Decision Aid Methodologies In Transportation

Lecture 5: Graph and Network

Graph

and

Networks

Shadi SHARIF AZADEH

Transport and Mobility Laboratory TRANSP-OR

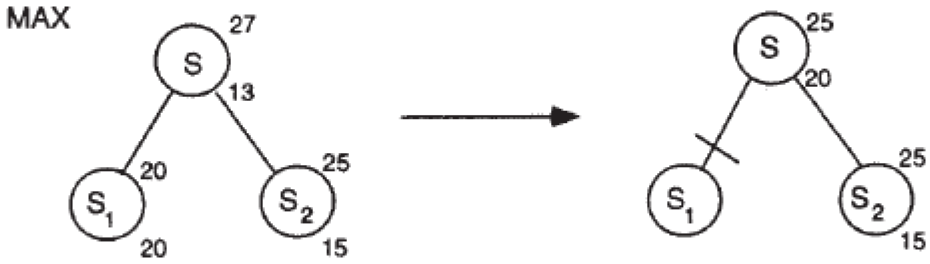
École Polytechnique Fédérale de Lausanne EPFL

TRANSP-OR

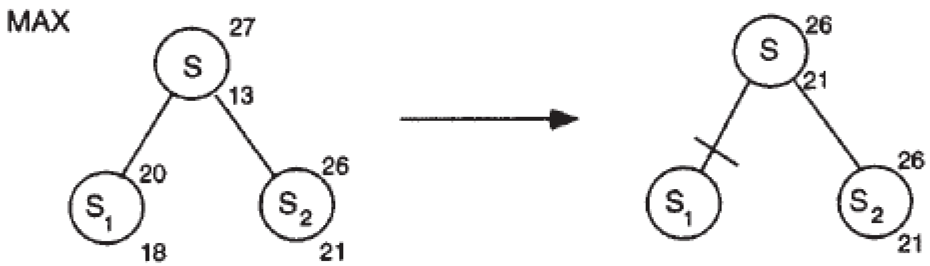
EPFL

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Branch and Bound



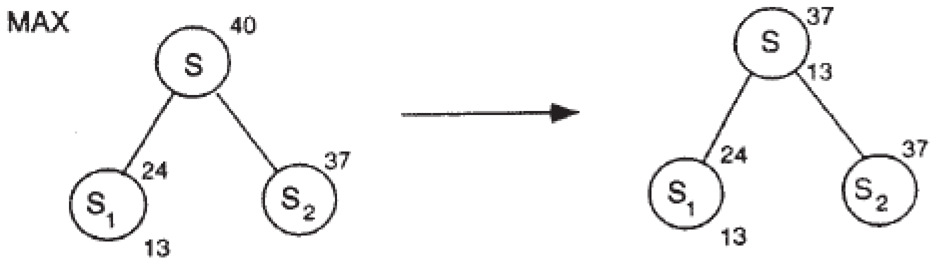
Pruned by optimality



Pruned by bound

Best Bound and Best Solution?

The relaxation of integer programming problem



No pruning possible

Three reasons that allow us to prune the tree and thus enumerate a large number of solutions

1) **Pruning by optimality**
2) **Pruning by bound**
3) **Pruning by infeasibility**

Branch and bound- example

Maximize $z = 5y_1 - 2y_2$

subject to $-y_1 + 2y_2 \leq 5$

$3y_1 + 2y_2 \leq 19$

$y_1 + 3y_2 \geq 9$

$y_1, y_2 \geq 0$ and integer

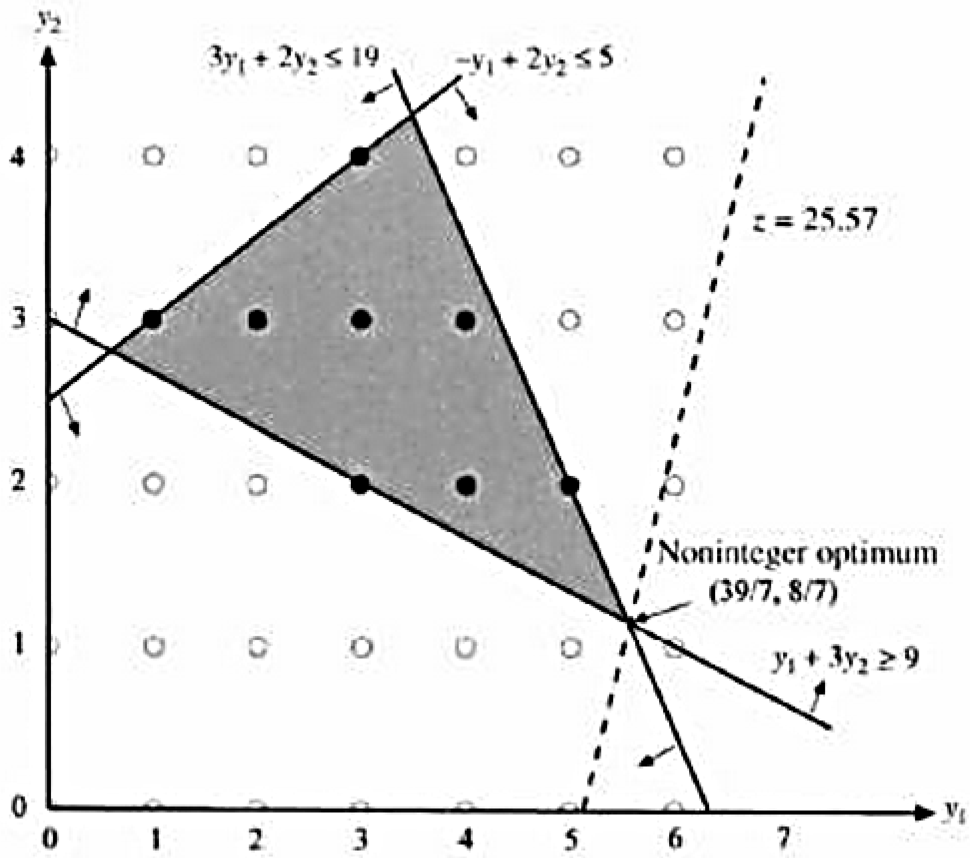
Integer Solution?

- No

Partitioning strategy?

- Variable Y_1?
- Variable Y_2?

- Sub problems?

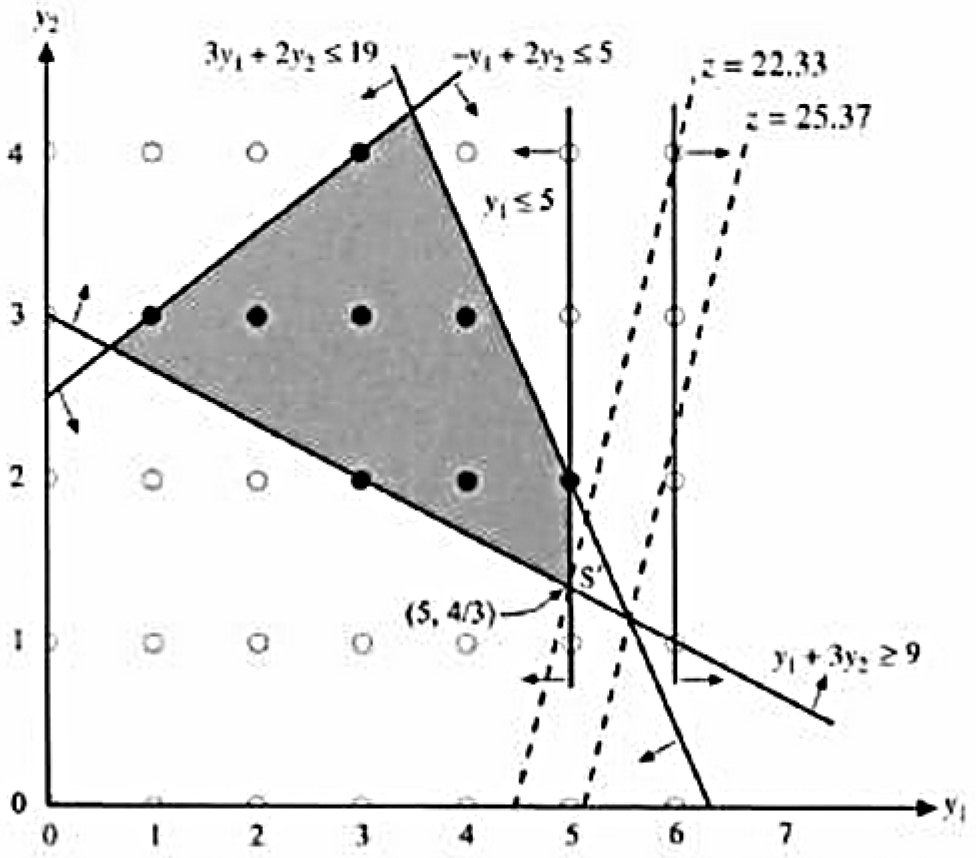


Branch and bound- example

Branch on Y1
Solution Y_1=(39/7)
Examine two sub problems

$Y_1 \geq 6$
$Y_1 \leq 5$

- Integer Solution?
  - No
- Partitioning strategy?
  - Variable Y_1
  - Variable Y_2



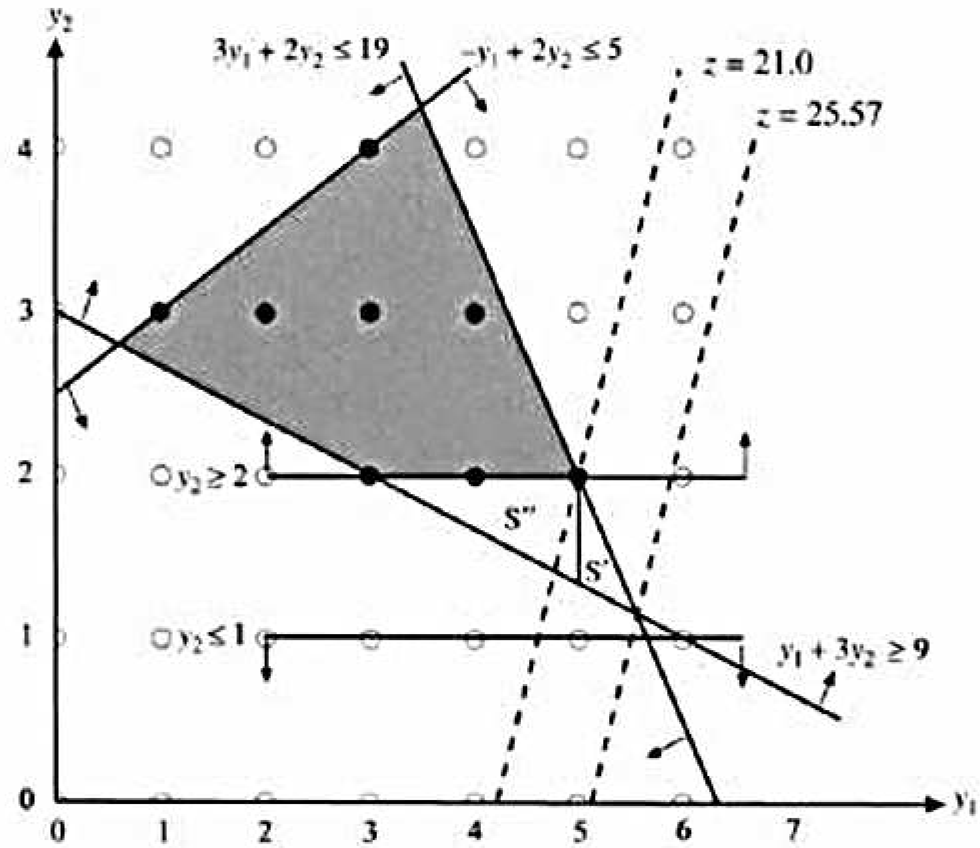
Branch and bound- example

Branch on Y_2
Solution Y_2=(8/7)
Examine two sub problems

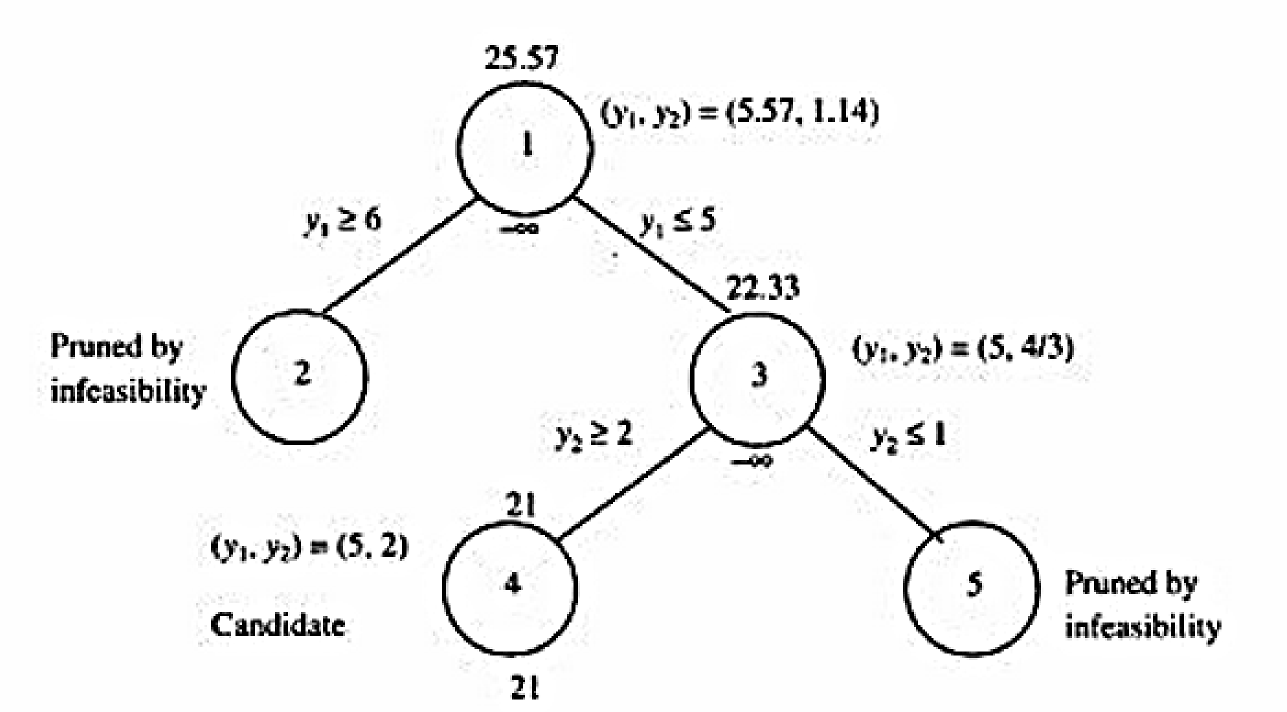
$Y_2 \geq 2$

$Y_2 \leq 1$

- Integer Solution?
  - Yes



Branch and bound- tree



Branch and bound- Practical tips

1) What relaxation should be used to provide upper bounds
    - Choose the model with best relaxation bound
2) How should the feasible region be separated into smaller regions
    - Which variable to branch
    - How to partition
    - How many sub problems
3) In what order should the sub problems be examined
    - Node selection
    - Which sub-problem should be examined first

Branch and bound- Practical tips

How should the feasible region be separated into smaller regions

When an LP solution contains several fractional values for integer variables, the decision about which integer variable should be chosen to branch is needed. The following rules are commonly used for choosing a branching variable:

1. Variable with fractional value closest to 0.5
2. Variable with highest impact on the objective function
3. Variable with the smallest index

Branch and bound- Practical tips

1. Variable with fractional value closest to 0.5

$$X + Y + Z = 1$$

LP Solution: $X = 0.9, Y = 0.05$, $Z = 0.05$
Sub-Problems?
3 nodes : $X = 1, Y = 1, Z = 1$
2 nodes: $X = 1, X = 0$

Which one is better?

Branch and bound- Practical tips

2. Variable with highest impact on the objective function

$$\text{Max } 100X + 10Y$$

LP Solution: $X = 4.3, Y = 2.4$
Partition:
Case 1: $X > 4$ and $X \leq 4$
Case 2: $X \geq 5$ and $X \leq 4$

Which one is better?

Branch and bound- Practical tips

3. Variable with the smallest index

$$\text{Max} \sum_{i=1}^{100} x_i + \sum_{i=1}^{10} y_i$$

Partition on x or y?

Branch and bound- Practical tips

which unpruned node to explore first

The most commonly used search strategies include

1- **depth-first (last-in-first-out)**

first solve the most recently generated sub problem

→ quickly obtain a primal feasible integer solution (solving by dual simplex)

2- **best-bound-first (best upper bound)**

branch on the active node with highest value of the objective function (for a maximization problem and vice-versa for the minimization problem)

→ The goal is to minimize the total number of nodes evaluated in the B&B tree

Branch and bound- Practical tips

Performance of these branching rules depends on the problem structure. In practice, a compromise between the two is usually adopted. That is , apply the depth-first strategy first to get one feasible integer solution, followed by a mixture of both strategies.

$$\begin{aligned}
\text{Maximize} \quad & z = -y_1 + 2y_2 + y_3 + 2x_1 \\
\text{subject to} \quad & y_1 + y_2 - y_3 + 3x_1 \leq 7 \\
& y_2 + 3y_3 - x_1 \leq 5 \\
& 3y_1 + x_1 \geq 2 \\
& y_1, y_2, y_3 \geq 0 \text{ and integer} \\
& x_1 \geq 0
\end{aligned}$$

Branch and bound- Practical tips

Branch-and-bound using depth-first

Node with best bound is chosen=n4

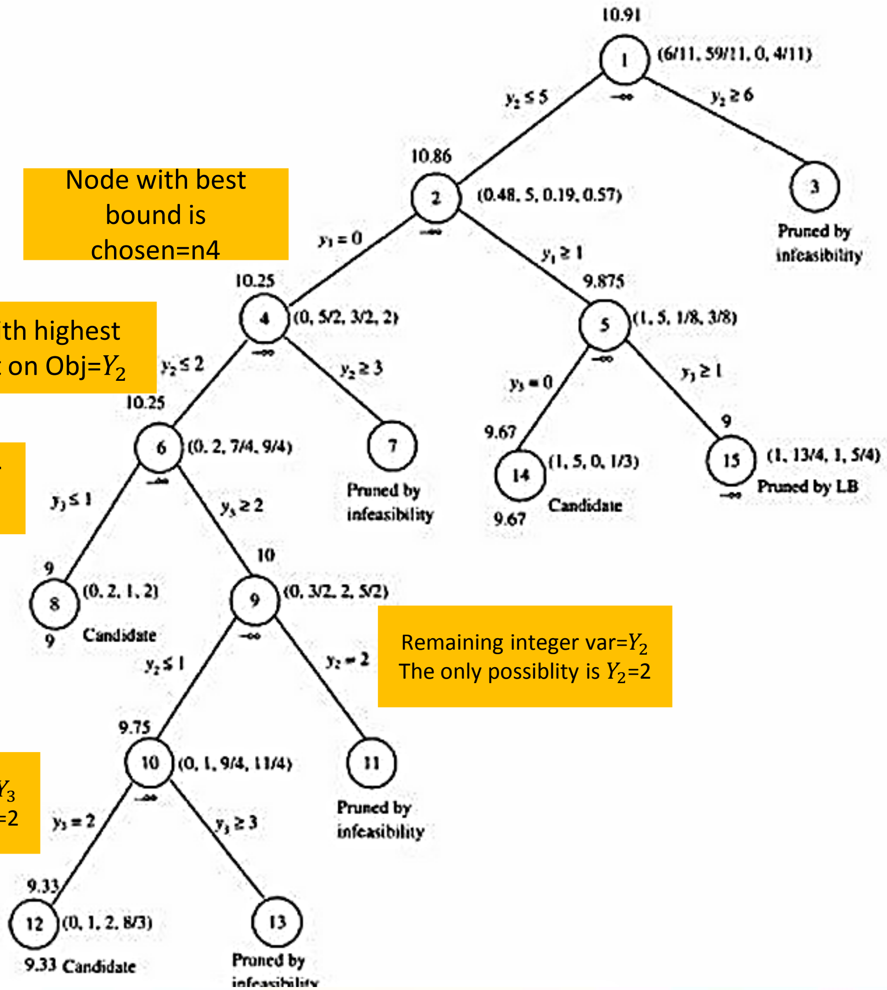
Var with highest impact on Obj=Y_2

Remaining integer var=Y_3

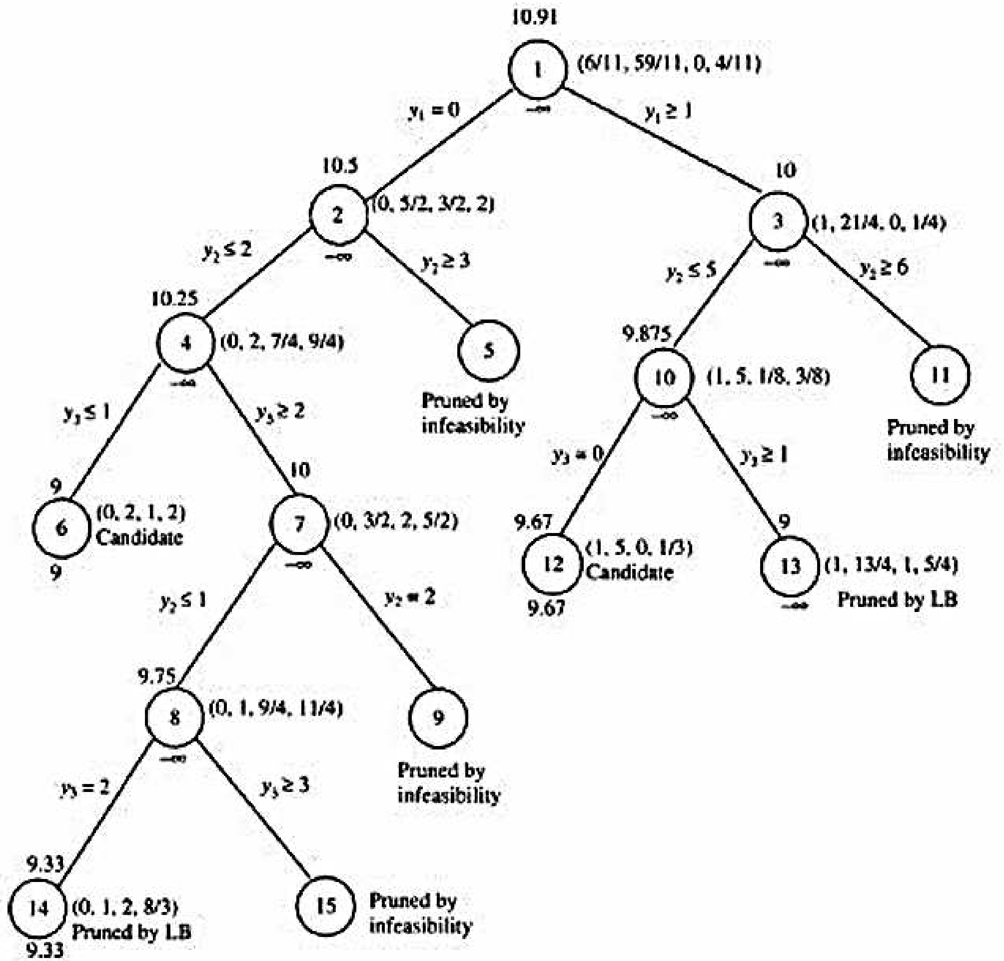
Var closest to 0.5 =Y_1

Remaining integer var=Y_2
The only possiblity is Y_2=2

Remaining integer var=Y_3
The only possiblity is Y_3=2



Branch and bound- Practical tips



Branch-and-bound using best bound first

Graph Theory-Shortest path problem

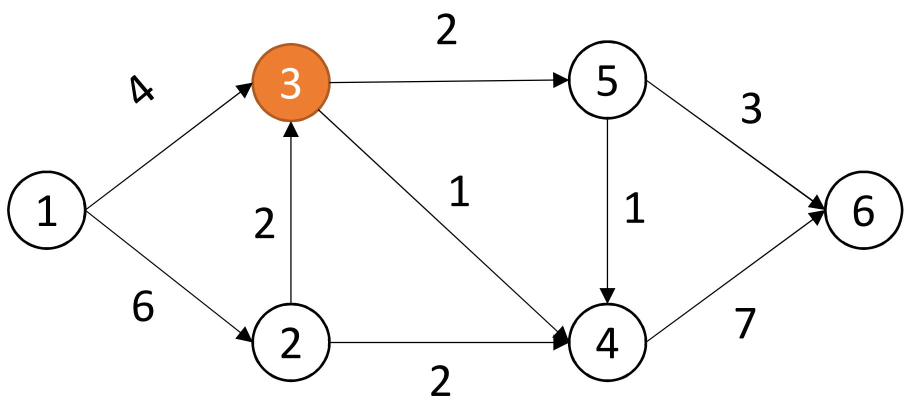
For all algorithm and notations $G = (V, A)$ represents the graph in which V is the set of nodes and A is the set of arcs.

Number of nodes $= n$ in our example graph we have 6 nodes

Number of arcs$= m$ in our example graph we have 9 arcs

We consider $V^{+(i)}$ as the set of imediate successor of node i and $V^{-(i)}$ as the set of immediate predecessor nodes

In our example graph $V^{+(3)} = \{5,4\}$ and $V^{-(3)} = \{1,2\}$

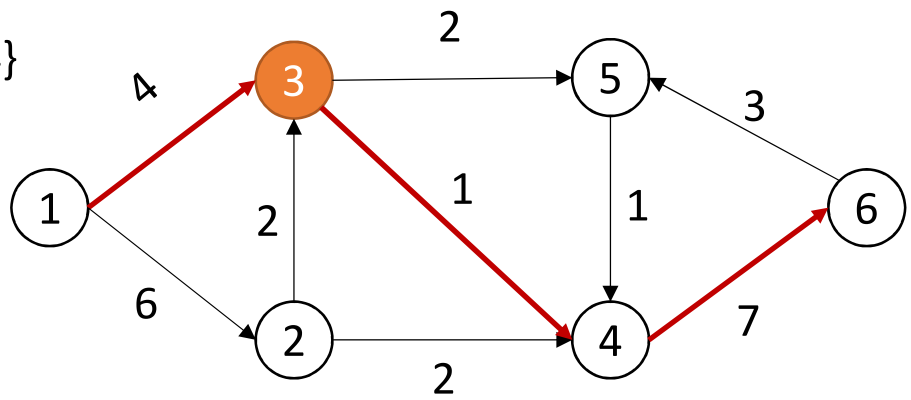


Graph Theory-Shortest path problem

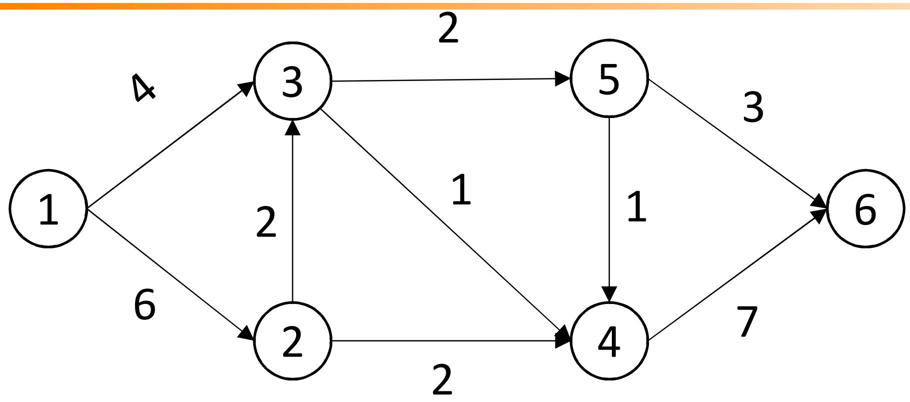
A **chain** of a graph G is an alternating sequence of vertices $x_0, x_1, \cdots, x_n$ beginning and ending with vertices in which each edge is incident with the two vertices immediately preceding and following it. If the first and the last nodes are the same we have the cycle.

Graph → Directed Graph
Chain → Path
Cycle → Directed cycle

Path={1,3,4,6}
Directed cycle={4,6,5,4}



Graph Theory-Shortest path problem



Mathematical model:

$$Z = \min \sum_{(i,j)\in A} c_{ij} x_{ij}$$

$$\sum_{k\in V^{+}(i)} x_{ik} - \sum_{k\in V^{-}(i)} x_{ki} = 1 \quad i = s$$

$$\sum_{k\in V^{+}(i)} x_{ik} - \sum_{k\in V^{-}(i)} x_{ki} = 0 \quad i \in V \setminus \{s,t\}$$

$$\sum_{k\in V^{+}(i)} x_{ik} - \sum_{k\in V^{-}(i)} x_{ki} = -1 \quad i = t$$

$$x_{ij} \geq 0 \quad \forall (i,j) \in A$$

Unimodality property

TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Graph Theory-Shortest path problem

Dijkstra

Hypothesis: all arcs has positive value
Find minimum distance from source to sink.
k is the index of a node.

(1) $\overline{S} := \{2,..,n\}; \pi(1) = 0;$ for all $k \neq 1$ do $\pi(\mathrm{k}) = \begin{cases} d_{1k} & \text{if } k \in V^+(1) \\ \infty & \text{otherwise} \end{cases}$

(2) determine k such as $\pi(\mathrm{k}) \leq \pi(y)$ for all y in $\overline{S}$ and consider $\overline{S} := \overline{S} - \{\mathrm{k}\}$

if $\overline{S} = \varnothing$ STOP

(3) for all y in $\overline{S} \cap V^+(\mathrm{k})$ do $\pi(y) := \min\{\pi(y), \pi(\mathrm{k}) + d_{ky}\}$ and return to (2)

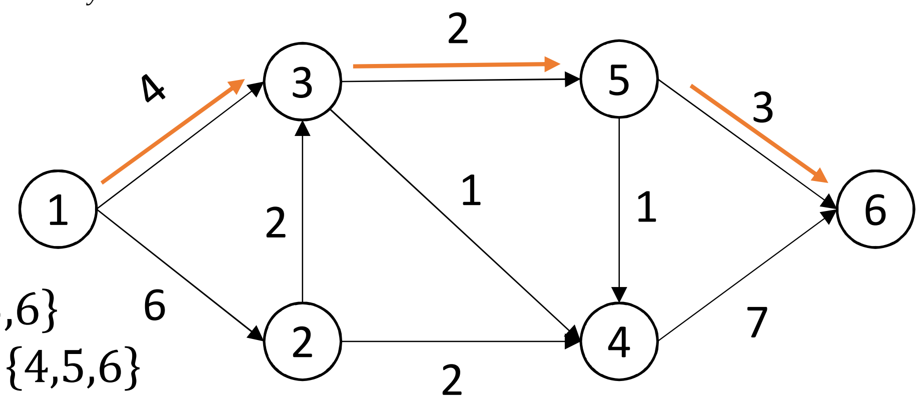
Graph Theory-Shortest path problem

(1) $\bar{S} := \{2,..,n\}; \pi(1) = 0;$ for all $k \neq 1$ do $\quad \pi(\mathrm{k}) = \begin{cases} d_{1k} & \text{if } k \in V^+(1) \\ \infty & \text{otherwise} \end{cases}$

(2) determine k such as $\pi(\mathrm{k}) \leq \pi(y)$ for all y in $\bar{S}$ and consider $\bar{S} := \bar{S} - \{\mathrm{k}\}$

if $\bar{S} = \varnothing$ STOP

(3) for all y in $\bar{S} \cap V^+(\mathrm{k})$ do $\pi(y) := \min\{\pi(y), \pi(\mathrm{k}) + d_{ky}\}$ and return to (2)



Initialize:

$\bar{S}$={2,3,4,5,6}

$\pi(1) = 0$

Iteration:

It1. $\pi(2) = 6, \quad \pi(3) = \min\{4, \infty\}, \quad \bar{S} = \{3,4,5,6\}$

It2. $\pi(3) = \min\{4,8\}, \pi(4) = \min\{8, \infty\}, \bar{S} = \{4,5,6\}$

It3. $\pi(4) = \min\{8,5, \infty\}, \pi(5) = \{6\}, \bar{S} = \{4,6\}$

It4. $\pi(4) = \min\{8,5,7\}, \pi(6) = \{9, \infty\}, \bar{S} = \{6\}$

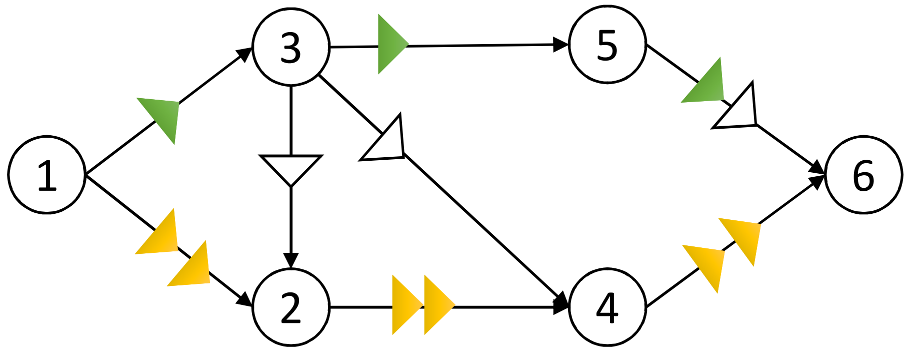
It5. $\pi(6) = \{9,12\} = 9$

TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Graph Theory-Maximum flow problem

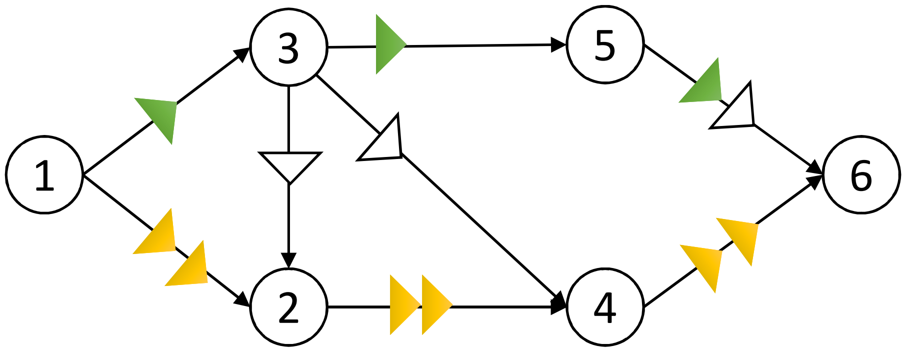
How many units of flow can be transferred from 1 to 6?



Find the maximum flow from a source to sink and repeat it until no flow exists.
One way is to iteratively find the paths between source to the sink that can simultaneously transfer the flow and calculate the maximum flow that these paths can handle. In the above case, there are 3 paths from 1 to 6; however, only two of them can transfer the flow (overall 3 units). Does this approach give us the optimal solution?

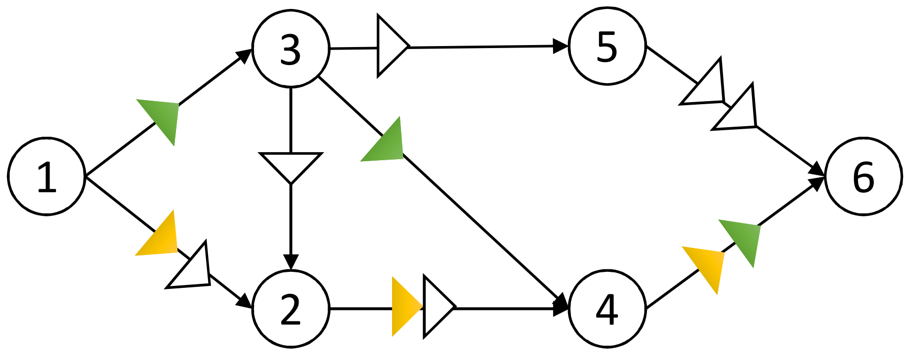
Graph Theory-Maximum flow problem

No, it may not find the maximum flow



Units transferred:
3

Blocking flow but
not maximum
{1,3,4,6}
{1,2,4,6}



Units transferred:
2

Graph Theory-Maximum flow problem

Some types of graphs:

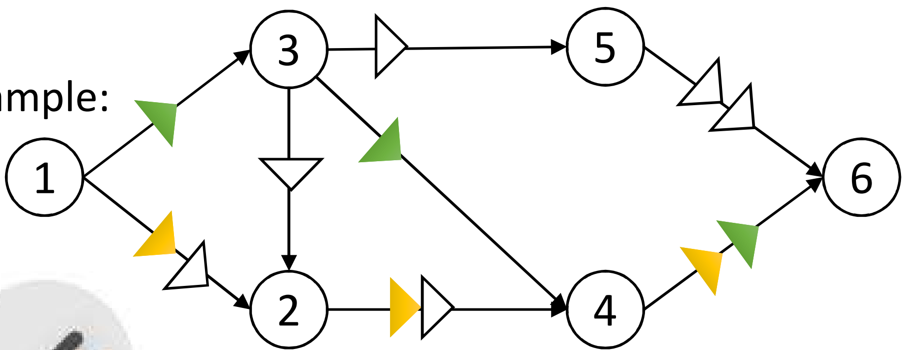
For a graph $G(V, A)$, each arc has a maximum capacity (Ca) and the amount of the flow on arc a is denoted by $f(a)$. The total flow passing from source to sink in the graph is presented by $G(f)$

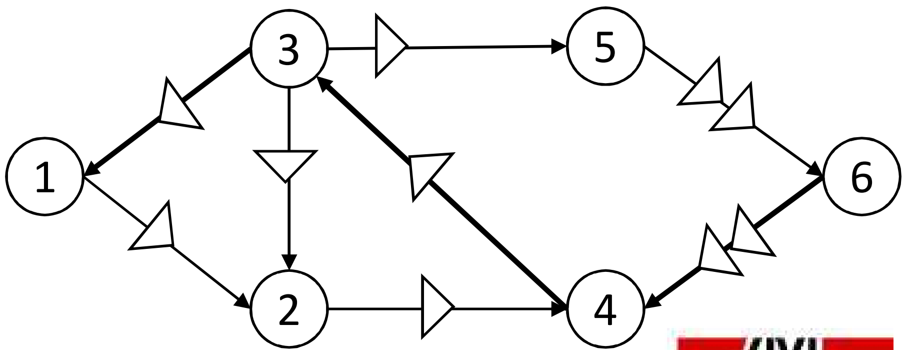
Residual graph ($G^*(f)$):

Based on the flow that passes through the graph node we can build a residual graph.

- Nodes: It has the same number of nodes as graph G
- Arcs: for each arc $a = (x, y)$ in G we generate one arc on G* based on the following possibilities:
  - If f(a)<Ca : we add an arc (x,y) with the capacity of Ca-f(a)
  - If f(a)=Ca: we add an arc from y to x with the capacity of Ca

Example:





Graph Theory-Maximum flow problem

Level graph ($\overline{G}$):

A graph is called Level, if we partition its nodes into two consecutive sub-sets of nodes, arcs in the current subsets must be connected to the nodes in another subset.

V_i *and* V_{i+1} are two concecutive subsets

M is a mega node of first subset

$\omega^+(M)$: represets arcs that are going out from mega node M

Algorithm:

(1) V_1:={s}; M:={s};i:=1;

(2) if $\omega^+(M)$ is empty then STOP

otherwise

add in $\overline{\text{G}}$ all the arcs of ω^+(M)

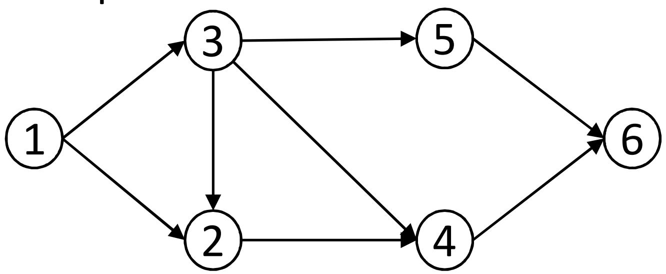
$\text{V}_{i+1}=\{x \mid \exists (y,x) \in \omega^+(\text{M})\}$

$\text{M}:=\text{M} \cup \text{V}_{i+1}$

i:=i+1 and GO TO (2)

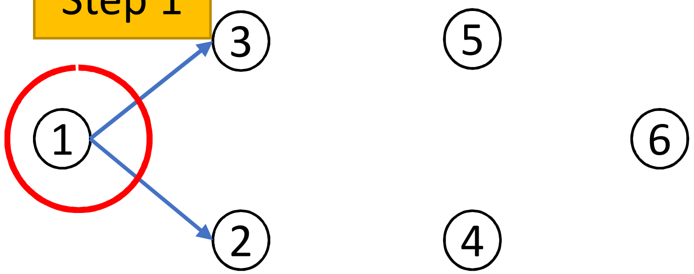
Graph Theory-Maximum flow problem

Example:

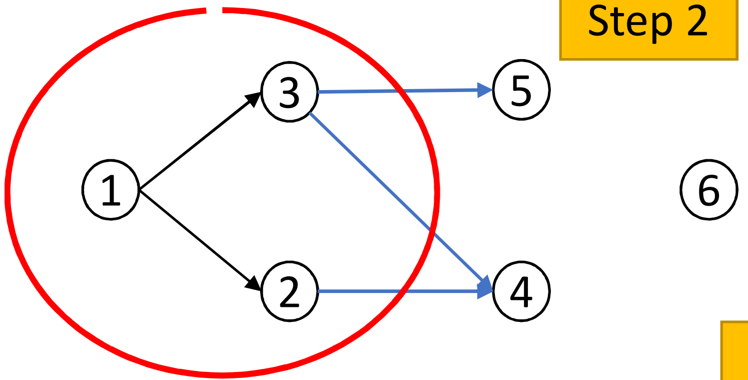


Graph G

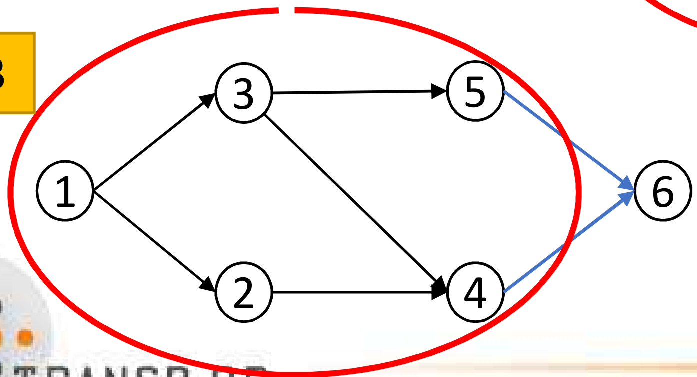
Step 1



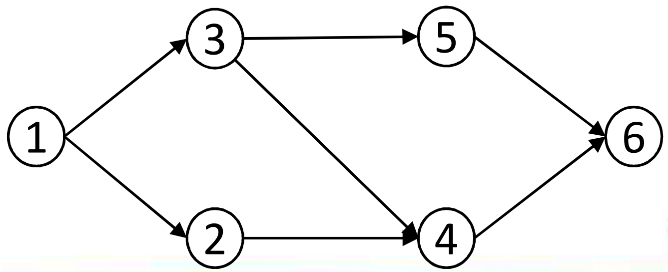
Step 2



Step 3



Level Graph $\overline{G}$

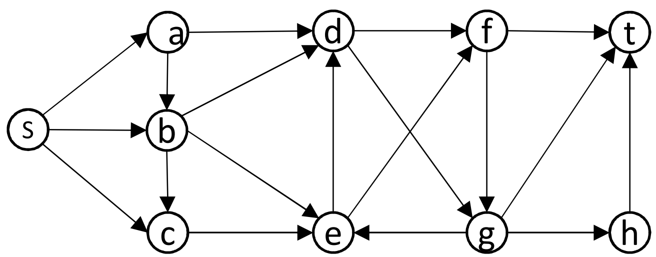


Graph Theory-Maximum flow problem

Dinic Algorithm:

(1) Determine a feasible flow f for the Graph named G(f).
If the flow is zero then we have G(0).
(2) Build the residual graph G*(f)
(3) Construct the level graph of G*(f) named $\overline{G^*(f)}$.
(4) Find blocking flow from source to sink in Graph $\overline{G^*(f)}$.
(5) If there is a blocking flow from source to sink then
Add all the blocking flows to G(f). Now we have G(f ') in which f ' > f. Replace G(f ') by G(f) and GOTO 2
Otherwise the solution is optimal

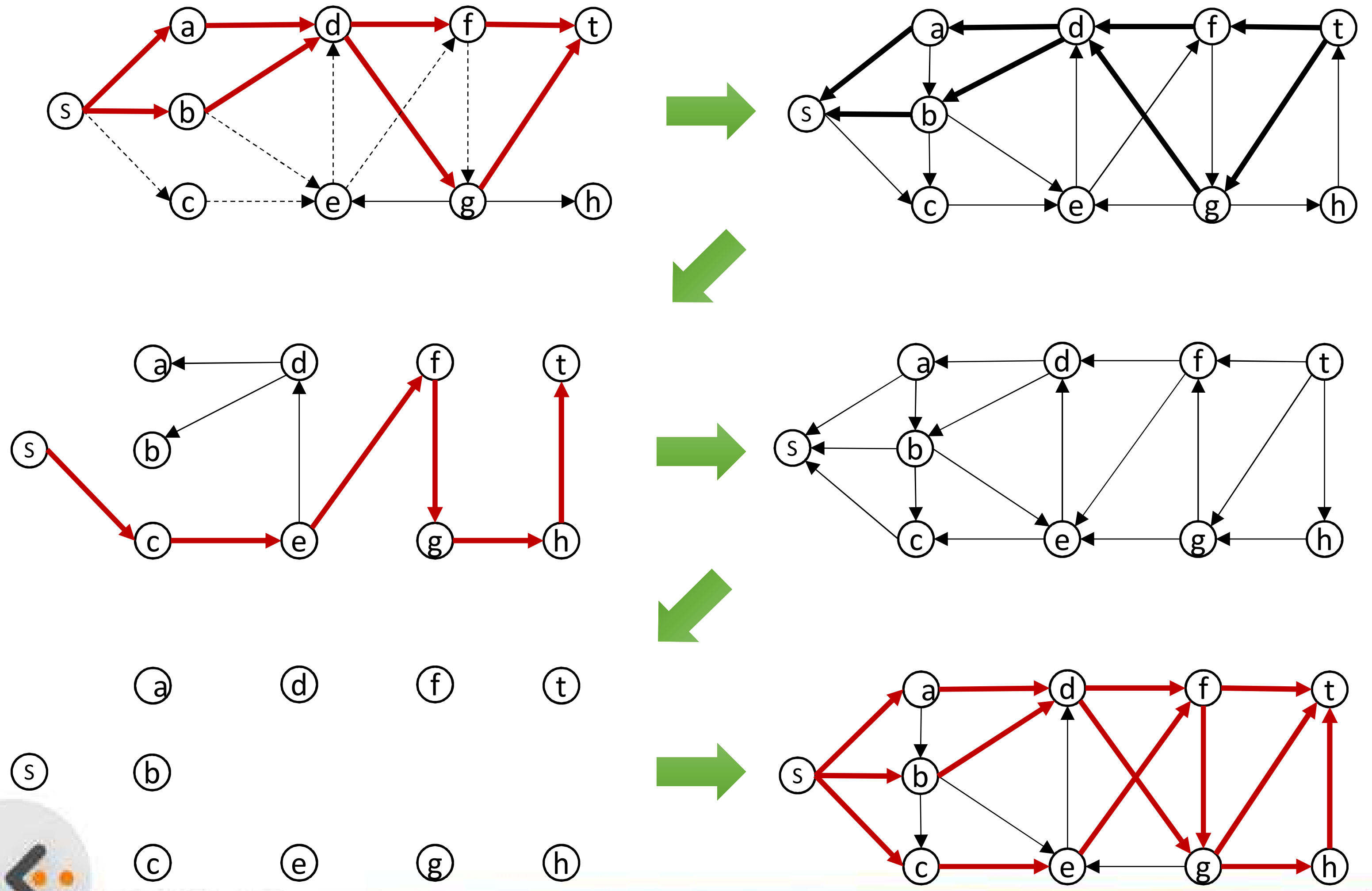
Example: Find the maximum flow for the following graph



TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Graph Theory-Maximum flow problem



Graph Theory-Minimum Cost Flow

Let $G = (N, A)$ be a directed network with a *cost* $C_{i,j}$ and a maximum *capacity* $U_{i,j}$ associated with every arc $(i, j) \in A$. We associate with each node $i \in N$ a number $b(i)$ which indicates its supply or demand depending on whether $b(i) > 0$ or $b(i) < 0$. The minimum cost flow problem can be stated as follows:

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

$$\sum_{i\in N} x_{ij} - \sum_{k \in N} x_{jk} = b_j \quad \text{for all } j \in N$$

$$l_{ij} \leq x_{ij} \leq u_{ij} \in A$$

Each node is:

Flow generation $b(i) < 0$

Flow consumption $b(i) > 0$

Flow conservation $b(i) = 0$

Graph Theory-Minimum Cost Flow

(1) Determine a feasible flow f

if no flow exists then STOP

(2) if G*(f) does not have a negative cycle then the current flow obtains minimum cost

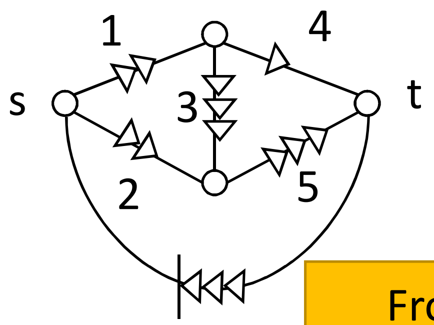
if not C is a directed negative cycle and $\Delta = \min_{(x,y)\in C} c^*_{(x,y)}$. for all arcs (x,y) in negative cost cycle (C) do

- increase the flow Δ unit on (x,y) if arc (x,y) exists in G

- decrease the flow Δ unit on (y,x) if arc (y,x) exists in G

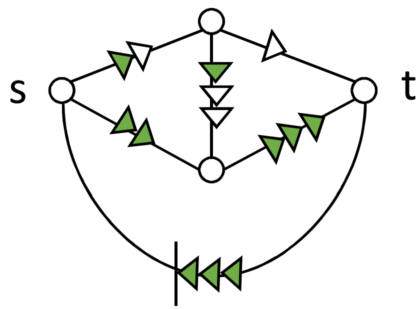
GOTO2

Graph Theory-Minimum Cost Flow



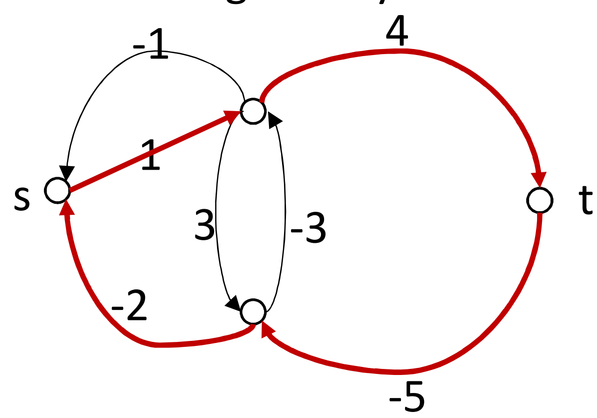
From this arc we must pass exactly 3 units of flow

Find a feasible flow



Total Cost=23

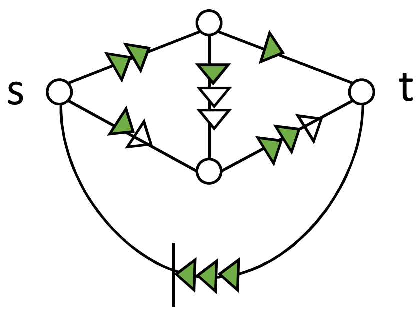
Find a negative cycle in G*



Arcs show the possibility of movement with associated cost

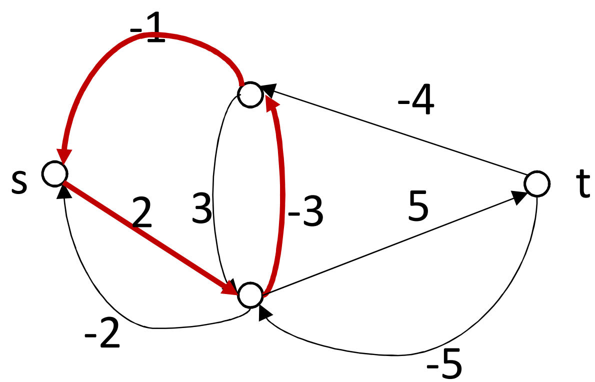
Graph Theory-Minimum Cost Flow

Update the flow

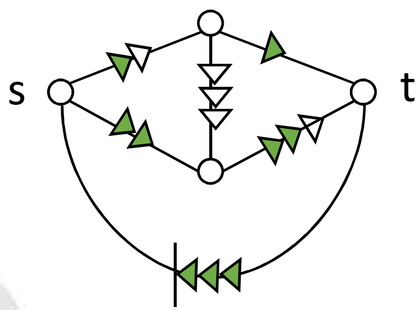


Total Cost=21

Find a negative cycle in G*



Update the flow



Total Cost=19

No negative Cycle the solution is optimal