Max-flow problems

G. Ferrari Trecate

Dipartimento di Ingegneria Industriale e dell'Informazione
Università degli Studi di Pavia

Industrial Automation

Introduction

Flow problem: model the transport of a product from a source to a destination in presence of transfer constraints

Def. A flow network is a digraph $G=(V,E)$ where each edge $(i,j)\in E$ has a maximal capacity $\kappa(i,j)\geq 0$. Moreover, a source node s and a destination node t are specified.

Standing assumption: $\delta^-(s)=\delta^+(t)=\emptyset$

Flows

Def. A function $x : E \to \mathbb{R}$ is a **flow**. A flow is **feasible** if

$$0 \le x(i,j) \le \kappa(i,j) \quad \forall (i,j) \in E \qquad (V1)$$

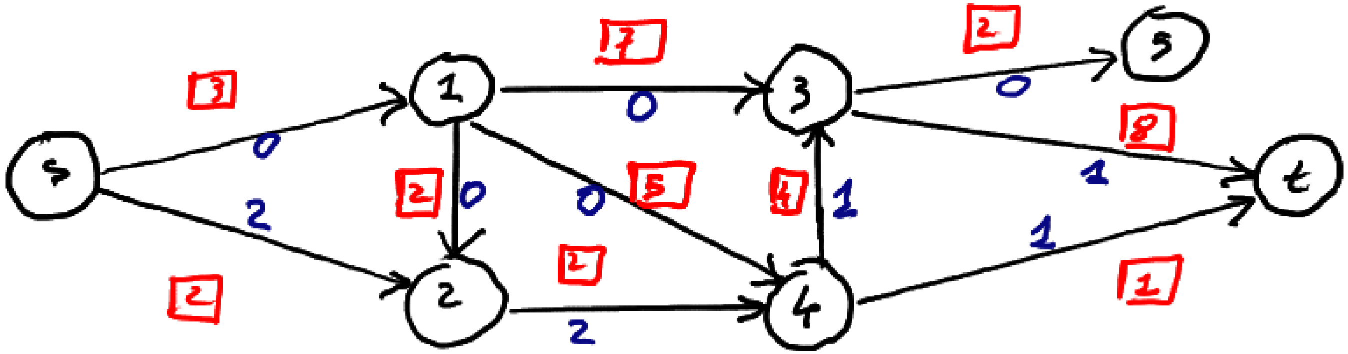
$$b(h,x) = 0 \quad \forall h \in V \setminus \{s,t\} \qquad (V2)$$

where
$$b(h,x) = \underbrace{\sum_{(h,j) \in \delta^+(h)} x(h,j)}_{\text{flow leaving } h} - \underbrace{\sum_{(i,h) \in \delta^-(h)} x(i,h)}_{\text{flow entering } h}$$

Rmk. (V1): capacity constraints

(V2): conservation law in all nodes except s (usually $b(s,x) \ge 0$) and t (usually $b(t,x) \le 0$)

Example : subway



Vertices : districts

$\boxed{K(i,j)}$: max n° of passengers that can transit from i to j in a day

$x(i,j)$: passengers moved in a day

Is the flow feasible ?

Is it maximal ?

NO. Setting $x(s,1) = 3$, $x(1,3) = 3$ and $x(3,t) = 4$, one carries 5 passengers from s to t.

Rmk. Conservation constraints ⇒ no passenger stuck in intermediate districts

Max-flow problem

$$\max_x \{ b(s,x) : x \text{ is a feasible flow} \}$$

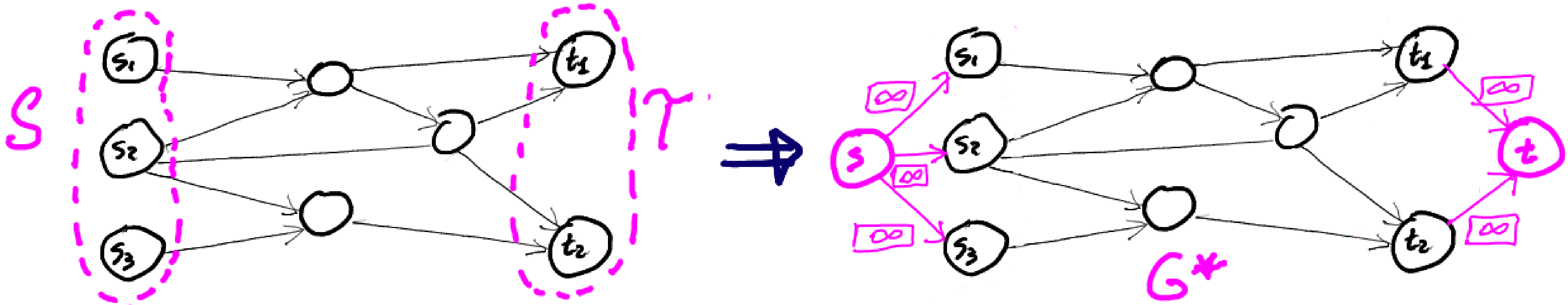
Rmk. • $\varphi_0 = b(s,x)$: flow value

• One has $b(t,x) = -\varphi_0$ (because of conservation constraints)

• Max-flow is an LP problem

Transformations for getting flow problems

Multiple source nodes (without incoming arcs) and multiple destination nodes (without outgoing arcs)



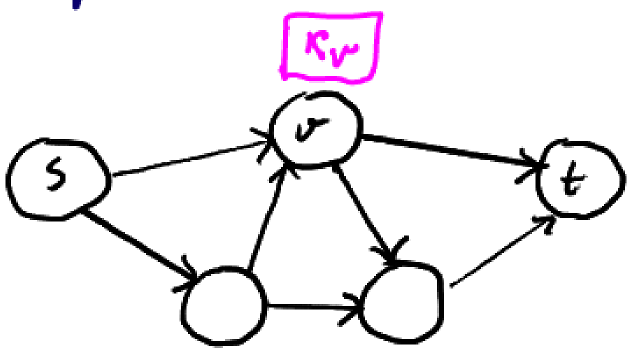
- Add fake source and destination nodes. Add fake arcs from s to all $v \in S$ and from each $v \in T$ to t with infinite capacity
  ↳ Solving the max-flow problem on G^* is equivalent to maximize the total flow from all nodes in S to all nodes in T.

Intermediate vertices with bounded capacity

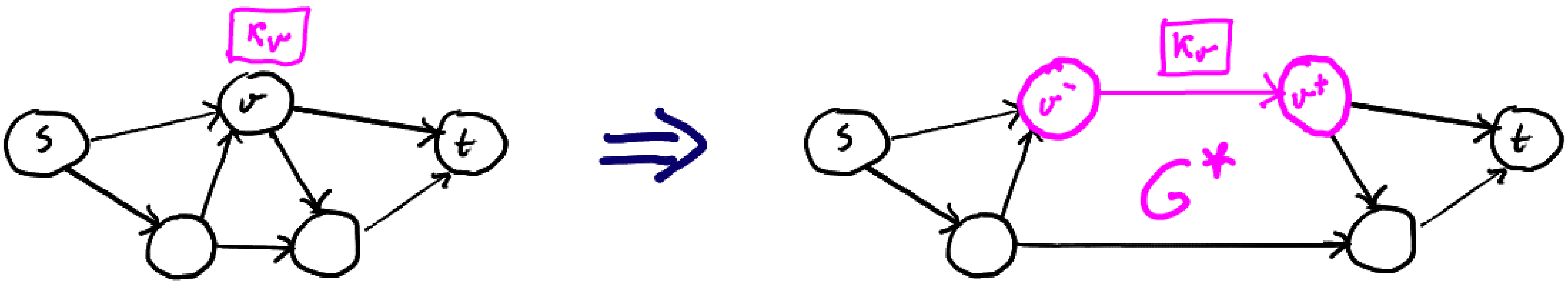
To a vertex v with maximal capacity κ_v corresponds the constraint

$$\sum_{(i,v)\in\delta^-(v)} x(i,v) = \sum_{(v,s)\in\delta^+(v)} x(v,s) \leq \kappa_v \qquad (\forall v)$$

Example



Idea:
1) Replace v with vertices v^+ and v^-
2) Replace all edges (i,v) with edges (i,v^-). Replace all edges (v,s) with edges (v^+,s)
3) Add an edge (v^-,v^+) with capacity κ_v



Every feasible flow in G^* verifies (VN) by construction

Properties of feasible flows

Def. Let $G = (V, E, \kappa)$ be a flow network. A cut of G is a partition $(S, V\setminus S)$ of V such that $s \in S$ and $t \in V\setminus S$.

Def. Let x be a feasible flow. The flow through the cut $(S, V\setminus S)$ is

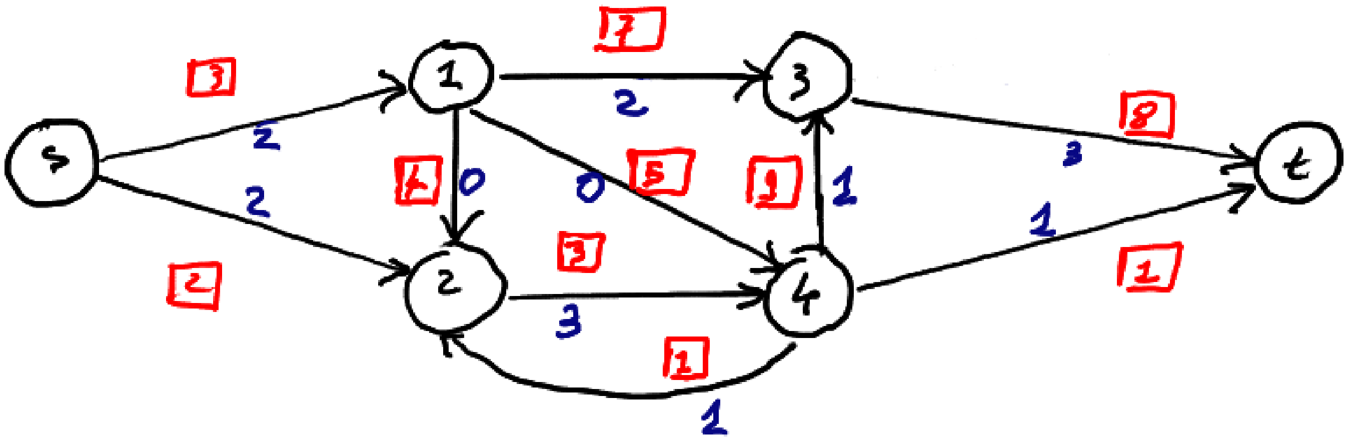
$$\varphi(S) = \sum_{(i,j)\in\delta^+(S)} x(i,j) - \sum_{(i,j)\in\delta^-(S)} x(i,j)$$

Rmk. The flow value verifies $\varphi_0 = \varphi(\{s\})$

Def. The capacity of the cut $(S, V\setminus S)$ is

$$\kappa(S) = \sum_{(i,j)\in\delta^+(S)} \kappa(i,j) \longrightarrow \text{outgoing capacity}$$

Example



$$S = \{s, 1, 2\} \longrightarrow \delta^+(S) = \{(1,3), (1,4), (2,4)\}$$

$$\delta^-(S) = \{(4,2)\}$$

$$\varphi(S) = (2+0+3) - 1 = 4$$

$$\varphi_0 = 4$$

$$\kappa(S) = 7+5+3 = 15$$

Thm. Let x be a feasible flow. For all cuts $(S, V\setminus S)$ it holds

$$\varphi(S) = \varphi_0 \qquad (F1)$$

$$\varphi(S) \leq \kappa(S) \qquad (F2)$$

Rmk.
- (F1): all cuts have the same flow
- (F2) $\Rightarrow$ Let φ_0^* be the maximal value of a feasible flow.
  Then,
  $$\varphi_0^* \leq \kappa(S^*)$$
  where $(S^*, V\setminus S^*)$ is a cut of minimal capacity.

Open problem: how to compute a solution to max-flow?

Residual network

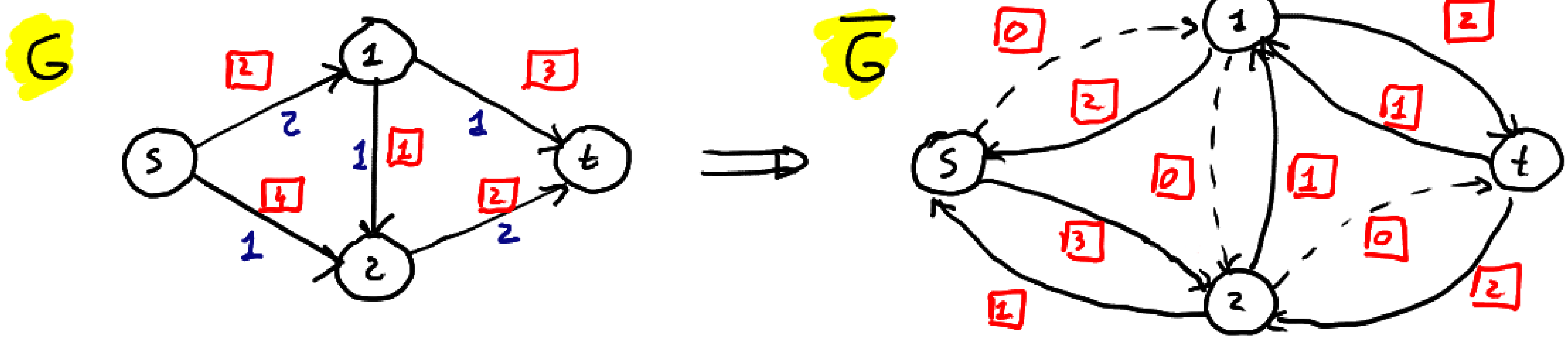
Def. Let x be a feasible flow. The corresponding residual network $\bar{G} = (\bar{V}, \bar{E}, \bar{\kappa})$ is obtained from $G = (V, E, \kappa)$ setting $\bar{V} = V$ and replacing each edge (i,j) with two edges

- a direct edge (i,j) with residual capacity $\bar{\kappa}(i,j) = \kappa(i,j) - x(i,j)$
- an inverse edge (j,i) " " " $\bar{\kappa}(j,i) = x(i,j)$

and removing edges with zero residual capacity.

Rmk. x feasible $\Rightarrow$ $\bar{\kappa}(i,j)$ and $\bar{\kappa}(j,i)$ are ≥ 0

Example



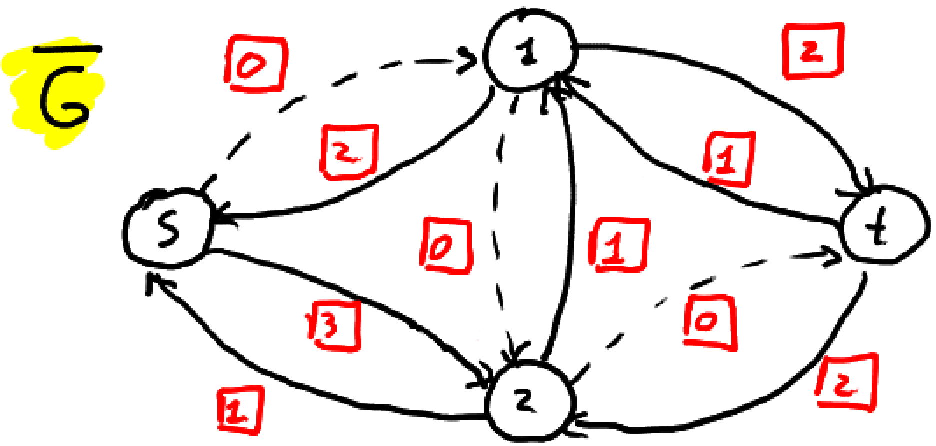
- direct edge (i,j): the flow from i to j in G can be increased of $\bar{\kappa}(i,j)$ at most
- inverse edge (j,i): the flow from i to j in G can be decreased of $\bar{\kappa}(j,i)$ at most

Def. A path from s to t in the residual network is an Augmenting Path (AP).

The existence of an AP Q means φ_0 can be increased. How much?
At most of

$$\delta = \min \{ \overrightarrow{\kappa}(u,v) : (u,v) \in Q \}$$

Previous example



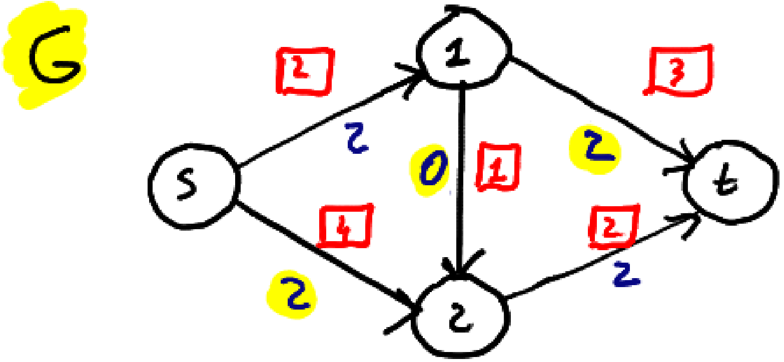
$Q = s\,2\,1\,t$ is an AP $\Rightarrow \delta = \min\{3,1,2\} = 1$

$(s,2)$: direct edge $\to$ increase $x(s,2)$ of δ

$(2,1)$: inverse edge $\to$ decrease $x(1,2)$ of δ

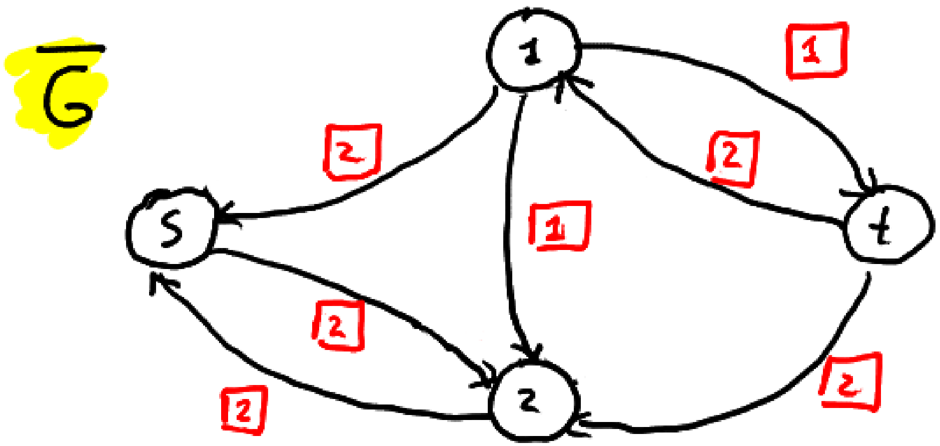
$(1,t)$: direct edge $\to$ increase $x(1,t)$ of δ

Updated feasible flow



→ Flow value $\varphi_0 = 4$

Can the flow value be further increased?



→ No AP in the new residual network... any conclusions?

Thm. A feasible flow is optimal for the max-flow problem if and only if t can not be reached from s in the residual network associated to x.

Ford-Fulkerson algorithm

Input: flow network $G=(V,E,\kappa)$

I) Init. Set $x(i,j)=0 \quad \forall (i,j)\in E, \quad \varphi_0=0$

II) Compute the residual network $\bar{G}=(\bar{V},\bar{E},\bar{\kappa})$ corresponding to x

III) Compute an AP Q. If it does not exist, STOP (φ_0 is the maximal flow value)

IV) Compute the maximal flow increment along Q

$$\delta=\min\{\bar{\kappa}(i,j):(i,j)\in Q\}$$

V) Update the flow in G according to the updated flow in Q

$$x(i,j)\leftarrow x(i,j)+\delta \quad \text{if } (i,j) \text{ is a direct edge in } Q$$

$$x(j,i)\leftarrow x(j,i)-\delta \quad \text{if } (i,j) \text{ is an inverse edge in } Q$$

$$\varphi_0\leftarrow\varphi_0+\delta$$

GO TO (II)

Computational complexity

Let $\kappa_{max} = \max\{\kappa(i,j) : (i,j) \in E\}$

Assumption: $\kappa(i,j) \in \mathbb{N}, \ \forall (i,j) \in E$

max n° of bits for coding a capacity

1) The size of an instance of max-flow is $O(m(\log_2 \kappa_{max}))$, $m = |E|$

2) At every iteration one has $x(i,j) \in \mathbb{N}$ and $\bar{\kappa}(i,j) \in \mathbb{N}$. This implies $\delta \geq 1$.

. The algorithm ends in at most φ_0^* iterations. Since

a) $\varphi_0^* \leq \kappa(\{s\}) = O(m\,\kappa_{max})$

b) An iteration takes $O(m)$ for updating the flow

the complexity of the algorithm is $O(m^2\, 2^{\log_2 \kappa_{max}})$

Rmk. Polinomial in m but exponential in the size of the instance!

Next: example on slides