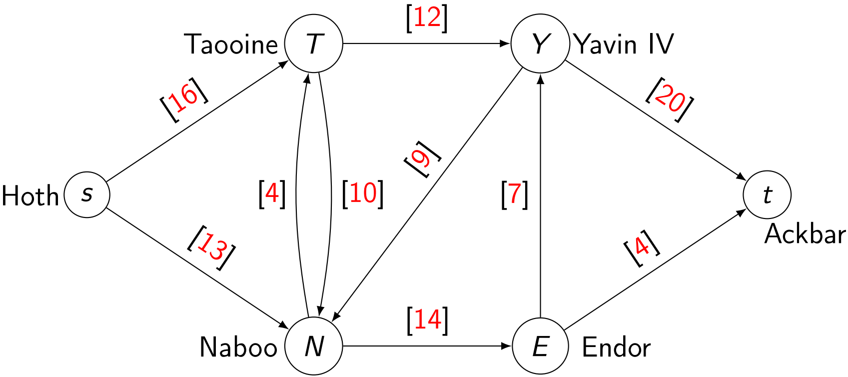
Chapter 3 : The maximum flow problem

ENSIIE - Operations Research Module

Dimitri Watel (dimitri.watel@ensiie.fr)

2022

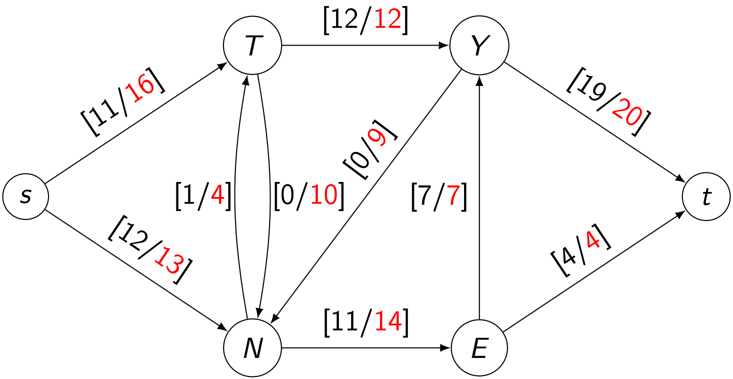
A road sizing problem



[x] : Maximum number of people on the road.

What is the max. number of people leaving s to t?

A road sizing problem



Answer : 23

Which algorithm solves this problem?

How to prove the result is optimal?

Problem formulation

Definition

A *transportation network* (also called *flow network*) is:

- a directed graph $G = (V, A)$;
- two nodes $s \in V$ (the source) and $t \in V$ (the sink);
- the capacities of the arcs $c : A \rightarrow \mathbb{R}^+$.

Problem formulation

Definition

A *transportation network* (also called *flow network*) is:

- a directed graph $G = (V, A)$;
- two nodes $s \in V$ (the source) and $t \in V$ (the sink);
- the capacities of the arcs $c : A \to \mathbb{R}^+$.

Definition

A *feasible flow* is a function f from A to $\mathbb{R}^+$ such that

- for every arc $a \in A$, $f(a) \in [0, c(a)]$ (capacity constraint)
- for every node $v \in V$ **except** s **and** t,
  $\sum_{a \in \gamma^-(v)} f(a) = \sum_{a \in \gamma^+(v)} f(a)$ (conservation constraint)

Remark

Theorem

If f is a feasible flow, then, for every subset of node $V' \subset V \backslash \{s, t\}$, the conservation constraint is satisfied:

$$\sum_{a \in \gamma^-(V')} f(a) = \sum_{a \in \gamma^+(V')} f(a).$$

(see the board for an example)

Problem formulation

Definition

The *value* v of the flow f is the flow entering the sink t, or the flow outgoing from the source s.

$$v = \sum_{a \in \gamma^-(t)} f(a) - \sum_{a \in \gamma^+(t)} f(a)$$

$$v = \sum_{a \in \gamma^+(s)} f(a) - \sum_{a \in \gamma^-(s)} f(a)$$

$$v = f_1 + f_2 - f_3 = f_4 - f_5 - f_6$$

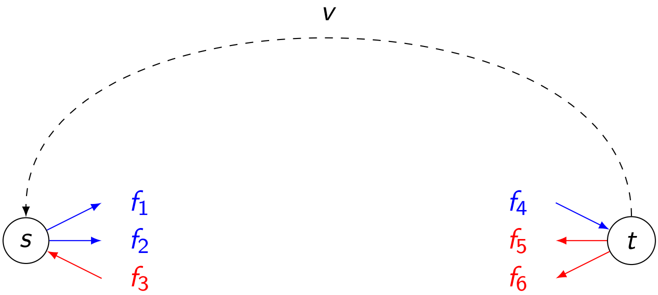
Problem formulation

The maximum flow problem

- Input : a flow network (G, s, t, c).
- Feasible solution : a feasible flow f on G (satisfies the capacity constraint and the conservation constraint)
- Optimal solution : a feasible solution f maximizing its value v.

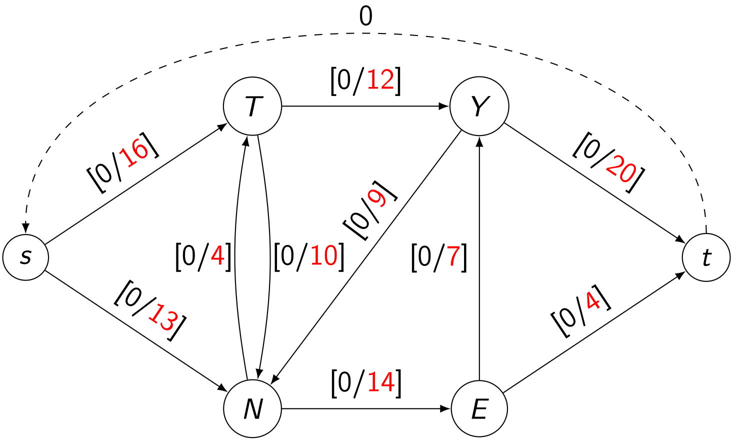
Remark

We can add a fictive arc from t to s such that the flow $f(t, s)$ always equals the value v of the flow f. In that case, the conservation constraint is also satisfied by s and t.



$$v + f_3 = f_1 + f_2$$
$$v + f_5 + f_6 = f_4$$

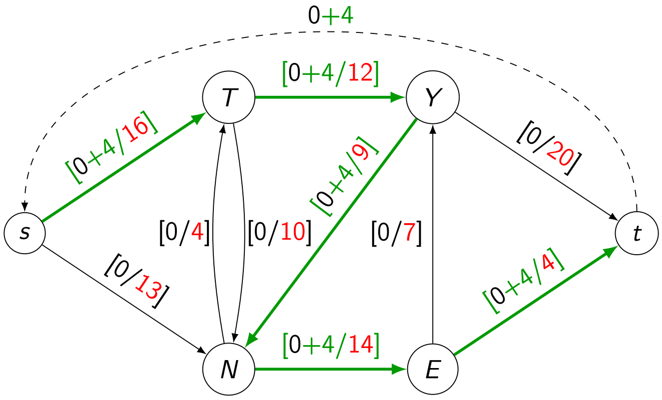
The nul flow



The nul flow is feasible.

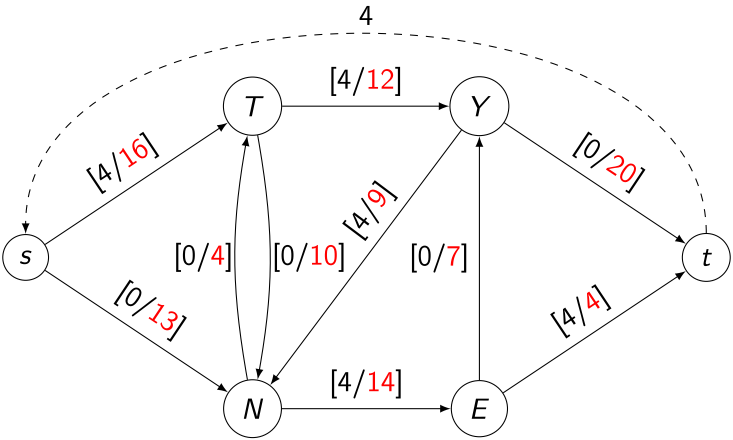
Improve the flow through a path

If we add α to every arc on a path from s to t, the value of the flow is increased by α.



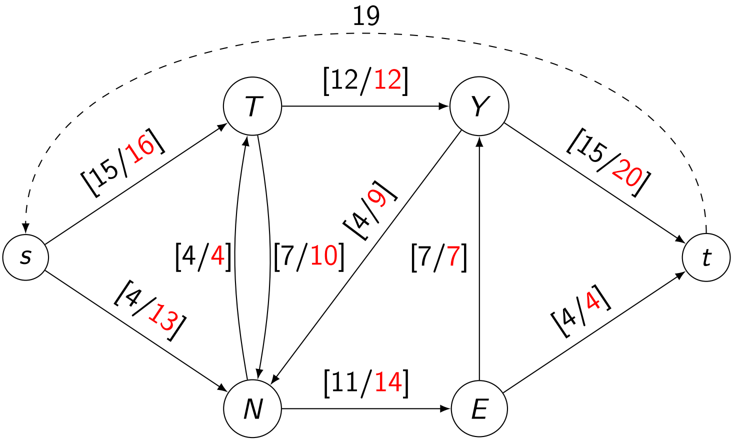
Add 4 to the flow.

Improve the flow through a path



To be continued ... on board

Improve the flow through a path



We are now stuck...

Complete flow

Definition

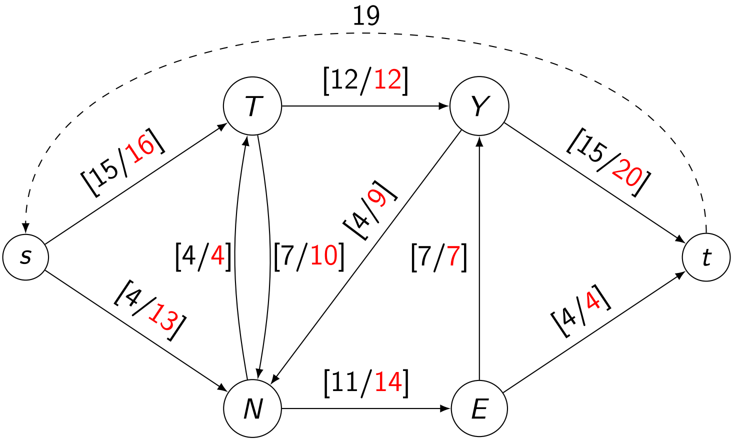
We say a flow f is *complete* if, in G, every path from s to t contains a saturated arc a, *i.e.* $f(a) = c(a)$.

WARNING

A complete flow is not necessarily a maximum flow.

Proof : according to the flow at the beggining of the course, the maximum flow is at least 23.

Improve the flow through a path



Idea : add 4 to (s, N) and (Y, t), and remove 4 to (Y, N). How to generalize this idea?

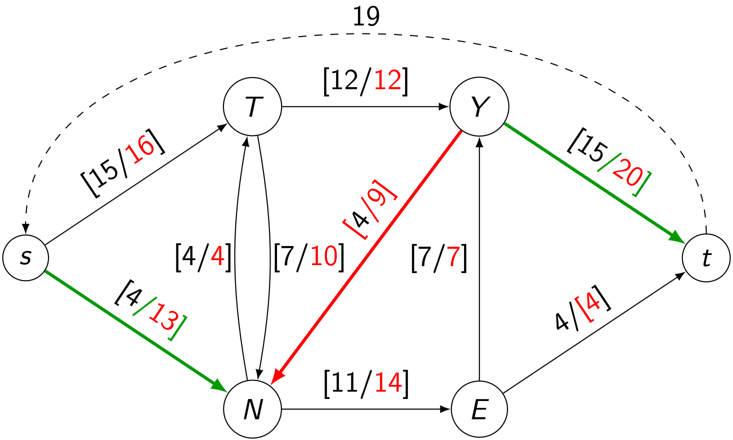
Augmenting path

Définition

Considering a network (G, s, t, c) and a feasible flot f, an *augmenting path* μ is an **(undirected)** path linking s and t such that:

- for every arc a of μ directed from s to t (the *right* direction), $f(a) < c(a)$. We write $a \in \mu^+$.
- for every arc a of μ directed from t to s (the *wrong* direction), $f(a) > 0$. We write $a \in \mu^-$.

Augmenting path



$\mu^+ = \{(s, N); (Y, t)\}; \mu^- = \{(Y, N)\}$

Augmenting path

Définition

Considering a network (G, s, t, c) and a feasible flot f, an *augmenting path* μ is an **(undirected)** path from s to t such that:

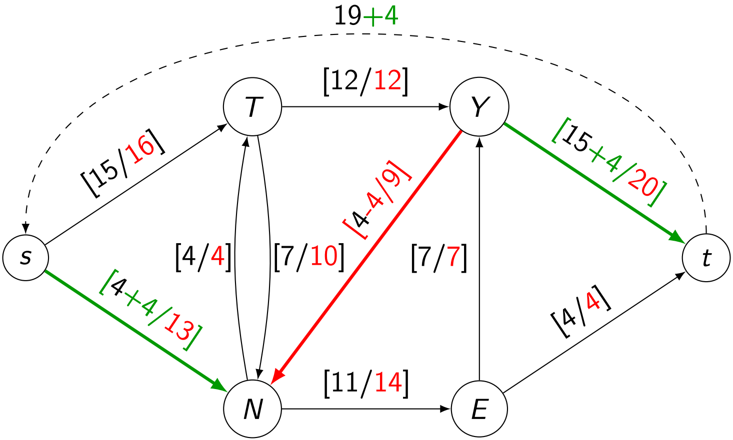
- for every arc a of μ directed from s to t (the *right* direction), $f(a) < c(a)$. We write $a \in \mu^+$.
- for every arc a of μ directed from t to s (the *wrong* direction), $f(a) > 0$. We write $a \in \mu^-$.

Such a path may increase the value of the flow by at least

$$v \rightarrow v + \min\left(\min_{a \in \mu^+}\{c(a) - f(a)\}; \min_{a \in \mu^-}\{f(a)\}\right)$$

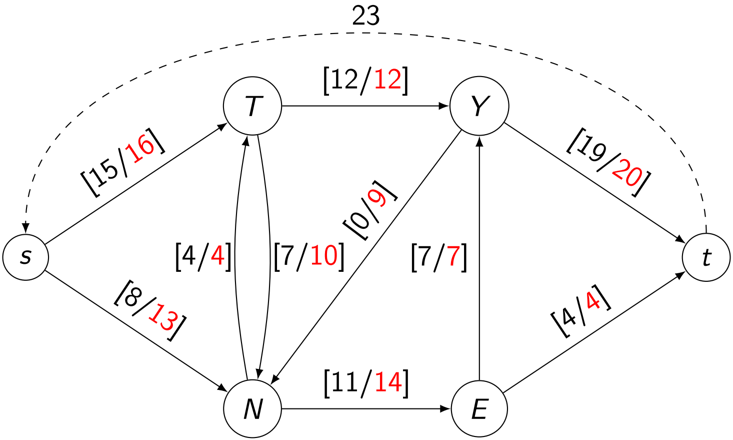
(We add flow to the arc on the right direction, and we remove flow from the arc on the wrong direction.)

Augmenting path



$$\min\left(\min_{a\in\mu^+}\{c(a)-f(a)\};\ \min_{a\in\mu^-}\{f(a)\}\right)=4$$

Augmenting path



No more augmenting path. The flow is maximum.

How to find a maximum flow?

Theorem

Given a flow network and a feasible flow, there is no augmenting path if and only if the flow is maximum.

Proof : later. We need stronger tools.

The Ford-Fulkerson algorithm : basis

Let (G, s, t, c) be a flow network

- Let f be a nul flow on G
- **While** there exists an augmenting path μ on f **do**
  - $dv \leftarrow \min\left(\min_{a \in \mu^+}\{c(a) - f(a)\}; \min_{a \in \mu^-}\{f(a)\}\right)$
  - $\forall a \in \mu^+, f(a) \leftarrow f(a) + dv$
  - $\forall a \in \mu^-, f(a) \leftarrow f(a) - dv$
- Return f

Find an augmenting path

1st version : the residual network

Definition

Let (G, s, t, c) be a flow network and a feasible flow f. We define the *residual network* as a graph $H = (V, B)$ and a weight $\omega : B \rightarrow \mathbb{N}$ over the arcs of H such that :

- for every arc $a = (u, v) \in A$ such that $f(a) < c(a)$, we add an arc $b = (u, v) \in B$, and the weight of b is $\omega(b) = c(a) - f(a)$
- for every arc $a = (u, v) \in A$ such that $f(a) > 0$, we add an arc $b = (v, u) \in B$, and the weight of b is $\omega(b) = f(a)$

(See the board for an example)

Find an augmenting path

1st version : the residual network

Lemma

There is an augmenting path in G if and only if there is a path in H from s to t.

(See the board for a proof and an example)

Find an augmenting path

2nd version : marking algorithm

1. Mark the source s with " + "
2. **For** every arc $(u, v) \in A$ **do**
   - **If** u is marked, v is not marked and $f(u, v) < c(u, v)$, **then** mark v with " $+(u)$"
   - **If** v is marked, u is not marked and $f(u, v) > 0$, **then** mark u with " $-(v)$"
3. **Restart** 2 if at least one node was marked.

(See the board for an example)

Find an augmenting path

2nd version : marking algorithm

Lemma

There is an augmenting path in G if and only if t is marked.

In that case, in order to build the path, we have to follow the markings backward from t.
(See the board for a proof and an example)

The minimum-cut problem

Definition

Let $(G = (V, A), s, t, c)$ be a flow network. A *cut* that separates the source and sink, or s-t cut, is a partition of V into two subsets $S \uplus T = V$ such that $s \in S$ and $t \in T$.
The weight of the cut is $c(S, T) = \sum_{u \in S} \sum_{v \in T} c(u, v)$.

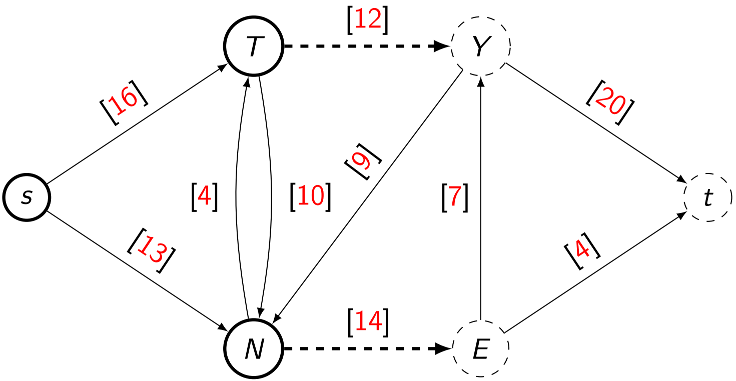
- The weight of an s-t cut is the sum of the capacities of the arcs entering T (similarly, the arcs leaving S)
- Removing those arcs **cut** every directed path from s to t.

The minimum-cut problem

The minimum-cut problem

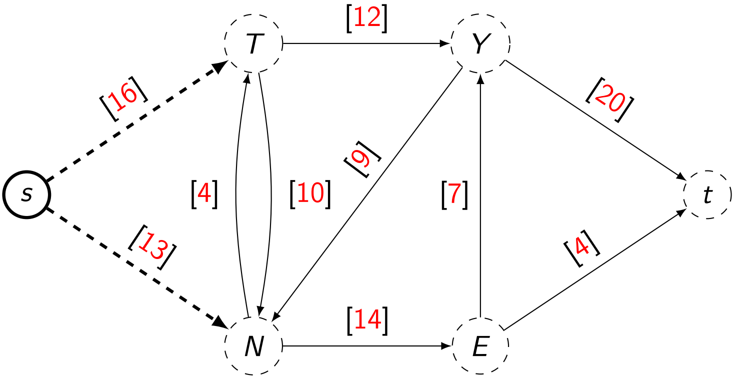
- Input : a flow network (G, s, t, c).
- Feasible solution: a cut (S, T) that separates s from t
- Optimal solution: a minimum weight s-t cut.

Examples



(S is bold, T is dashed)
Weight $c(S, T)$: 12+14=26

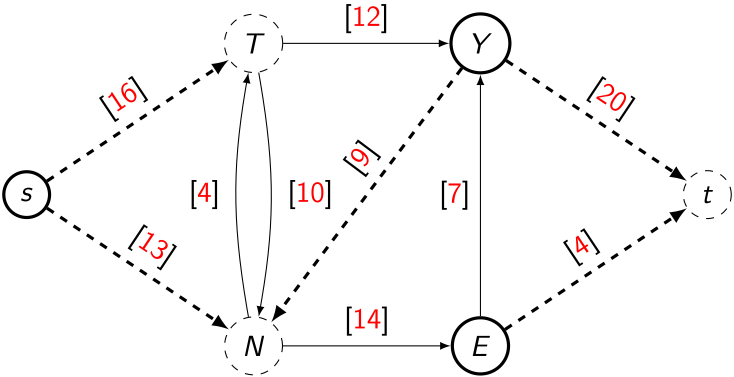
Examples



(S is bold, T is dashed)

Weight $c(S, T)$: 13+16=29

Examples



(S is bold, T is dashed)

Weight $c(S, T)$: 13+16+9+20+4=62

The (weak) max-flow-min-cut theorem

Theorem

Let (G, s, t, c) be a flow network, f be a feasible flow on G of value v, and (S, T) be an $s - t$ cut, then $c(S, T) \geq v$.

Proof : the flow entering t is no more than the flow entering T.

The (strong) max-flow-min-cut theorem

Theorem

Let (G, s, t, c) be a flow network, f be a maximum on G of value v, and (S, T) be a minimum $s - t$ cut, then $c(S, T) = v$.

The proof shows also this theorem :

Theorem

Given a flow network and a feasible flow, there is no augmenting path if and only if the flow is maximum.

(Proof on board)

Compute a minimum cut

When the Ford-Fulkerson algorithm stops, 1st version.

- We proved that, in that case, there is no path from s to t in the residual network H.
- Let S be the set of nodes x such that there is a path from s to x in H, and let T be the other nodes.
- (S, T) is a minimum cut.

Compute a minimum cut

When the Ford-Fulkerson algorithm stops, 2nd version.

- In that case, we proved that t is not marked.
- Let S be the set of marked nodes and let T be the other nodes.
- (S, T) is a minimum $s - t$ cut.