The Feasibility Problem: Find a feasible flow



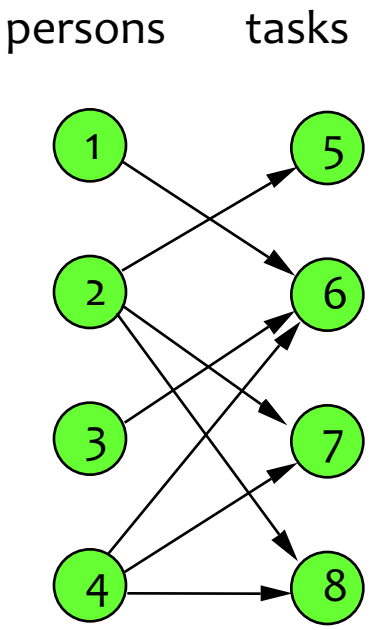
Is there a way of shipping from the warehouses to the retailers to satisfy demand?

Transformation to a max-flow problem



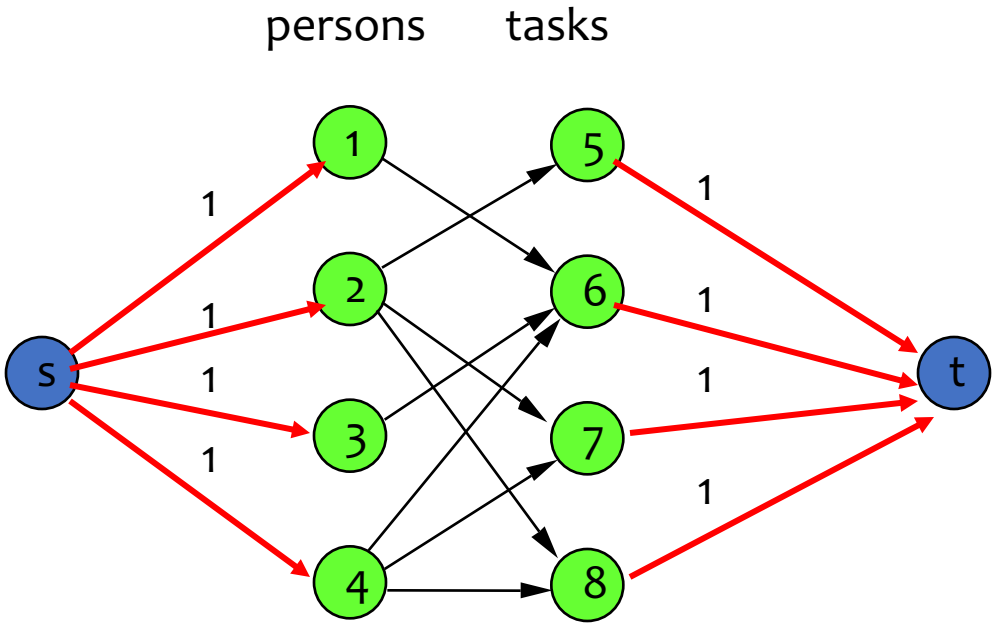
There is a 1-1 correspondence with flows from s to t with 24 units (why 24?) and feasible flows for the transportation problem.

The Feasibilty Problem: Find a matching



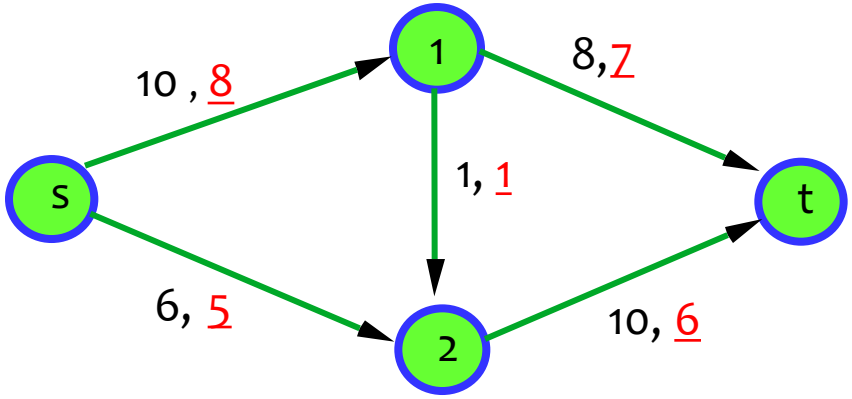
Is there a way of assigning persons to tasks so that each person is assigned a task, and each task has a person assigned to it?

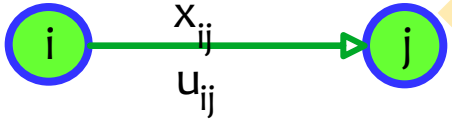
Transformation to a max-flow problem

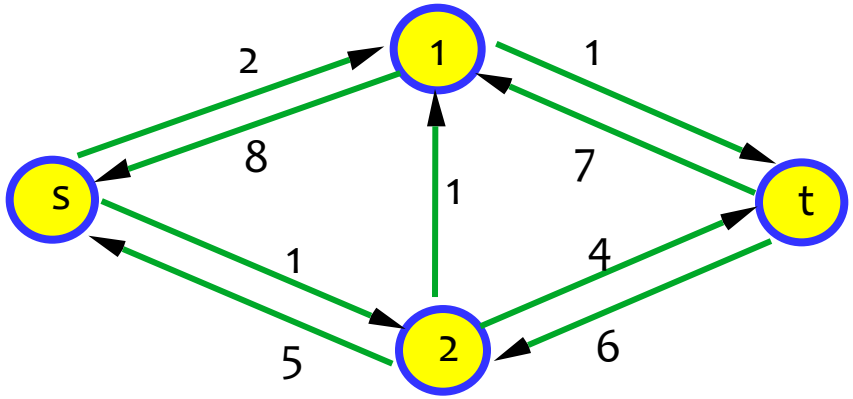


Does the maximum flow from s to t have 4 units?

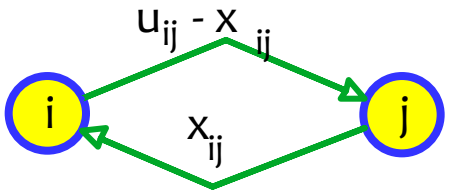
Residual Network







The *Residual Network* G(x)

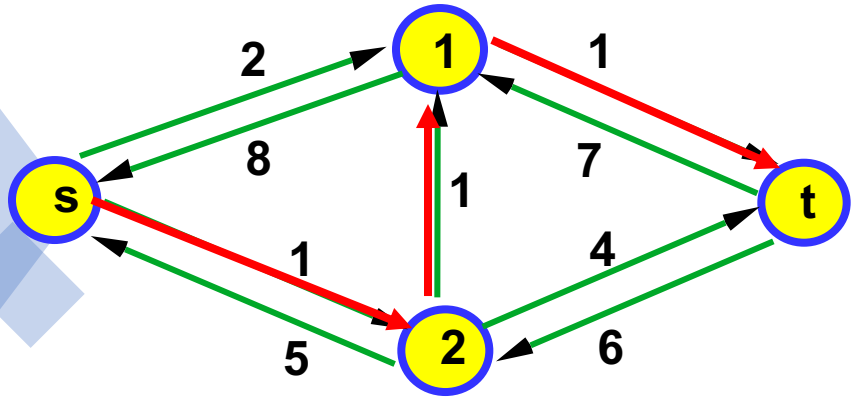


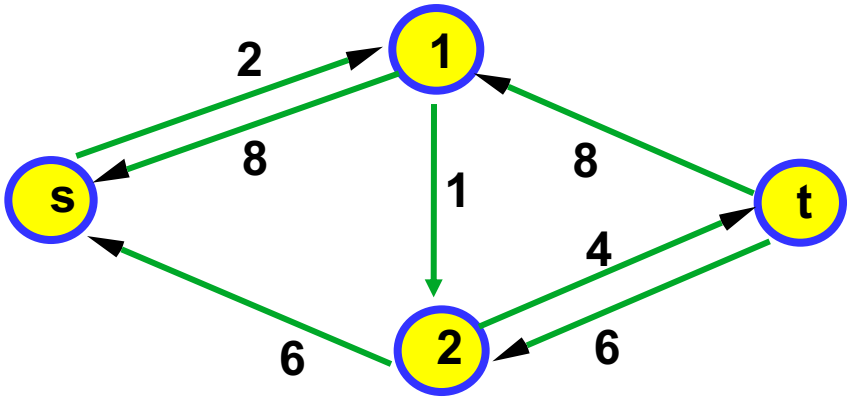
We let r_{ij} denote the *residual capacity* of arc (i,j)

A Useful Idea: Augmenting Paths

- An *augmenting path* is a path from s to t in the residual network.
- The *residual capacity* of the augmenting path P is $\delta(P) = \min\{r_{ij} : (i,j) \in P\}$.
- To *augment along P* is to send $\delta(P)$ units of flow along each arc of the path. We modify x and the residual capacities appropriately.

$r_{ij} := r_{ij} - \delta(P)$ and $r_{ji} := r_{ji} + \delta(P)$ for $(i,j) \in P$.

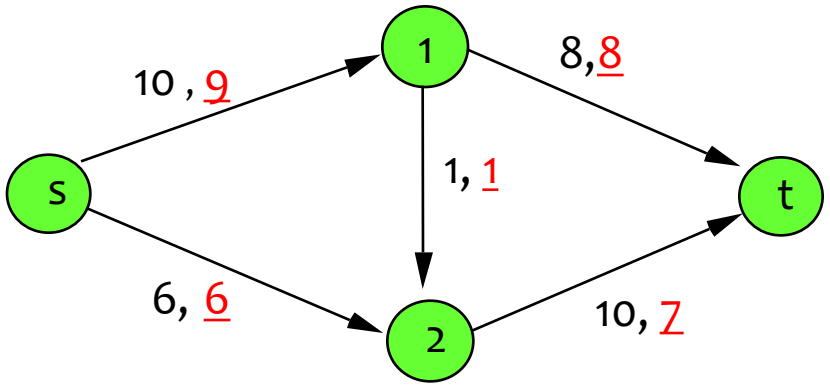




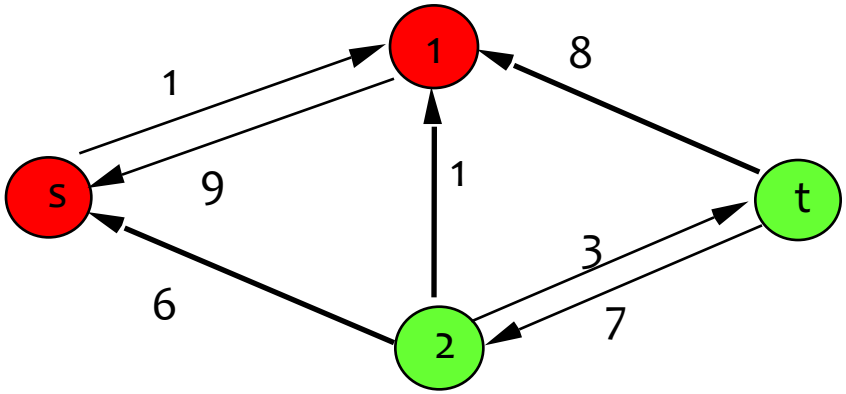
The Ford Fulkerson Maximum Flow Algorithm

```
Begin
    x := 0;
    create the residual network G(x);
    while there is some directed path from s to t in G(x) do
    begin
        let P be a path from s to t in G(x);
        Δ := δ(P);
        send Δ units of flow along P;
        update the r's;
    end
end {the flow x is now maximum}.
```

How do we know when a flow is optimal?



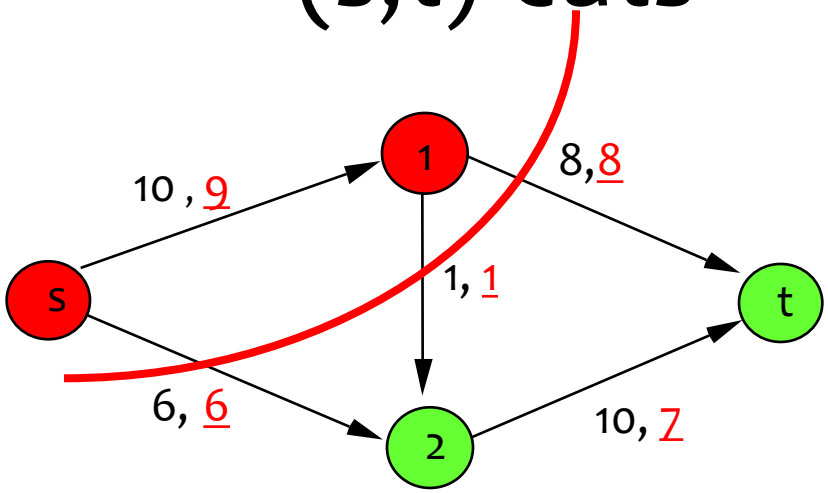
There is no augmenting path in the residual network.



reachable from s in G(x)

not reachable from s in G(x)

(s,t)-cuts



An *(s,t)-cut* in a network G = (N,A) is a partition of N into two disjoint subsets S and T such that s ∈ S and t ∈ T, e.g., S = { s, 1 } and T = { 2, t }.

The *capacity* of a cut (S,T) is

$$CAP(S,T) = \sum_{i \in S} \sum_{j \in T} u_{ij}$$

(s-t)-cuts

- *Observation: If x is any feasible flow and if (S,T) is an (s,t)-cut, then the flow v(x) from source to sink in x is at most CAP(S,T).*

- *In other words, capacity of any cut is an upper bound on the the value of any feasible flow*

Max-flow Min-cut Theorem

Theorem. (Optimality conditions for max flows):

A flow x is maximum if and only if there is no augmenting path in G(x).

Proof.

Suppose that there is an augmenting path in G(x). Then x is not maximum.

Max-flow Min-cut Theorem (cont'd)

Suppose there is no augmenting path in G(x).

Claim: Let S be the set of nodes reachable from s in G(x). Let T = N\S. Then there is no arc in G(x) from S to T.

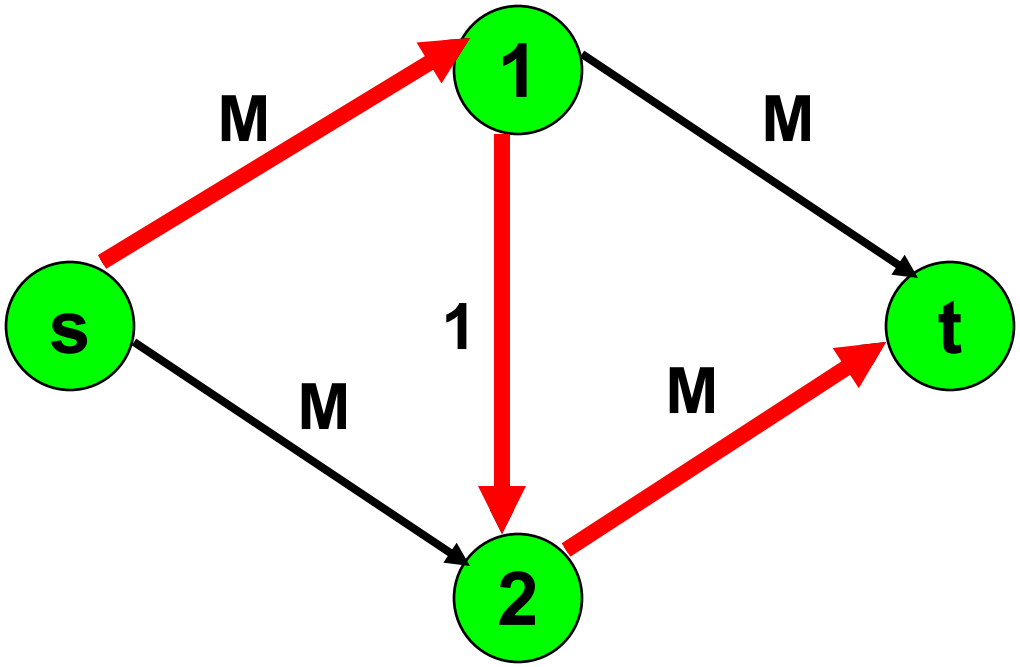
Thus,

$i \in S$ and $j \in T \Rightarrow x_{ij} = u_{ij}$

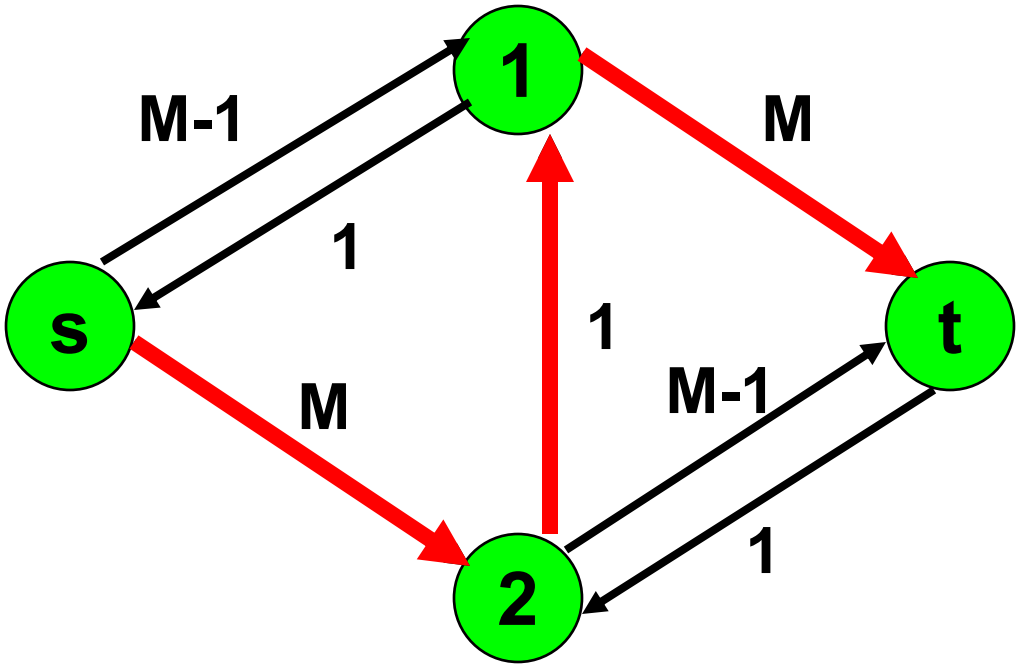
$i \in T$ and $j \in S \Rightarrow x_{ij} = 0$.

Thus flow across this cut = cap(S,T) and thus is optimal!

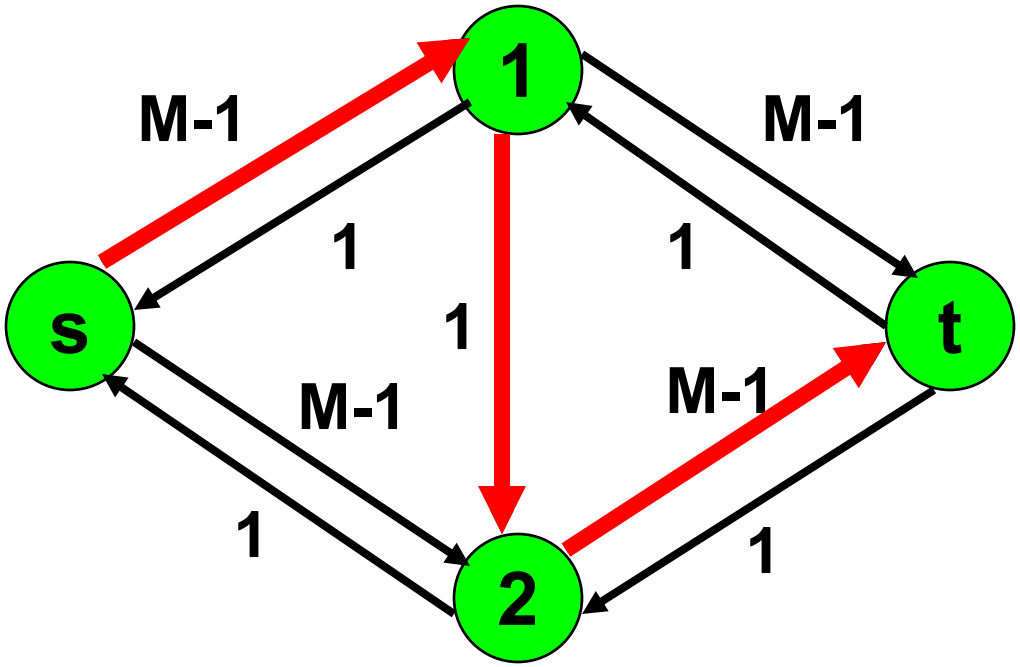
A simple and very bad example



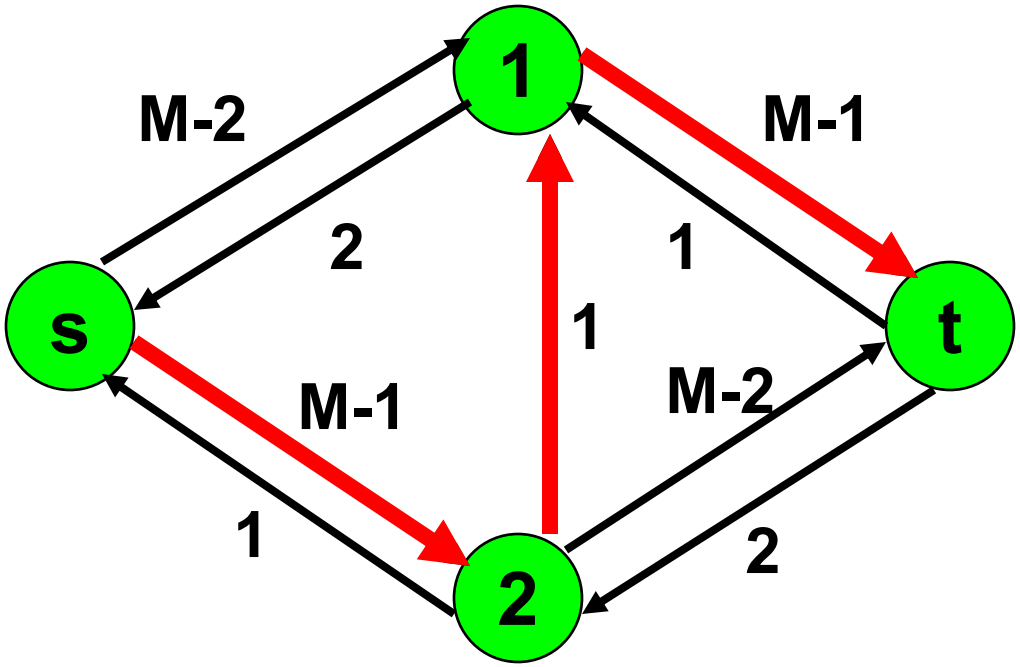
After 1 augmentation



After two augmentations



After 3 augmentations



And so on

After 2M augmentations