
15.082J and 6.855J
March 4, 2003

Introduction to Maximum Flows

The Max Flow Problem

G = (N,A)

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

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

s = source node

t = sink node

Maximize v

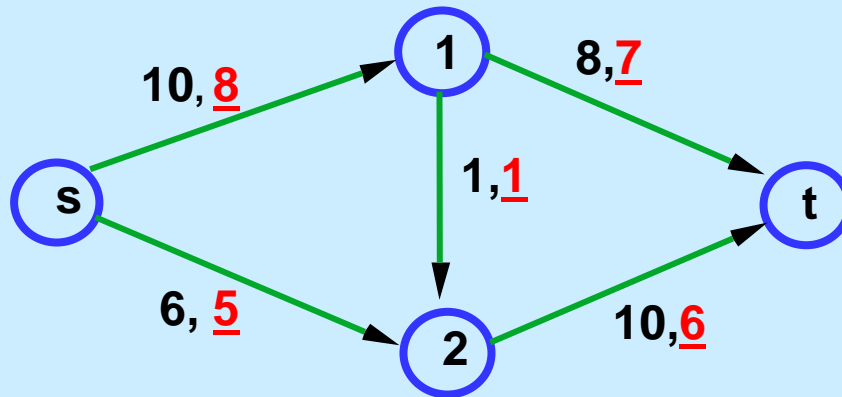
Subject to $\sum_j x_{ij} - \sum_k x_{ki} = 0$ for each $i \neq s,t$

$$\sum_j x_{sj} = v$$

$$0 \leq x_{ij} \leq u_{ij} \text{ for all } (i,j) \in A.$$

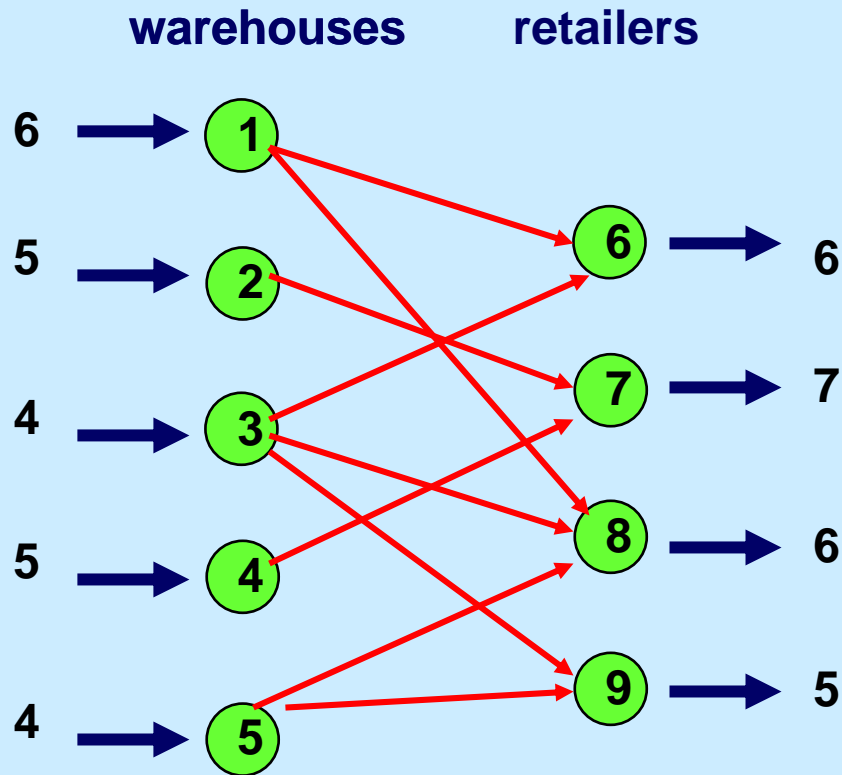
Maximum Flows

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



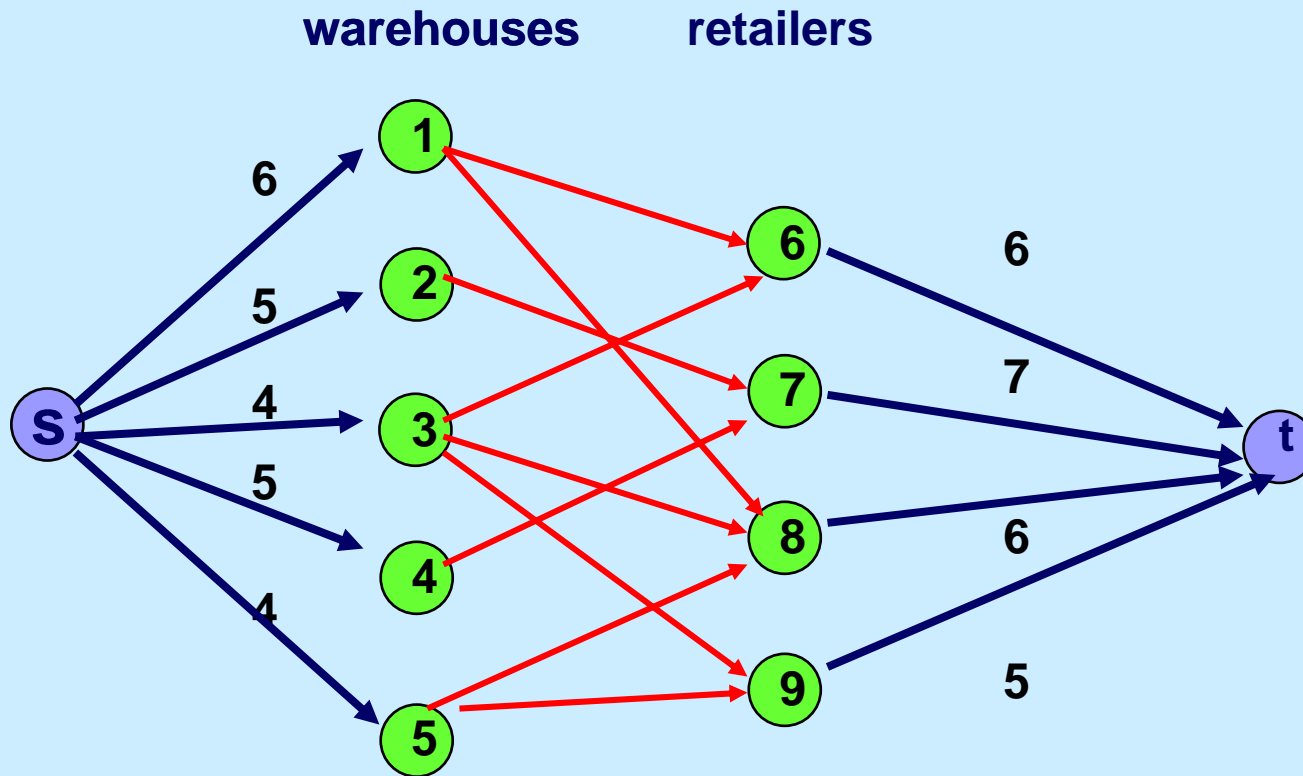
A max flow problem. Capacities and a non-optimum flow.

The feasibility problem: find a feasible flow



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

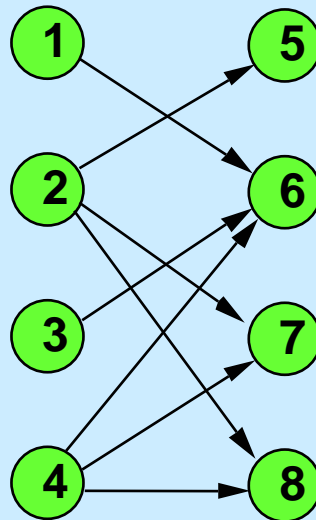
Transformation to a max flow problem



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

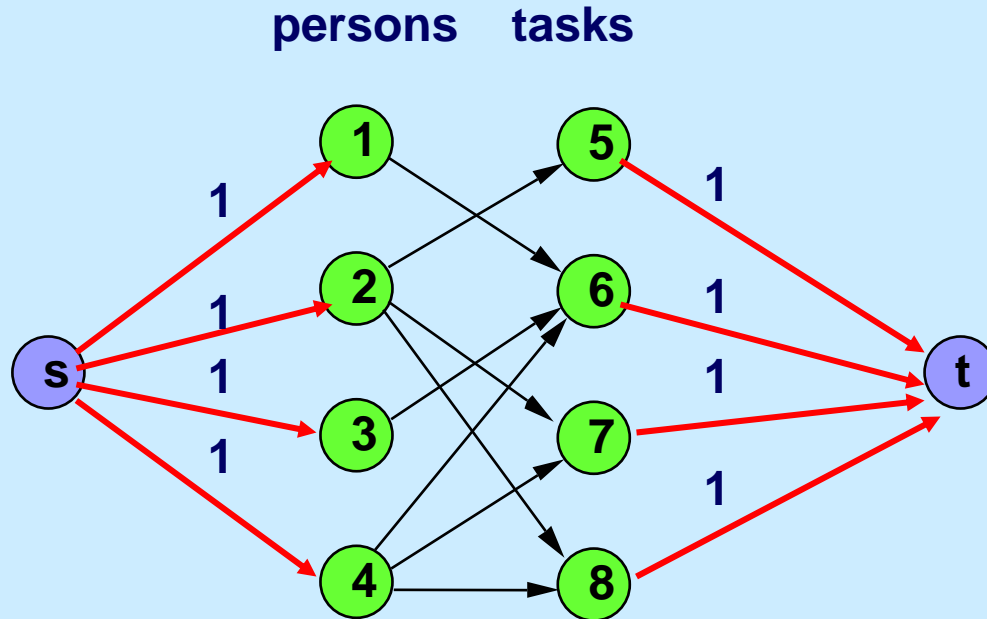
The feasibility problem: find a matching

persons tasks



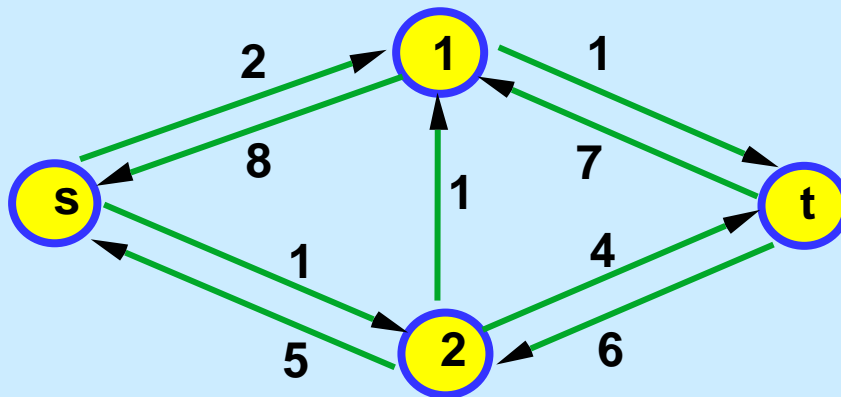
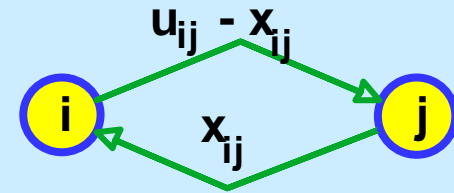
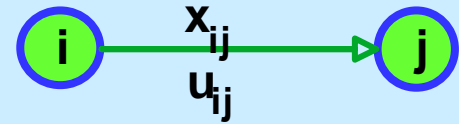
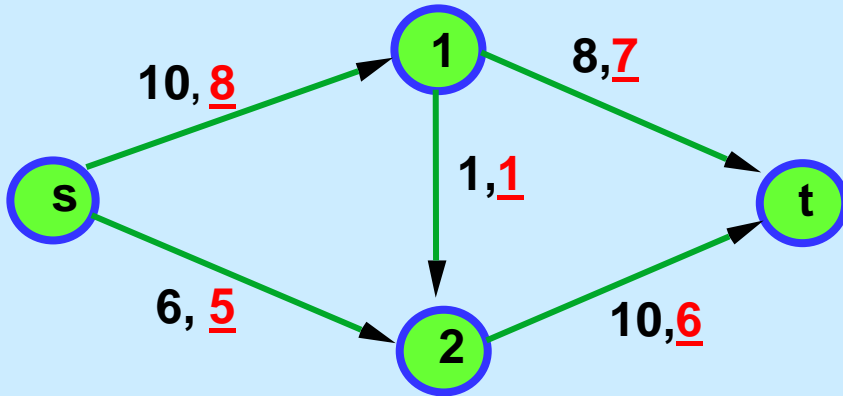
Is there a way of assigning persons to tasks so that each person is assigned a task, and each task has a person assigned to it?

Transformation to a maximum flow problem



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

The Residual Network



The **Residual Network** $G(x)$

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

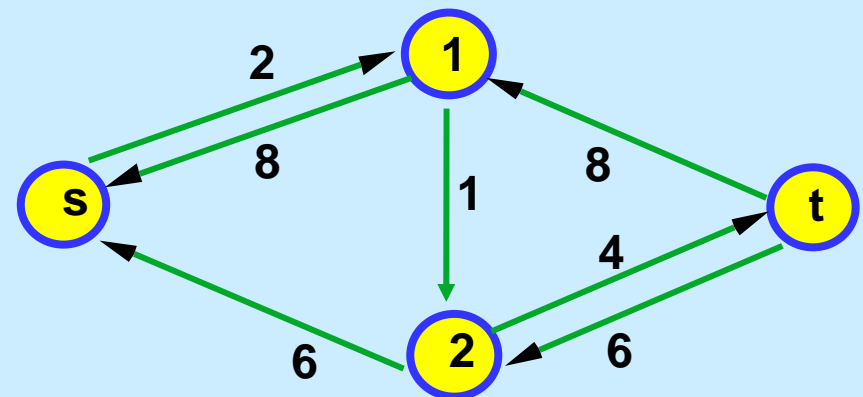
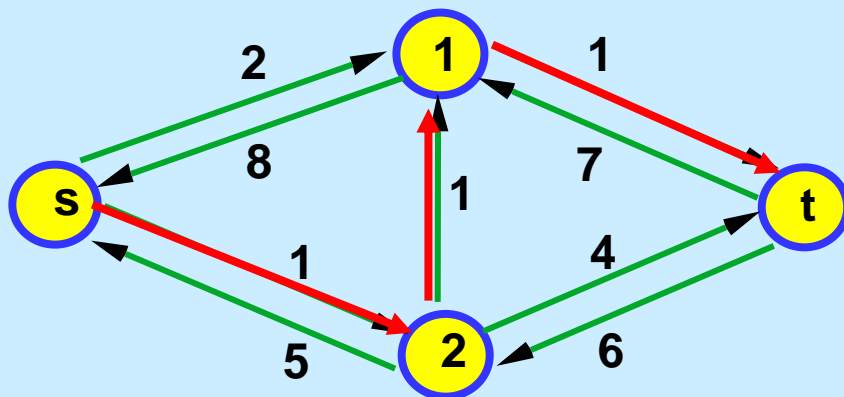
A Useful Idea: Augmenting Paths

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 $d(P)$ 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 $G(x);$

$\Delta := \delta(P);$

send Δ units of flow along P;

update the r's;

end

end {the flow x is now maximum}.

**Ford-
Fulkerson
Algorithm
Animation**

Proof of Correctness of the Algorithm

Assume that all data are integral.

Lemma: *At each iteration all residual capacities are integral.*

Proof. It is true at the beginning. Assume it is true after the first $k-1$ augmentations, and consider augmentation k along path P .

The residual capacity Δ of P is the smallest residual capacity on P , which is integral.

After updating, we modify residual capacities by 0, or Δ , and thus residual capacities stay integral.

Theorem. The Ford-Fulkerson Algorithm is finite

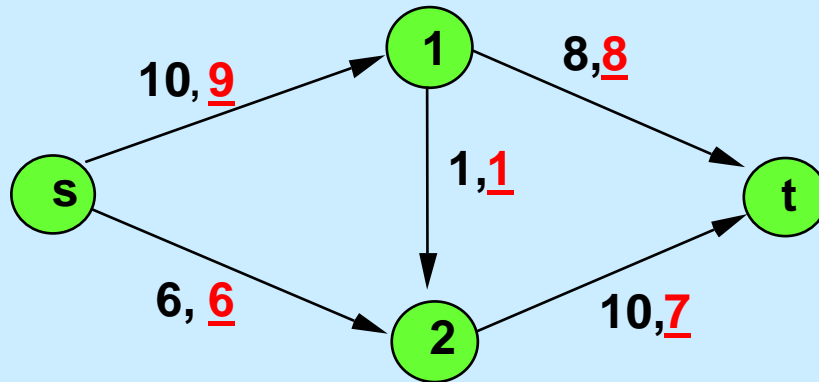
Proof. The capacity of each augmenting path is at least 1.

The augmentation reduces the residual capacity of some arc (s, j) and does not increase the residual capacity of (s, i) for any i .

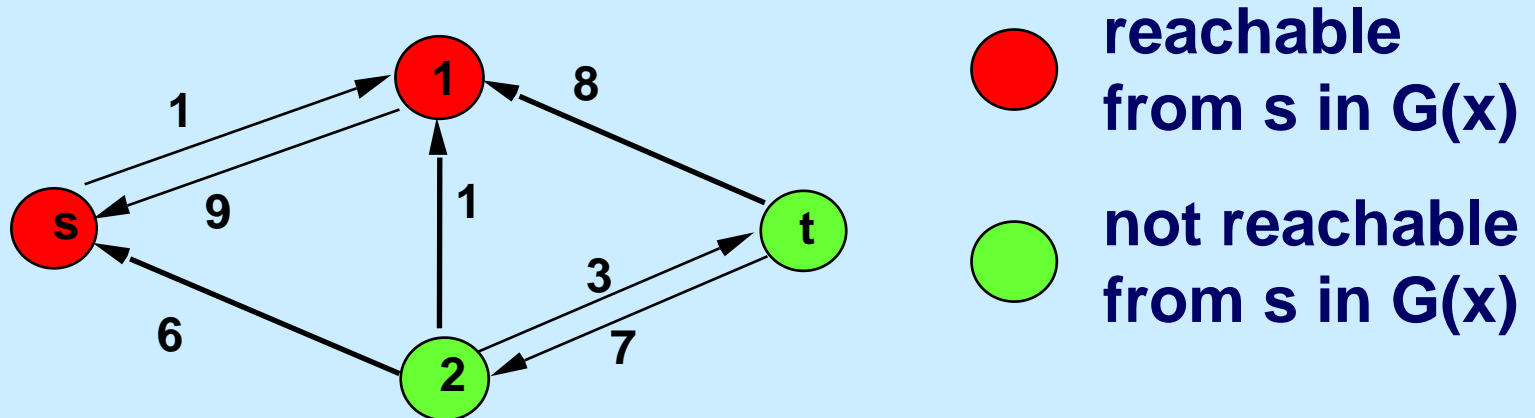
So, the sum of the residual capacities of arcs out of s keeps decreasing, and is bounded below by 0.

Number of augmentations is $O(nU)$, where U is the largest capacity in the network.

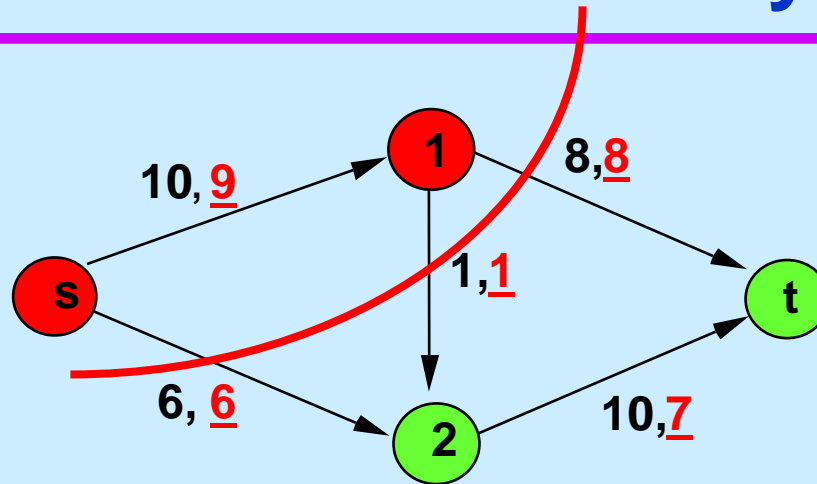
How do we know when a flow is optimal?



METHOD 1. There is no augmenting path in the residual network.



Method 2: Cut Duality Theory



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

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

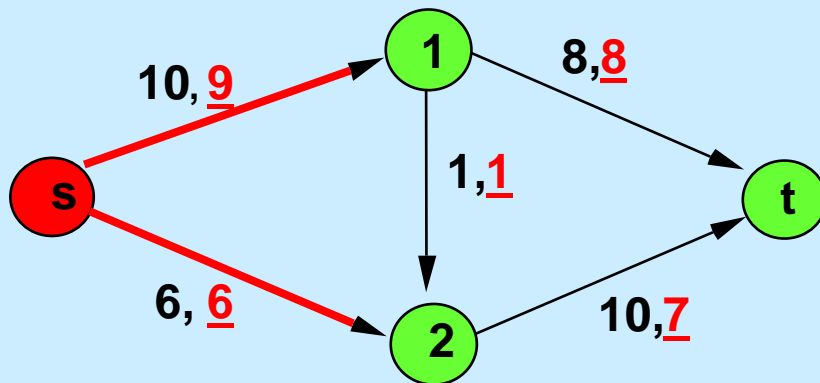
$$\text{CAP}(S,T) = \sum_{i \in S} \sum_{j \in T} u_{ij}$$

Weak Duality Theorem for the Max Flow Problem

Theorem. 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 $CAP(S, T)$.

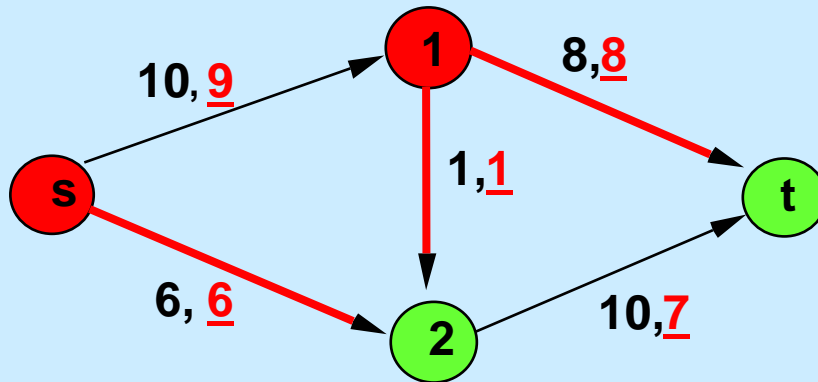
PROOF. We define the **flow across the cut** (S, T) to be

$$F_x(S, T) = \sum_{i \in S} \sum_{j \in T} x_{ij} - \sum_{i \in S} \sum_{j \in T} x_{ji}$$

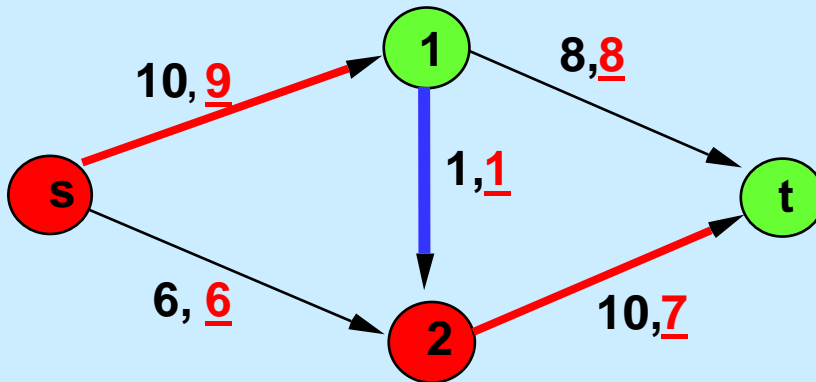


If $S = \{s\}$, then the flow across (S, T) is $9 + 6 = 15$.

Flows Across Cuts



If $S = \{s, 1\}$, then
the flow across
(S, T) is
 $8 + 1 + 6 = 15$.



If $S = \{s, 2\}$, then
the flow across
(S, T) is
 $9 + 7 - 1 = 15$.

More on Flows Across Cuts

Claim: Let (S, T) be any s - t cut. Then $F_x(S, T) = v =$ flow into t .

Proof. Add the conservation of flow constraints for each node $i \in S - \{s\}$ to the constraint that the flow leaving s is v . The resulting equality is $F_x(S, T) = v$.

$$\sum_j x_{ij} - \sum_k x_{ki} = 0 \quad \text{for each } i \in S \setminus s$$

$$\sum_j x_{sj} = v$$



More on Flows Across Cuts

Claim: The flow across (S,T) is at most the capacity of a cut.

Proof. If $i \in S$, and $j \in T$, then $x_{ij} \leq u_{ij}$. If $i \in T$, and $j \in S$, then $x_{ij} \geq 0$.

$$F_x(S,T) = \sum_{i \in S} \sum_{j \in T} x_{ij} - \sum_{i \in S} \sum_{j \in T} x_{ji}$$

$$\text{Cap}(S,T) = \sum_{i \in S} \sum_{j \in T} u_{ij} - \sum_{i \in S} \sum_{j \in T} 0$$

Max Flow Min Cut Theorem

Theorem. (Optimality conditions for max flows).
The following are equivalent.

1. *A flow x is maximum.*
2. *There is no augmenting path in $G(x)$.*
3. *There is an s - t cutset (S, T) whose capacity is the flow value of x .*

Corollary. (Max-flow Min-Cut). *The maximum flow value is the minimum value of a cut.*

Proof of Theorem. **1 \Rightarrow 2.** (not 2 \Rightarrow not 1)

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

Continuation of the proof.

3 \Rightarrow 1. Let $v = F_x(S, T)$ be the flow from s to t . By assumption, $v = \text{CAP}(S, T)$. By weak duality, the maximum flow is at most $\text{CAP}(S, T)$. Thus the flow is maximum.

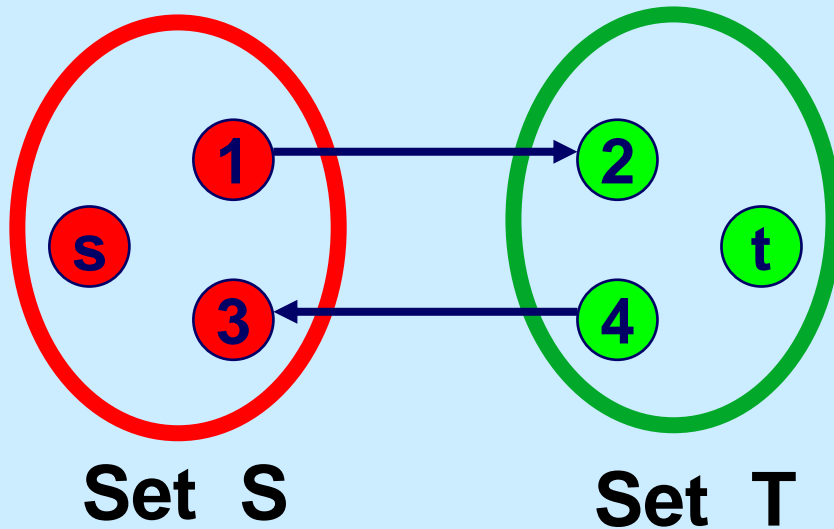
2 \Rightarrow 3. Suppose there is no augmenting path in $G(x)$.

Claim: Let S be the set of nodes reachable from s in $G(x)$. Let $T = N \setminus S$. 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 $j \in S \Rightarrow x_{ij} = 0.$

Final steps of the proof

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



There is no arc from S to T in $G(x)$

$$x_{12} = u_{12}$$

$$x_{43} = 0$$

It follows that

$$F_x(S, T) = \sum_{i \in S} \sum_{j \in T} x_{ij} - \sum_{i \in S} \sum_{j \in T} x_{ji}$$

$$= \sum_{i \in S} \sum_{j \in T} u_{ij} - \sum_{i \in S} \sum_{j \in T} 0 = \text{CAP}(S, T)$$

Review

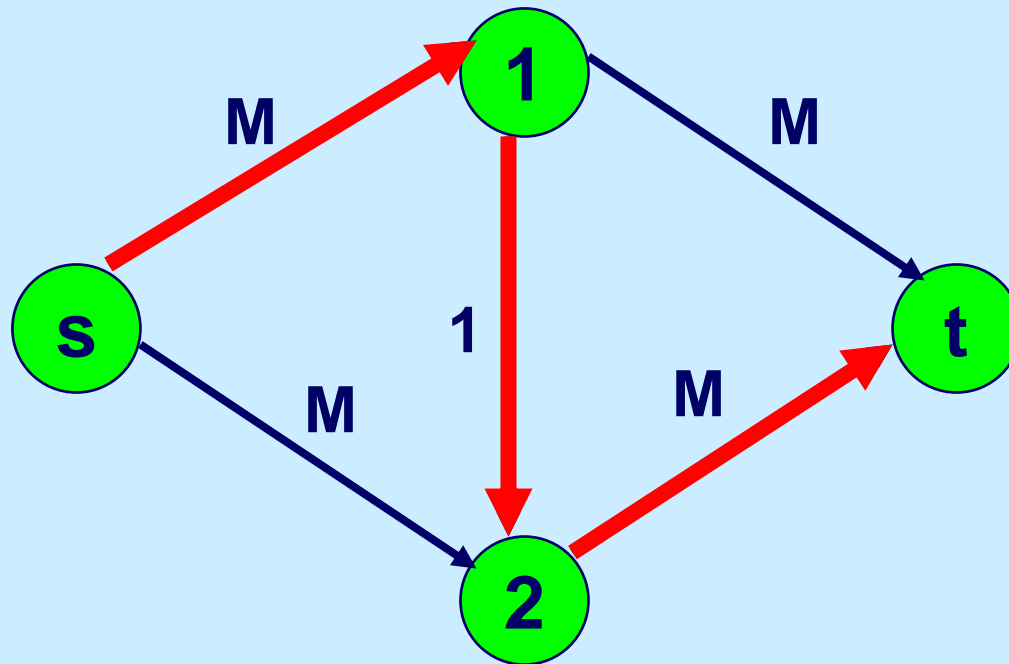
Corollary. If the capacities are finite integers, then the Ford-Fulkerson Augmenting Path Algorithm terminates in finite time with a maximum flow from s to t .

Corollary. If the capacities are finite rational numbers, then the Ford-Fulkerson Augmenting Path Algorithm terminates in finite time with a maximum flow from s to t . (why?)

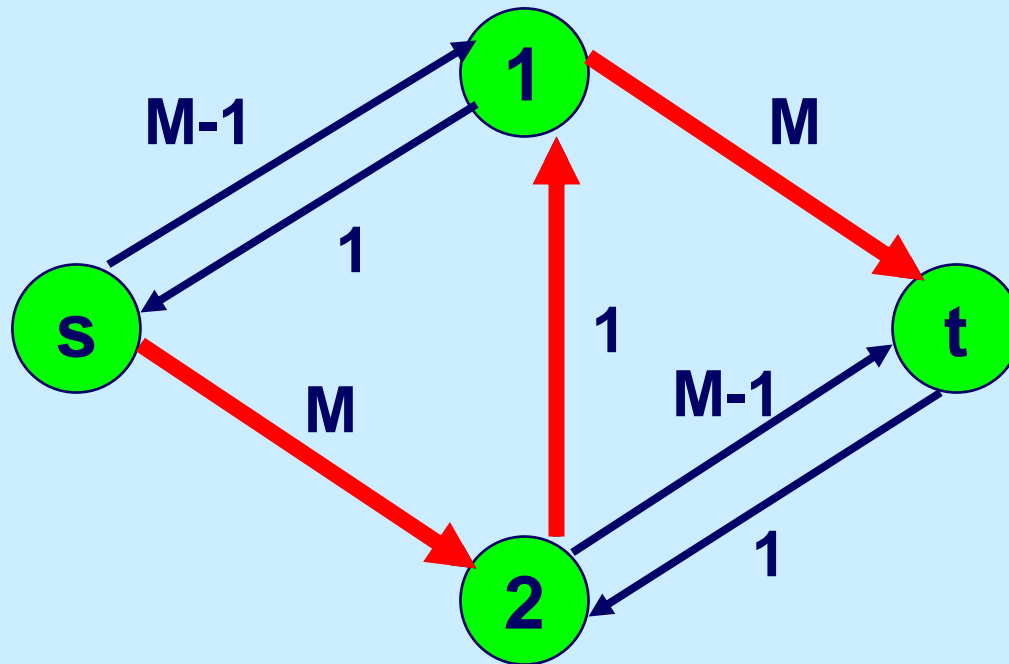
Corollary. To obtain a minimum cut from a maximum flow x , let S denote all nodes reachable from s in $G(x)$.

Remark. This does not establish finiteness if $u_{ij} = \infty$ or if capacities may be irrational.

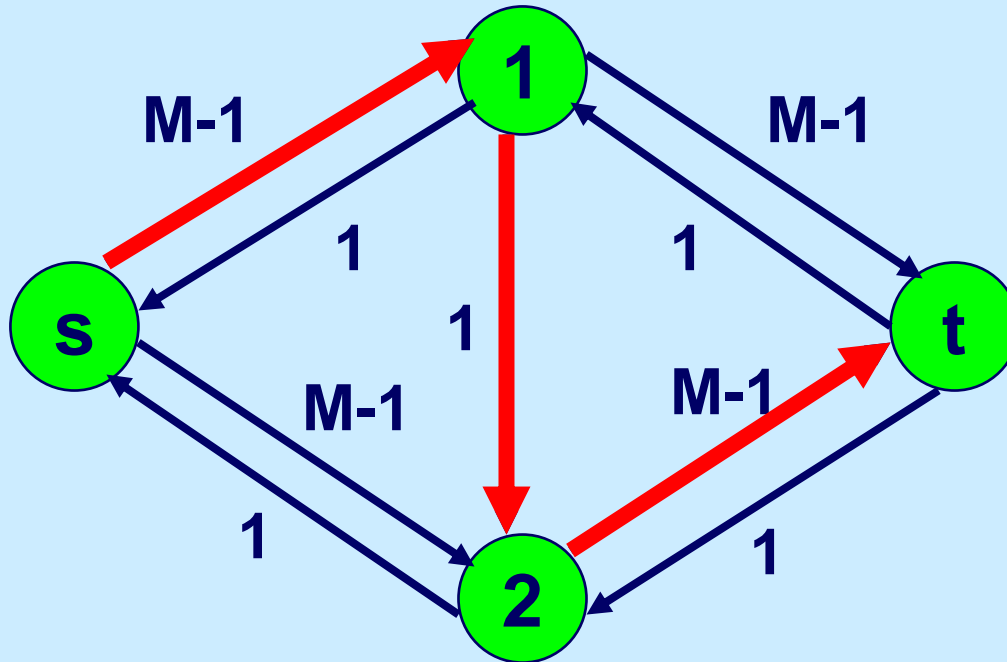
A simple and very bad example



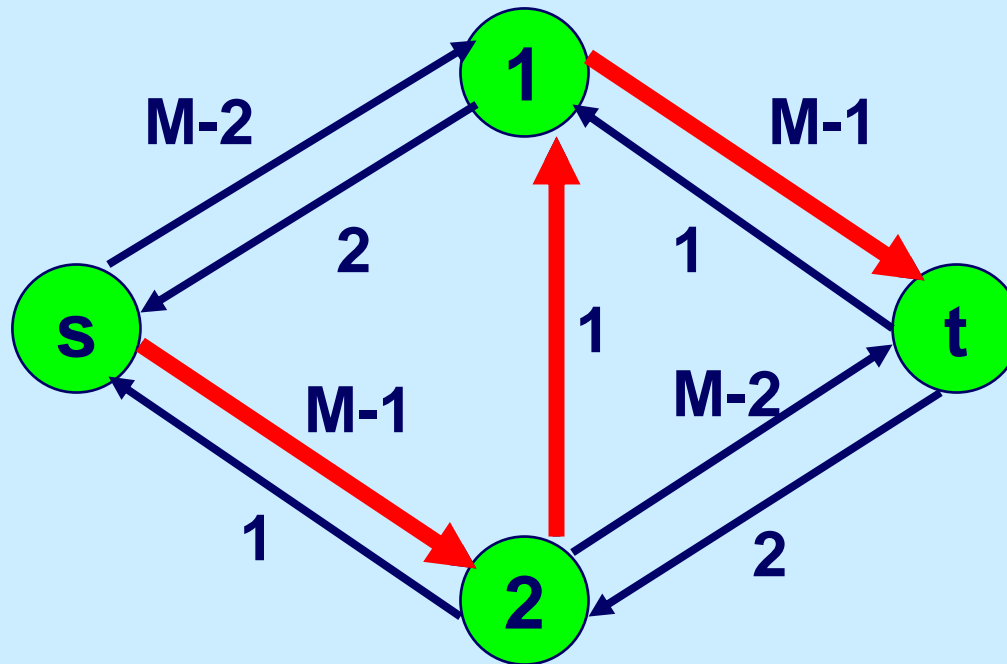
After 1 augmentation



After two augmentations



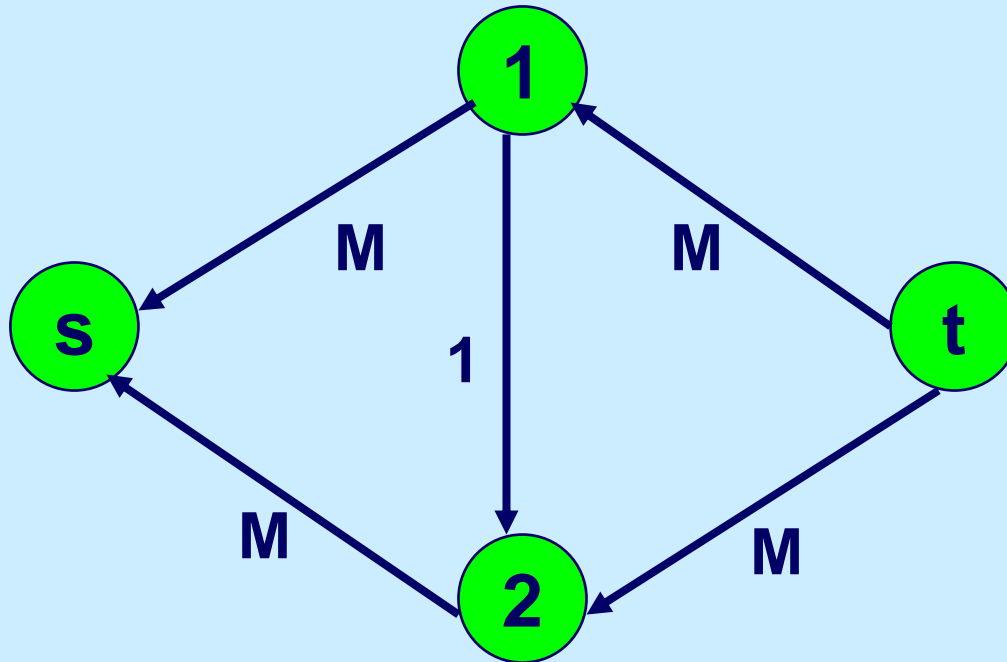
After 3 augmentations



And so on



After 2M augmentations



An even worse example

In Exercise 6.48, there is an example that takes an infinite number of augmentations on irrational data, and does not converge to the correct flow.

But we shall soon see how to solve max flows in a polynomial number of operations, even if data can be irrational.

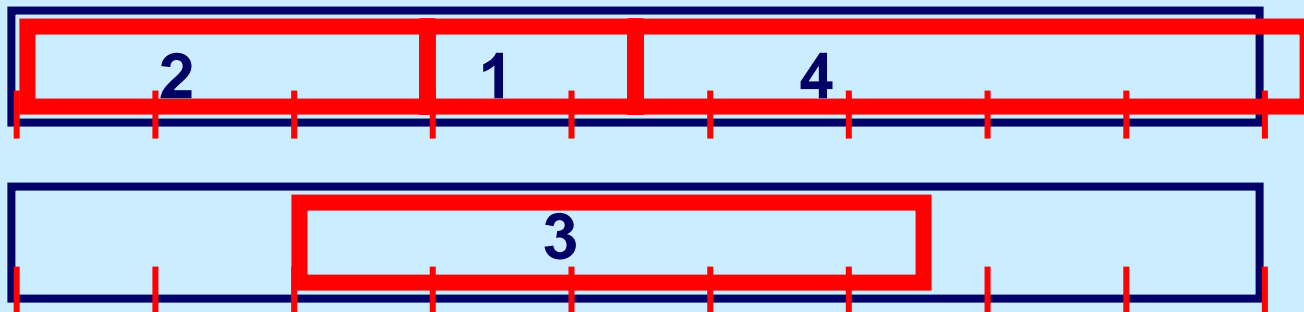
Summary and Extensions

1. **Augmenting path theorem**
2. **Ford-Fulkerson Algorithm**
3. **Duality Theory.**
4. **Computational Speedups.**

Application 6.4 Scheduling on Uniform Parallel Machines

Job(j)	1	2	3	4
Processing Time	1.5	3	4.5	5
Release Time	2	0	2	4
Due Date	5	4	7	9

Suppose there are 2 parallel machines

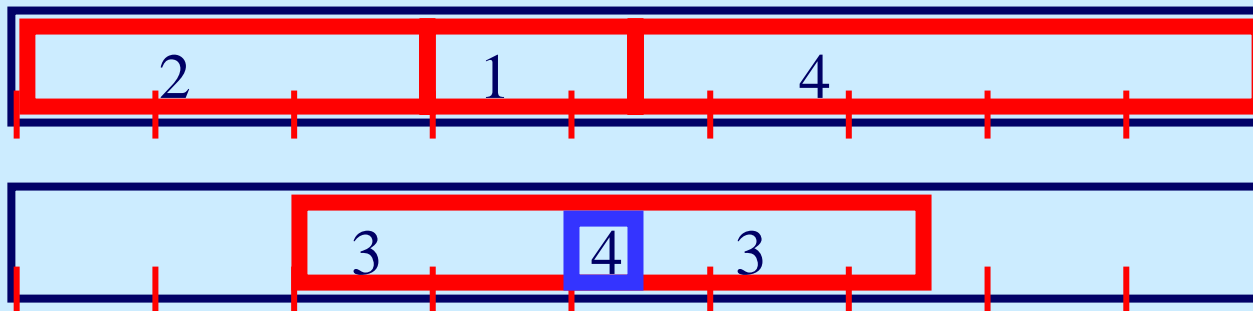


No schedule is possible unless **preemption** is allowed

Application 6.4 Scheduling on Uniform Parallel Machines

Job(j)	1	2	3	4
Processing Time	1.5	3	4.5	5
Release Time	2	0	2	4
Due Date	5	4	7	9

Suppose there are 2 parallel machines

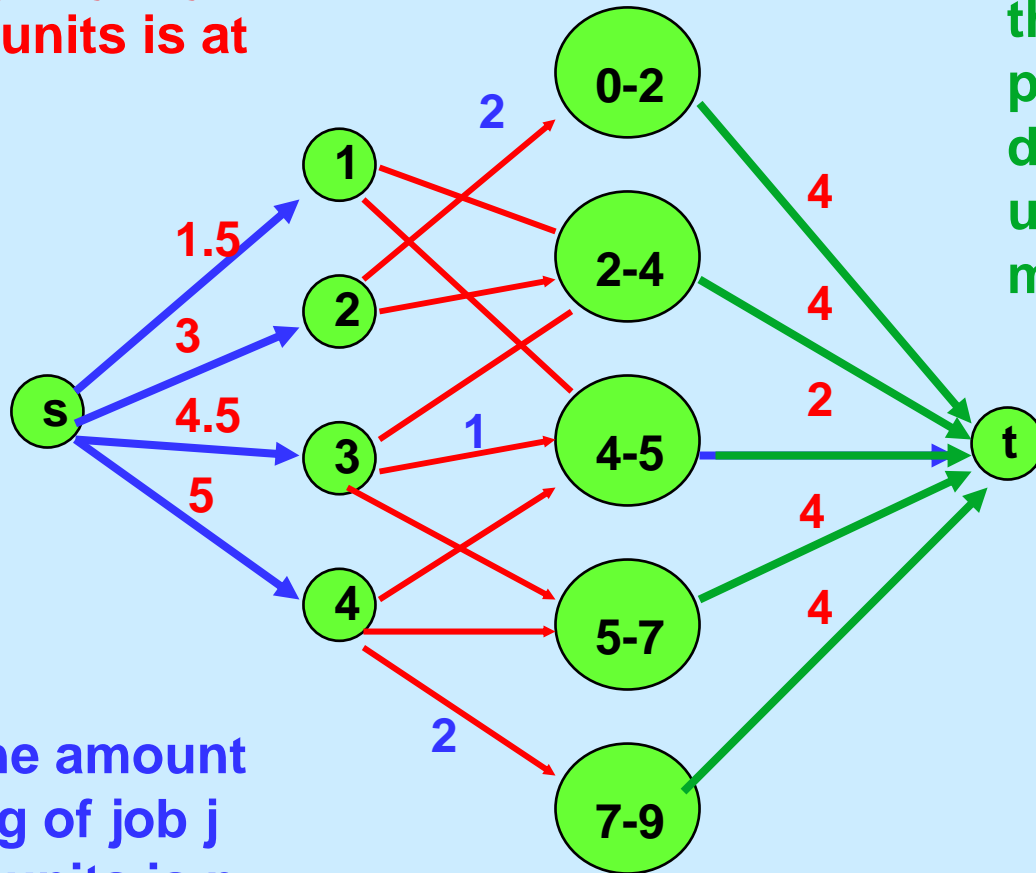


A feasible schedule with **preemption allowed**

Transformation into a maximum flow problem

red arcs: the amount of processing of job j during t time units is at most t .

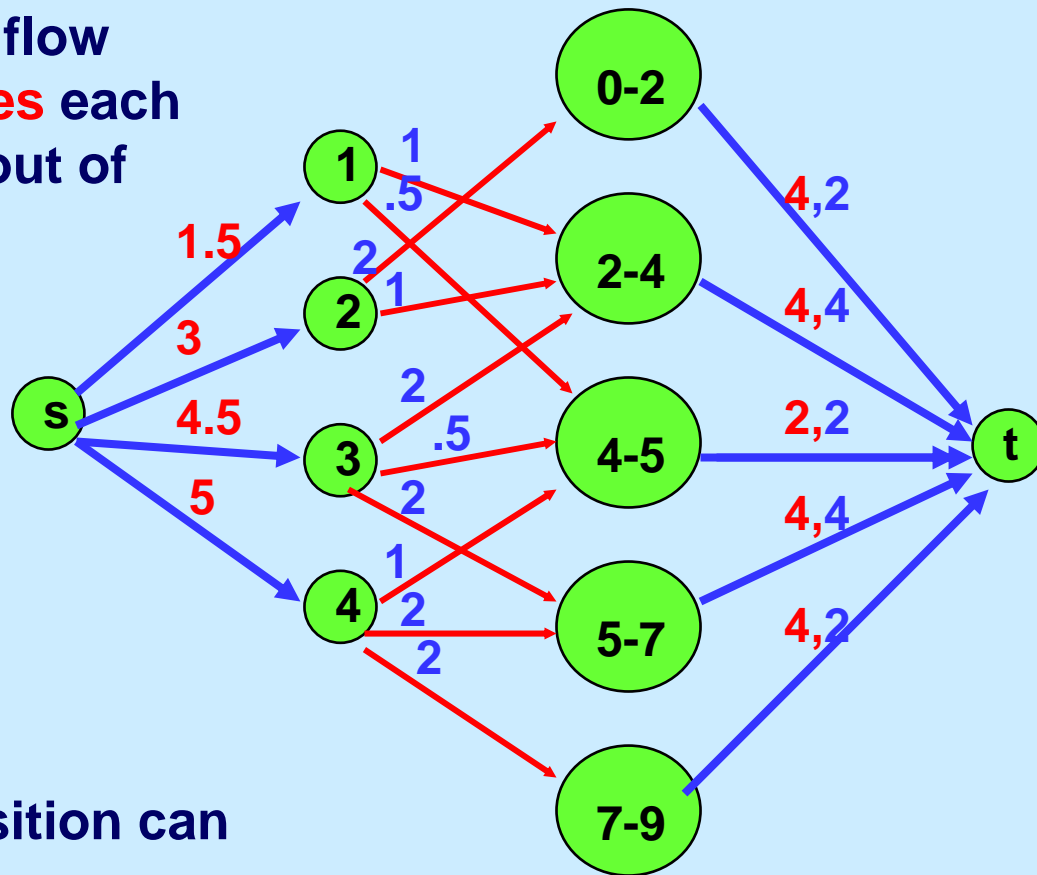
green arcs: the amount of processing during t time units is at most Mt .



blue arcs: the amount of processing of job j over all time units is p_j .

A maximum flow

There is a feasible schedule if there is a maximum flow that **saturates** each of the arcs out of s .



Flow decomposition can be used to transform flows into schedules