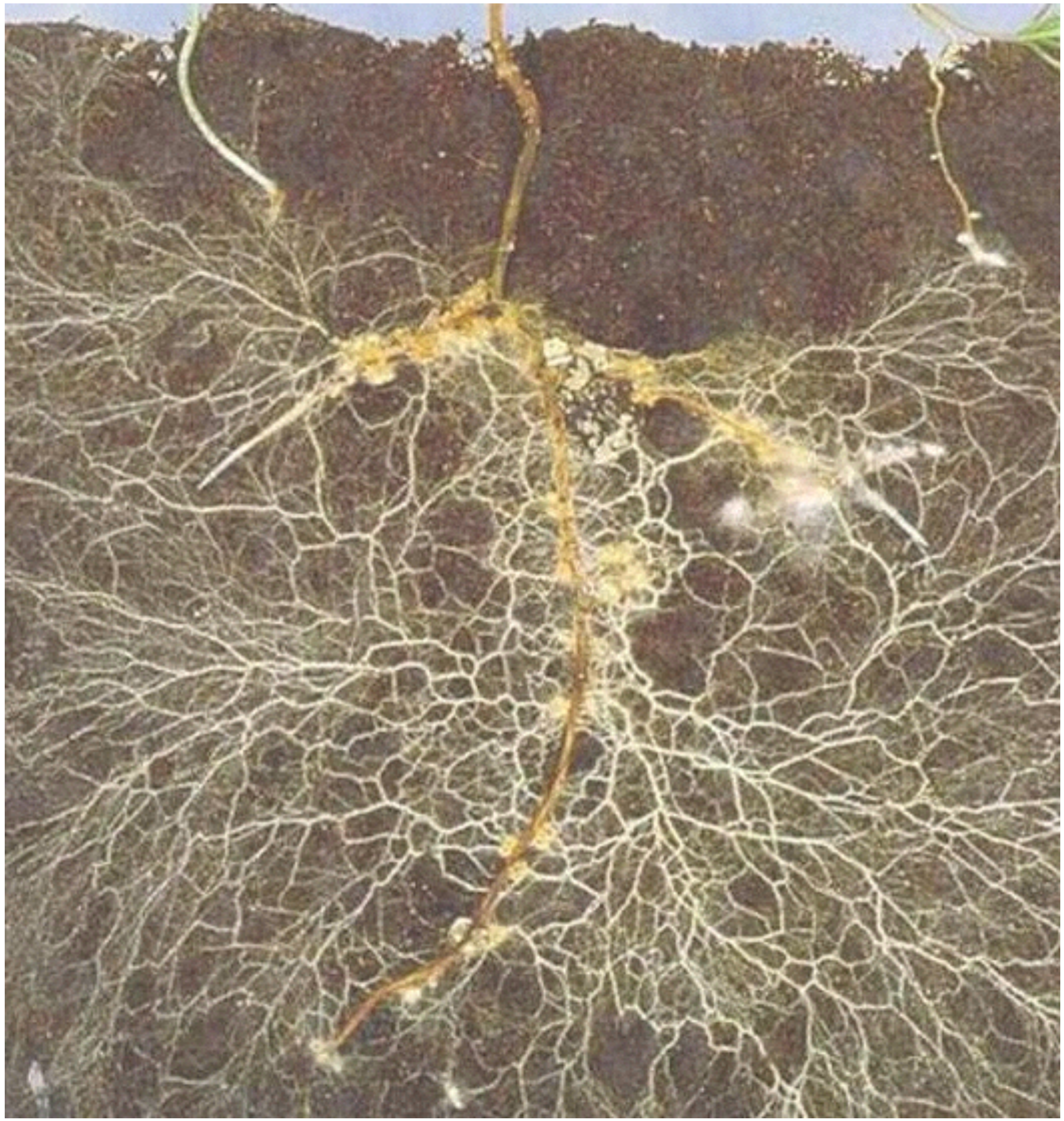
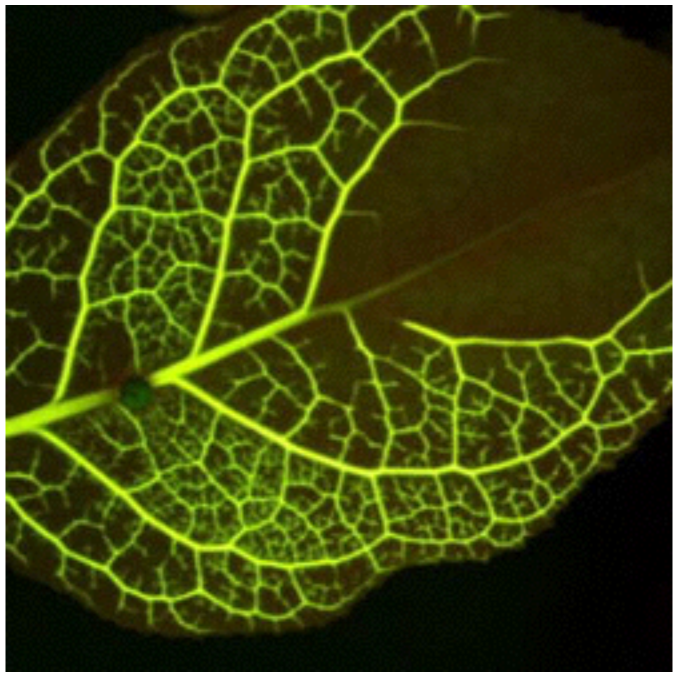
Transport problems

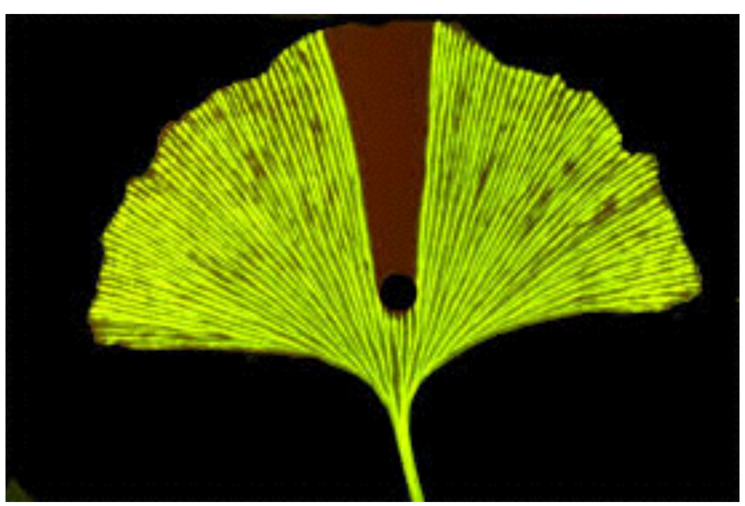
18.S995 - L32

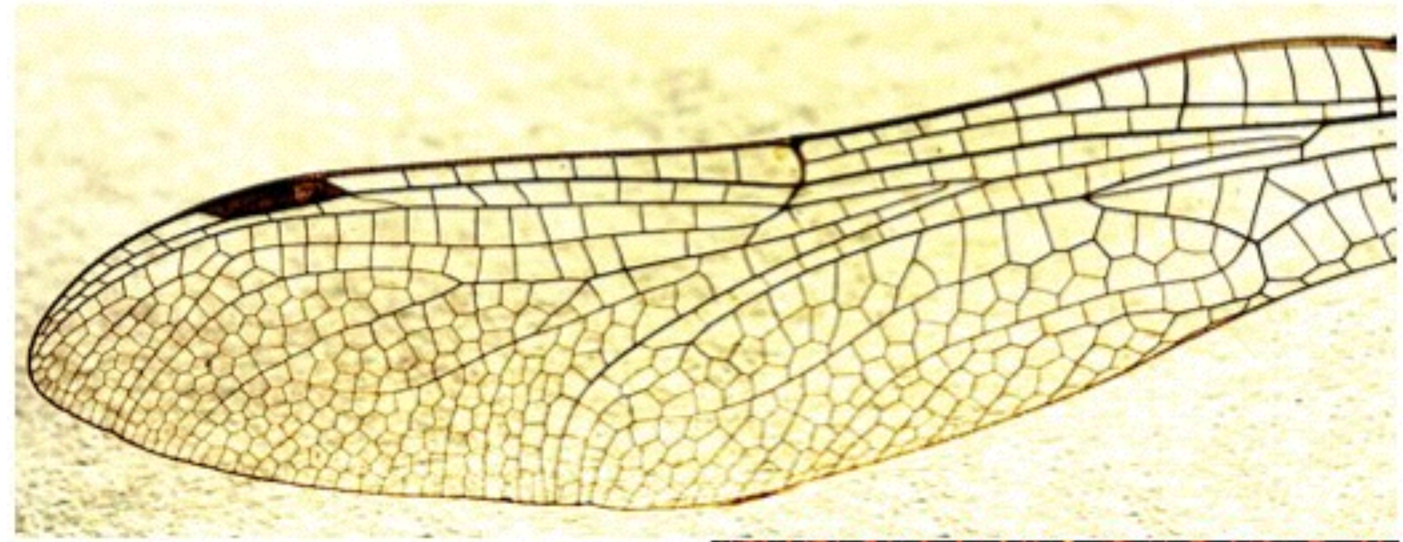
dunkel@math.mit.edu

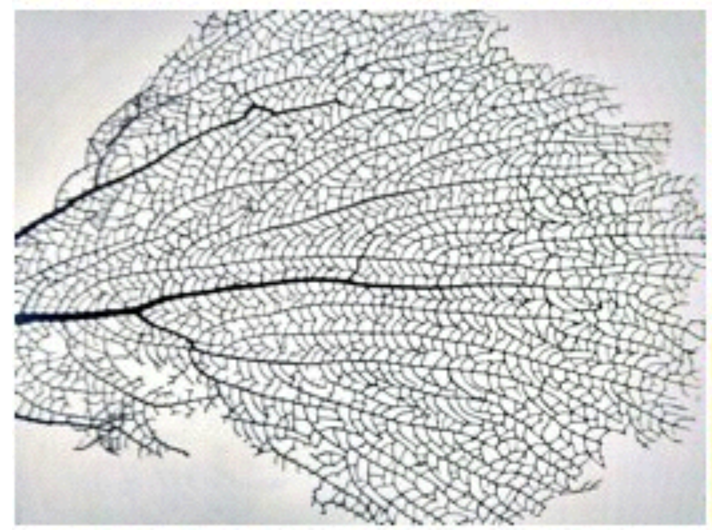
Root systems

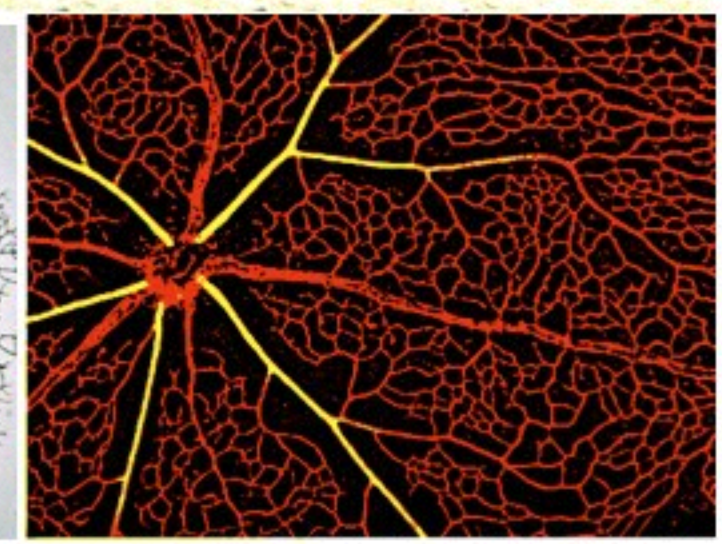


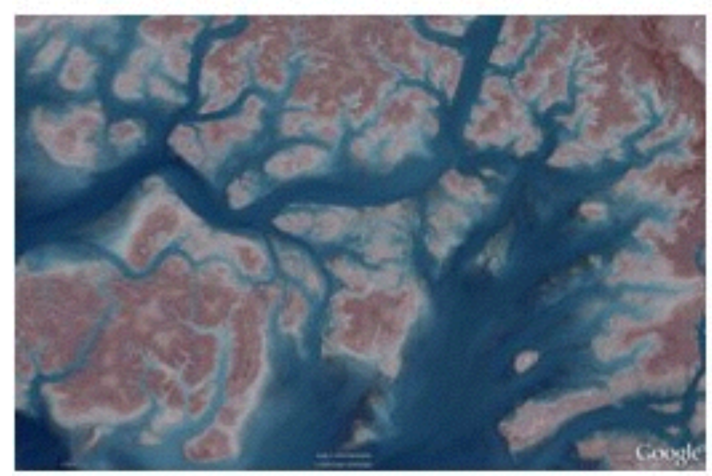


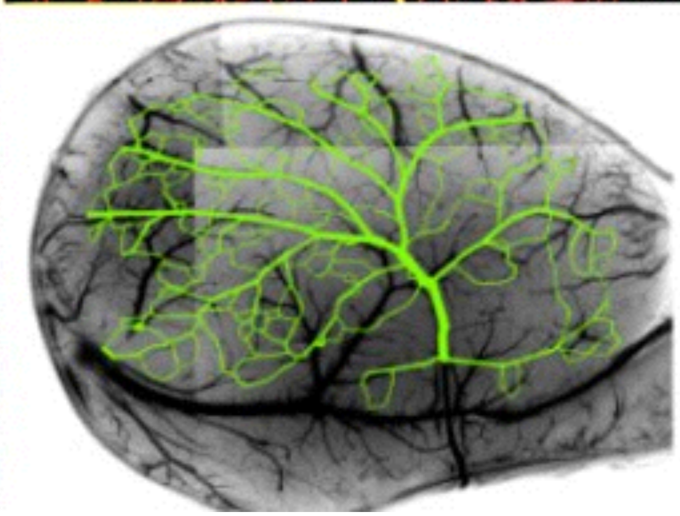






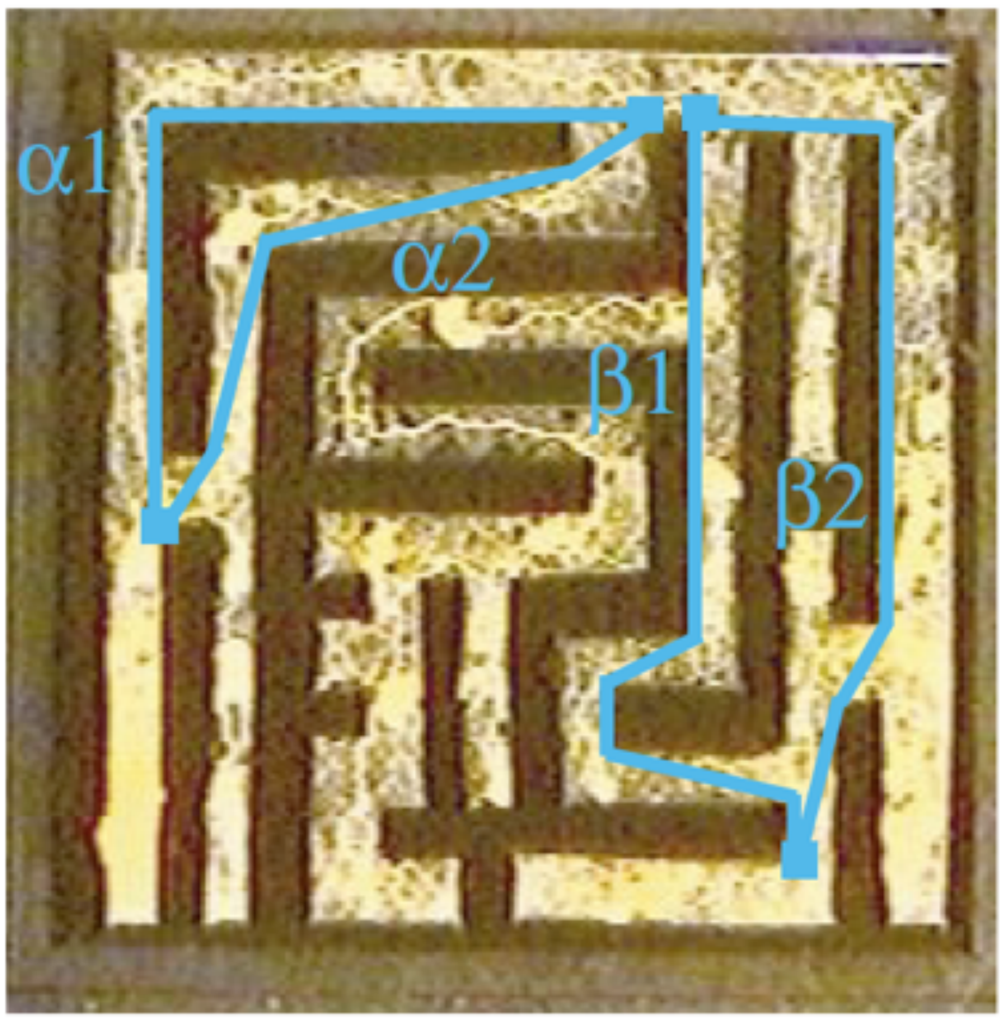




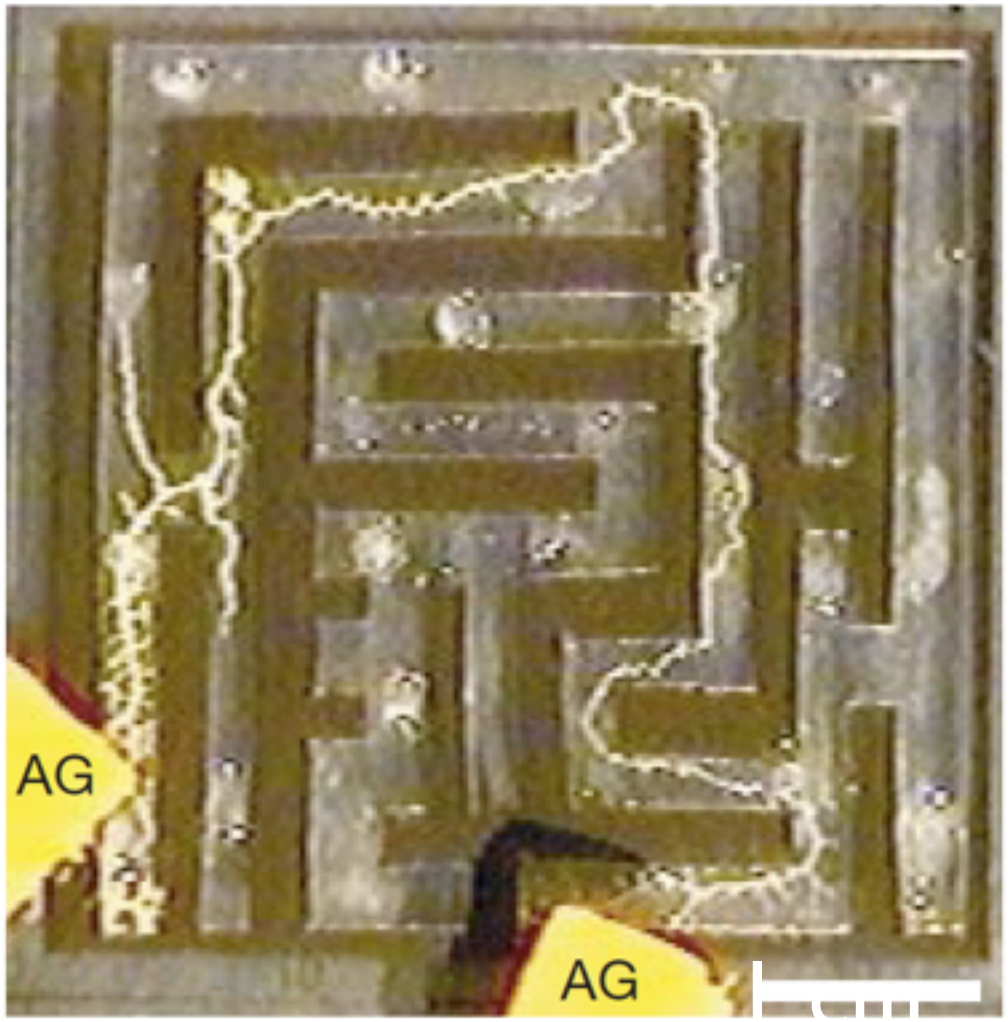


Katifori lab, MPI Goettingen

Maze solving



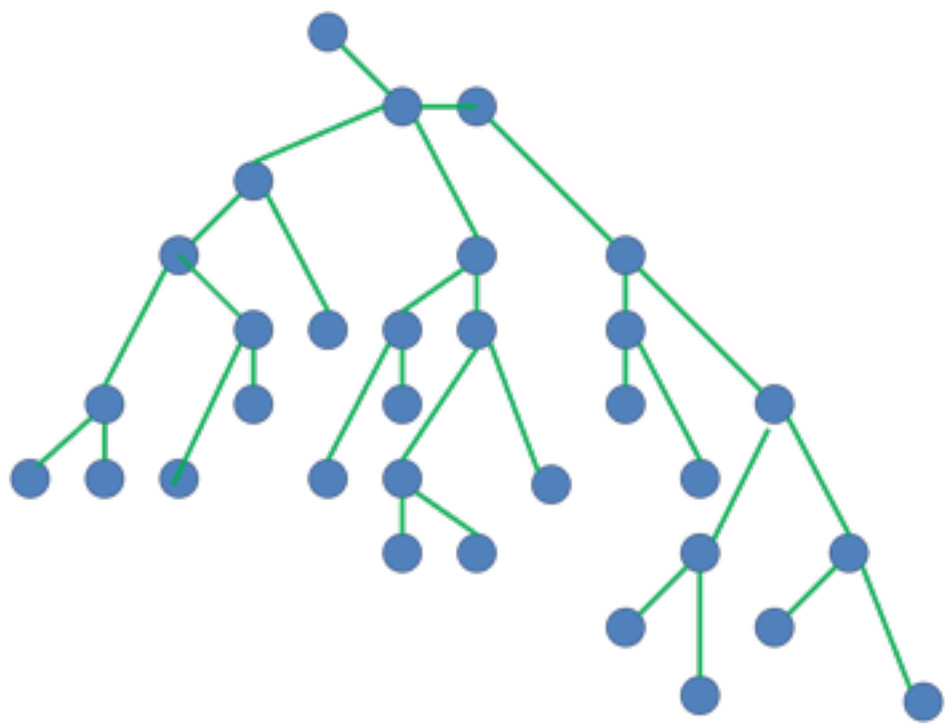
time = 0

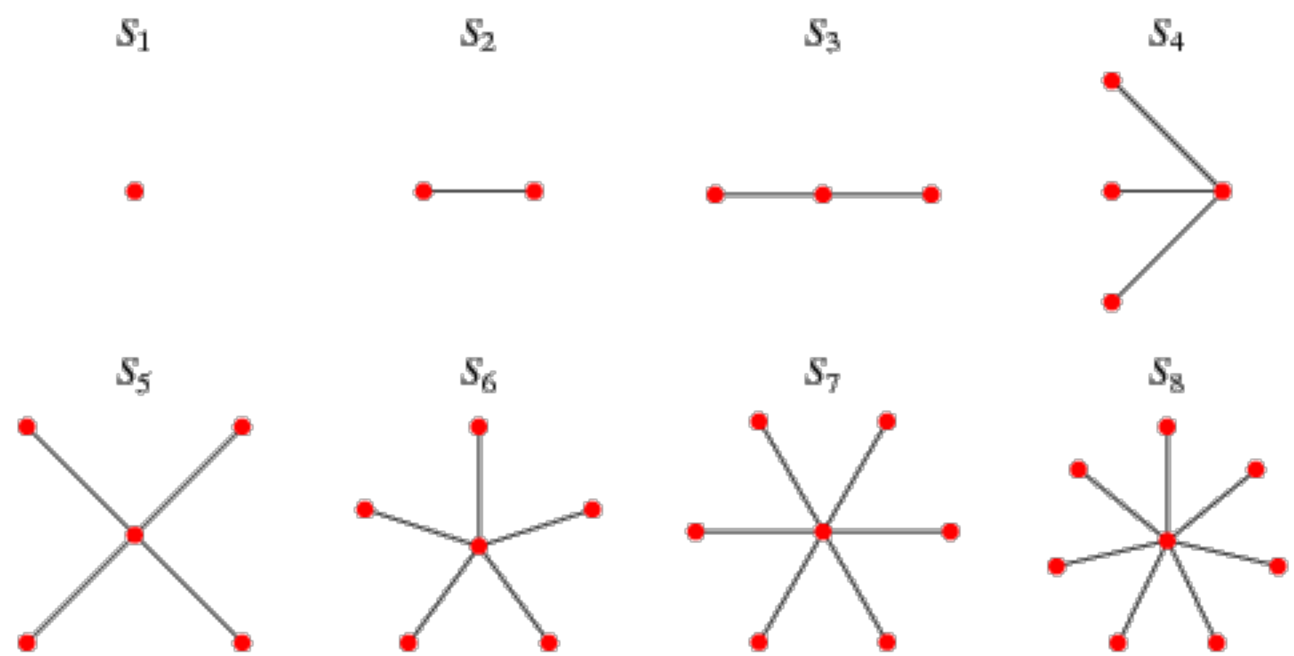


after 8 hours

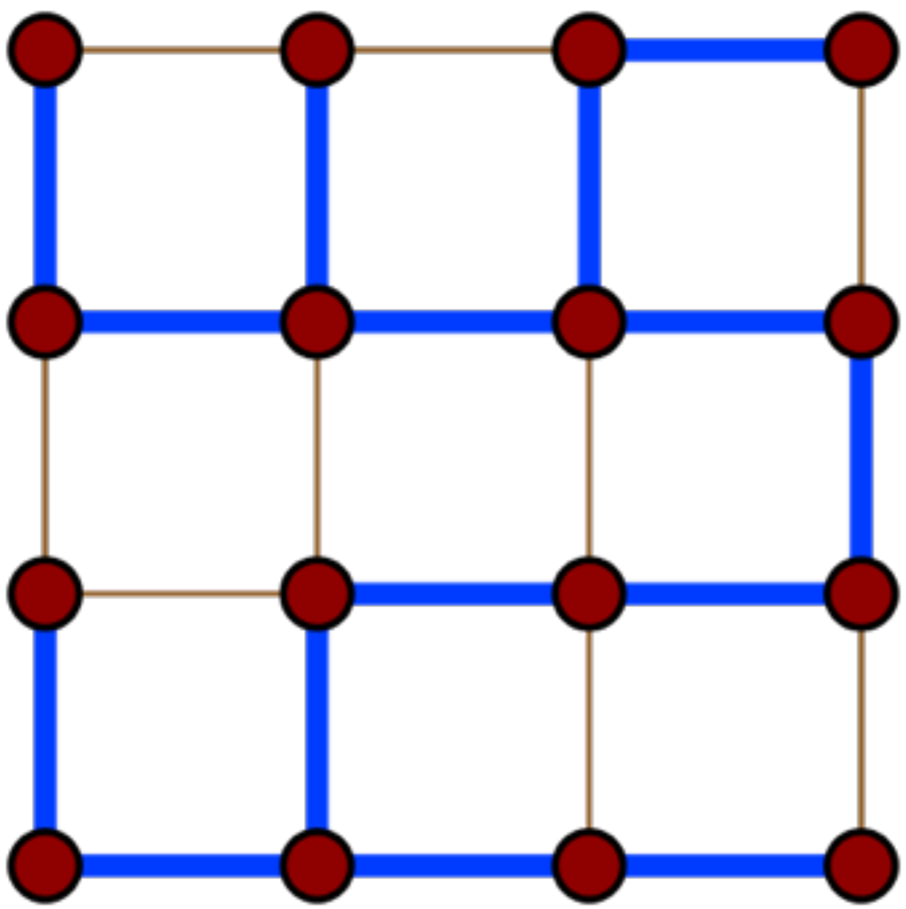
Nakagaki *et al* (2000) Nature

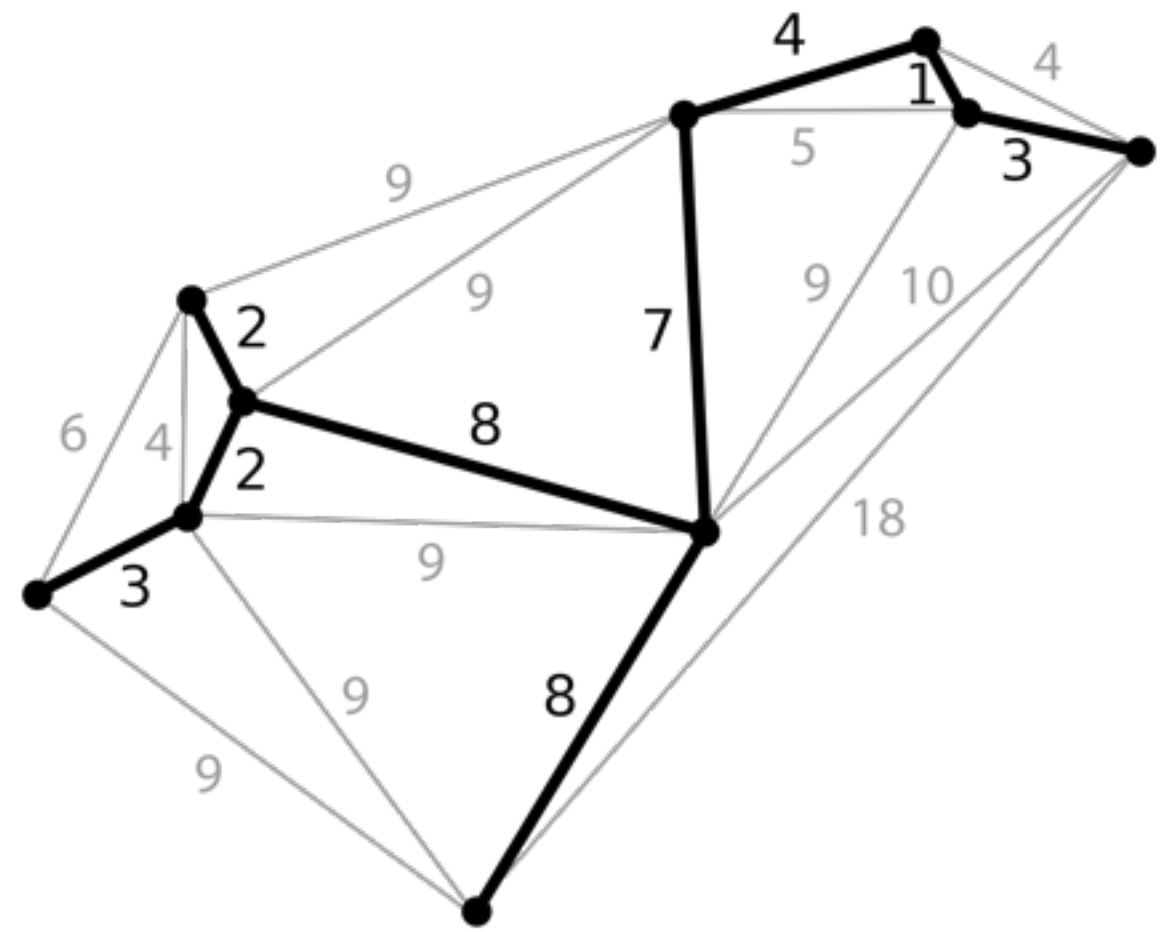
Tree graphs





Spanning trees





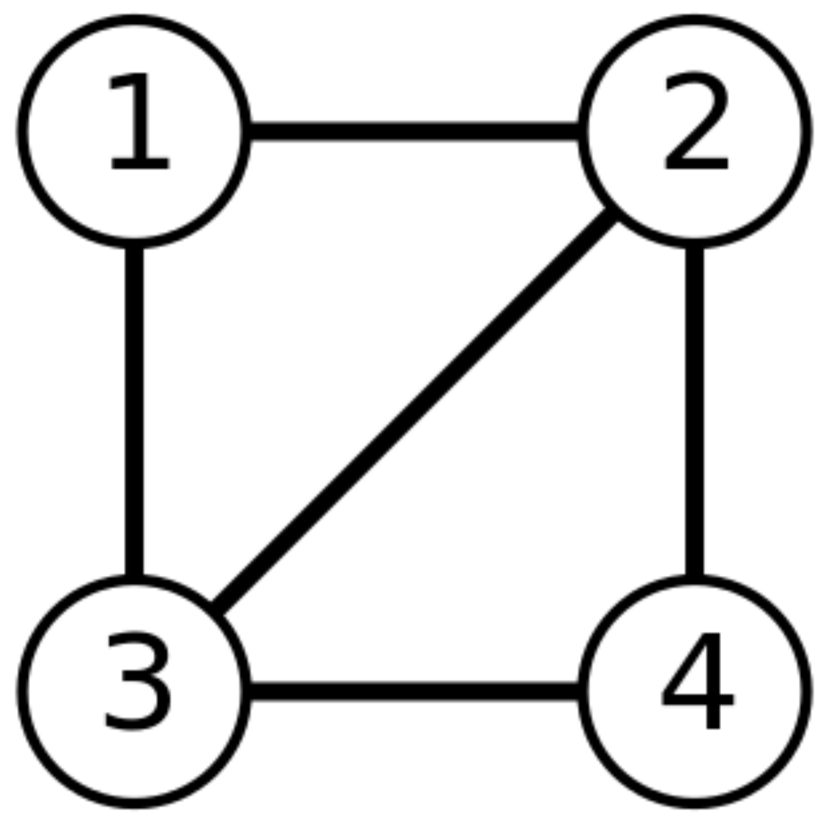
Minimal spanning tree

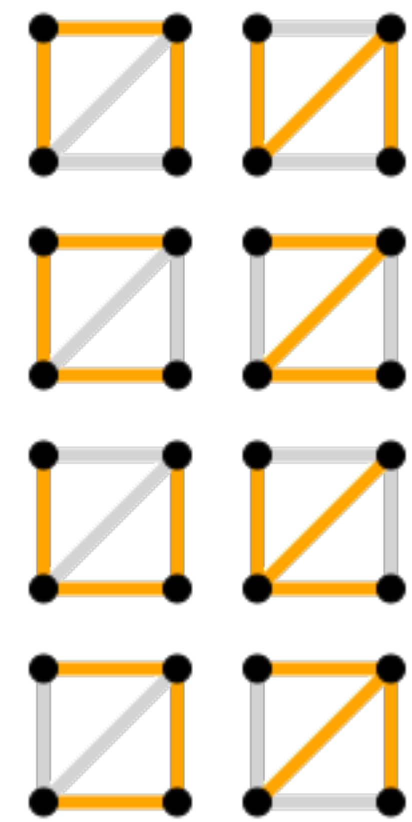
Kirchhoff's theorem

Number of spanning trees $$t(G) = \frac{1}{n}\lambda_1 \lambda_2 \cdots \lambda_{n-1} \,.$$

$$L = \begin{bmatrix} 2 & -1 & -1 & 0 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 3 & -1 \\ 0 & -1 & -1 & 2 \end{bmatrix}.$$

$$L_{11} = \begin{bmatrix} 3 & -1 & -1 \\ -1 & 3 & -1 \\ -1 & -1 & 2 \end{bmatrix}.$$

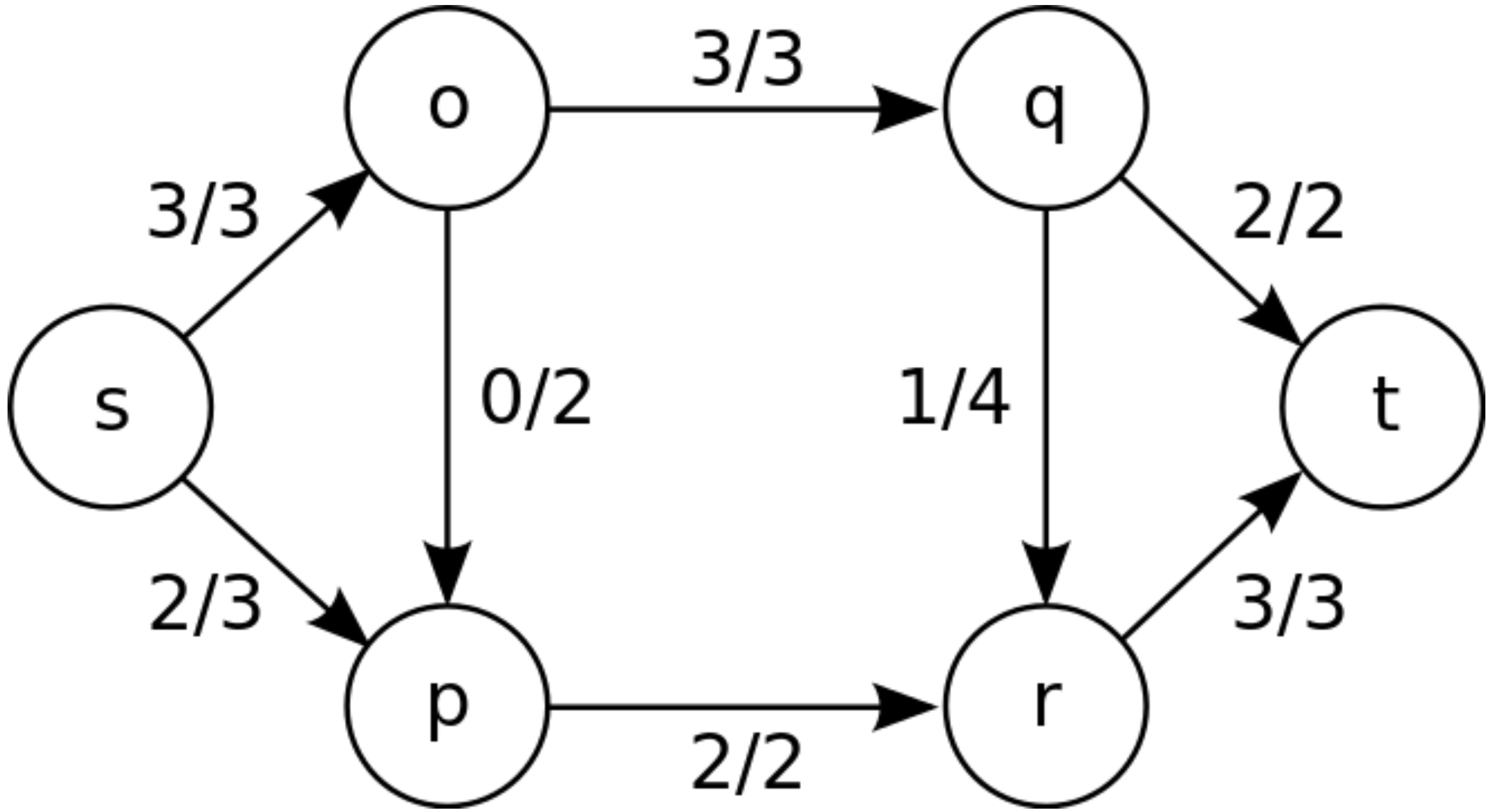




sufficient to consider since row and column sums of Laplacian are zero

$$\det L_{11} = 8$$

Max-Flow Min-Cut



An example of a flow network with a maximum flow. The source is *s*, and the sink *t*. The numbers denote flow and capacity.

Deterministic transport

Maximum Flow and Minimum Cut

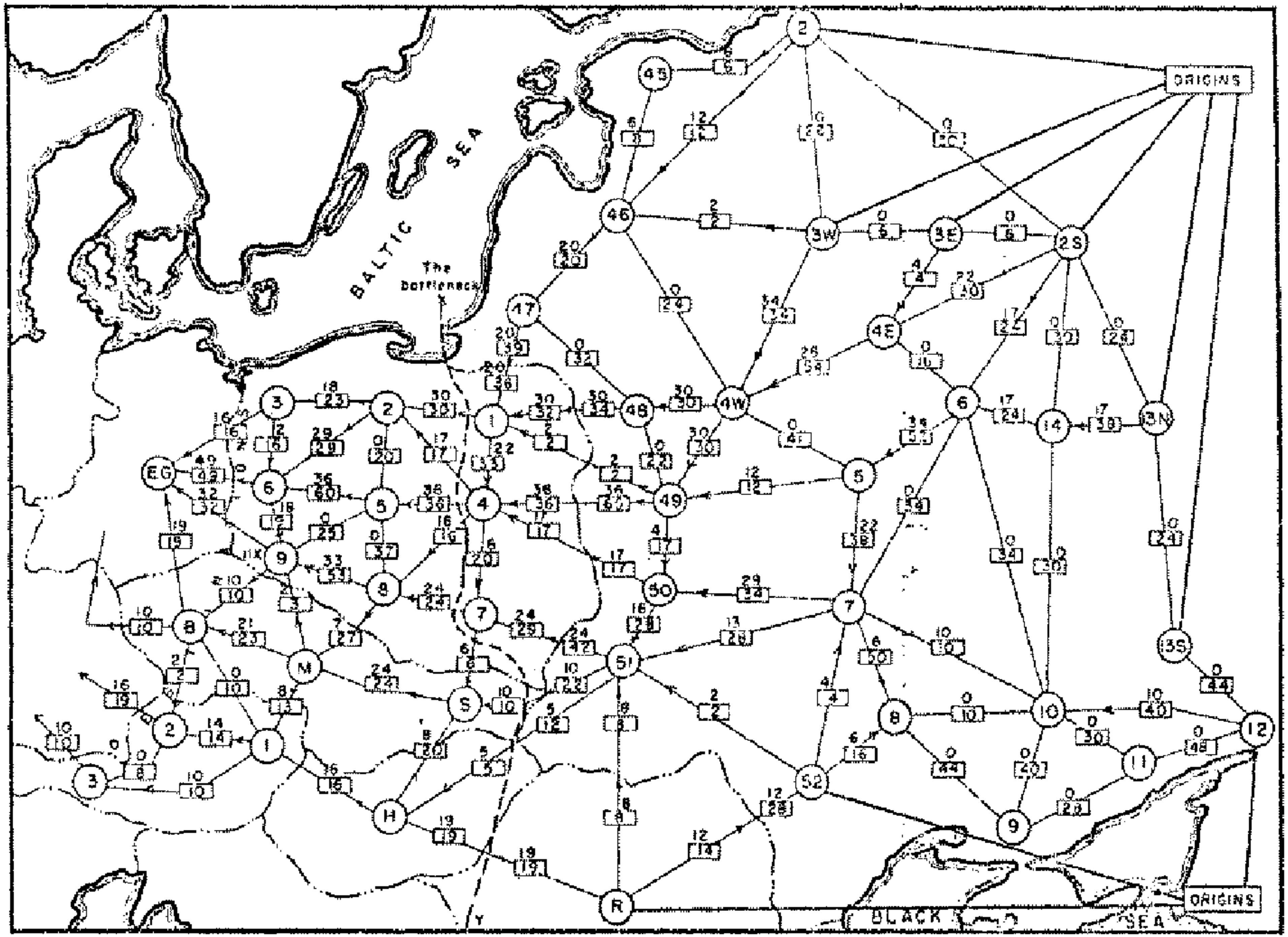
Max flow and min cut.

- Two very rich algorithmic problems.
- Cornerstone problems in combinatorial optimization.
- Beautiful mathematical duality.

Nontrivial applications / reductions.

- Network connectivity.
- Bipartite matching.
- Data mining.
- Open-pit mining.
- Airline scheduling.
- Image processing.
- Project selection.
- Baseball elimination.
- Network reliability.
- Security of statistical data.
- Distributed computing.
- Egalitarian stable matching.
- Distributed computing.
- Many many more . . .

Soviet Rail Network, 1955



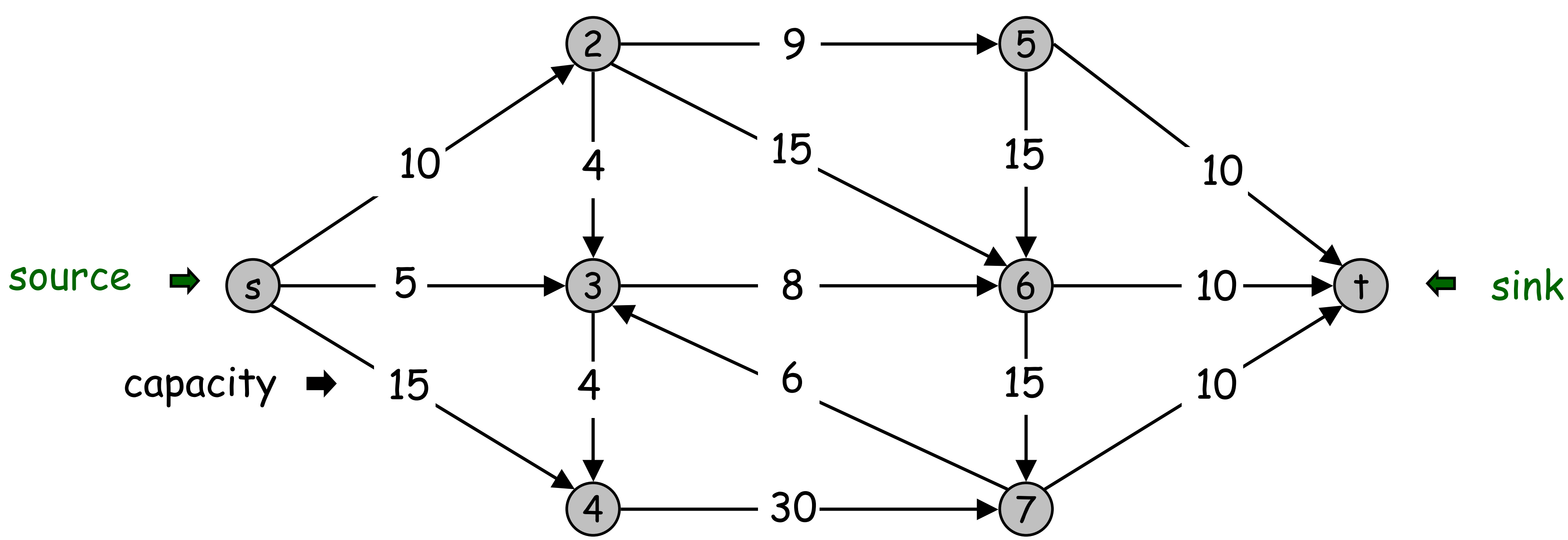
Source: *On the history of the transportation and maximum flow problems.*
Alexander Schrijver in Math Programming, 91: 3, 2002.

Minimum Cut Problem

Network: abstraction for material FLOWING through the edges.

- Directed graph.
- Capacities on edges.
- Source node s, sink node t.

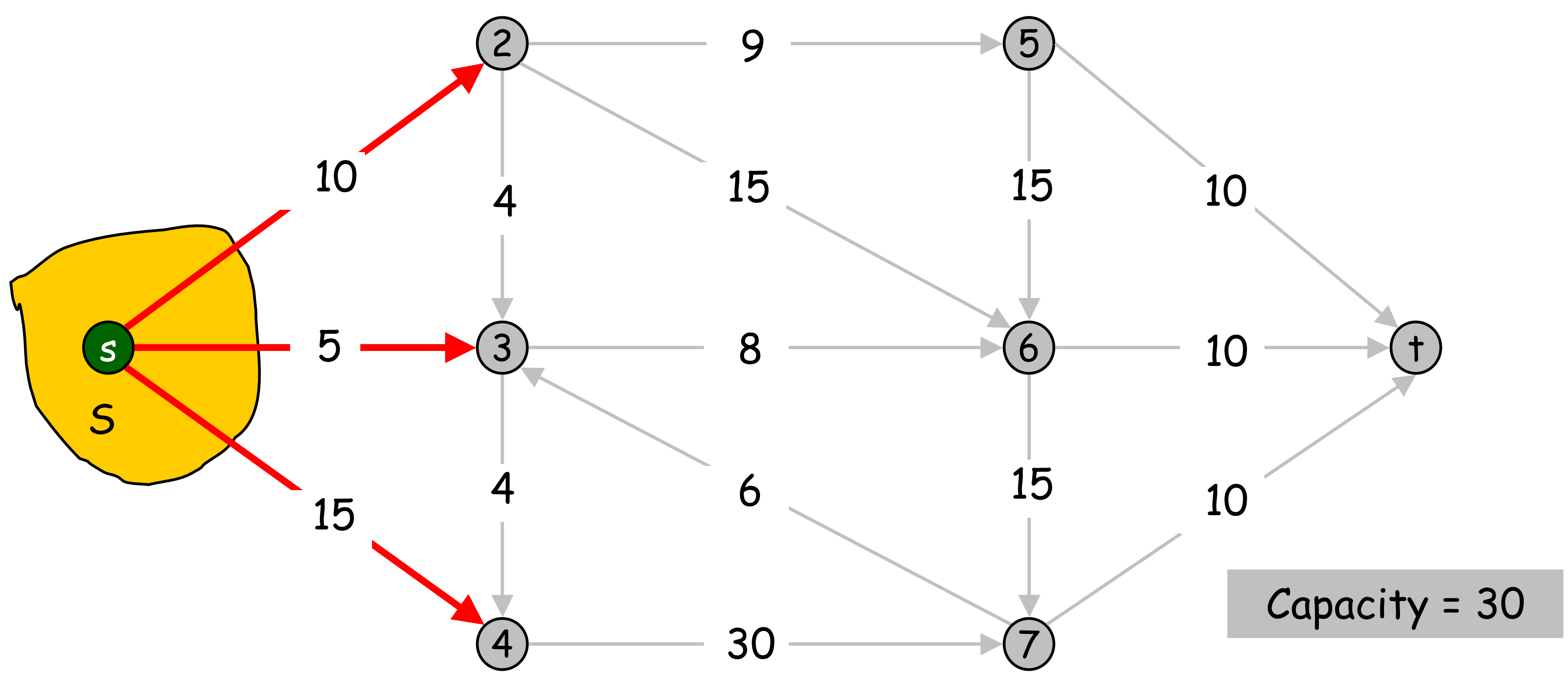
Min cut problem. Delete "best" set of edges to disconnect t from s.



Cuts

A cut is a node partition (S, T) such that s is in S and t is in T.

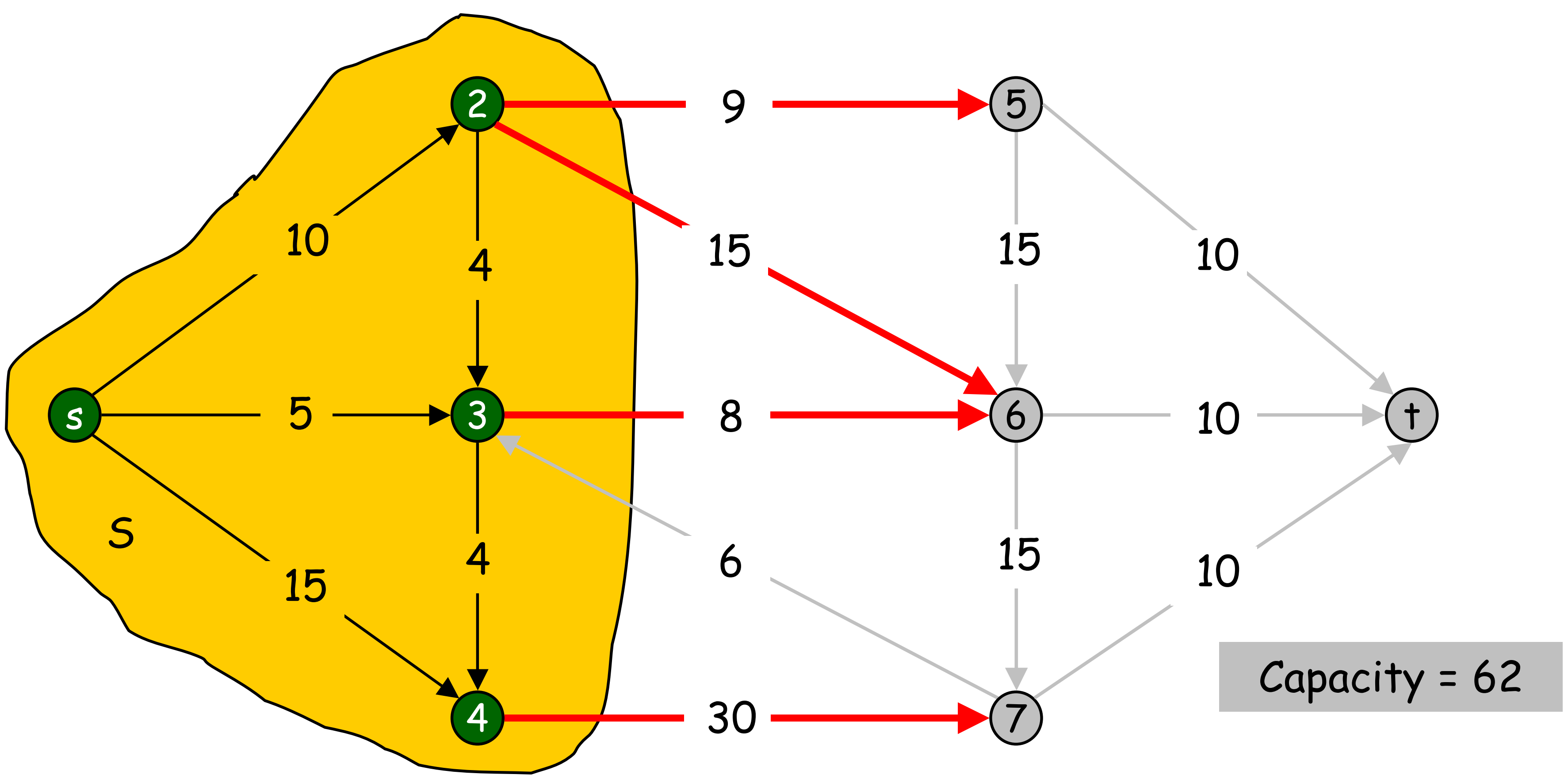
- capacity(S, T) = sum of weights of edges leaving S.



Cuts

A cut is a node partition (S, T) such that s is in S and t is in T.

- capacity(S, T) = sum of weights of edges leaving S.

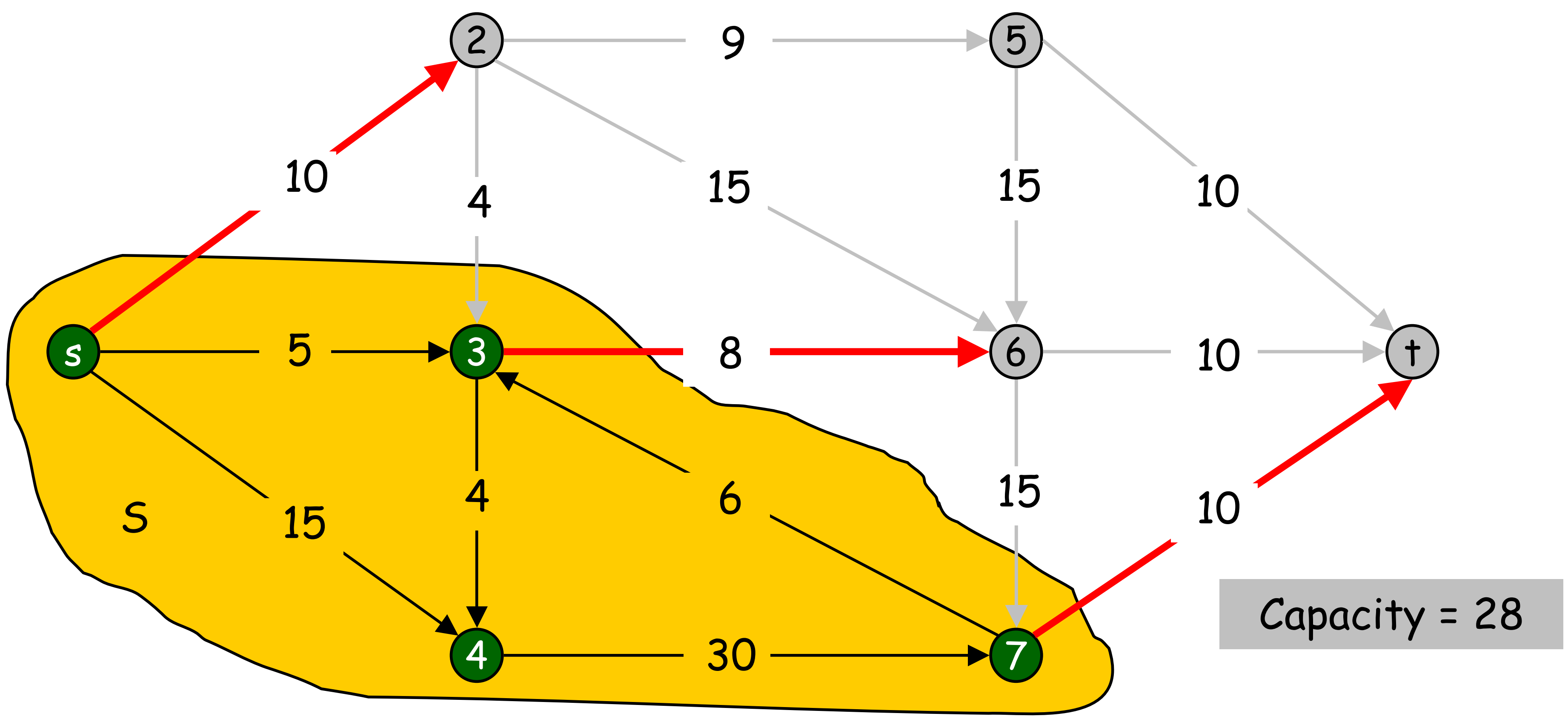


Minimum Cut Problem

A cut is a node partition (S, T) such that s is in S and t is in T.

- capacity(S, T) = sum of weights of edges leaving S.

Min cut problem. Find an s-t cut of minimum capacity.



Maximum Flow Problem

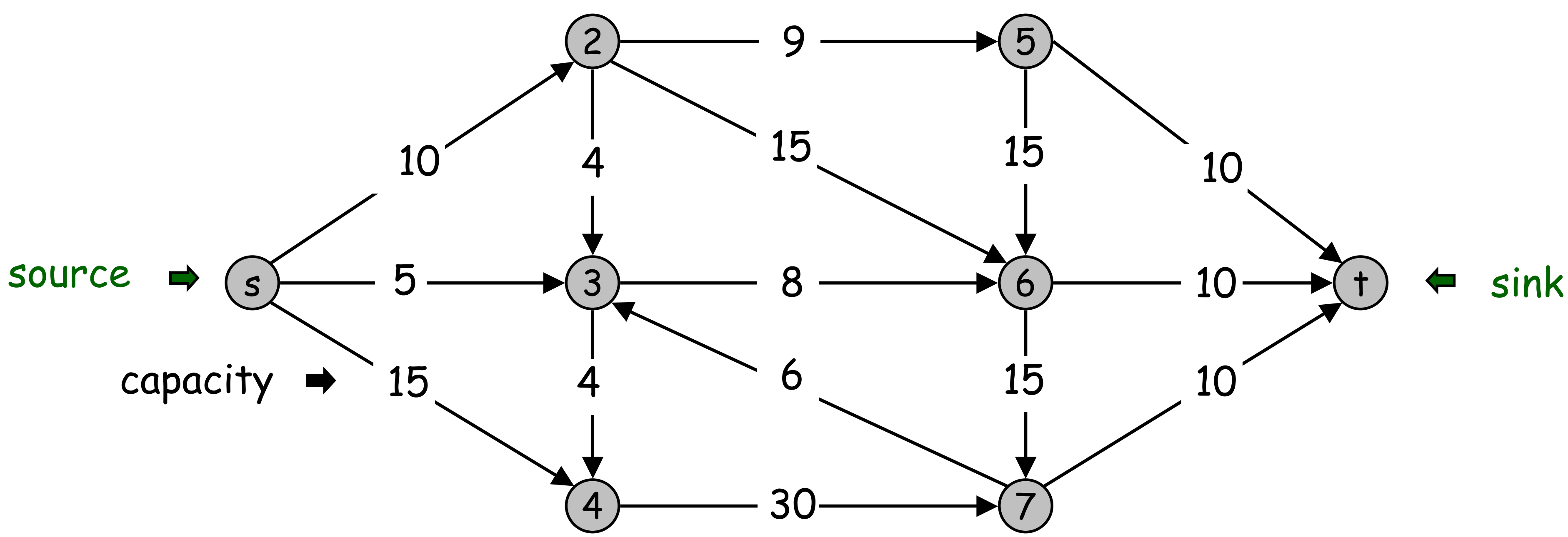
Network: abstraction for material FLOWING through the edges.

- Directed graph.
- Capacities on edges.
- Source node s, sink node t.

same input as min cut problem

Max flow problem. Assign flow to edges so as to:

- Equalize inflow and outflow at every intermediate vertex.
- Maximize flow sent from s to t.

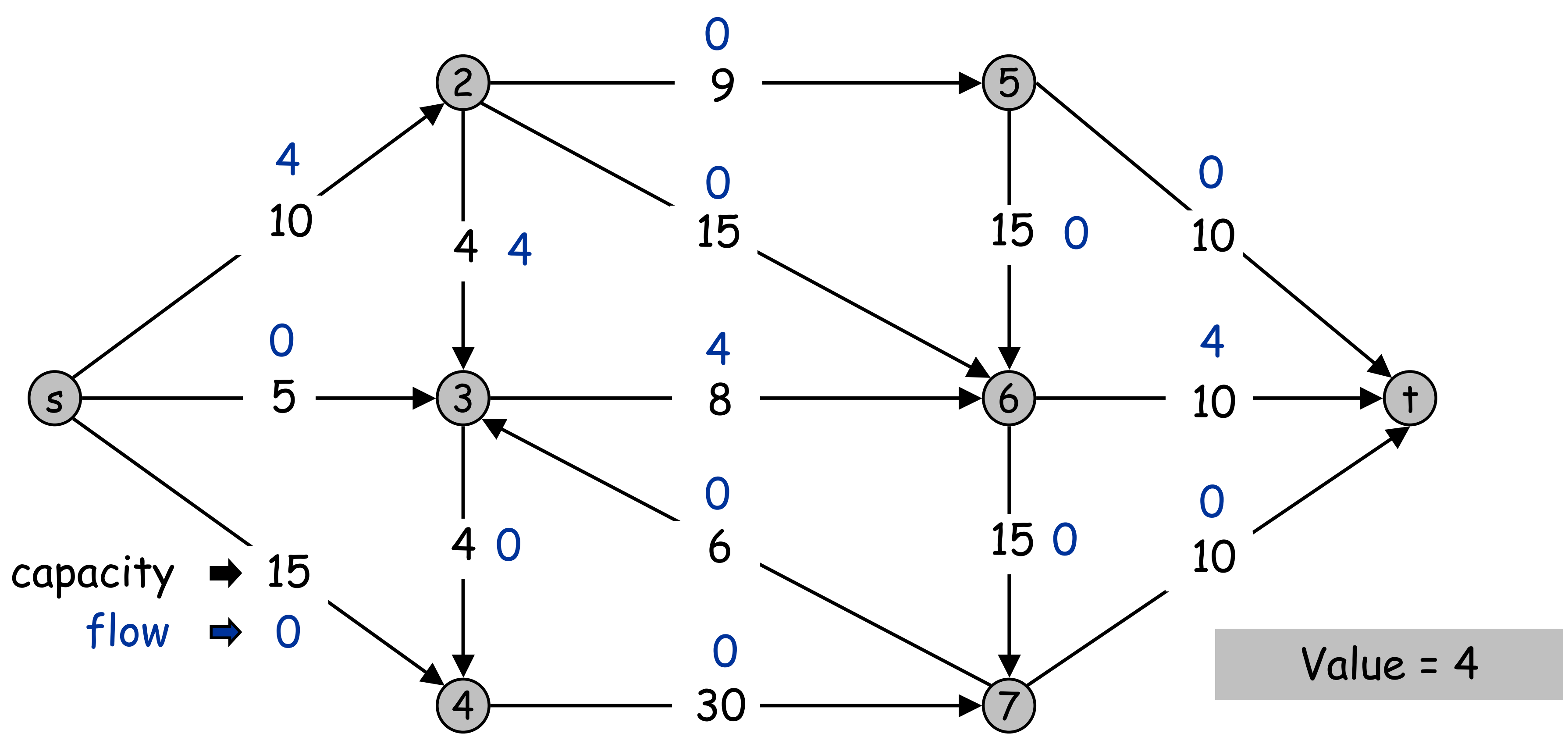


Flows

A flow f is an assignment of weights to edges so that:

- Capacity: $0 \leq f(e) \leq u(e)$.
- Flow conservation: flow leaving v = flow entering v.

except at s or t

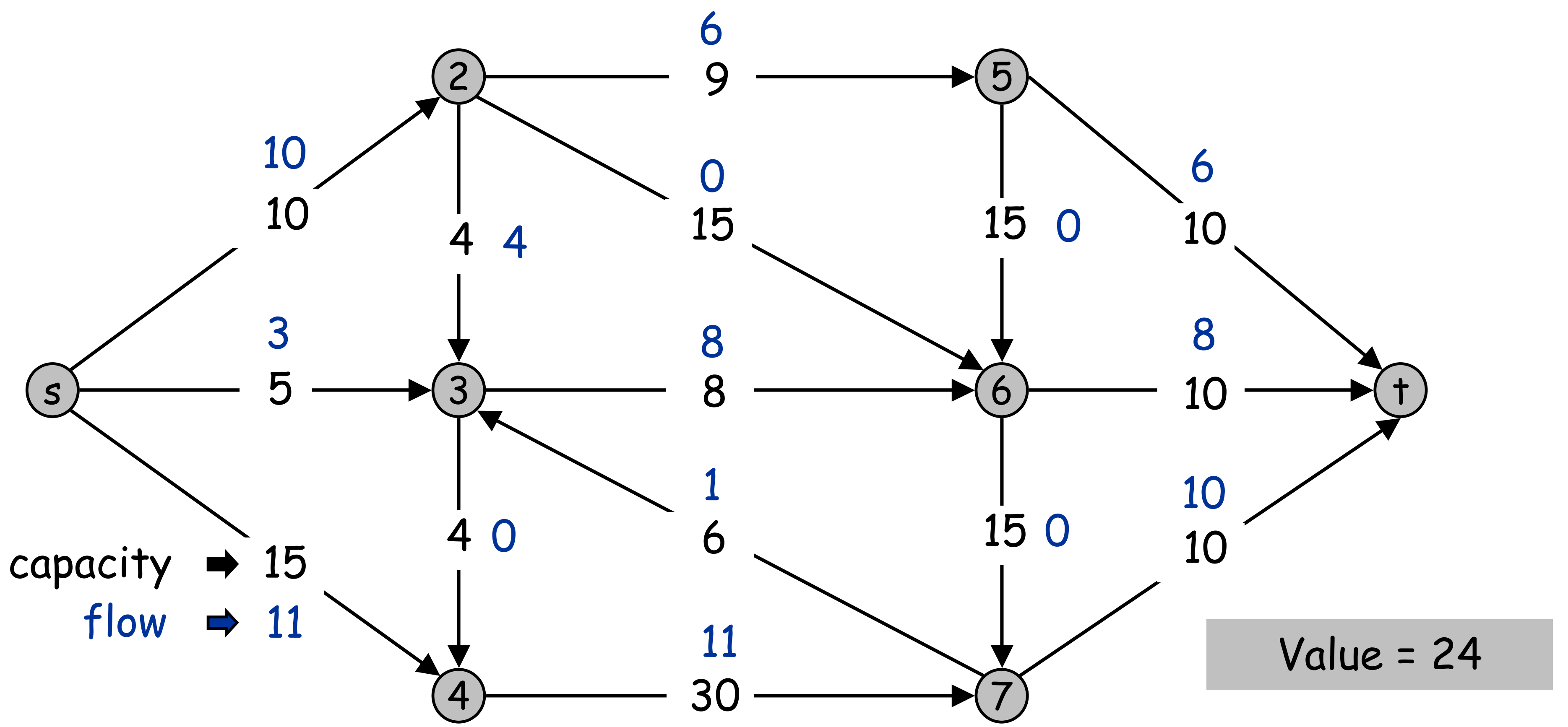


Flows

A flow f is an assignment of weights to edges so that:

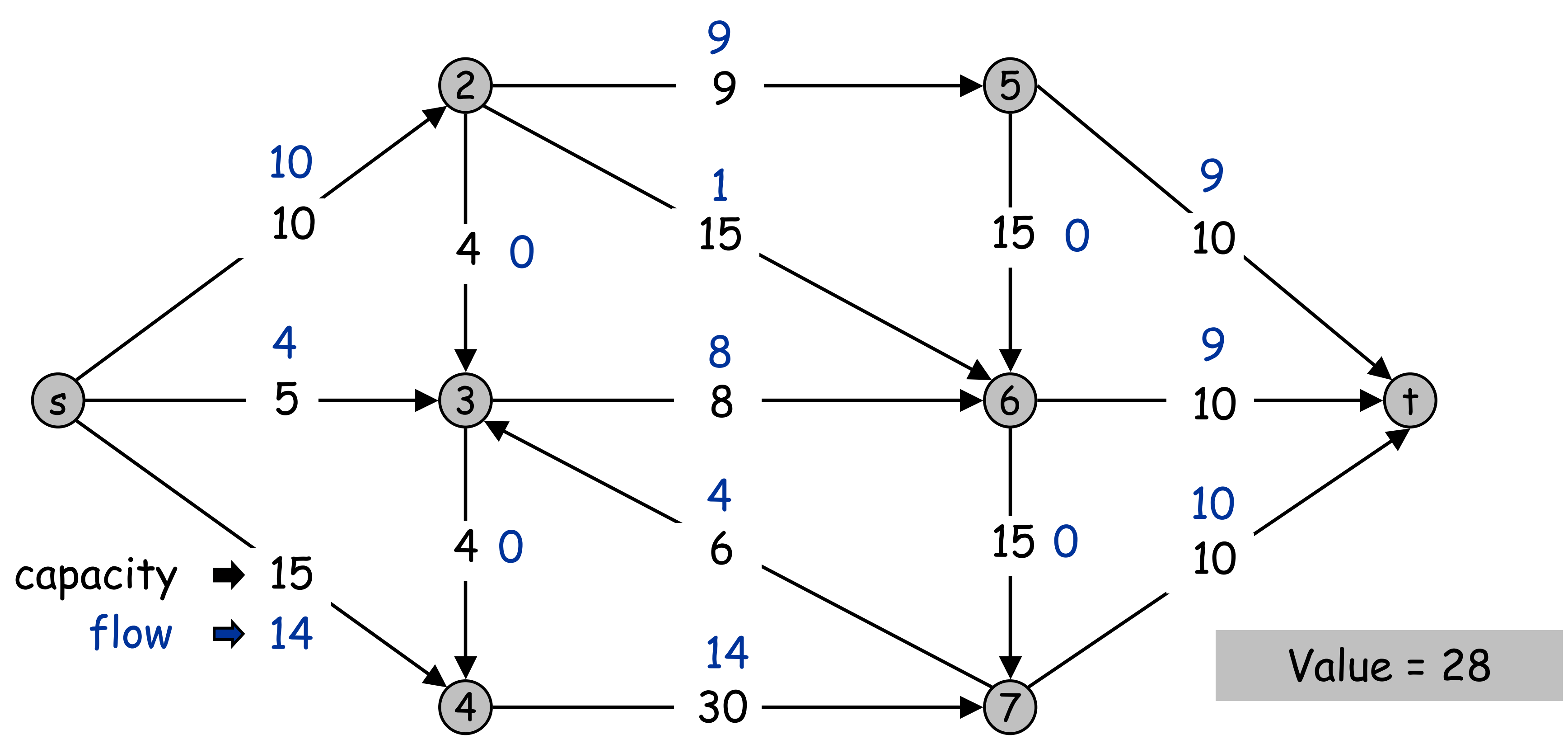
- Capacity: $0 \leq f(e) \leq u(e)$.
- Flow conservation: flow leaving v = flow entering v.

except at s or t



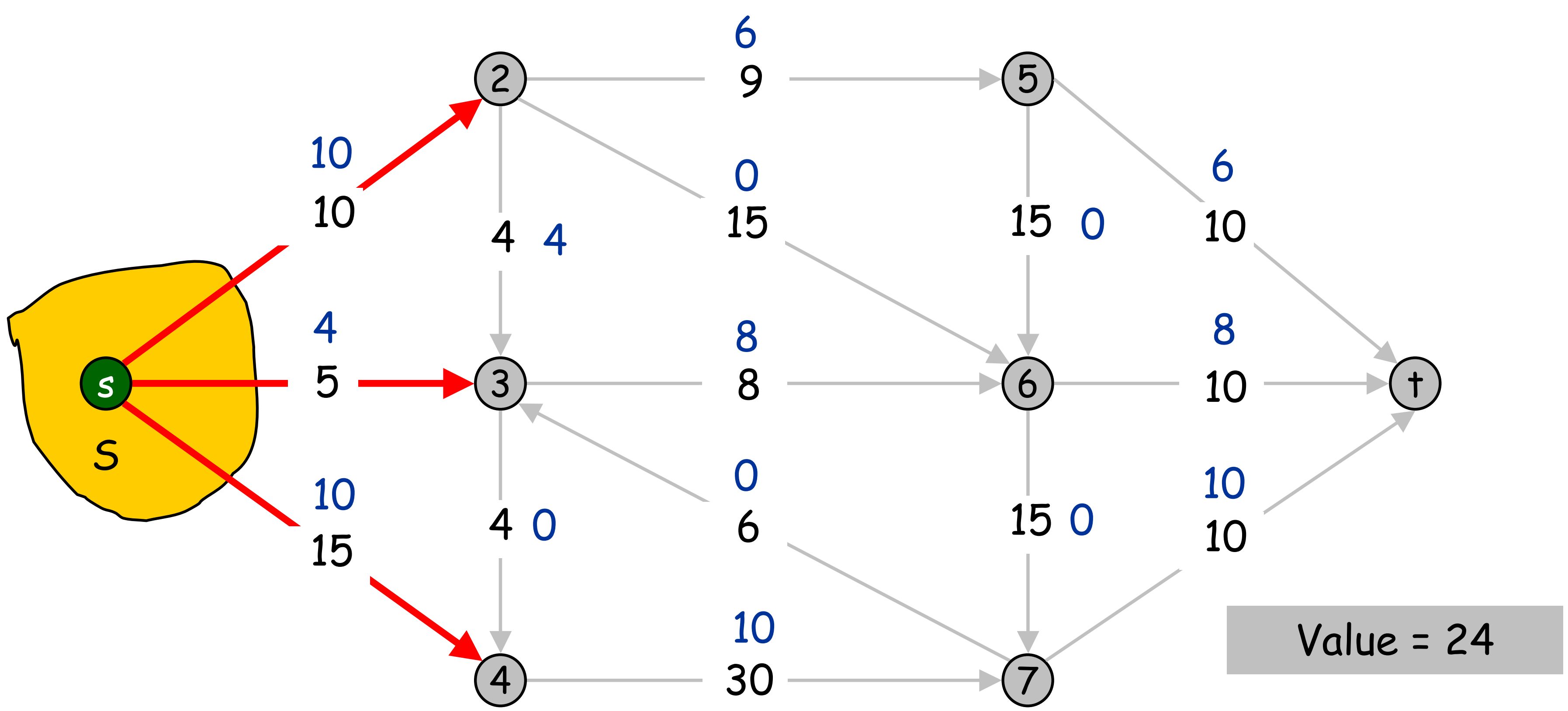
Maximum Flow Problem

Max flow problem: find flow that maximizes net flow into sink.



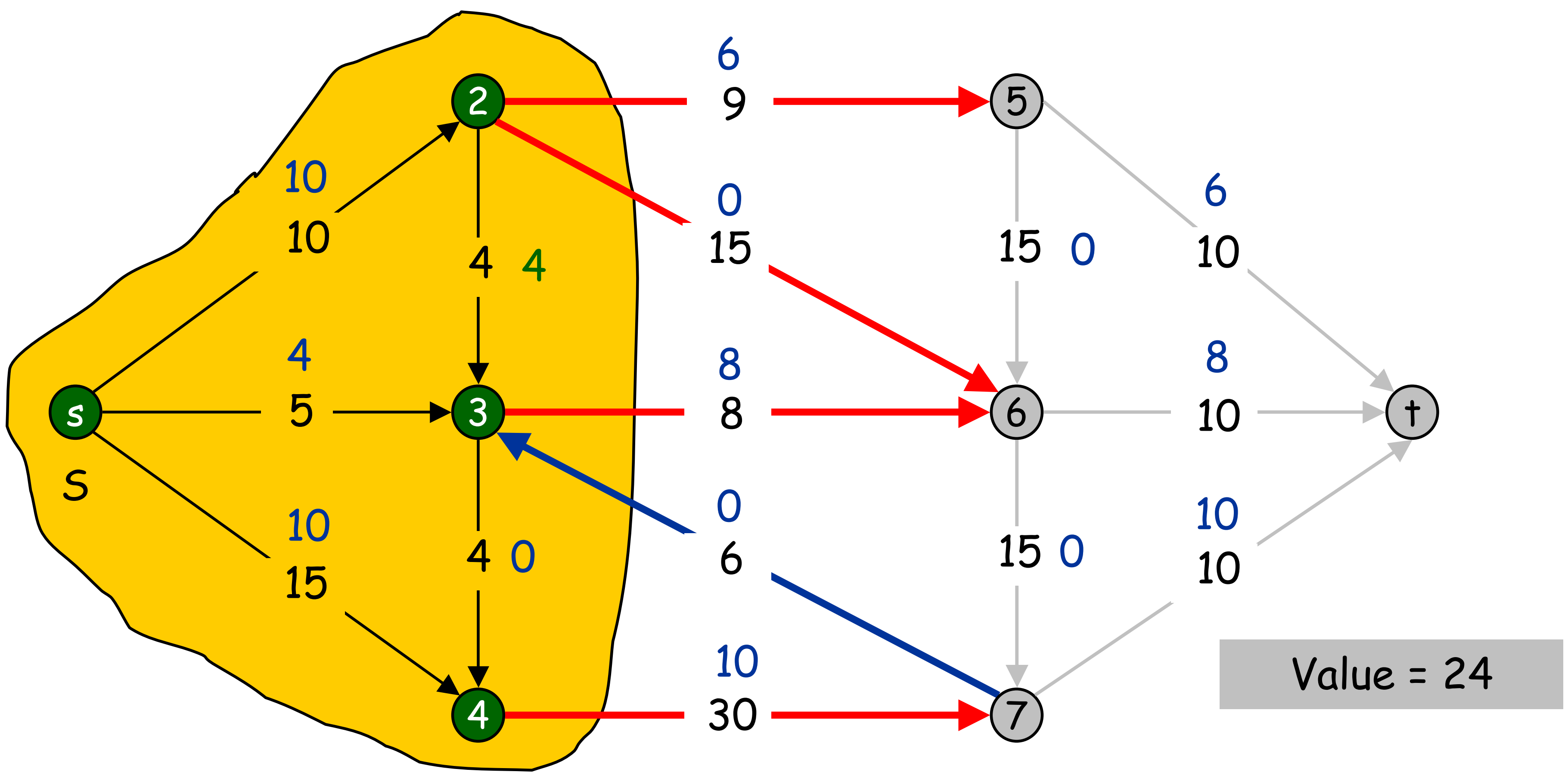
Flows and Cuts

Observation 1. Let f be a flow, and let (S, T) be any s-t cut. Then, the net flow sent across the cut is equal to the amount reaching t.



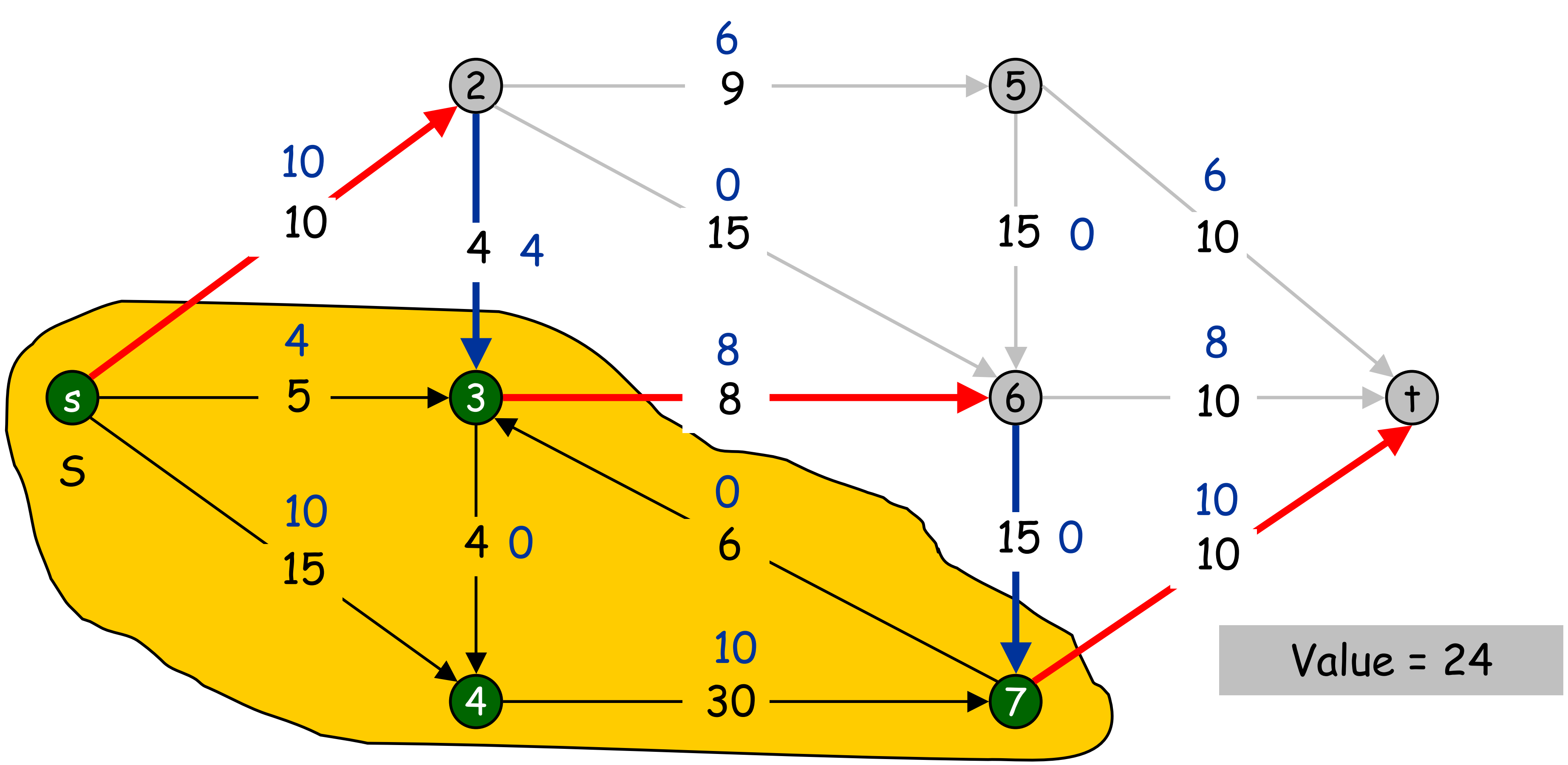
Flows and Cuts

Observation 1. Let f be a flow, and let (S, T) be any s-t cut. Then, the net flow sent across the cut is equal to the amount reaching t.



Flows and Cuts

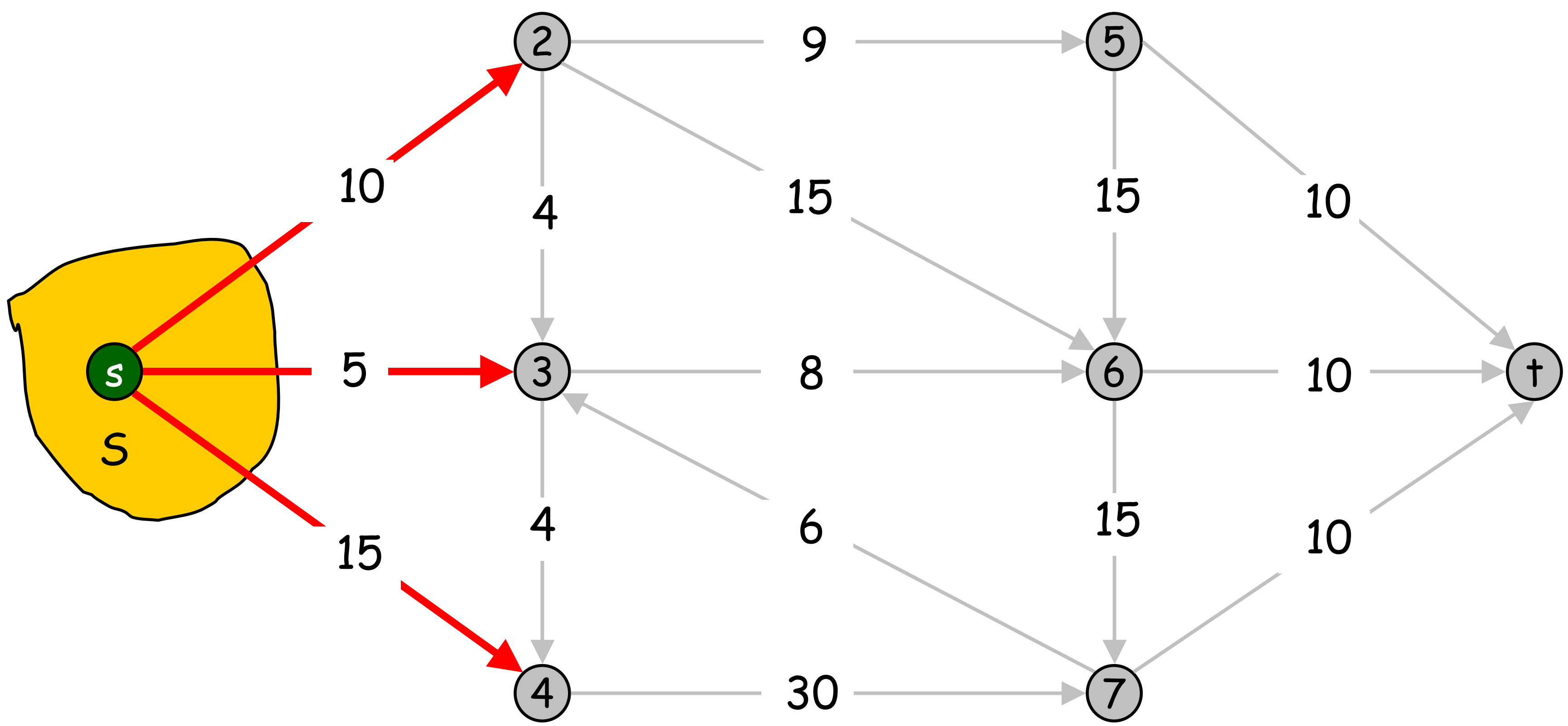
Observation 1. Let f be a flow, and let (S, T) be any s-t cut. Then, the net flow sent across the cut is equal to the amount reaching t.



Flows and Cuts

Observation 2. Let f be a flow, and let (S, T) be any s-t cut. Then the value of the flow is at most the capacity of the cut.

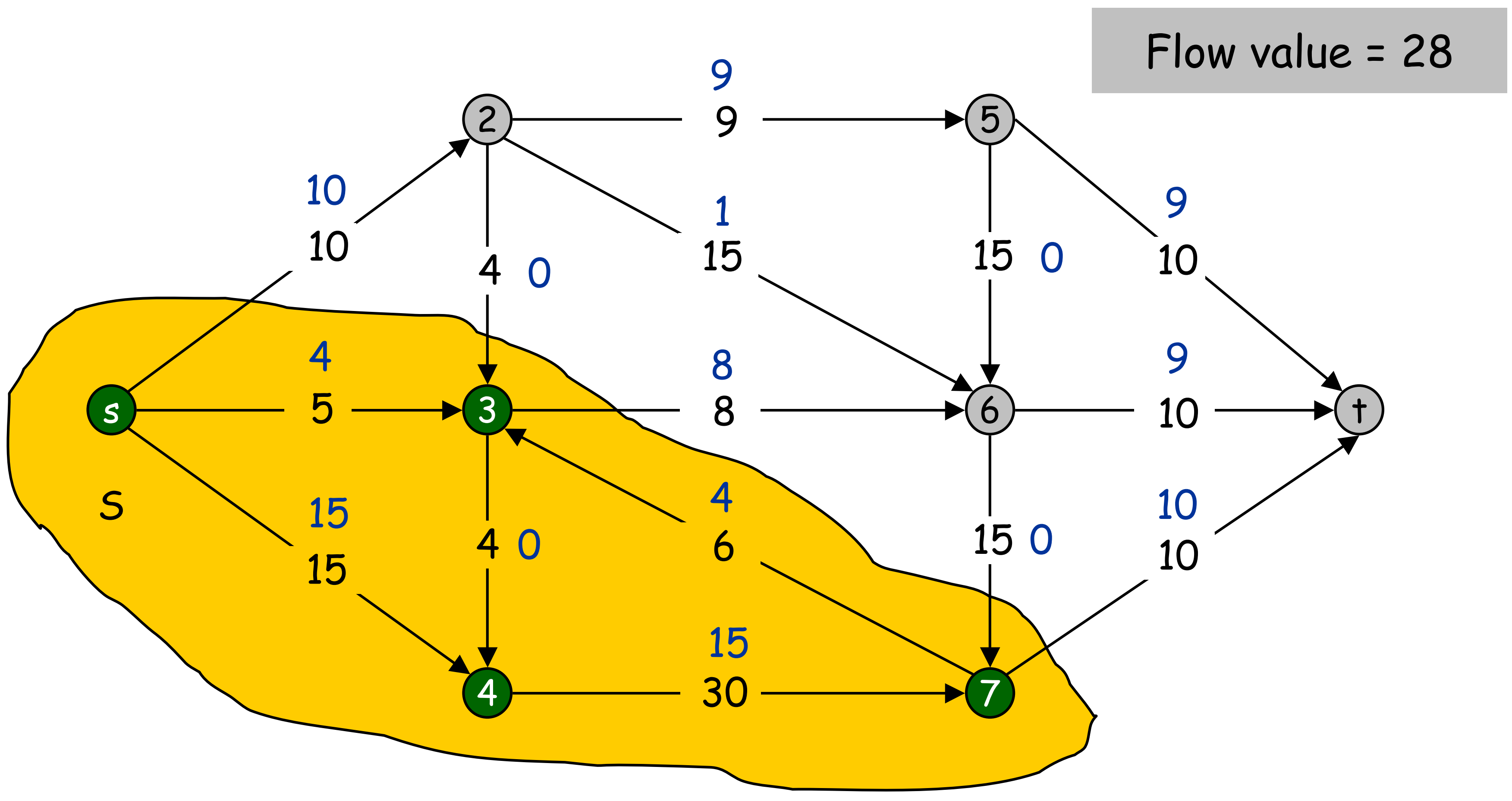
Cut capacity = 30 $\Rightarrow$ Flow value $\leq$ 30



Max Flow and Min Cut

Observation 3. Let f be a flow, and let (S, T) be an s-t cut whose capacity equals the value of f. Then f is a max flow and (S, T) is a min cut.

Cut capacity = 28 ⇒ Flow value ≤ 28

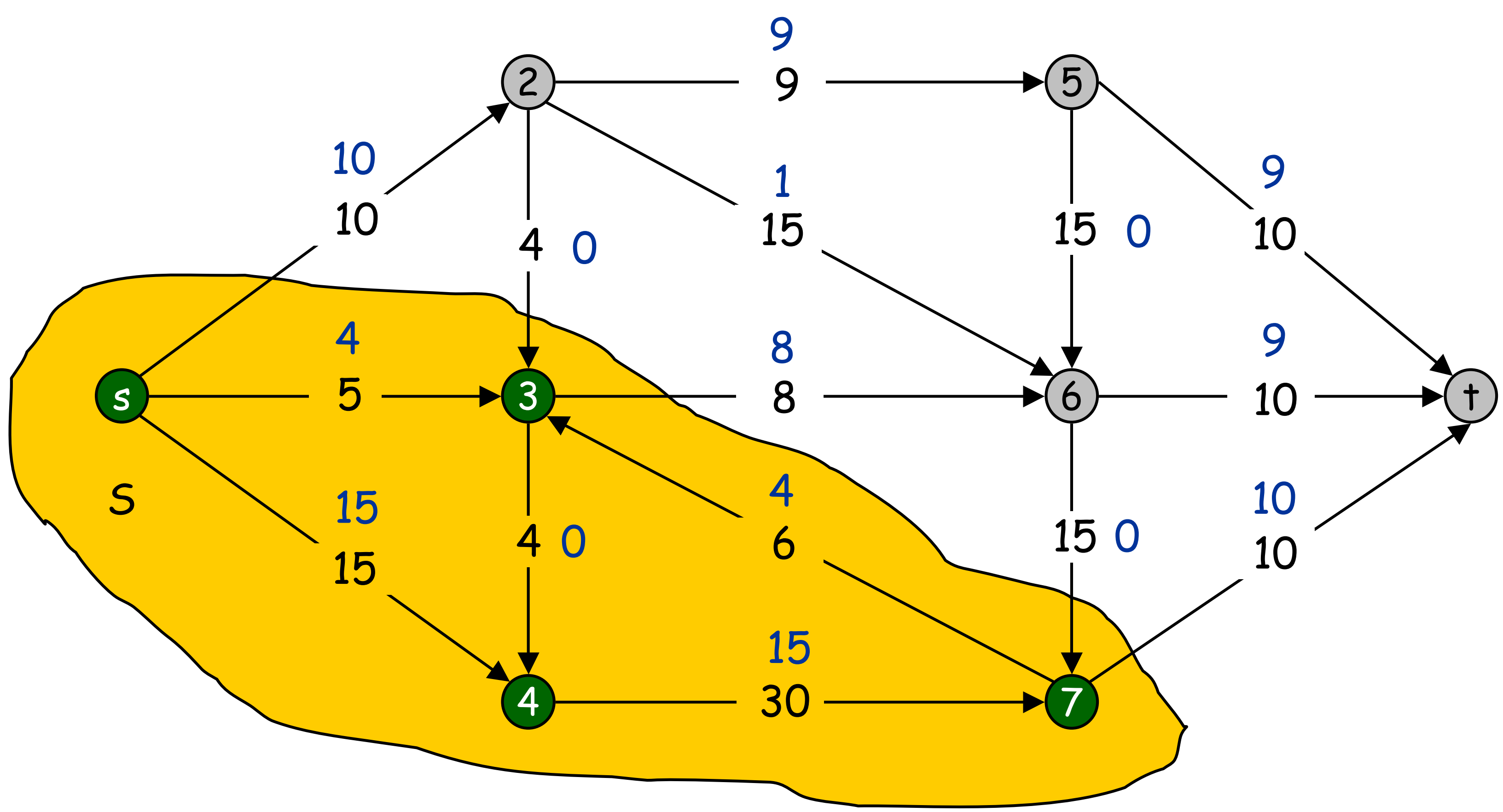


Max-Flow Min-Cut Theorem

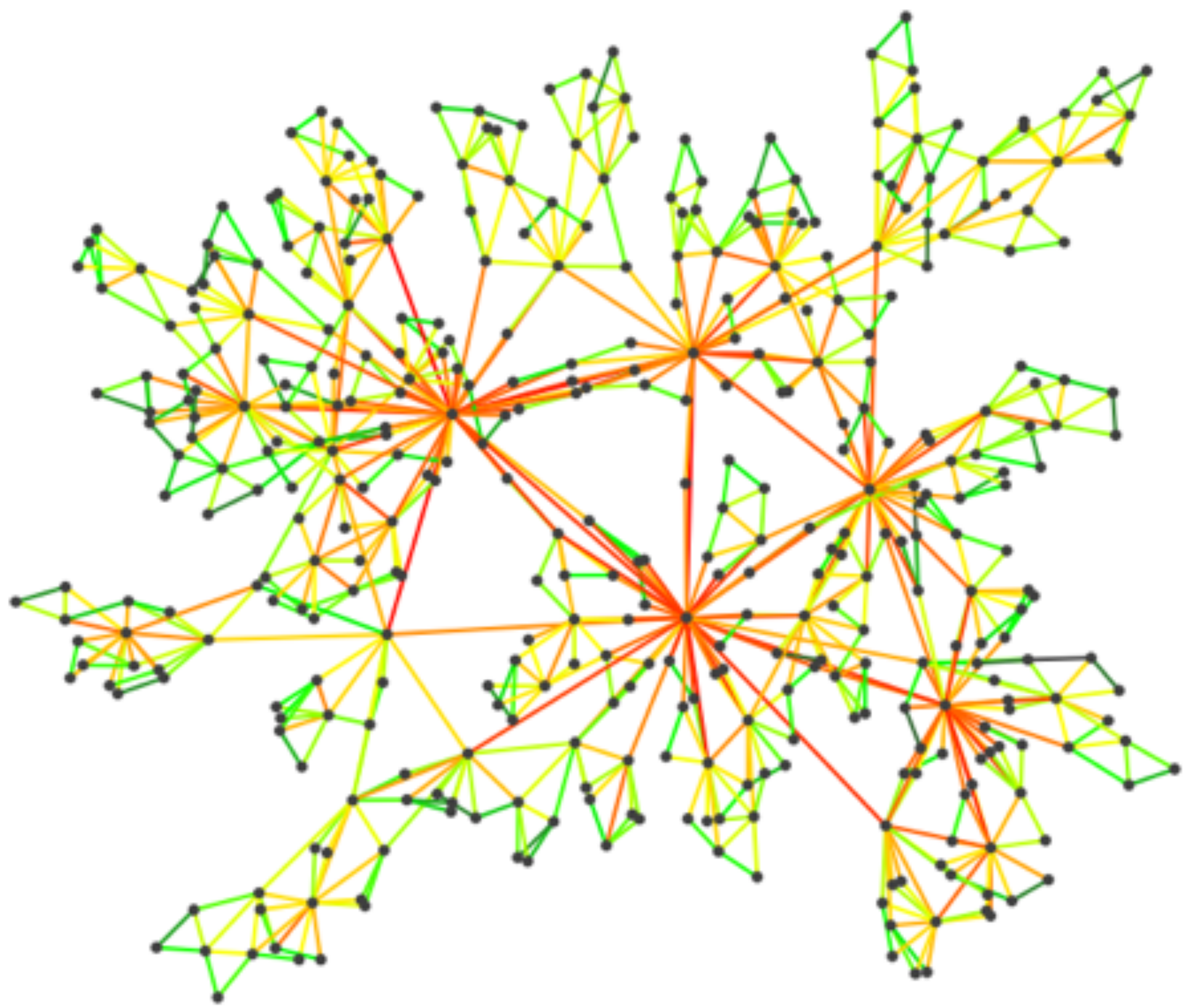
Max-flow min-cut theorem. (Ford-Fulkerson, 1956): In any network, the value of max flow equals capacity of min cut.

- Proof IOU: we find flow and cut such that Observation 3 applies.

Min cut capacity = 28 ⇔ Max flow value = 28



Random transport



RANDOM WALKS ON GRAPHS: A SURVEY

L. LOVÁSZ

Combinatorics,
Paul Erdős is Eighty (Volume 2)
Keszthely (Hungary), 1993, pp. 1–46.

Let $G = (V, E)$ be a connected graph with n nodes and m edges. Consider a *random walk* on G: we start at a node v_0; if at the t-th step we are at a node v_t, we move neighbor of v_t with probability $1/d(v_t)$. Clearly, the sequence of random nodes $(v_t : t = 0, 1, \ldots)$ is a Markov chain. The node v_0 may be fixed, but may itself be drawn from some initial distribution P_0. We denote by P_t the distribution of v_t:

$$P_t(i) = \text{Prob}(v_t = i).$$

We denote by $M = (p_{ij})_{i,j \in V}$ the matrix of transition probabilities of this Markov chain. So

$$p_{ij} = \begin{cases} 1/d(i), & \text{if } ij \in E, \\ 0, & \text{otherwise.} \end{cases} \tag{1.1}$$

Let A_G be the adjacency matrix of G and let D denote the diagonal matrix with $(D)_{ii} = 1/d(i)$, then $M = DA_G$. If G is d-regular, then $M = (1/d)A_G$. The rule of the walk can be expressed by the simple equation

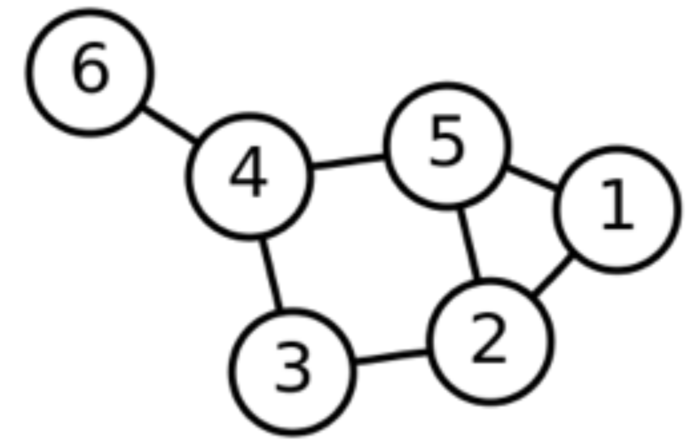
$$P_{t+1} = M^T P_t,$$

(the distribution of the t-th point is viewed as a vector in $\mathbb{R}^V$), and hence

$$P_t = (M^T)^t P_0.$$

It follows that the probability p_{ij}^t that, starting at i, we reach j in t steps is given by the ij-entry of the matrix M^t.

$$\pi(v) = \frac{d(v)}{2m}$$



$$D^{-1} = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$M = DA$$

Combinatorics,
Paul Erdős is Eighty (Volume 2)
Keszthely (Hungary), 1993, pp. 1–46.

2. Main parameters

We now formally introduce the measures of a random walk that play the most important role in the quantitative theory of random walks, already mentioned in the introduction.

(a) The *access time* or *hitting time* H_{ij} is the expected number of steps before node j is visited, starting from node i. The sum

$$\kappa(i,j) = H(i,j) + H(j,i)$$

is called the *commute time:* this is the expected number of steps in a random walk starting at i, before node j is visited and then node i is reached again. There is also a way to express access times in terms of commute times, due to Tetali [63]:

$$H(i,j) = \frac{1}{2}\left(\kappa(i,j) + \sum_u \pi(u)[\kappa(u,j) - \kappa(u,i)]\right). \qquad (2.1)$$

This formula can be proved using either eigenvalues or the electrical resistance formulas (sections 3 and 4).

(b) The *cover time* (starting from a given distribution) is the expected number of steps to reach every node. If no starting node (starting distribution) is specified, we mean the worst case, i.e., the node from which the cover time is maximum.

(c) The *mixing rate* is a measure of how fast the random walk converges to its limiting distribution. This can be defined as follows. If the graph is non-bipartite, then $p_{ij}^{(t)} \to d_j/(2m)$ as $t \to \infty$, and the mixing rate is

$$\mu = \limsup_{t\to\infty} \max_{i,j} \left| p_{ij}^{(t)} - \frac{d_j}{2m} \right|^{1/t}.$$

(For a bipartite graph with bipartition $\{V_1, V_2\}$, the distribution of v_t oscillates between "almost proportional to the degrees on V_1'' and "almost proportional to the degrees on V_2''. The results for bipartit graphs are similar, just a bit more complicated to state, so we ignor this case.)

Combinatorics,
Paul Erdős is Eighty (Volume 2)
Keszthely (Hungary), 1993, pp. 1–46.

Example 2. As another example, let us determine the access times and cover times for a complete graph on nodes $\{0, \ldots, n-1\}$. Here of course we may assume that we start from 0, and to find the access times, it suffices to determine $H(0,1)$. The probability that we first reach node 1 in the t-th step is clearly $\left(\frac{n-2}{n-1}\right)^{t-1} \frac{1}{n-1}$, and so the expected time this happens is

$$H(0,1) = \sum_{t=1}^{\infty} t \left(\frac{n-2}{n-1}\right)^{t-1} \frac{1}{n-1} = n-1.$$

The cover time for the complete graph is a little more interesting, and is closely related to the so-called Coupon Collector Problem (if you want to collect each of n different coupons, and you get every day a random coupon in the mail, how long do you have to wait?). Let τ_i denote the first time when i vertices have been visited. So $\tau_1 = 0 < \tau_2 = 1 < \tau_3 < \ldots < \tau_n$. Now $\tau_{i+1} - \tau_i$ is the number of steps while we wait for a new vertex to occur — an event with probability $(n-i)/(n-1)$, independently of the previous steps. Hence

$$E(\tau_{i-1} - \tau_i) = \frac{n-1}{n-i},$$

and so the cover time is

$$E(\tau_n) = \sum_{i=1}^{n-1} E(\tau_{i+1} - \tau_i) = \sum_{i=1}^{n-1} \frac{n-1}{n-i} \approx n \log n.$$