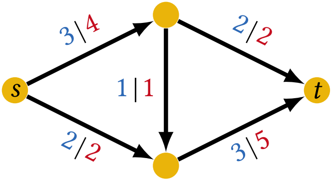
Introduction to

# Linear and Combinatorial Optimization

# 10 Maximum Flows

## 10.1 Network Flows

**Given:** Digraph $D = (V, A)$, arc capacities $u : A \longrightarrow \mathbb{R}_{\geq 0}$, nodes $s, t \in V$.




capacities

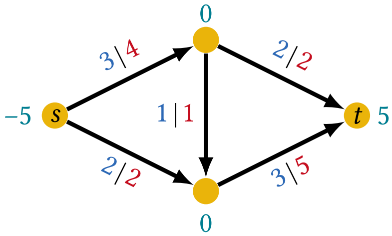
flow values

**Definition 10.1** A flow in $D$ is a function $x : A \longrightarrow \mathbb{R}_{\geq 0}$. Flow $x$ in $D$

- i obeys arc capacities and is called feasible, if $x(a) \leq u(a)$ for all $a \in A$;
- ii has excess $\mathrm{ex}_x(v) := x(\delta^-(v)) - x(\delta^+(v))$ at node $v \in V$;
- iii satisfies flow conservation at node $v \in V$ if $\mathrm{ex}_x(v) = 0$;
- iv is a circulation if it satisfies flow conservation at all nodes $v \in V$;
- v is an $s$-$t$-flow of value $\mathrm{ex}_x(t)$ if it satisfies flow conservation at each node $v \in V \setminus \{s, t\}$ and if $\mathrm{ex}_x(t) \geq 0$.

## Basic Definitions

**Given:** Digraph $D = (V, A)$, arc capacities $u : A \longrightarrow \mathbb{R}_{\geq 0}$, nodes $s, t \in V$.




capacities

flow values

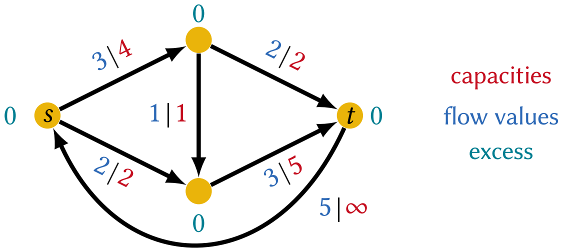
excess

**Definition 10.1** A flow in $D$ is a function $x : A \longrightarrow \mathbb{R}_{\geq 0}$. Flow $x$ in $D$

- i obeys arc capacities and is called feasible, if $x(a) \leq u(a)$ for all $a \in A$;
- ii has excess $\mathrm{ex}_x(v) := x(\delta^-(v)) - x(\delta^+(v))$ at node $v \in V$;
- iii satisfies flow conservation at node $v \in V$ if $\mathrm{ex}_x(v) = 0$;
- iv is a circulation if it satisfies flow conservation at all nodes $v \in V$;
- v is an $s$-$t$-flow of value $\mathrm{ex}_x(t)$ if it satisfies flow conservation at each node $v \in V \setminus \{s, t\}$ and if $\mathrm{ex}_x(t) \geq 0$.

# Basic Definitions

**Given:** Digraph $D = (V, A)$, arc capacities $u : A \longrightarrow \mathbb{R}_{\geq 0}$, nodes $s, t \in V$.




**Definition 10.1** A flow in $D$ is a function $x : A \longrightarrow \mathbb{R}_{\geq 0}$. Flow $x$ in $D$

i obeys arc capacities and is called feasible, if $x(a) \leq u(a)$ for all $a \in A$;

ii has excess $\mathrm{ex}_x(v) := x(\delta^-(v)) - x(\delta^+(v))$ at node $v \in V$;

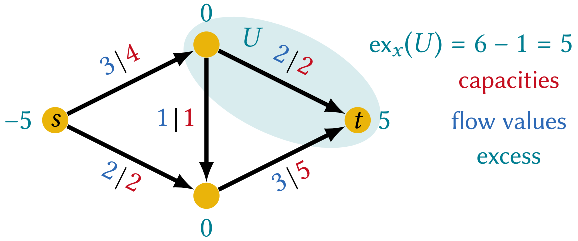
iii satisfies flow conservation at node $v \in V$ if $\mathrm{ex}_x(v) = 0$;

iv is a circulation if it satisfies flow conservation at all nodes $v \in V$;

v is an $s$-$t$-flow of value $\mathrm{ex}_x(t)$ if it satisfies flow conservation at each node $v \in V \setminus \{s, t\}$ and if $\mathrm{ex}_x(t) \geq 0$.

# Maximum $s$-$t$-Flows

The maximum $s$-$t$-flow problem asks for a feasible $s$-$t$-flow in $D$ of maximum value.




- the excess of $U \subseteq V$ is defined as $\mathrm{ex}_x(U) := x(\delta^-(U)) - x(\delta^+(U))$

**Lemma 10.2** For a flow $x$ and $U \subseteq V$ it holds that $\mathrm{ex}_x(U) = \sum_{v \in U} \mathrm{ex}_x(v)$. In particular, the value of an $s$-$t$-flow $x$ is equal to

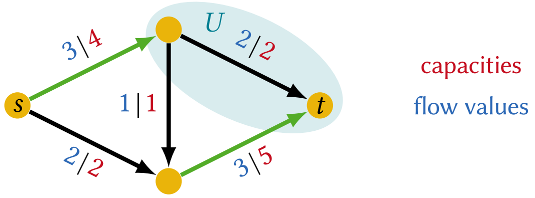
$$\mathrm{ex}_x(t) = -\mathrm{ex}_x(s) = \mathrm{ex}_x(U) \quad \text{for each } U \subseteq V \setminus \{s\} \text{ with } t \in U.$$

**Proof:**
$$\begin{aligned}\mathrm{ex}_x(U) &= \textstyle\sum_{a \in \delta^-(U))} x(a) - \sum_{a \in \delta^+(U)} x(a) \\ &= \textstyle\sum_{v \in U} \Big( \sum_{a \in \delta^-(v)} x(a) - \sum_{a \in \delta^+(v)} x(a) \Big) \\ &= \textstyle\sum_{v \in U} \mathrm{ex}_x(v)\end{aligned}$$

□

# $s$-$t$-Flows and $s$-$t$-Cuts

For $U \subseteq V \setminus \{s\}$ with $t \in U$, the subset of arcs $\delta^-(U)$ is called an $s$-$t$-cut.




**Lemma 10.3** Let $U \subseteq V \setminus \{s\}$ with $t \in U$. The value of a feasible $s$-$t$-flow $x$ is at most the capacity $u(\delta^-(U))$ of the $s$-$t$-cut $\delta^-(U)$. Equality holds if and only if $x(a) = u(a)$ for all $a \in \delta^-(U)$ and $x(a) = 0$ for all $a \in \delta^+(U)$.

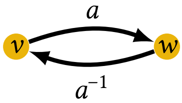
**Proof:**

$$\begin{aligned}
\mathrm{ex}_x(t) &= \mathrm{ex}_x(U) && \text{by Lemma 10.2}\\
&= x(\delta^-(U)) - x(\delta^+(U))\\
&\le x(\delta^-(U)) && \text{with equality iff } x(\delta^+(U)) = 0\\
&\le u(\delta^-(U)) && \text{with equality iff } x(\delta^-(U)) = u(\delta^-(U))
\end{aligned}$$

□

## Residual Graph and Residual Arcs

For $a = (v, w) \in A$, let $a^{-1} := (w, v)$ be the corresponding backward arc and $A^{-1} := \{a^{-1} \mid a \in A\}$.

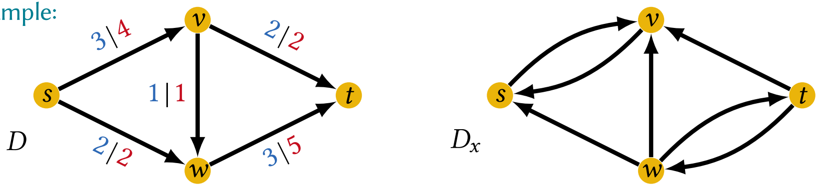



**Definition 10.4** For a feasible flow $x$, the set of residual arcs is given by

$$A_x := \{a \in A \mid x(a) < u(a)\} \cup \{a^{-1} \in A^{-1} \mid x(a) > 0\} \ .$$

The digraph $D_x := (V, A_x)$ is called the residual graph of $x$.

Example:



# Optimality Criterion for Maximum $s$-$t$-Flows 

- a dipath in $D_x$ is called $x$-augmenting path

**Lemma 10.5** If $x$ is a feasible $s$-$t$-flow such that $D_x$ does not contain an $s$-$t$-dipath, then $x$ is a maximum $s$-$t$-flow.

**Proof:**

- let $S := \{v \in V : \text{ there is an } s\text{-}v\text{-dipath in } D_x\}$, let $U := V \setminus S$
- by construction $s \in S$, $t \in U$
- Claim 1: $x_a = u(a)$ for all $a \in \delta^-(U) = \delta^+(S)$
  - for all $(v, w) \in \delta^+(S)$, there is an $s$-$v$-dipath in $D_x$
  - if $x(a) < u_a$, then $a \in A_x$ and there is an $s$-$w$-dipath in $D_x$
- Claim 2: $x(a) = 0$ for all $a \in \delta^+(U) = \delta^-(S)$
  - for all $(v, w) \in \delta^-(S)$, there is an $s$-$w$-dipath in $D_x$
  - if $x(a) > 0$, then $a^{-1} = (w, v) \in A_x$ and there is an $s$-$v$-dipath in $D_x$
- Lemma 10.3 implies the result $\square$

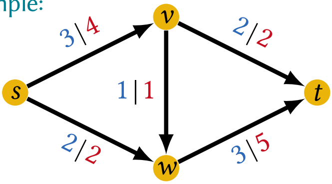
## Residual Capacities

**Definition 10.6** Let $x$ be a feasible flow. For $a \in A$, define

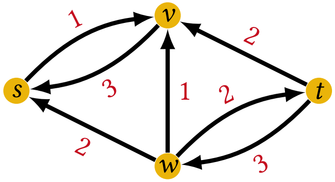
$$u_x(a) := u(a) - x(a) \quad \text{if } a \in A_x, \quad \text{and} \quad u_x(a^{-1}) := x(a) \quad \text{if } a^{-1} \in A_x.$$

The value $u_x(a)$ is called residual capacity of arc $a \in A_x$.

Example:




$D$, $u$, $x$



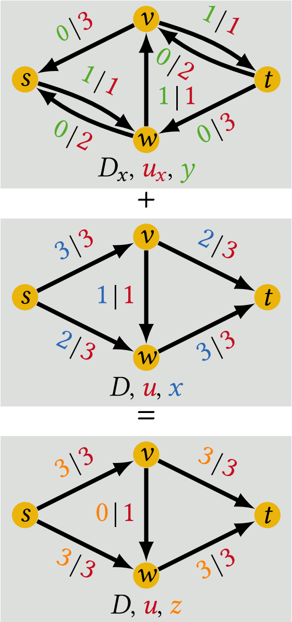

$D_x$, $u_x$

# Adding and Subtracting Flows

- let $x$ be feasible flow in $(D, u)$ and $y$ be feasible flow in $(D_x, u_x)$, then $z := x + y$ defined as

$$z(a) := x(a) + y(a) - y(a^{-1}) \quad \text{for } a \in A$$

  yields a feasible flow $z$ in $D$
- $\mathrm{ex}_z(v) = \mathrm{ex}_x(v) + \mathrm{ex}_y(v)$ for each $v \in V$



# Adding and Subtracting Flows

- let $x$ be feasible flow in $(D, u)$ and $y$ be feasible flow in $(D_x, u_x)$, then $z := x + y$ defined as

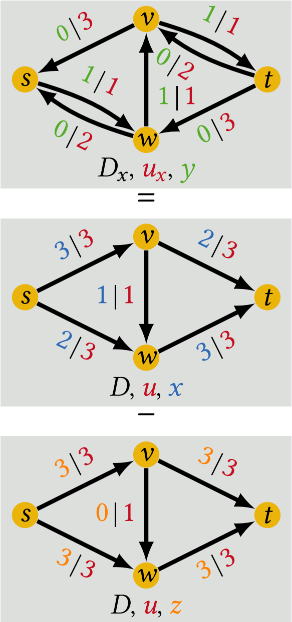
$$z(a) := x(a) + y(a) - y(a^{-1}) \quad \text{for } a \in A$$

  yields a feasible flow $z$ in $D$
- $\mathrm{ex}_z(v) = \mathrm{ex}_x(v) + \mathrm{ex}_y(v)$ for each $v \in V$

- if $x$, $z$ are feasible flows in $(D, u)$, then $y := z - x$ defined as

$$y(a) := \max\{0, z(a) - x(a)\} \text{ for } a \in A \cap A_x$$
$$y(a^{-1}) := \max\{0, x(a) - z(a)\} \text{ for } a^{-1} \in A^{-1} \cap A_x$$

  is a feasible flow $y$ in $D_x$
- $\mathrm{ex}_y(v) = \mathrm{ex}_z(v) - \mathrm{ex}_x(v)$ for each $v \in V$



Introduction to

# Linear and Combinatorial Optimization

# 10 Maximum Flows

## 10.2 Max-Flow-Min-Cut-Theorem

# Max-Flow Min-Cut Theorem

**Theorem 10.7** The maximum $s$-$t$-flow value equals the minimum capacity of an $s$-$t$-cut.

**Proof:**

- let $x$ be a maximal $s$-$t$-flow
- if $D_x$ does not contain an $s$-$t$-dipath, we construct an $s$-$t$-cut whose capacity equals the flow value of $x$ as in the proof of Lemma 10.5
- if $D_x$ contains an $s$-$t$-dipath $P$, let $\delta := \min_{a \in P} u_x(a) > 0$
- define $s$-$t$-flow in $D_x$ by

$$y(a) = \begin{cases} \delta & \text{if } a \in P \\ 0 & \text{otherwise.} \end{cases}$$

- $y$ is a feasible $s$-$t$ flow in $D$ of value $\delta$ in $D_x$
- $z = x + y$ is a feasible $s$-$t$ flow of value $\mathrm{ex}_x(t) + \mathrm{ex}_y(t) = \mathrm{ex}_x(t) + \delta > \mathrm{ex}_x(t)$
- contradicts maximality of $x$

$\square$

# Ford and Fulkerson's Alleged Motivation



In their seminal book *'Flows in Networks'* (1962), Ford & Fulkerson write:

> *"The mathematical problem … of determining a maximal flow … comes up naturally in the study of transportation or communication networks. It was posed to the authors in the spring of 1955 by T. E. Harris, who, in conjunction with General F. S. Ross (Ret.), had formulated a simplified model of railway traffic flow, and pinpointed this particular problem as the central one suggested by the model [11]."*

Let us look into this paper of Harris and Ross.

NOTICE: THIS DOCUMENT CONTAINS INFORMATION AFFECTING THE NATIONAL DEFENSE OF THE UNITED STATES WITHIN THE MEANING OF THE ESPIONAGE LAWS, TITLE 18, U.S.C., SECTIONS 793 and 794. THE TRANSMISSION OR THE REVELATION OF ITS CONTENTS IN ANY MANNER TO AN UNAUTHORIZED PERSON IS PROHIBITED BY LAW.

UNCLASSIFIED

| AD NUMBER | |
|---|---|
| AD093458 | |
| **CLASSIFICATION CHANGES** | |
| TO: | unclassified |
| FROM: | confidential |
| **LIMITATION CHANGES** | |
| TO: | Approved for public release, distribution unlimited |
| FROM: | |
| **AUTHORITY** | |
| USAF 11th Wing ltr., 13 May 1999; Same | |

THIS PAGE IS UNCLASSIFIED

SECRET

U. S. AIR FORCE
# PROJECT RAND

## RESEARCH MEMORANDUM

# FUNDAMENTALS OF A METHOD FOR EVALUATING RAIL NET CAPACITIES (U)

T. E. Harris
F. S. Ross

RM-1573

October 24, 1955 Copy No. 37

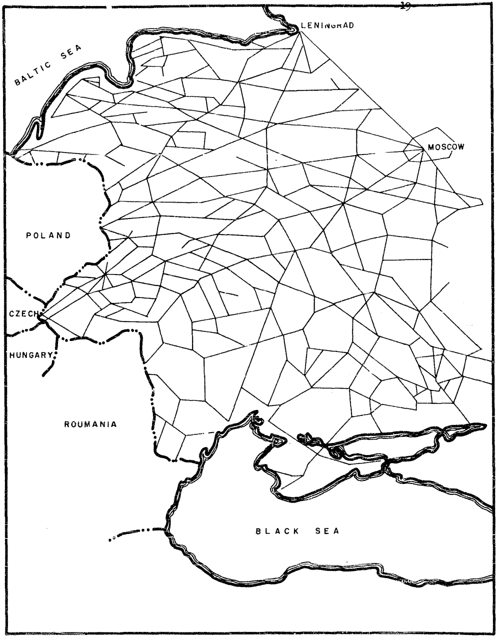



Fig. 1—The railway system of western Russia

# Historical Remark: True Motivation



Unlike what Ford and Fulkerson say, the interest of Harris and Ross was not to find a maximum flow. They write:

> *"Air power is an effective means of interdicting an enemy's rail system, and such usage is a logical and important mission for this arm. ... The present paper describes the fundamentals of a method intended to help the specialist who is engaged in estimating railway capabilities, so that he might more readily accomplish this purpose and thus assist the commander and his staff with greater efficiency than is possible at present."*

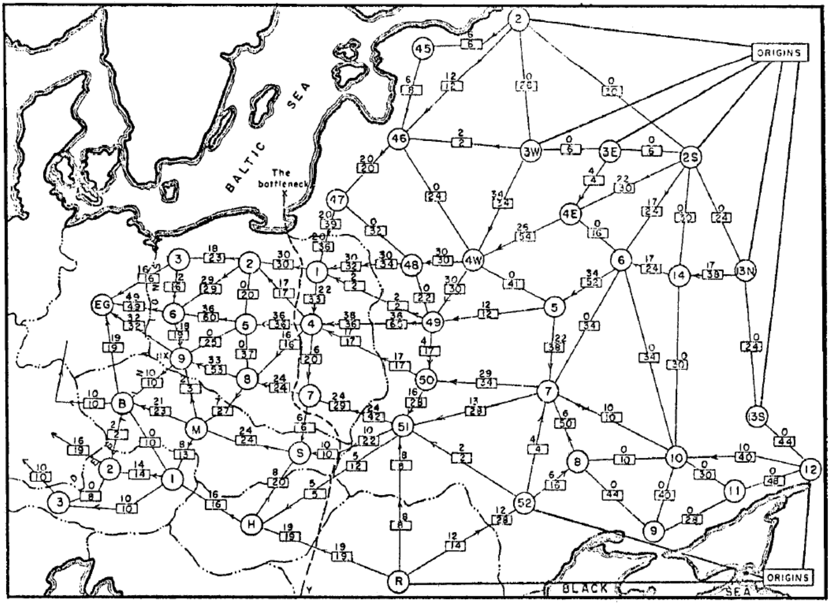



**Fig. 2.** From Harris and Ross [11]: Schematic diagram of the railway network of the Western Soviet Union and Eastern European countries, with a maximum flow of value 163,000 tons from Russia to Eastern Europe, and a cut of capacity 163,000 tons indicated as "The bottleneck"

Introduction to

# Linear and Combinatorial Optimization

# 10 Maximum Flows

## 10.3 Ford-Fulkerson-Algorithm

**Corollary 10.8** A feasible $s$-$t$-flow $x$ is maximum if and only if $D_x$ does not contain an $s$-$t$-dipath.

### Ford-Fulkerson Algorithm

i set $x := 0$;

ii while there is an $s$-$t$-dipath $P$ in $D_x$

iii     set $x := x + \delta \cdot \chi^P$ with $\delta := \min\{u_x(a) \mid a \in P\}$;

$\chi^P \in \{0, 1, -1\}^A$ is the characteristic vector of dipath $P$ defined by

$$\chi^P(a) := \begin{cases} 1 & \text{if } a \in P, \\ -1 & \text{if } a^{-1} \in P, \\ 0 & \text{otherwise,} \end{cases} \qquad \text{for all } a \in A.$$

# Termination of the Ford-Fulkerson Algorithm 

**Theorem 10.9**

a If all capacities are rational, then the algorithm terminates with a maximum $s$-$t$-flow.

b If all capacities are integral, it finds an integral maximum $s$-$t$-flow.

**Proof:** b we prove by induction that $x$ is integral throughout the run of the algorithm

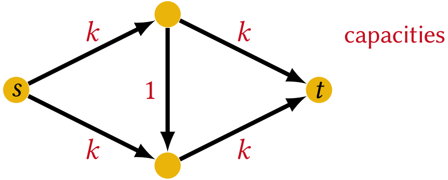
- $x \equiv 0$ at the beginning
- if $x$ is integral, then all residual capacities $u_x(a)$ are integral
- thus $\delta$ is integral and $x + \delta\chi^P$ is integral
- flow value increases in each iteration by at least 1
- the value of any feasible flow is bounded by the finite capacity of an $s$-$t$-cut
- the algorithm terminates after a finite number of iterations

a reduce to integral capacities by scaling

# Running Time of Ford-Fulkerson Algorithm 

Observation: If an arbitrary $x$-augmenting path is chosen in every iteration, the Ford-Fulkerson Algorithm might behave rather badly.

Example:




Remark: There exist instances with finite irrational capacities where the Ford-Fulkerson Algorithm never terminates and the flow value converges to a value that is strictly smaller than the maximum flow value (see exercise).

**Theorem 10.10** If all capacities are integral and the maximum flow value is $K < \infty$, then the Ford-Fulkerson Algorithm terminates after at most $K$ iterations. Its running time is $O(m \cdot K)$ in this case (i.e., pseudo-polynomial).

**Proof:** In each iteration the integral flow value increases by at least 1. □

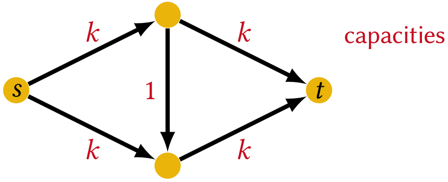
# Edmonds-Karp Algorithm

A variant of the Ford-Fulkerson Algo. is the Edmonds-Karp Algorithm:

- In each iteration, choose shortest $s$-$t$-dipath in $D_x$ (using BFS).

(Here, the length of a dipath $P$ is the number of arcs in $P$.)

Example. For the following digraph, the algorithm needs two iterations.




**Theorem 10.11** The Edmonds-Karp Algorithm terminates after at most $n \cdot m$ iterations; its running time is $O(n \cdot m^2)$.

- $d_x(v, w)$ := length of a shortest $v$-$w$-dipath in $D_x$

**Lemma 10.12** Throughout the iterations of the algorithm, for each $v \in V$ the distances $d_x(s, v)$ and $d_x(v, t)$ never decrease.

**Proof:**

- for a contradiction, assume that there are consecutive iterations with flows $x$ and $x'$ such that $d_x(s, v) > d_{x'}(s, v)$
- choose $v$ with this property and $d_{x'}(s, v) > 0$ minimal
- let $P$ be a shortest $s$-$v$-path in $D_{x'}$ and $w$ be the predecessor of $v$ on $P$

$$d_x(s, v) > d_{x'}(s, v) = d_{x'}(s, w) + 1 \geq d_x(s, w) + 1$$

- implies that $a = (w, v) \in A_{x'} \setminus A_x$
- in $D_x$ flow was augmented along a shortest $s$-$t$-dipath containing $a^{-1} = (v, w)$, thus $d_x(s, w) = d_x(s, v) + 1$
- we obtain $d_x(s, v) > d_x(s, w) + 1 = d_x(s, v) + 2$, a contradiction
- analogous argument for $d_x(v, t)$ $\square$

## Prerequisites for the Proof of Thm. 10.11 (Cont.)



- $\tilde{A}_x := \{a \in A_x \mid a \text{ is on some shortest } s\text{-}t\text{-path in } D_x\}$.

**Lemma 10.13** If $d_x(s,t) = d_{x'}(s,t)$ for two consecutive flows $x$ and $x'$, then $\tilde{A}_{x'} \subsetneq \tilde{A}_x$.

**Proof:**

- for a contradiction, suppose that $d_x(s,t) = d_{x'}(s,t)$, but there is $a = (v,w) \in \tilde{A}_{x'} \setminus \tilde{A}_x$, then, by Lemma 10.12,

$$d_x(s,t) = d_{x'}(s,t) = d_{x'}(s,v) + 1 + d_{x'}(w,t) \geq d_x(s,v) + 1 + d_x(w,t)$$

- since $a \notin \tilde{A}_x$, in $D_x$ flow was augmented along shortest $s$-$t$-dipath containing $a^{-1} = (w,v)$, thus, we obtain the contradiction

$$\begin{aligned} d_x(s,t) &= d_x(s,w) + 1 + d_x(v,t) \\ &= (d_x(s,v) - 1) + 1 + (d_x(w,t) - 1) \\ &= d_x(s,v) + d_x(w,t) - 1 \end{aligned}$$

- thus $\tilde{A}_{x'} \subseteq \tilde{A}_x$, for the bottleneck arc $a \in \tilde{A}_x$ on the shortest $s$-$t$-dipath in $D_x$ chosen by the algorithm, we have $a \notin \tilde{A}_{x'}$, thus, $\tilde{A}_{x'} \subsetneq \tilde{A}_x$ □

# Proof of Thm. 10.11



**Proof:**

- by Lemma 10.12, $d_x(s,t)$ is non-decreasing and $1 \le d_x(s,t) \le n-1$ throughout the algorithm
- by Lemma 10.13, there can be at most $m$ consectuive iterations without increase of $d_x(s,t)$
- the algorithm has at most $nm$ iterations

# 10 Maximum Flows

## 10.4 Flows and LPs

# Arc-Based LP Formulation for Maximum Flows



Straightforward LP formulation of the maximum $s$-$t$-flow problem:

$$
\begin{array}{lll}
\max & \displaystyle\sum_{a\in\delta^+(s)} x_a - \sum_{a\in\delta^-(s)} x_a & \\
\text{s.t.} & \displaystyle\sum_{a\in\delta^-(v)} x_a - \sum_{a\in\delta^+(v)} x_a = 0 & \text{for all } v \in V \setminus \{s,t\} \\
& x_a \leq u(a) & \text{for all } a \in A \\
& x_a \geq 0 & \text{for all } a \in A
\end{array}
$$

Dual LP:

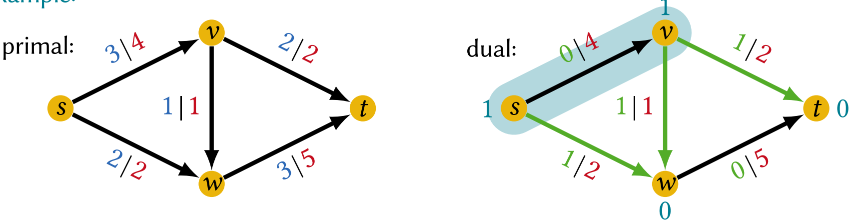
$$
\begin{array}{lll}
\min & \displaystyle\sum_{a\in A} u(a) \cdot z_a & \\
\text{s.t.} & y_w - y_v + z_{(v,w)} \geq 0 & \text{for all } (v,w) \in A \\
& y_s = 1, \quad y_t = 0 & \\
& z_a \geq 0 & \text{for all } a \in A
\end{array}
$$

# Dual Solutions and $s$-$t$-Cuts

$$
\begin{array}{lll}
\min & \displaystyle\sum_{a\in A} u(a)\cdot z_a & \\
\text{s.t.} & y_w - y_v + z_{(v,w)} \geq 0 & \text{for all } (v,w)\in A \\
& y_s = 1, \quad y_t = 0 & \\
& z_a \geq 0 & \text{for all } a \in A
\end{array}
$$

- $s$-$t$-cut $\delta^+(U)$, $U \subseteq V \setminus \{t\}$, $s \in U$ yields feasible dual solution $(y,z)$ of value $u(\delta^+(U))$ where $y = \chi^U$ and $z = \chi^{\delta^+(U)}$

Example:



# Dual Solutions and $s$-$t$-Cuts

$$
\begin{aligned}
\min \quad & \sum_{a \in A} u(a) \cdot z_a & \\
\text{s.t.} \quad & y_w - y_v + z_{(v,w)} \geq 0 & \text{for all } (v,w) \in A \\
& y_s = 1, \quad y_t = 0 & \\
& z_a \geq 0 & \text{for all } a \in A
\end{aligned}
$$

- $s$-$t$-cut $\delta^+(U)$, $U \subseteq V \setminus \{t\}$, $s \in U$ yields feasible dual solution $(y, z)$ of value $u(\delta^+(U))$ where $y = \chi^U$ and $z = \chi^{\delta^+(U)}$

**Theorem 10.14** There exists an $s$-$t$-cut $\delta^+(U)$ (with $U \subseteq V \setminus \{t\}$, $s \in U$) such that the corresponding dual solution $(y, z)$ is an optimal dual solution.

**Proof:** follows directly from the Max-Flow-Min-Cut-Theorem. □

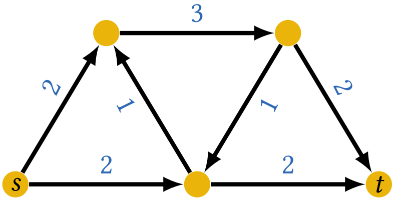
# Flow Decomposition

**Theorem 10.15** For an $s$-$t$-flow $x$ in $D$, there exist $s$-$t$-dipaths $P_1, \ldots, P_k$ and dicycles $C_1, \ldots, C_\ell$ in $D$ with $k + \ell \leq m$ and $y_{P_1}, \ldots, y_{P_k}, y_{C_1}, \ldots, y_{C_\ell} \geq 0$ with

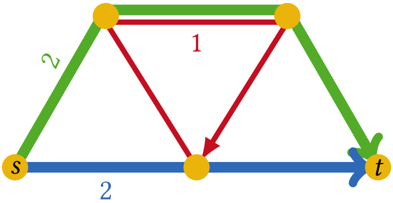
$$x_a = \sum_{i\,:\, a \in P_i} y_{P_i} + \sum_{j\,:\, a \in C_j} y_{C_j} \qquad \text{for all } a \in A.$$

Moreover, the value of $x$ is $\sum_{i=1}^{k} y_{P_i}$.

Example: arc flow




flow decomposition

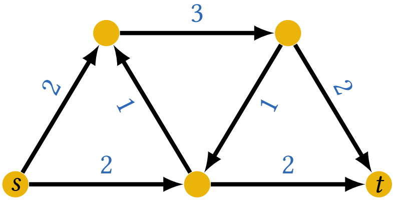


**Theorem 10.15** For an $s$-$t$-flow $x$ in $D$, there exist $s$-$t$-dipaths $P_1, \dots, P_k$ and dicycles $C_1, \dots, C_\ell$ in $D$ with $k + \ell \leq m$ and $y_{P_1}, \dots, y_{P_k}, y_{C_1}, \dots, y_{C_\ell} \geq 0$ with

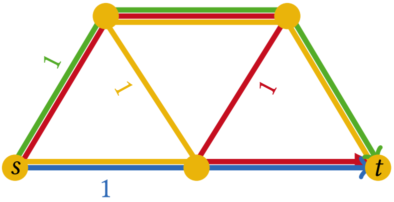
$$x_a = \sum_{i\,:\,a \in P_i} y_{P_i} + \sum_{j\,:\,a \in C_j} y_{C_j} \qquad \text{for all } a \in A.$$

Moreover, the value of $x$ is $\sum_{i=1}^{k} y_{P_i}$.

Example: arc flow




alternative flow decomposition



**Theorem 10.15** For an $s$-$t$-flow $x$ in $D$, there exist $s$-$t$-dipaths $P_1, \ldots, P_k$ and dicycles $C_1, \ldots, C_\ell$ in $D$ with $k + \ell \leq m$ and $y_{P_1}, \ldots, y_{P_k}, y_{C_1}, \ldots, y_{C_\ell} \geq 0$ with

$$x_a = \sum_{i\,:\,a \in P_i} y_{P_i} + \sum_{j\,:\,a \in C_j} y_{C_j} \qquad \text{for all } a \in A.$$

Moreover, the value of $x$ is $\sum_{i=1}^{k} y_{P_i}$.

**Proof:** by induction on the support of $x$

- choose $a = (v, w) \in A$ with $x(a) > 0$ arbitrarily
- if $v \neq s$, by flow conservation, there $a' \in \delta^-(v)$ with $x(a') > 0$
- if $w \neq t$, by flow conservation, there is $a'' \in \delta^+(w)$ with $x(a'') > 0$
- iterating this argument yields either an $s$-$t$-dipath or a dicycle $Q$
- set $y_Q := \min_{a \in Q} x(a)$, $x' = x - y_Q \cdot \chi^Q$ $\square$

- for an $s$-$t$-flow $x$ with flow decomposition as in Theorem 10.15, let $x'_a := \sum_{i\,:\,a \in P_i} y_{P_i}$ for all $a \in A$. Then $x'$ is an $s$-$t$-flow of the same value as $x$ and $x'_a \leq x_a$ for all $a \in A$.

# Path-Based LP Formulation

Let $\mathcal{P}$ be the set of all $s$-$t$-dipaths in $D$.

$$
\begin{aligned}
\max \quad & \sum_{P \in \mathcal{P}} y_P \\
\text{s.t.} \quad & \sum_{P \in \mathcal{P} \,:\, a \in P} y_P \le u(a) && \text{for all } a \in A \\
& y_P \ge 0 && \text{for all } P \in \mathcal{P}
\end{aligned}
$$

Dual LP:

$$
\begin{aligned}
\min \quad & \sum_{a \in A} u(a) \cdot z_a \\
\text{s.t.} \quad & \sum_{a \in P} z_a \ge 1 && \text{for all } P \in \mathcal{P} \\
& z_a \ge 0 && \text{for all } a \in A
\end{aligned}
$$

Remarks.

Notice that $|\mathcal{P}|$ and thus the number of variables of the primal LP and the number of constraints of the dual LP can be exponential in $n$ and $m$.

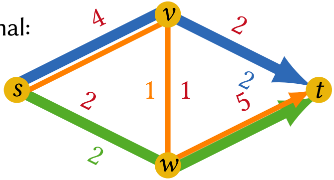
# Dual Solutions and $s$-$t$-Cuts

$$\begin{array}{lll} \min & \sum_{a \in A} u(a) \cdot z_a & \\ \text{s.t.} & \sum_{a \in P} z_a \geq 1 & \text{for all } P \in \mathcal{P} \\ & z_a \geq 0 & \text{for all } a \in A \end{array}$$

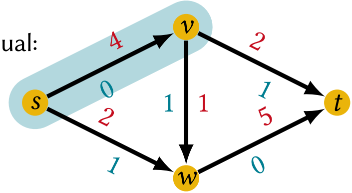
- $s$-$t$-cut $\delta^+(U)$, $U \subseteq V \setminus \{t\}$, $s \in U$ yields feasible dual solution $z$ of value $u(\delta^+(U))$ where $z = \chi^{\delta^+(U)}$

Example:

primal:




dual:



# Dual Solutions and $s$-$t$-Cuts

$$
\begin{aligned}
\min \quad & \sum_{a \in A} u(a) \cdot z_a & & \\
\text{s.t.} \quad & \sum_{a \in P} z_a \geq 1 & & \text{for all } P \in \mathcal{P} \\
& z_a \geq 0 & & \text{for all } a \in A
\end{aligned}
$$

- $s$-$t$-cut $\delta^+(U)$, $U \subseteq V \setminus \{t\}$, $s \in U$ yields feasible dual solution $z$ of value $u(\delta^+(U))$ where $z = \chi^{\delta^+(U)}$

**Theorem 10.16** There exists an $s$-$t$-cut $\delta^+(U)$ (with $U \subseteq V \setminus \{t\}$, $s \in U$) such that the corresponding dual solution $z$ is an optimal dual solution.

Introduction to

# Linear and Combinatorial Optimization

# 10 Maximum Flows

## 10.5 Preflow-Push-Algorithm

# Another Algorithmic Approach

- a feasible flow $x$ is a maximum $s$-$t$-flow if it fulfills two conditions:
  - i $\mathrm{ex}_x(v) = 0$ for all $v \in V \setminus \{s, t\}$; (flow conservation)
  - ii there is no $s$-$t$-dipath in $D_x$.
- Ford-Fulkerson and Edmonds-Karp always fulfill the first condition and terminate as soon as the second condition is fulfilled.
- the Goldberg-Tarjan Algorithm (or Push-Relabel Algorithm, or Preflow-Push Algorithm) always fulfills the second condition and terminates as soon as the first condition is fulfilled.

**Definition 10.17**

- i A flow $x$ is called preflow if $\mathrm{ex}_x(v) \geq 0$ for all $v \in V \setminus \{s\}$.
- ii A node $v \in V \setminus \{s, t\}$ is called active if $\mathrm{ex}_x(v) > 0$.

- a preflow is feasible if it satisfies the capacity constraints

# Valid Labelings and Admissible Arcs



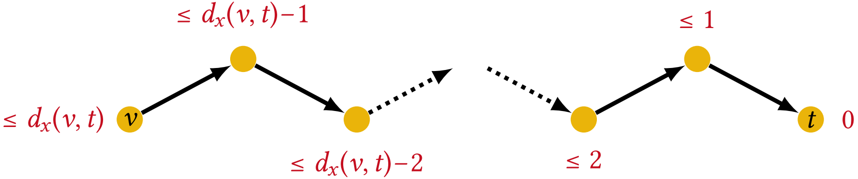
**Definition 10.18** Let $x$ be a preflow. A function $d : V \longrightarrow \mathbb{Z}_{\geq 0}$ with $d(s) = n$ and $d(t) = 0$ is called valid labeling if $d(v) \leq d(w) + 1$ for all $(v, w) \in A_x$.
An arc $(v, w) \in A_x$ is called admissible if $v$ is active and $d(v) = d(w) + 1$.

- $d_x(v, w)$ := length of a shortest $v$-$w$-dipath in $D_x$ (in terms of # arcs)

**Observation 10.19** Let $x$ be a feasible preflow and $d$ a valid labeling. Then $d_x(v, t) \geq d(v)$ for all $v \in V$.

**Proof:**

- consider $v \in V$ with $d_x(v, t) < \infty$, i.e., there is a shortest $v$-$t$-dipath in $D_x$



# Valid Labelings and Admissible Arcs (Cont.)

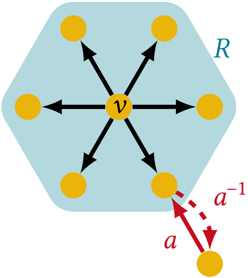
**Lemma 10.20** Let $x$ be a feasible preflow and $d$ a valid labeling.

a There is a $v$-$s$-dipath in $D_x$ for every active node $v$.

b There is no $s$-$t$-dipath in $D_x$.

**Proof:** a

- let $v$ be an active node
- $R := \{w \in V \mid \text{ there is a } v\text{-}w\text{-dipath in } D_x\}$
- we show that $s \in R$
- by construction $x(a) = 0$ for all $a \in \delta^-(R)$
- $\sum_{w\in R} \mathrm{ex}_x(w) = \mathrm{ex}_x(R) = \sum_{a\in\delta^-(R)} x(a) - \sum_{a\in\delta^+(R)} x(a) \leq 0$
- $\sum_{w\in R} \mathrm{ex}_x(w) \leq 0$ and $\mathrm{ex}_x(v) > 0$ together imply that $s \in R$




b follows from $d(s) = n$ and Observation 10.19

### Goldberg-Tarjan Algorithm

1 for all $a \in \delta^+(s)$ set $x(a) := u(a)$;
for all $a \in A \setminus \delta^+(s)$ set $x(a) := 0$;
set $d(s) := n$; for all $v \in V \setminus \{s\}$ set $d(v) := 0$;

2 while there is an active node $v$ do

3 if there is no admissible arc $a \in \delta^+_{D_x}(v)$ then `Relabel`(v);

4 choose an admissible arc $a \in \delta^+_{D_x}(v)$ and `Push`(a);

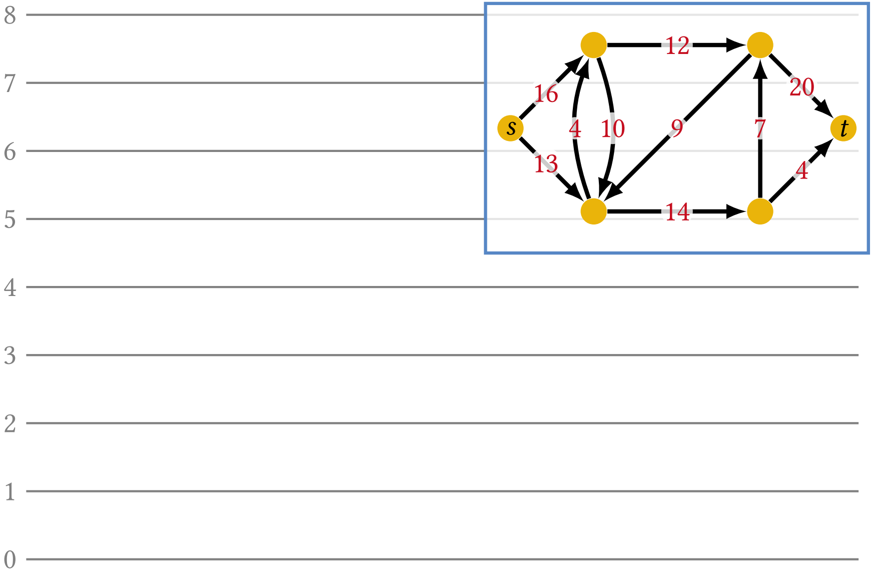
### `Relabel`(v)

set $d(v) := \min\{d(w) + 1 \mid (v, w) \in A_x\}$;

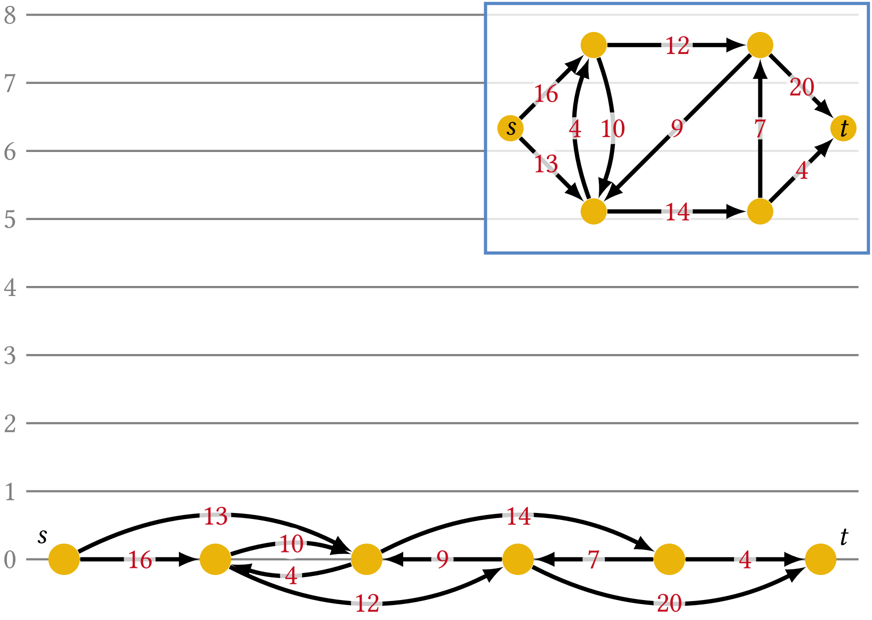
### `Push`(a)

augment $x$ along arc $a$ by $\gamma := \min\{\mathrm{ex}_x(v), u_x(a)\}$; $(a \in \delta^+_{D_x}(v))$

# Example

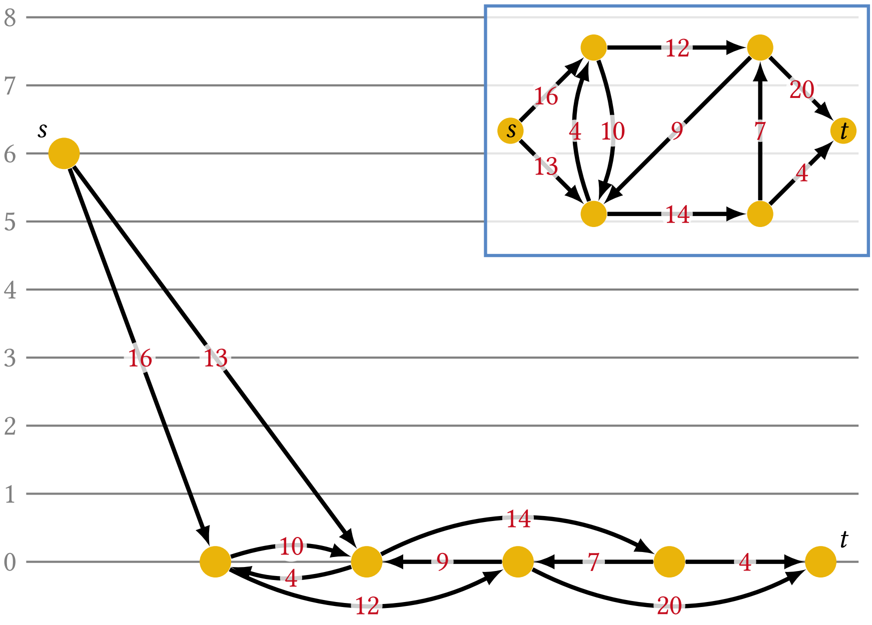


# Example



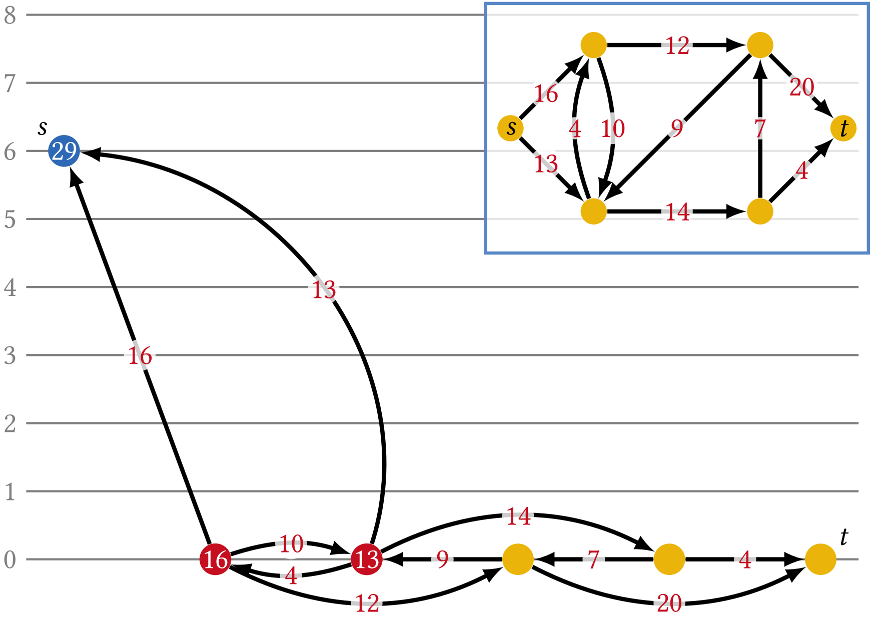
# Example





# Example

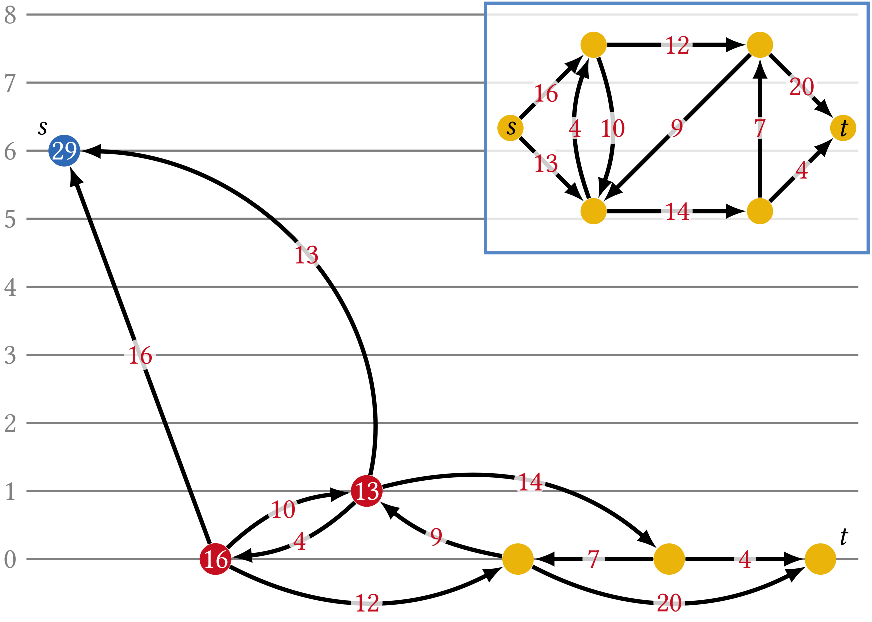




8
7
6
5
4
3
2
1
0
s
29
16
13
t
16
13
10
4
12
14
9
7
4
20
s
t
16
13
4
10
12
9
20
7
14
4

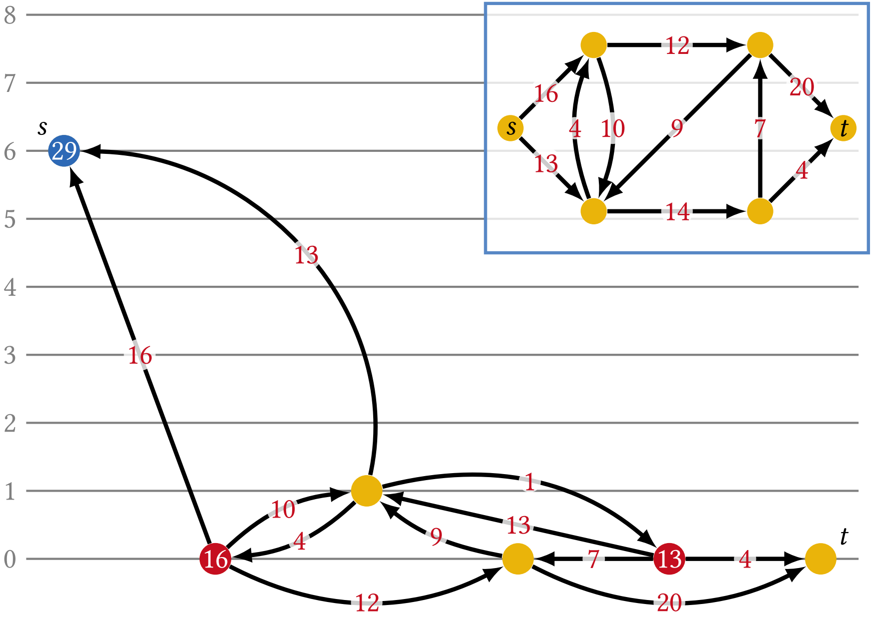
## Example



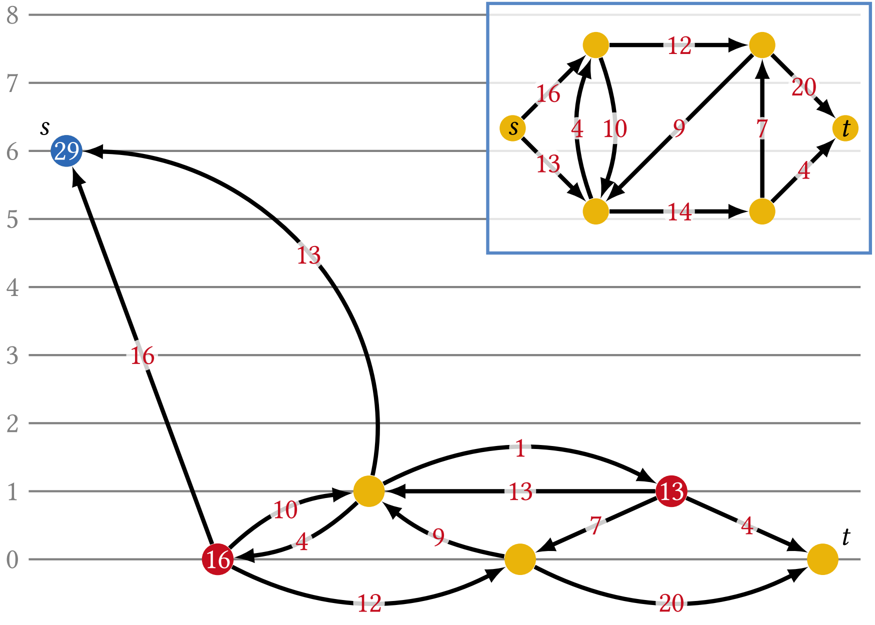


8
7
6
5
4
3
2
1
0
s
29
16
13
13
10
4
14
9
7
4
12
20
t
16
10
12
20
s
16
4
10
9
7
t
13
4
14

# Example

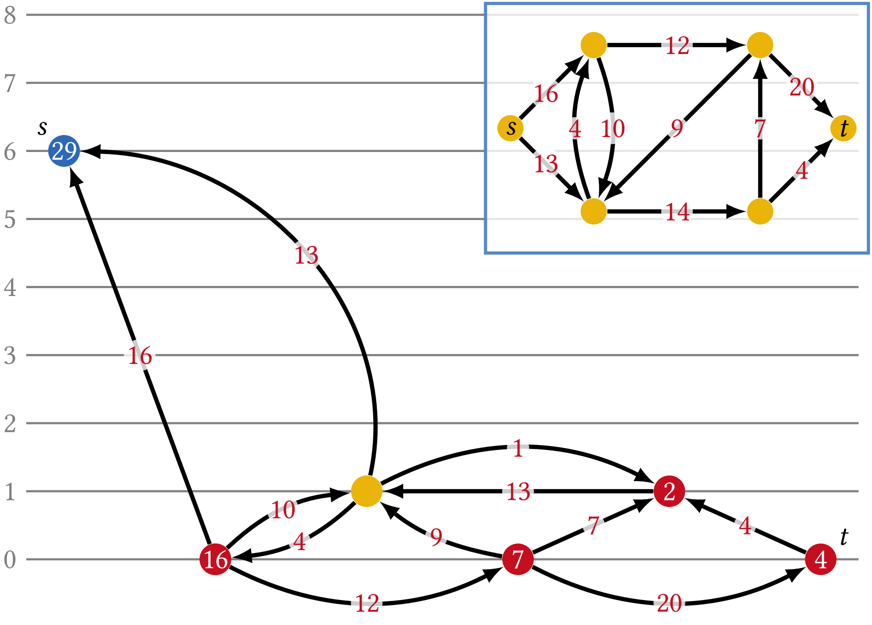

8
7
6
5
4
3
2
1
0
s
29
16
13
10
4
1
13
9
7
13
4
t
12
20
s
t
16
13
12
4
10
9
7
20
4
14

# Example



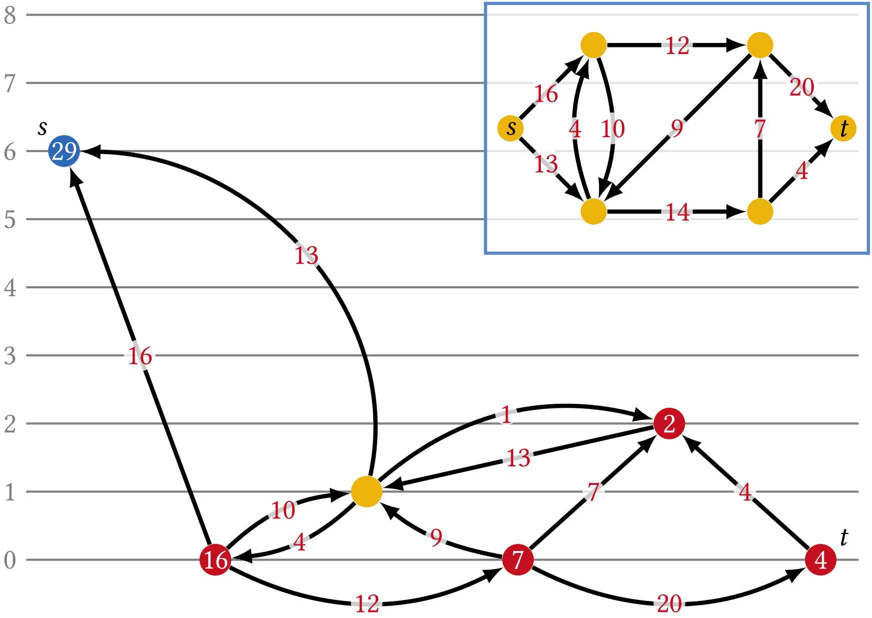
# Example




8
7
6
5
4
3
2
1
0
s
29
16
13
10
4
1
13
2
9
7
16
7
4
4
t
12
20
16
12
20
s
t
4
10
9
7
13
4
14

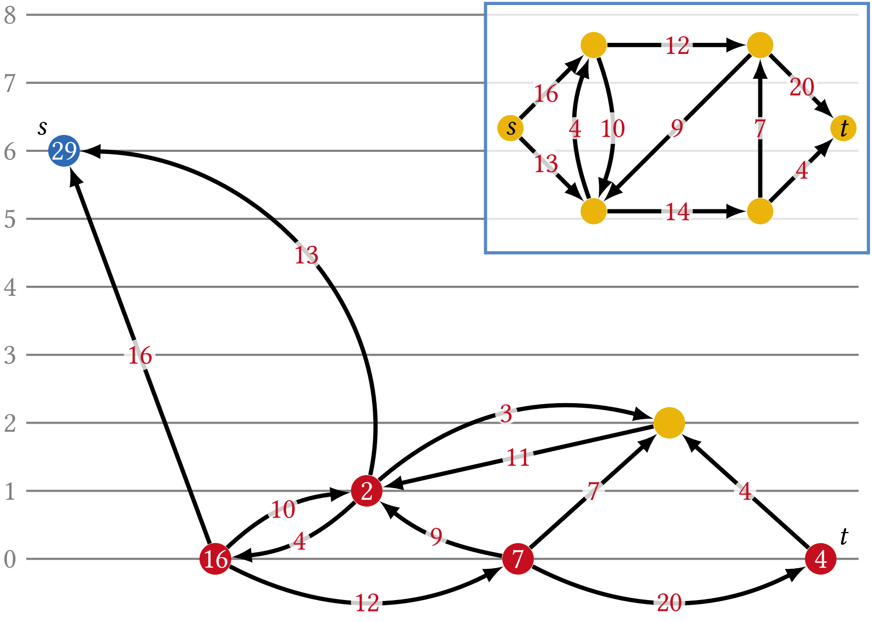
# Example





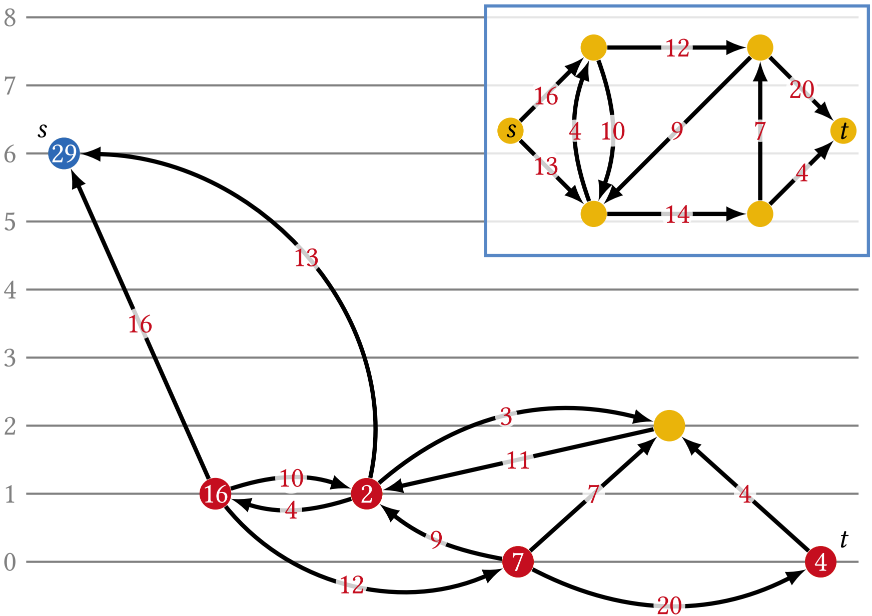
8
7
6
5
4
3
2
1
0
s
29
16
13
1
2
13
10
4
9
7
4
t
16
12
7
20
4
12
s
16
13
4
10
9
7
20
t
4
14

## Example

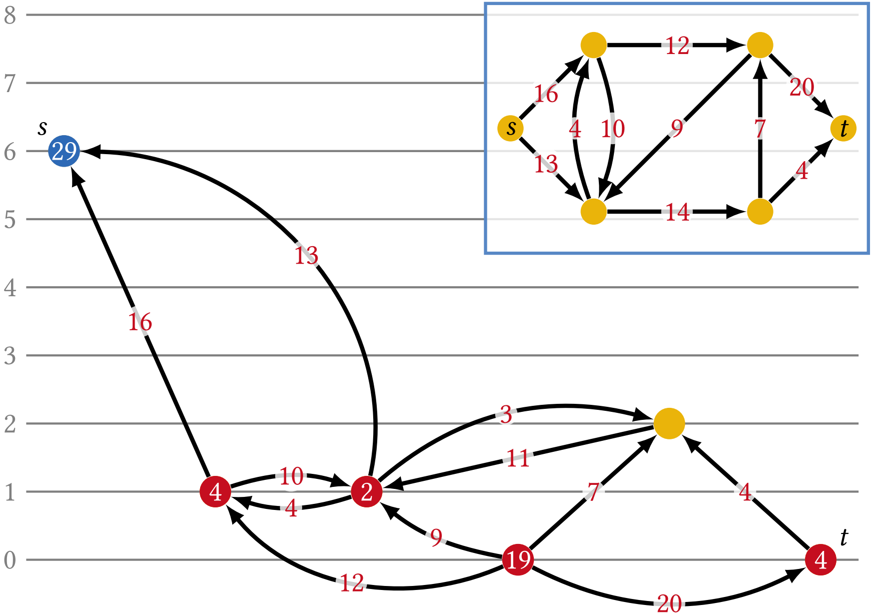


## Example



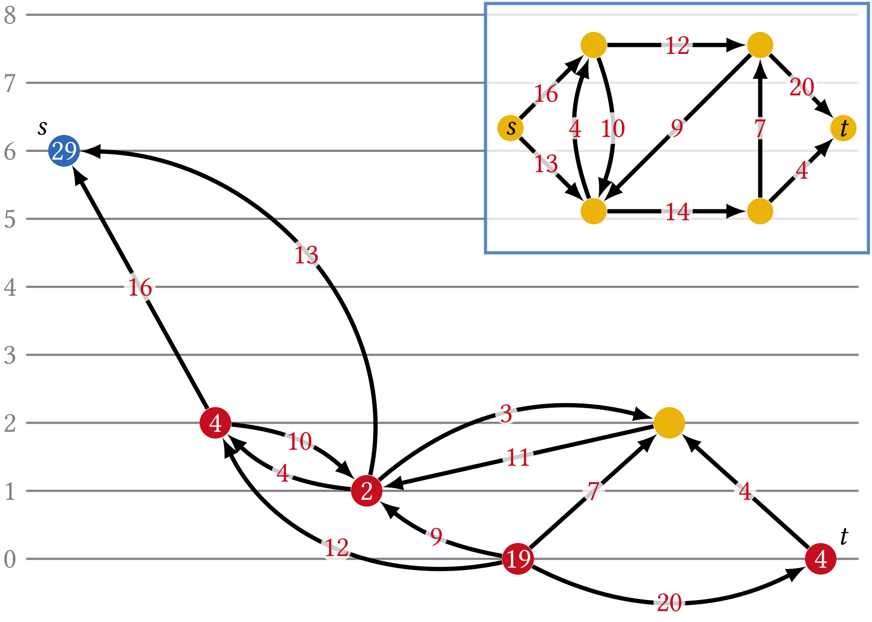

8
7
6
5
4
3
2
1
0
s
29
16
13
16
2
10
4
3
11
7
4
9
7
12
20
4
t
s
t
12
16
20
4
10
9
7
13
4
14

# Example


8
7
6
5
4
3
2
1
0
s
29
t
16
13
10
4
3
11
7
9
12
20
4
2
19
4
16
13
12
4
10
9
7
20
4
14

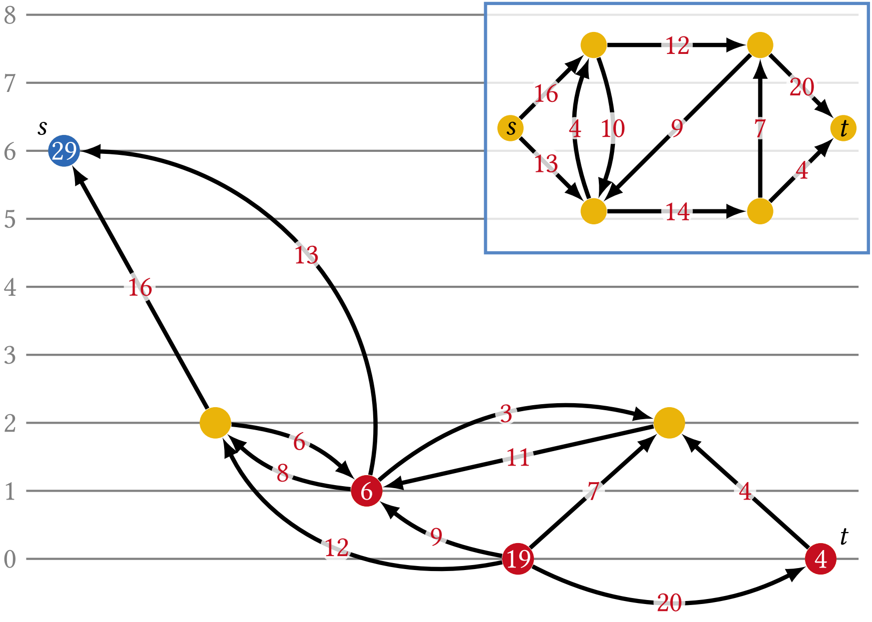
# Example




8
7
6
5
4
3
2
1
0
s
29
16
13
4
10
4
2
3
11
7
4
9
12
19
20
4
t
12
16
20
s
4
10
9
7
t
13
4
14

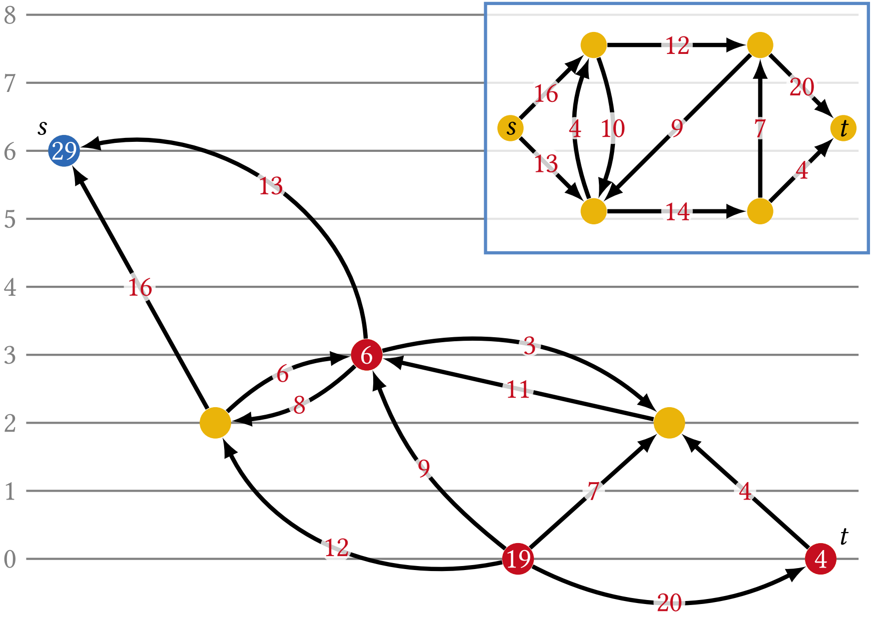
# Example




8
7
6
5
4
3
2
1
0
s
29
16
13
6
8
3
11
6
7
4
9
12
19
20
4
t
12
16
20
s
4
10
9
7
t
13
4
14

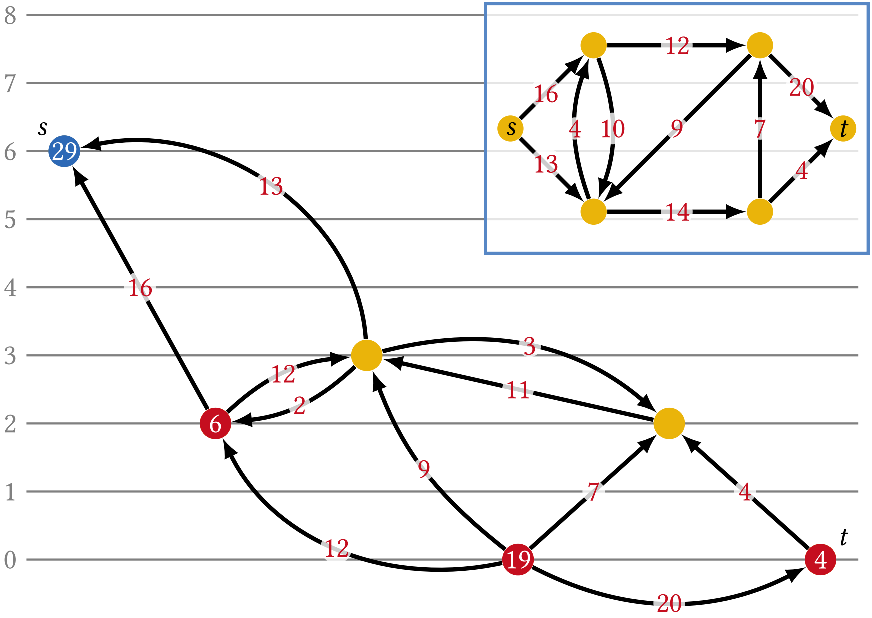
## Example





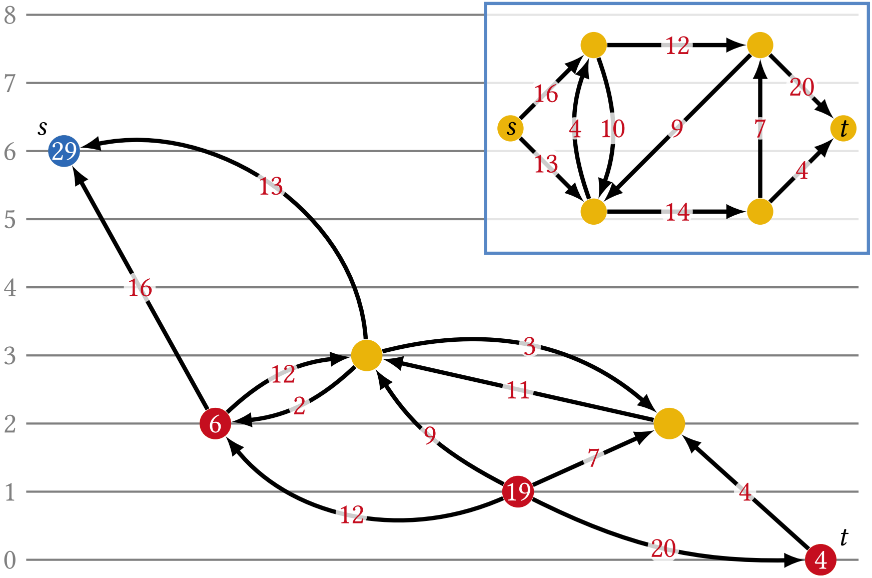
8
7
6
5
4
3
2
1
0
s
29
13
16
6
3
6
8
11
9
7
4
12
19
20
4
t
12
16
20
s
4
10
9
7
t
13
4
14

# Example


8
7
6
5
4
3
2
1
0
s
29
16
13
6
12
2
3
11
9
7
4
12
19
20
4
t
s
t
16
13
4
10
12
9
7
20
4
14

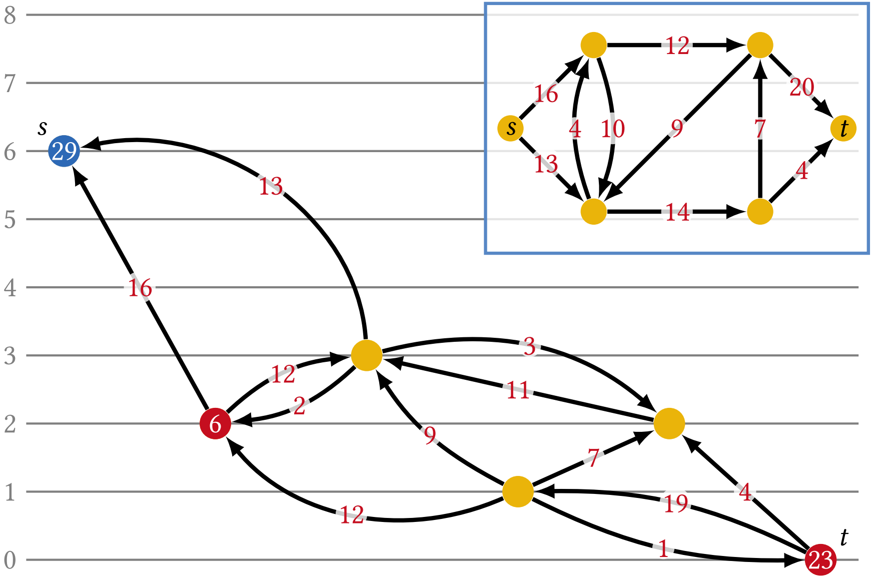
# Example





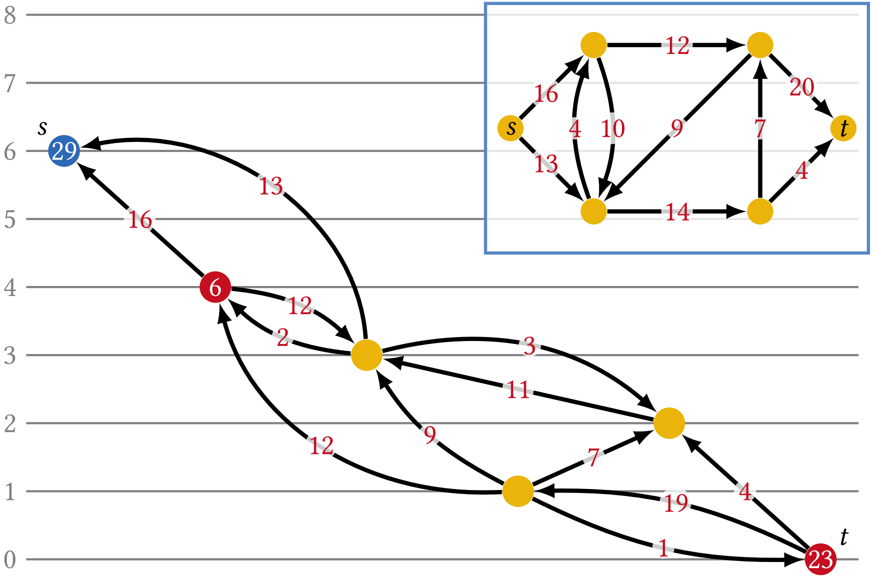
8
7
6
5
4
3
2
1
0
s
29
t
4
6
19
16
13
12
2
3
11
9
7
4
12
20
s
t
16
13
4
10
12
9
14
7
20
4

## Example


8
7
6
5
4
3
2
1
0
s
29
6
23
t
16
13
12
2
3
11
9
7
12
19
4
1
s
t
16
13
12
4
10
9
7
20
14
4

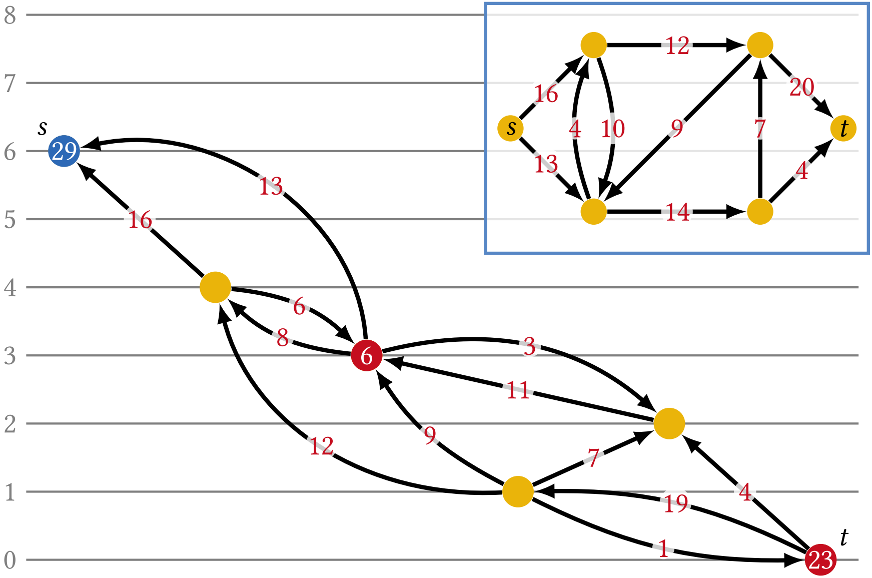
# Example




8
7
6
5
4
3
2
1
0
s
29
6
23
t
16
13
12
2
3
11
9
12
7
19
4
1
s
t
16
13
4
10
12
9
7
20
14
4

# Example

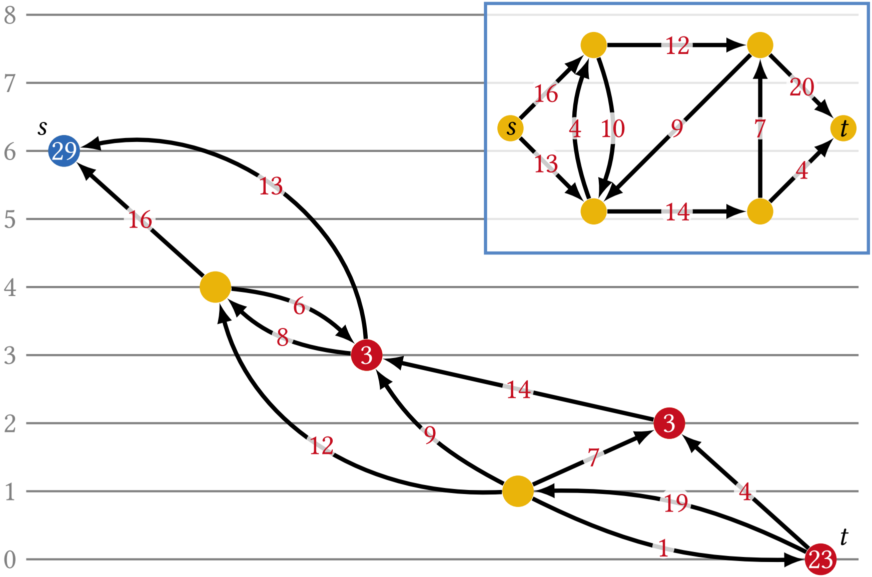




8
7
6
5
4
3
2
1
0
s
29
16
13
6
8
6
3
11
9
12
7
4
19
1
23
t
s
16
13
12
4
10
9
7
20
4
14
t

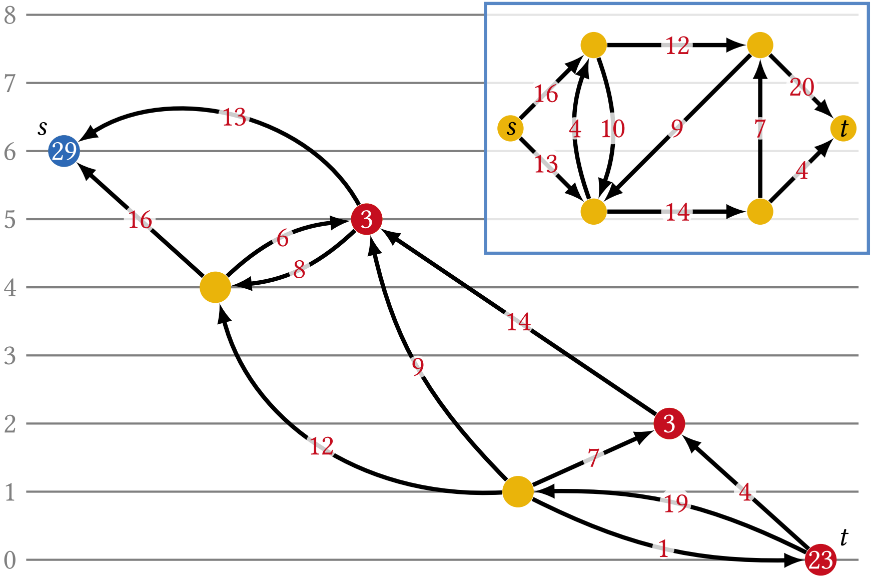
# Example





8
7
6
5
4
3
2
1
0
s
29
16
13
6
8
3
14
3
12
9
7
19
4
1
t
23
s
16
13
4
10
12
9
7
20
14
4
t

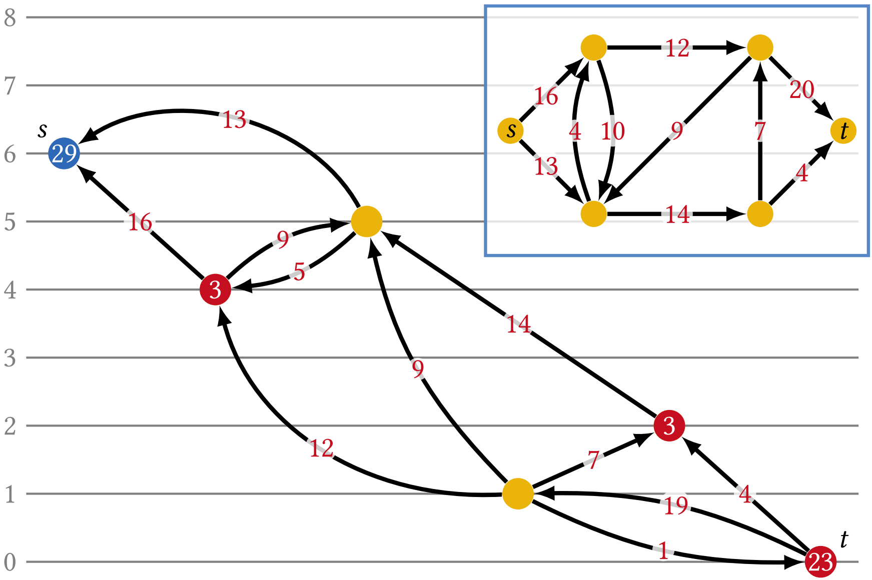
## Example



8
7
6
5
4
3
2
1
0
s
29
t
23
3
3
13
16
6
8
14
9
12
7
19
4
1
s
t
12
16
20
4
10
9
7
13
4
14

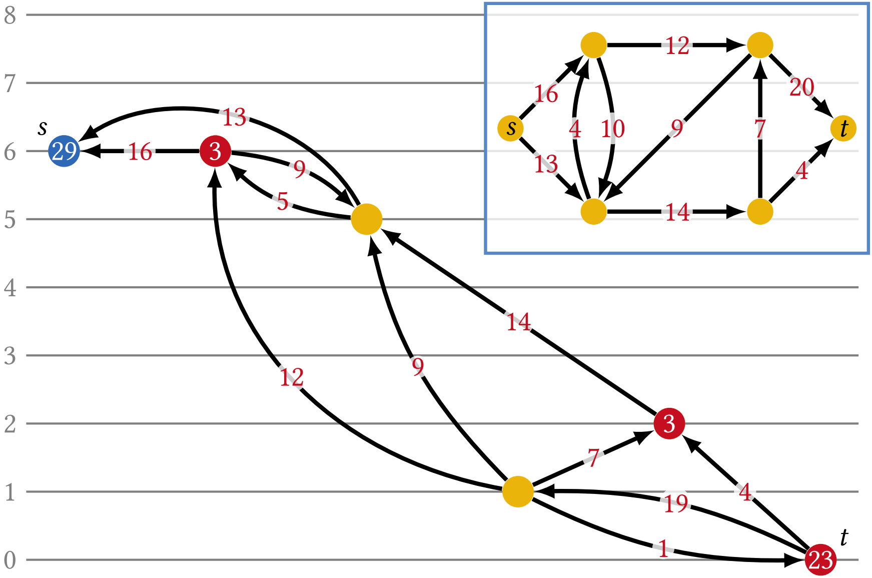
# Example





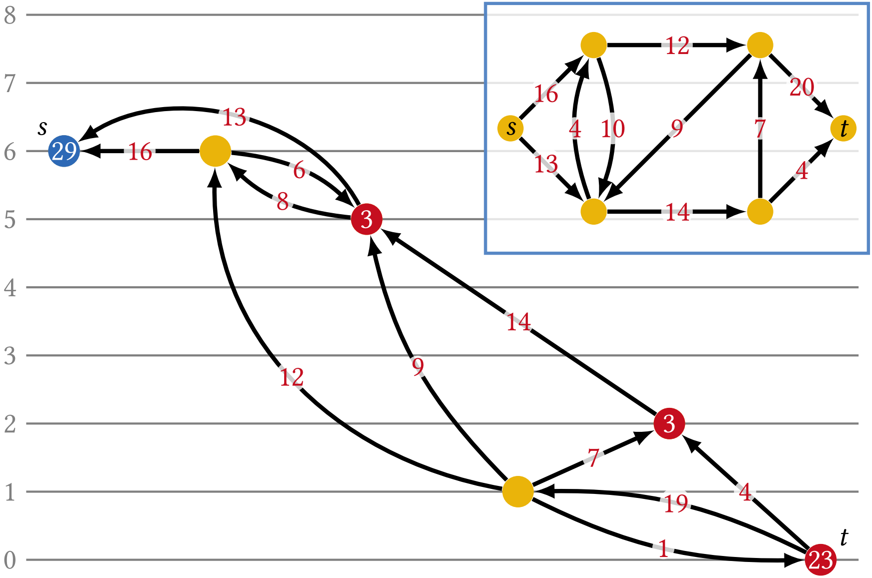
8
7
6
5
4
3
2
1
0
s
29
13
16
9
5
3
14
9
12
3
7
4
19
1
23
t
12
16
20
s
4
10
9
7
t
13
4
14

# Example


8
7
6
5
4
3
2
1
0
s
29
16
3
13
9
5
14
12
9
3
7
4
19
1
23
t
s
16
13
4
10
12
9
7
20
14
4
t

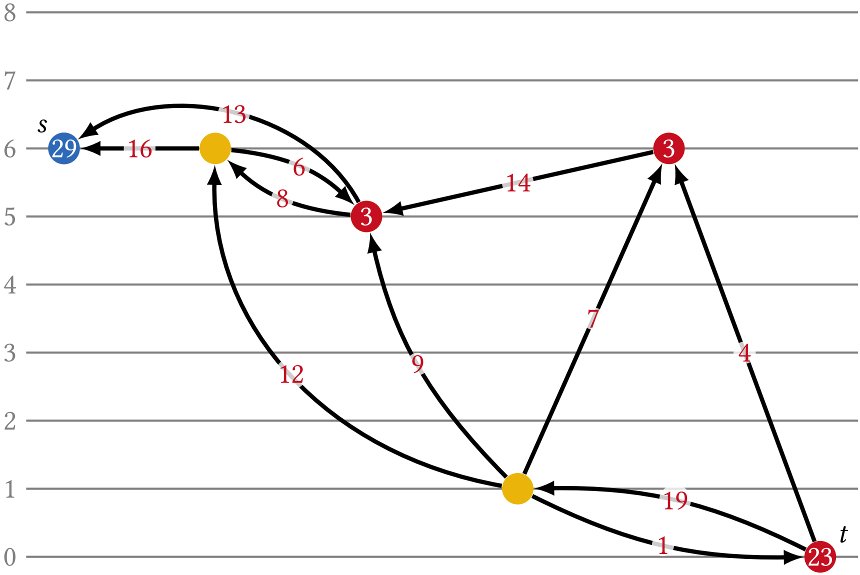
# Example




8
7
6
5
4
3
2
1
0
s
29
16
13
6
8
3
14
12
9
3
7
19
4
1
23
t
12
16
20
s
4
10
9
7
t
13
4
14

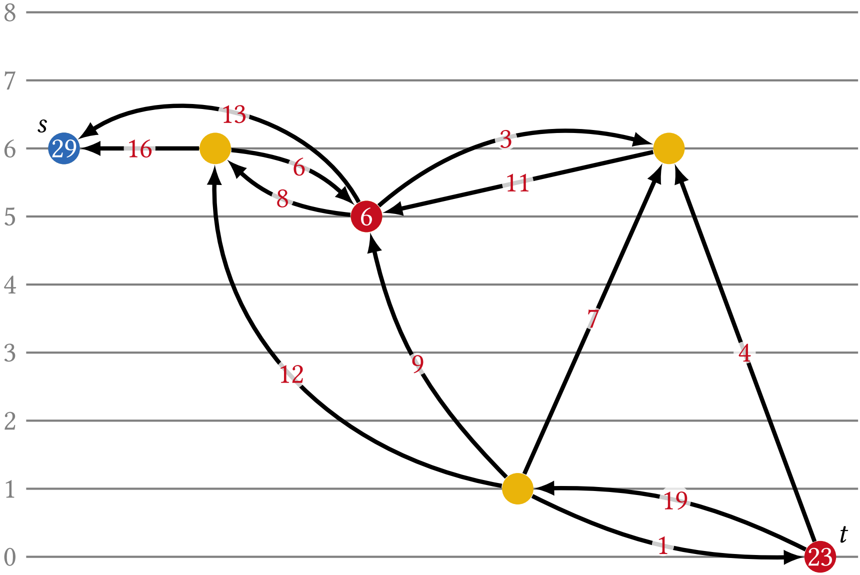
# Example





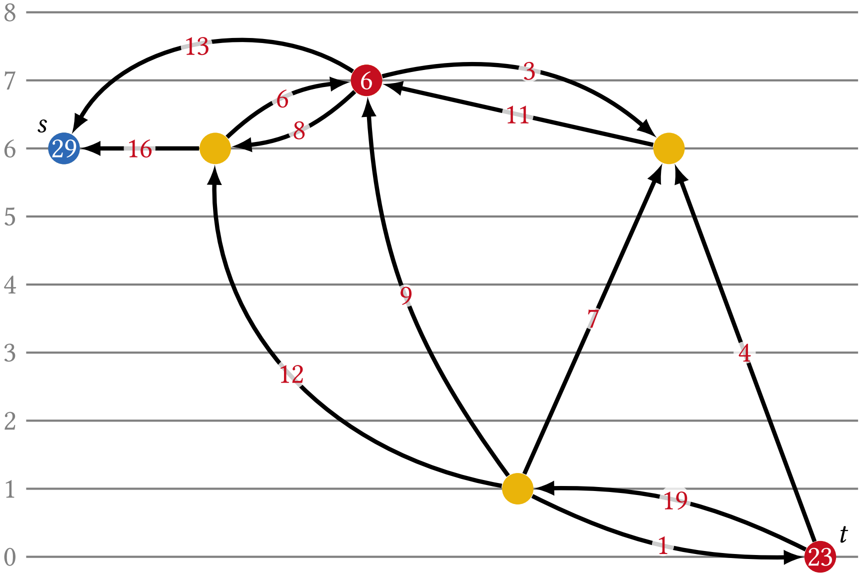
# Example





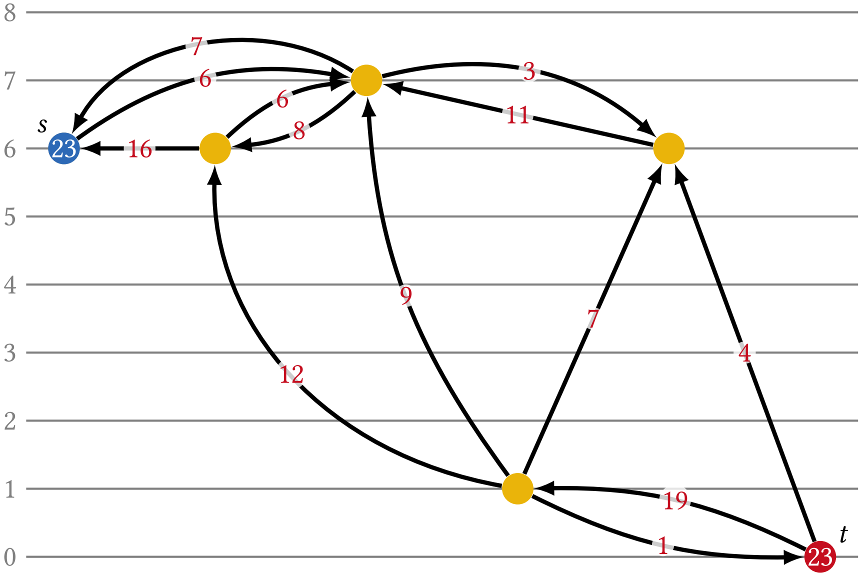
# Example





# Example





**Lemma 10.21** At any stage of the algorithm, $x$ is a feasible preflow and $d$ a valid labeling.

**Proof:** feasible preflow

- algorithm starts with feasible preflow
- $x$ is only modified during `Push` operations
- `Push` operations obviously maintain capacity constraints and $\mathrm{ex}_x(v) \geq 0$

valid labeling

- algorithm starts with valid labeling
- `Relabel` operation sets $d(v) := \min\{d(w) + 1 \mid (v, w) \in A_x\}$ so that after relabeling $d(v) \leq d(w) + 1$ for all $(v, w) \in A_x$
- `Push` for arc $a = (v, w)$ is applied only if $d(v) = d(w) + 1$, this may add the backward arc $(w, v)$ to $A_x$
- in that case $d(w) = d(v) - 1$, thus, $d(w) \leq d(v) + 1$ □

# Analysis (Cont.)

**Corollary 10.22** After termination of the algorithm, $x$ is a maximum $s$-$t$-flow.

**Proof:**

- at termination $\mathrm{ex}_x(v) = 0$ for all $v \in V \setminus \{s, t\}$, i.e., $x$ is an $s$-$t$-flow
- optimality follows since there is no $s$-$t$-dipath in $D_x$ by Lemma 10.20 b □

**Lemma 10.23**

a A label $d(v)$ is never decreased by the algorithm.

b Calling `Relabel`(v) strictly increases $d(v)$.

c $d(v) \leq 2n - 1$ throughout the algorithm.

d The number of `Relabel` operations is at most $2n^2$.

# Proof of Lemma 10.23

**Proof:** a $d(v)$ is non-decreasing, b $\texttt{Relabel}(v)$ strictly increases $d(v)$

- only $\texttt{Relabel}$ operations change $d(v)$
- since $d$ is valid labeling $d(v) \leq d(w) + 1$ for all $(v, w) \in A_x$, when $\texttt{Relabel}(v)$ is called $d(v) < d(w) + 1$ for all $(v, w) \in A_x$
- setting $d(v) := \min\{d(w) + 1 \mid (v, w) \in A_x\}$ increases $d(v)$ by at least 1

c $d(v) \leq 2n - 1$

- $d(v)$ is only increased when $v$ is active
- if $v$ is active, by Lemma 10.20, there is a $v$-$s$-dipath $P$ in $D_x$
- for $P = v_0, a_0, v_1, a_1, \dots, a_k, v_k$ with $v_0 = v$ and $v_k = s$, we have $d_{v_i} \leq d(v_{i+1}) + 1$ for all $i$, hence,

$$d(v) \leq d(s) + k \leq n + (n - 1) = 2n - 1$$

d the number of $\texttt{Relabel}$ operations $\leq 2n^2$

- by b and c, for each of the $n - 2$ nodes $v \in V \setminus \{s, t\}$, $\texttt{Relabel}(v)$ is called at most $2n - 1$ times

# Bounding the Number of Push Operations



- Push operation on arc $a$ is called saturating if, after the Push, arc $a$ has disappeared from the residual graph $D_x$
- otherwise, the Push operation is called nonsaturating and node $v$ with $a \in \delta^+(v)$ is no longer active

**Lemma 10.24** The number of saturating Push operations is in $O(m \cdot n)$.

**Proof:** we show that on any arc $a \in A \cup A^{-1}$ there can be at most $n$ saturating Push operations

- let $a = (v, w)$ arbitrary
- at a saturating Push on $a$ we have $d(v) = d(w) + 1$, and $a$ disappears from $D_x$
- arc $a$ can only reappear in $D_x$ after flow has been pushed through $a^{-1}$ which only happens if $d(w) = d(v) + 1$
- thus, between two consecutive Push operations, the labels of $v$ and $w$ have increased by at least 2
- by Lemma 10.23 c this can happen at most $n$ times $\square$

## Bounding the Number of Push Operations

**Lemma 10.25** The number of nonsaturating Push operations is at most $O(m \cdot n^2)$.

**Proof:** see exercise

**Lemma 10.26** If the algorithm always chooses an active node $v$ with $d(v)$ maximum, then the number of nonsaturating Push operations is in $O(n^3)$.

**Proof:** we show that for any node $v \in V$ there can be at most $2n^2$ nonsaturating Push operations

- after the nonsaturating Push, $v$ becomes inactive and since flow is sent to nodes of smaller label, no node with label larger than $d(v)$ can become active before the next Relabel operations
- $v$ can only become active again after a Relabel operation
- by Lemma 10.23 this can happen at most $2n^2$ times □

# Running Time of the Goldberg-Tarjan Algorithm

**Theorem 10.27** The Goldberg-Tarjan Algorithm finds a maximum $s$-$t$-flow in $O(mn^2)$ time.

**Theorem 10.28** If the algorithm always chooses an active node $v$ with $d(v)$ maximum, its running time is $O(n^3)$.

## Remarks

- If the algorithm always chooses an active node $v$ with $d(v)$ maximum, one can show that the number of nonsaturating `Push` operations and thus the total running time is at most $O(n^2\sqrt{m})$.
- The currently best known running time of a maximum $s$-$t$-flow algorithm is $O(nm)$ (Orlin 2013).