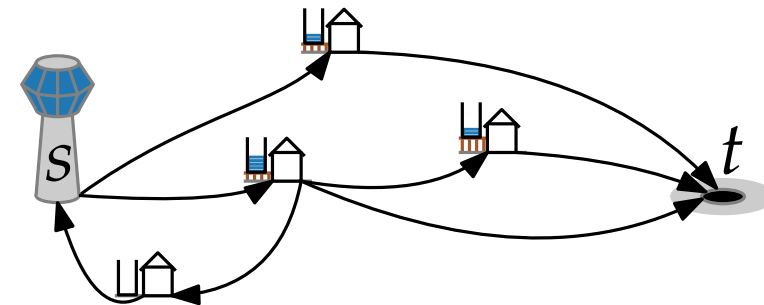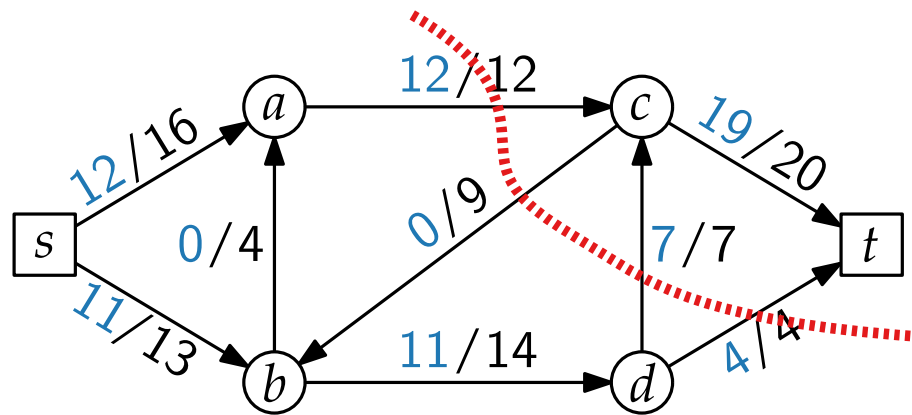# Advanced Algorithms
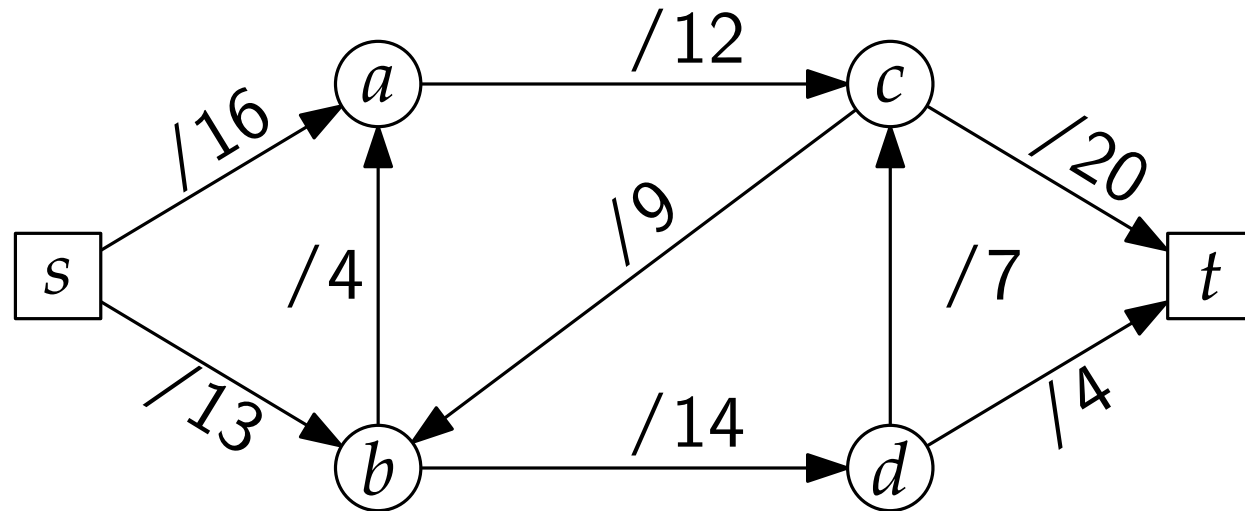
Maximum Flow Problem
Push–Relabel Algorithm

Johannes Zink · WS23/24

Flow Networks

A **flow network** $G = (V, E)$ is a digraph (short for "**di**rected **graph**") with
- unique **source** s and **sink** t,
- no antiparallel edges, and
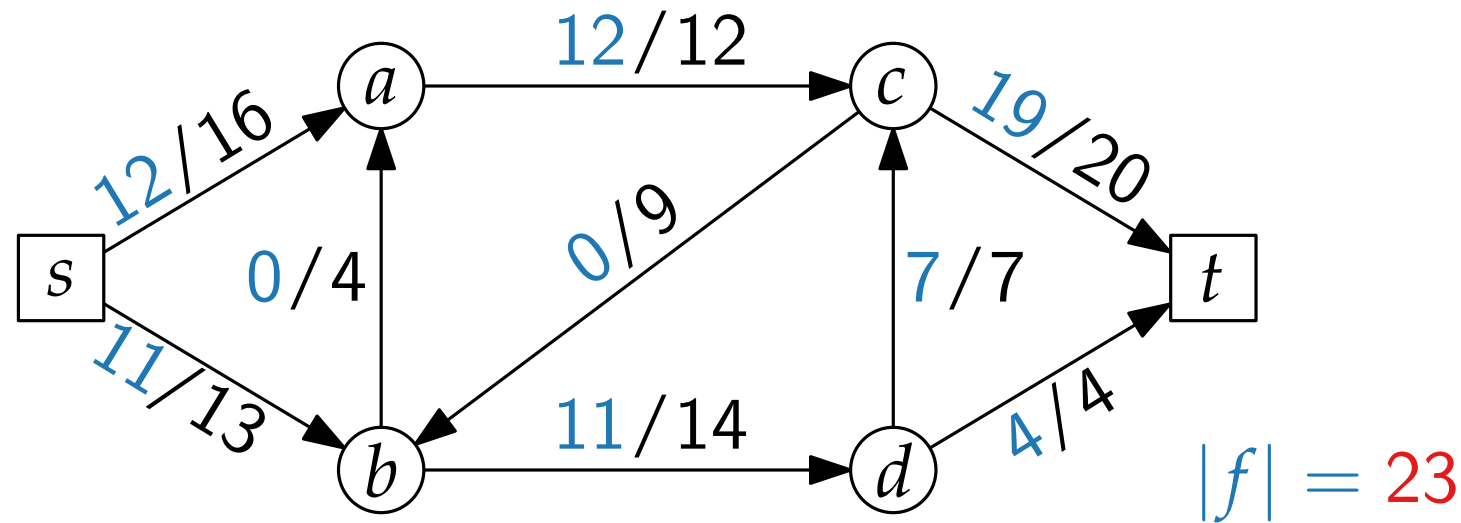- a **capacity** $c(u, v) \geq 0$ for every $(u, v) \in E$.

Flow

An s–t **flow** in G is a real-valued function $f \colon V \times V \to \mathbb{R}$ that satisfies

- **flow conservation**,

  $\sum\limits_{v \in V} f(v, u) = \sum\limits_{v \in V} f(u, v)$ for all $u \in V \setminus \{s, t\}$, and

- **capacity constraint**, $0 \leq f(u, v) \leq c(u, v)$.

$|f| = 23$

> **Maximum flow problem.** Given a flow network G with source s and sink t, find an s–t flow of maximum value.

The **value** $|f|$ of an s–t flow f is defined as
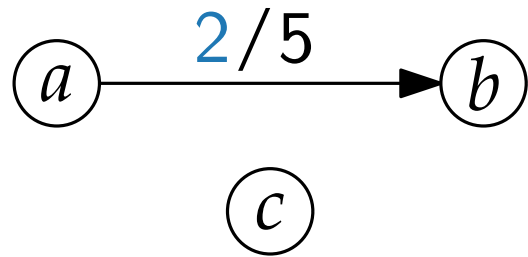
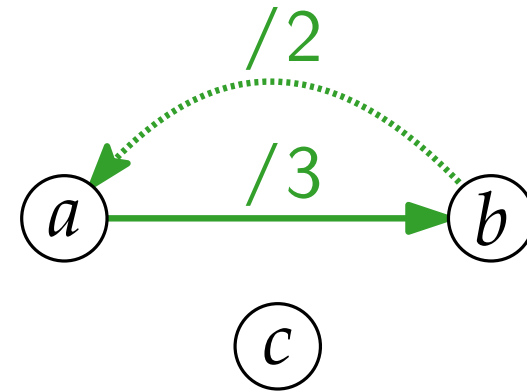$$|f| = \sum_{v \in V} f(s, v) - \sum_{v \in V} f(v, s).$$

By How Much May Flow Change?

Given G and f, the **residual capacity** c_f for a pair $u, v \in V$ is

$$c_f(u, v) = \begin{cases} c(u, v) - f(u, v) & \text{if } (u, v) \in E \\ f(v, u) & \text{if } (v, u) \in E \\ 0 & \text{otherwise.} \end{cases}$$

$$c_f(a, b) = 3$$
$$c_f(b, a) = 2$$
$$c_f(a, c) = 0$$

Residual Networks & Augmenting Paths

The **residual network** $G_f = (V, E_f)$ for a flow network G with s–t flow f has
- $E_f = \{(u, v) \in V \times V \mid c_f(u, v) > 0\}$.

An **augmenting path** is an st-path in G_f. $\Rightarrow$ use to increase f
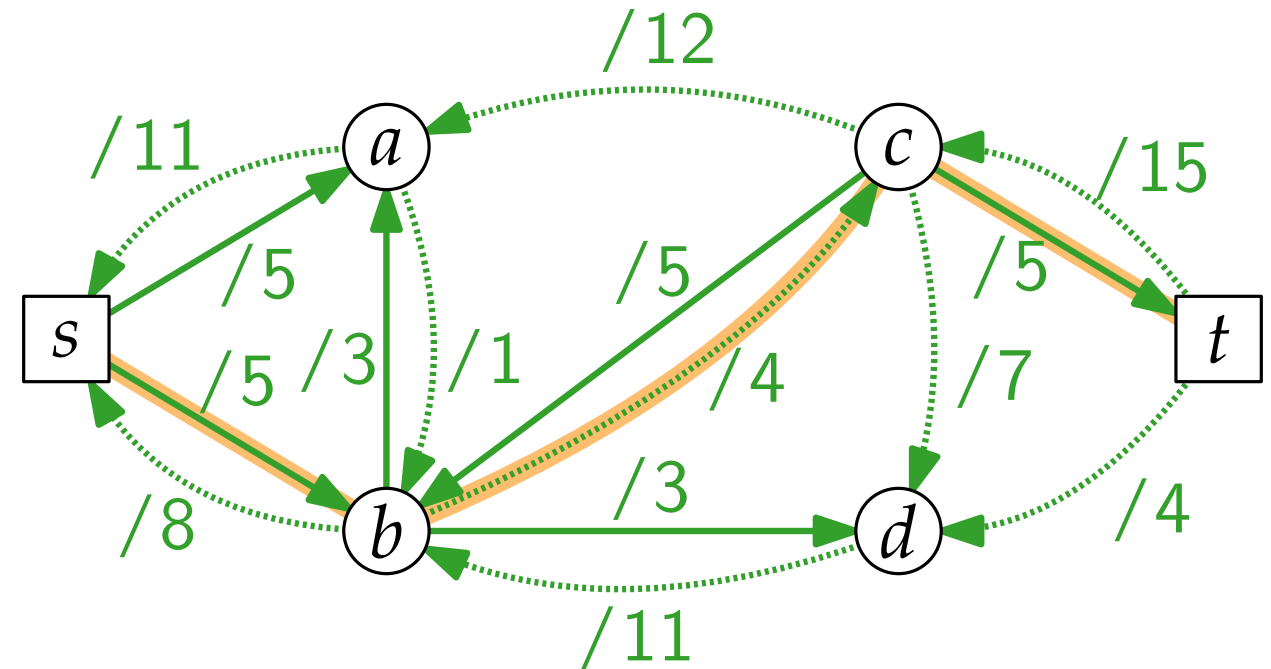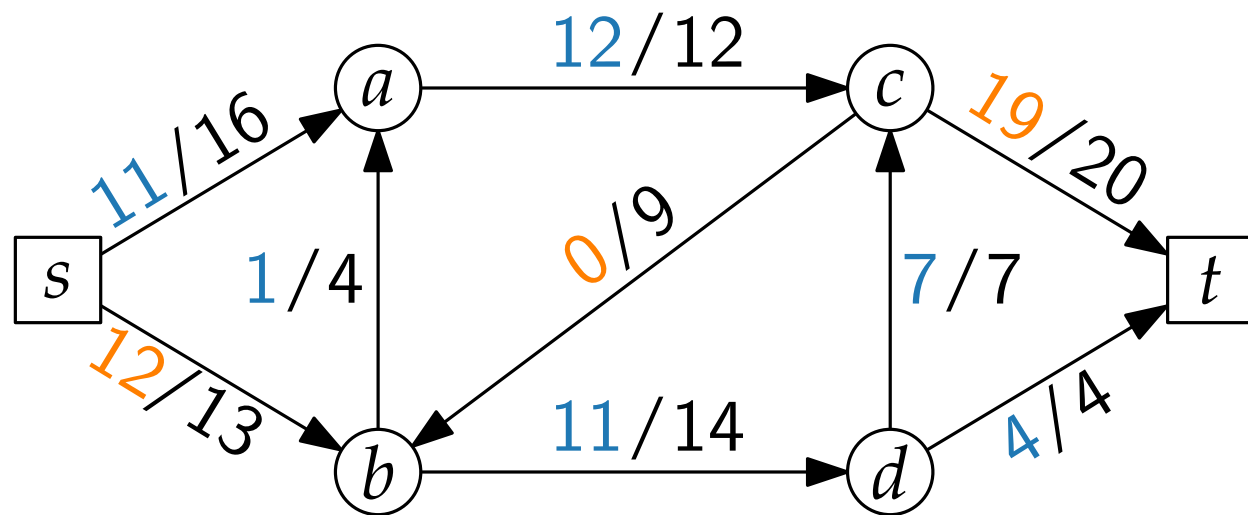
$$c_f(u, v) = \begin{cases} c(u, v) - f(u, v) & \text{if } (u, v) \in E \\ f(v, u) & \text{if } (v, u) \in E \\ 0 & \text{otherwise.} \end{cases}$$

flow/capacity

/c_f non-satured edges

reverse edges

The Algorithms of Ford–Fulkerson and Edmonds–Karp

EdmondsKarp

~~FordFulkerson~~$(G = (V, E), c, s, t)$:

> **foreach** $uv \in E$ **do**
> > $f(u, v) = 0$

$\left.\right\}$ initializing zero flow

shortest

> **while** G_f contains augmenting path p **do**
> > $\Delta = \min_{uv \in p} c_f(uv)$ $\left.\right\}$ residual capacity of p
> >
> > **foreach** $uv \in p$ **do**
> > > **if** $uv \in E$ **then**
> > > > $f(u, v) = f(u, v) + \Delta$
> > >
> > > **else**
> > > > $f(v, u) = f(v, u) - \Delta$

$\left.\right\}$ augmentation along p

> **return** f $\left.\right\}$ return max flow

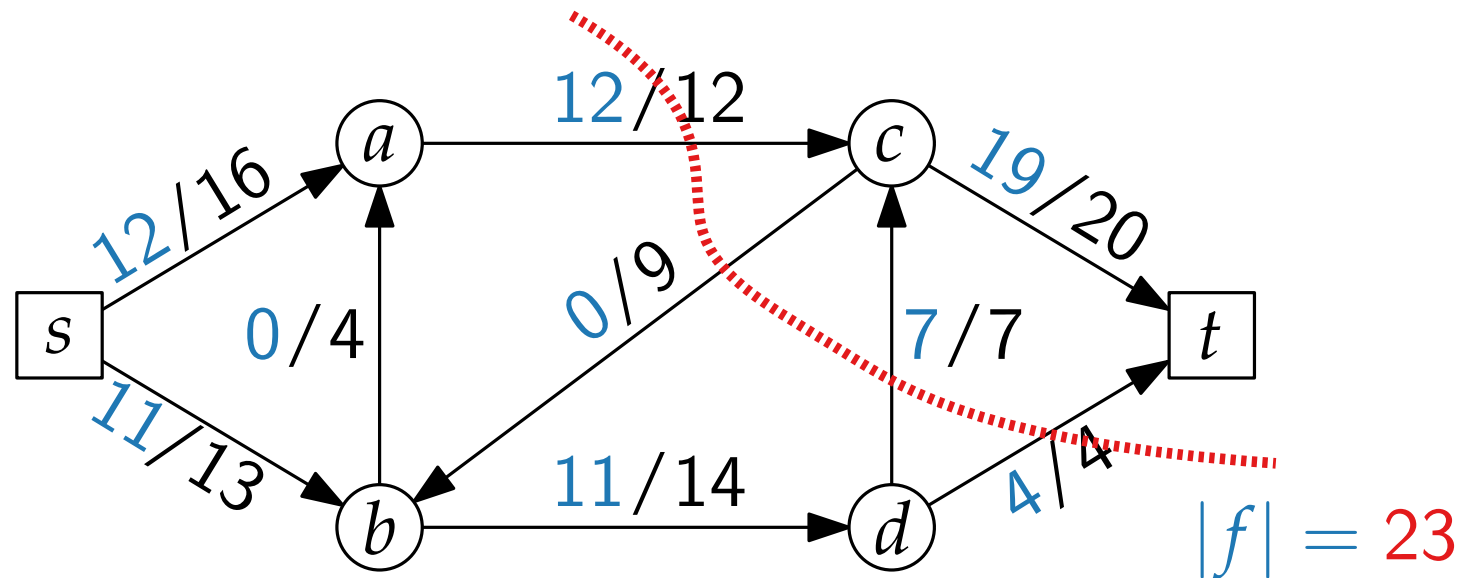- Ford–Fulkerson runs in $\mathcal{O}(|E||V| \max_{e \in E} c(e))$ and Edmonds–Karp in $\mathcal{O}(|E|^2|V|)$ time.

The Max-Flow Min-Cut Theorem

Theorem.
For an s–t flow f in a flow network G, the following conditions are equivalent:
- f is a maximum s–t flow in G.
- G_f contains no augmenting paths.
- $|f| = c(S, T)$, which is the capacity of at least one s–t cut (S, T) of G.

$|f| = 23$

The Push–Relabel Idea

[1988]

A (New) Approach to the Maximum-Flow Problem

ANDREW V. GOLDBERG

Massachusetts Institute of Technology, Cambridge, Massachusetts

AND

ROBERT E. TARJAN

Princeton University, Princeton, New Jersey, and AT&T Bell Laboratories, Murray Hill, New Jersey

Abstract. All previously known efficient maximum-flow algorithms work by finding augmenting paths, either one path at a time (as in the original Ford and Fulkerson algorithm) or all shortest-length augmenting paths at once (using the layered network approach of Dinic). An alternative method based on the *preflow* concept of Karzanov is introduced. A preflow is like a flow, except that the total amount

for the next phase. Our algorithm abandons the idea of finding a flow in each phase and also abandons the idea of global phases. Instead, our algorithm maintains a preflow in the original network and pushes local flow excess toward the sink along what it estimates to be shortest paths in the residual graph. This pushing of flow changes the residual graph, and paths to the sink may become saturated. Excess that cannot be moved to the sink is returned to the source, also along estimated shortest paths. Only when the algorithm terminates does the preflow become a flow, and then it is a maximum flow.
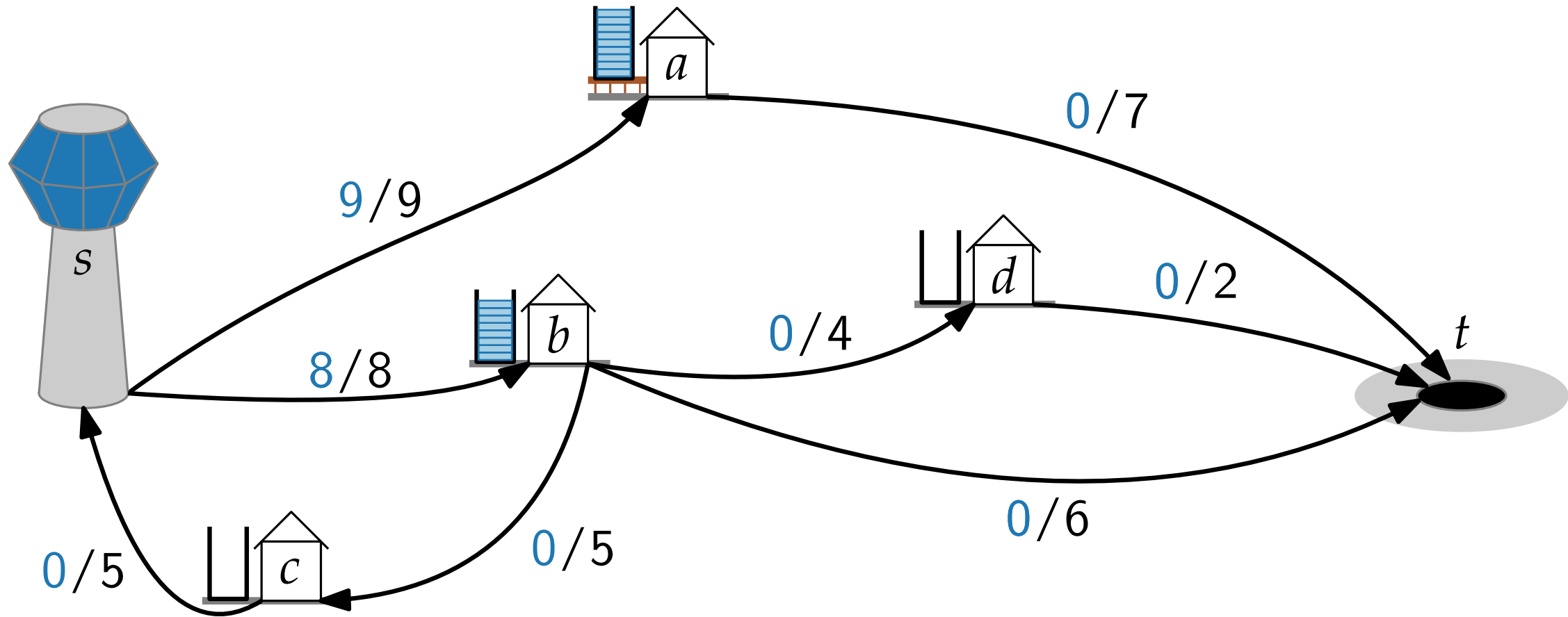
The Push–Relabel Idea

The Push–Relabel Idea

$9/9$

$7/7$

$8/8$

$4/4$

$2/2$

$0/6$

$0/5$

$0/5$
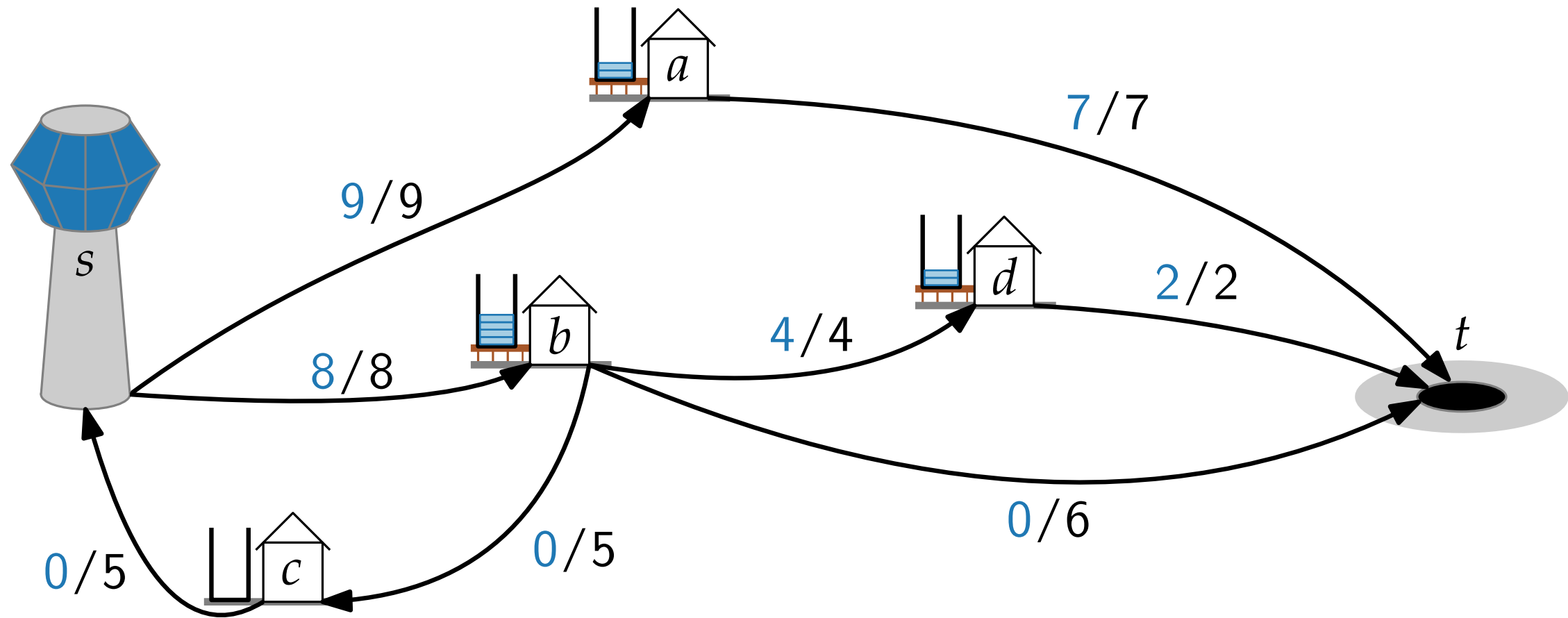
The Push–Relabel Idea
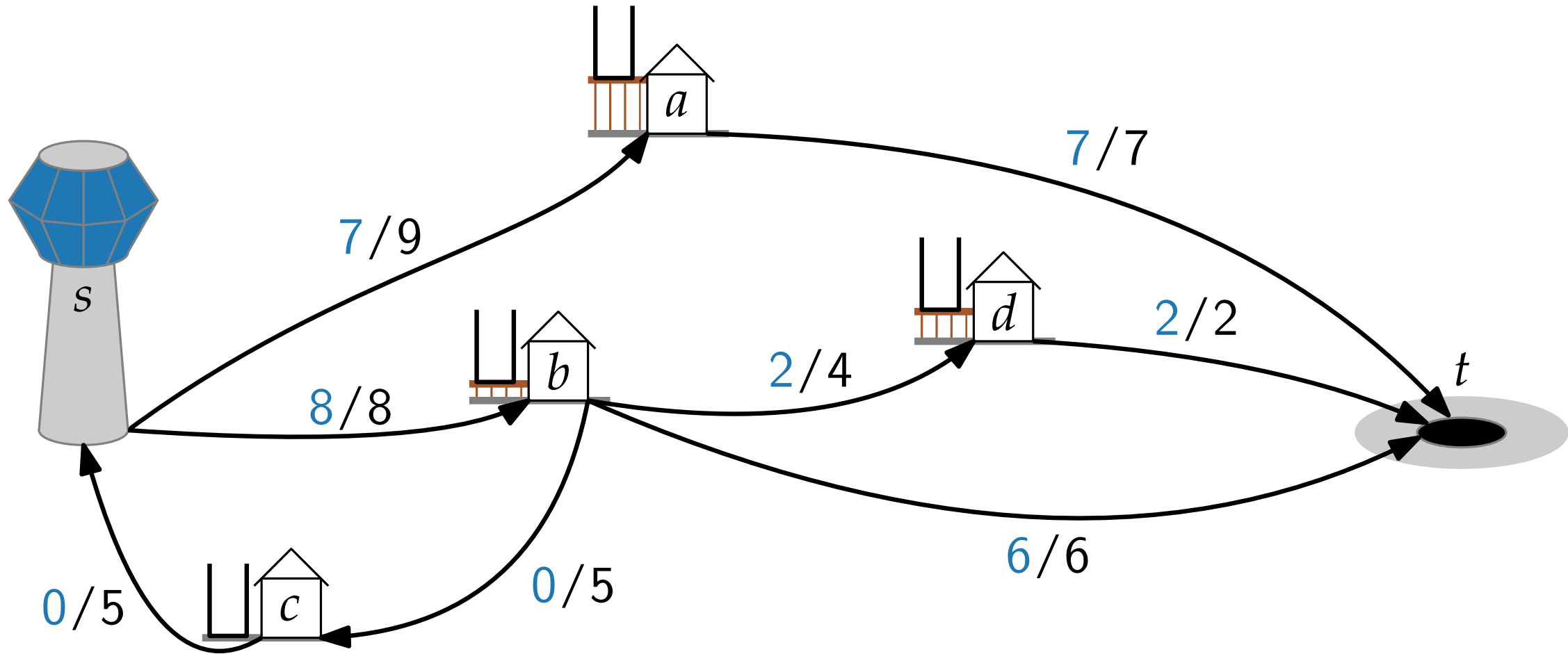
Preflow, Excess Flow, and Height

A **preflow** in G is a real-value function $f\colon V \times V \to \mathbb{R}$ that satisfies the capacity constraint and, for each $u \in V \setminus \{s\}$,

■ $\displaystyle\sum_{v \in V} f(v, u) - \sum_{v \in V} f(u, v) \geq 0$.

The **excess flow** of a vertex u is

■ $\displaystyle e(u) = \sum_{v \in V} f(v, u) - \sum_{v \in V} f(u, v)$.

A vertex u is called **overflowing**, when $e(u) > 0$.

For a flow network G with preflow f, a **height function** is a function $h\colon V \to \mathbb{N}$ such that

■ $h(s) = |V|$,

■ $h(t) = 0$, and

■ $h(u) \leq h(v) + 1$ for every residual edge $(u, v) \in E_f$.
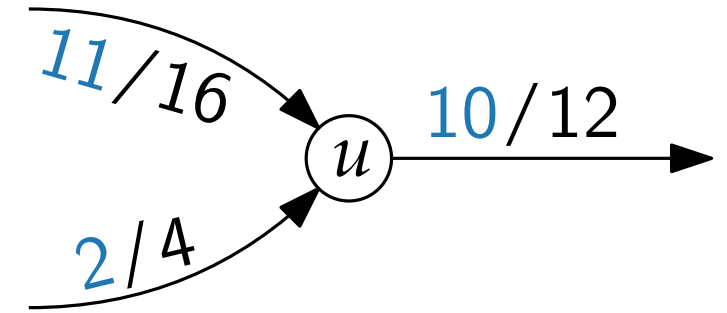
$11/16$

$10/12$

$2/4$

$e(u) = 3$
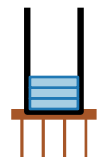
The PUSH Operation

Condition: u is overflowing, $c_f(u, v) > 0$, and $h(u) = h(v) + 1$.

Effect: Push $\min(e(u), c_f(u, v))$ overflow from u to v.

PUSH(u, v):

$\quad \Delta = \min(e(u), c_f(u, v))$

$\quad$ **if** $(u, v) \in E$ **then**

$\quad\quad\mid \quad f(u, v) = f(u, v) + \Delta$

$\quad$ **else**

$\quad\quad\lfloor \quad f(v, u) = f(v, u) - \Delta$

$\quad e(u) = e(u) - \Delta$

$\quad e(v) = e(v) + \Delta$

Example.

$$\text{PUSH}(u, v)$$
$$\Delta = 4$$

$e(u) = 5 \qquad e(v) = 1$
$h(u) = 4 \qquad h(v) = 3$

$e(u) = 1 \qquad e(v) = 5$
$h(u) = 4 \qquad h(v) = 3$
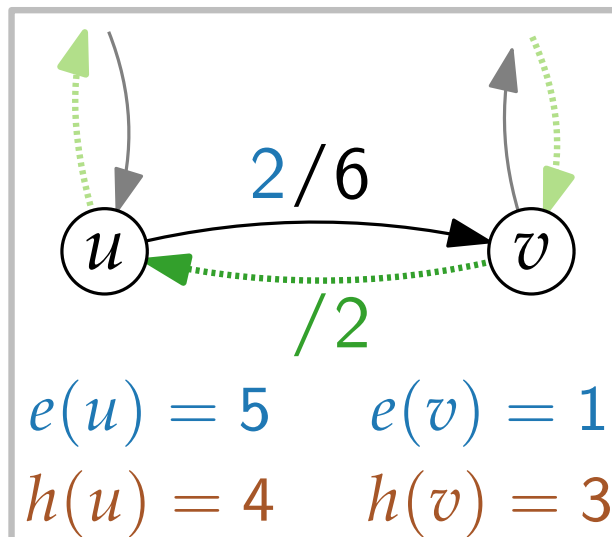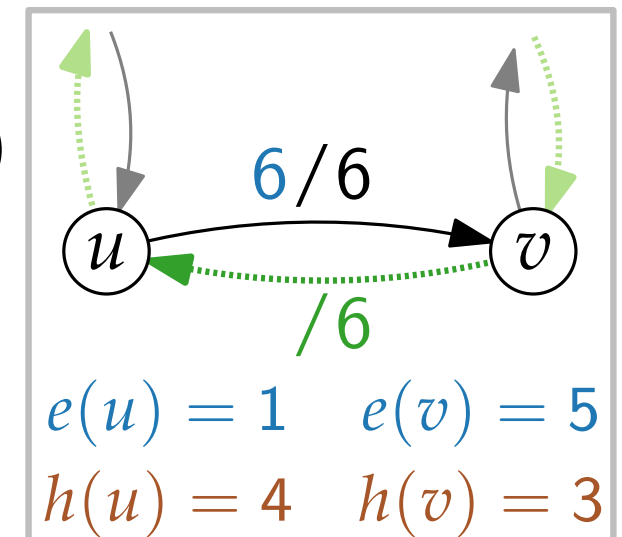
The RELABEL Operation

Condition: u is overflowing and $h(u) \leq h(v)$ for every $v \in V$ with $(u, v) \in E_f$.
Effect: Increase the height of u.

RELABEL(u):

$\quad h(u) = 1 + \min\{h(v) \colon v \in V \text{ with } (u, v) \in E_f\}$

Example.
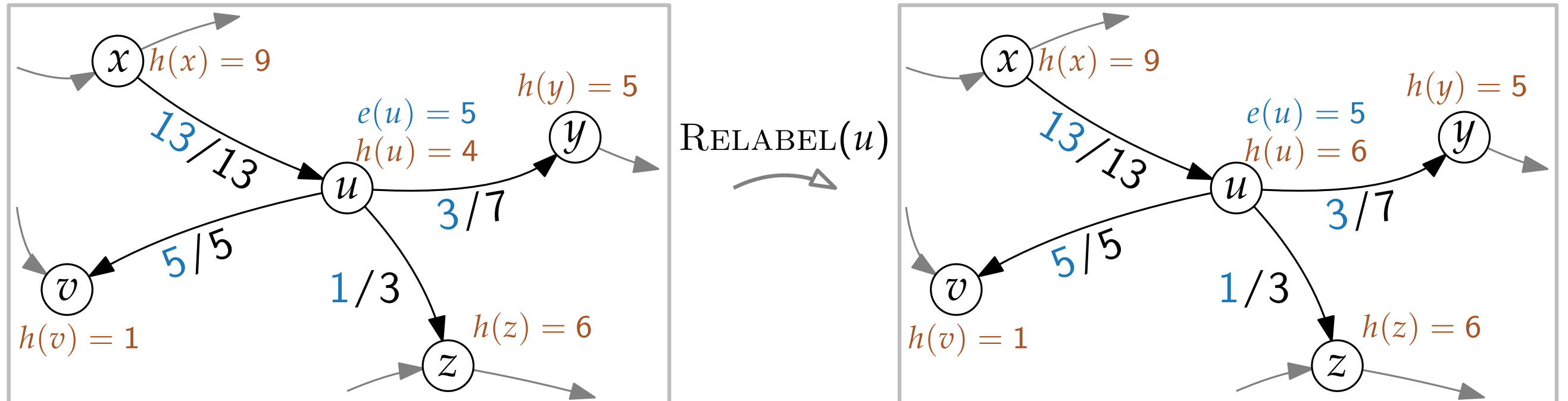
The PUSH–RELABEL Algorithm

PUSH–RELABEL(G, c, s, t):

$\quad$ INITPREFLOW(G, c, s)
$\quad$ **while** $\exists$ applicable PUSH or RELABEL operation X **do**
$\quad\quad\lfloor$ apply X

INITPREFLOW(G, c, s)

$\quad$ **foreach** $v \in V$ **do** $h(v) = 0$; $e(v) = 0$
$\quad$ $h(s) = |V|$
$\quad$ **foreach** $(u, v) \in E$ **do** $f(u, v) = 0$
$\quad$ **foreach** v such that $(s, v) \in E$ **do**
$\quad\quad\lfloor$ $f(s, v) = c(s, v)$
$\quad\quad\lfloor$ $e(v) = c(s, v)$

- initializes heights
- pushes max flow over every edge that leaves s

Correctness

<div style="border: 3px solid green; padding: 10px;">

Part 1.

If the algorithm terminates, the preflow is a maximum flow.

- The algorithm maintains f as a preflow and h as a height function

- If an overflowing vertex exists, the algorithm can continue..

- The sink t is not reachable from source s in G_f.

</div>

Part 2.

The algorithm terminates and the heights stay finite.

- Find upper bound on heights.

- Find upper bound for the number of calls to RELABEL.

- Find upper bound for the number of calls to PUSH.

Maintaining the Preflow

Lemma 1.
The push–relabel algorithm maintains a preflow f.

Proof.

- INITPREFLOW initializes a preflow f. ✓

- $\text{RELABEL}(u)$ does not affect f. ✓

- $\text{PUSH}(u, v)$ maintains f as a preflow. ✓

Height function:
- $h(s) = |V|$
- $h(t) = 0$
- $h(u) \leq h(v) + 1 \quad \forall (u, v) \in E_f$

Condition: u is overflowing,
$c_f(u, v) > 0$, and $h(u) = h(v) + 1$.

$\text{PUSH}(u, v)$:

$\quad \Delta = \min(e(u), c_f(u, v))$
$\quad \textbf{if } (u, v) \in E \textbf{ then}$
$\quad\quad | \quad f(u, v) = f(u, v) + \Delta$
$\quad \textbf{else}$
$\quad\quad | \quad f(v, u) = f(v, u) - \Delta$
$\quad e(u) = e(u) - \Delta$
$\quad e(v) = e(v) + \Delta$

Condition: u is overflowing and
$h(u) \leq h(v) \; \forall v \in V$ with $(u, v) \in E_f$.

$\text{RELABEL}(u)$:

$\quad h(u) = 1 + \min\{h(v) \colon (u, v) \in E_f\}$

Maintaining the Height Function

Lemma 2.
The push–relabel algorithm maintains h as a height function.

Proof.

- $\textsc{InitPreflow}$ initializes h as a height function. ✓

- Under $\textsc{Push}(u, v)$, h remains a height function:

    - If (v, u) is added to E_f, then
      $h(v) = h(u) - 1 < h(u) + 1.$ ✓
    - If (u, v) is removed from E_f, then ✓.

- Under $\textsc{Relabel}(u)$, h remains a height function: ✓

    - $(u, v) \in E_f$, then $h(u) \leq h(v) + 1$
    - $(w, u) \in E_f$, then $h(w) < h(u) + 1$

Height function:
- $h(s) = |V|$
- $h(t) = 0$
- $h(u) \leq h(v) + 1 \quad \forall (u, v) \in E_f$

Condition: u is overflowing,
$c_f(u, v) > 0$, and $h(u) = h(v) + 1$.
$\textsc{Push}(u, v)$:

$\Delta = \min(e(u), c_f(u, v))$
if $(u, v) \in E$ **then**
$\quad | \quad f(u, v) = f(u, v) + \Delta$
else
$\quad \lfloor \quad f(v, u) = f(v, u) - \Delta$
$e(u) = e(u) - \Delta$
$e(v) = e(v) + \Delta$

Condition: u is overflowing and
$h(u) \leq h(v) \; \forall v \in V$ with $(u, v) \in E_f$.
$\textsc{Relabel}(u)$:

$h(u) = 1 + \min\{h(v) \colon (u, v) \in E_f\}$

Continuation

Lemma 3.
If a vertex u is overflowing, either a $\textsc{Push}$ or a $\textsc{Relabel}$ operation applies to u.

Proof.
By Lemma 2, h is a height function. Thus, we have
- $h(u) \leq h(v) + 1$ for all v with $(u, v) \in E_f$.

If no $\textsc{Push}$ operation is valid for $(u, v) \in E_f$, then
- $h(u) \leq h(v)$ for all v with $(u, v) \in E_f$.

Therefore, $\textsc{Relabel}(u)$ is applicable.

Height function:
- $h(s) = |V|$
- $h(t) = 0$
- $h(u) \leq h(v) + 1 \quad \forall (u, v) \in E_f$

Condition: u is overflowing,
$\quad c_f(u, v) > 0$, and $h(u) = h(v) + 1$.
$\textsc{Push}(u, v)$:
$\quad \Delta = \min(e(u), c_f(u, v))$
$\quad$ **if** $(u, v) \in E$ **then**
$\quad \quad f(u, v) = f(u, v) + \Delta$
$\quad$ **else**
$\quad \quad f(v, u) = f(v, u) - \Delta$
$\quad e(u) = e(u) - \Delta$
$\quad e(v) = e(v) + \Delta$

Condition: u is overflowing and
$\quad h(u) \leq h(v) \; \forall v \in V$ with $(u, v) \in E_f$.
$\textsc{Relabel}(u)$:
$\quad h(u) = 1 + \min\{h(v) : (u, v) \in E_f\}$

Reachability of the Sink

> **Lemma 4.**
>
> During the execution of the push–relabel algorithm, there is no path from s to t in G_f.

Height function:
- $h(s) = |V|$
- $h(t) = 0$
- $h(u) \leq h(v) + 1 \quad \forall (u, v) \in E_f$

Proof.

Suppose there is a path $s = v_0, v_1, \ldots, v_k = t$ in G_f.
Then

- $(v_i, v_{i+1}) \in E_f$ for $0 \leq i \leq k - 1$, and

- $h(v_i) \leq h(v_{i+1}) + 1$ for $0 \leq i \leq k - 1$.

$$\Rightarrow h(s) \leq h(t) + k = k$$

But since $k < |V|$, it follows that $h(s) < |V|$. ✗

Correctness of the Algorithm (Part I)

> **Theorem 5.**
> When the push–relabel algorithm terminates, the computed preflow f is a maximum flow.

Proof.

■ By Lemma 3, the algorithm stops when there is no overflowing vertex.

■ By Lemma 1, f is a preflow.

$\Rightarrow f$ is a flow.

■ By Lemma 2, h is a height function.

■ So by Lemma 4, there is no s–t path in G_f.

$\Rightarrow$ By the Max-Flow Min-Cut Theorem, the flow f is a maximum flow.

Correctness

Part 1. ✓

If the algorithm terminates, the preflow is maximum flow.

■ The algorithm maintains f as a preflow and h as a height function.

■ If an overflowing vertex exists, the algorithm can continue.

■ Sink t is not reachable from source s in G_f.

Part 2.

The algorithm terminates and the heights stay finite.

■ Find upper bound on heights.

■ Find upper bound for the number of calls to RELABEL.

■ Find upper bound for the number of calls to PUSH.

Reachability of the Source in the Residual Graph

> **Lemma 6.**
> For every overflowing vertex v,
> there is a path from v to s in G_f.

Proof.

- Let S_v be the set of vertices reachable from v in G_f.

- Suppose that $s \notin S_v$.

- Since f is a preflow and $s \notin S_v$, we have $\displaystyle\sum_{w \in S_v} e(w) \geq 0$.

- Since $v \in S_v$, we even have $\displaystyle\sum_{w \in S_v} e(w) > 0$.

- There is an edge (u, w) with $u \notin S_v, w \in S_v$ and $f(u, w) > 0$.

- But then $c_f(w, u) > 0$, meaning u is reachable from v. ✗

s

S_v

u
$f(u, w) > 0$
$c_f(w, u) > 0$
w
v
$e(v) > 0$
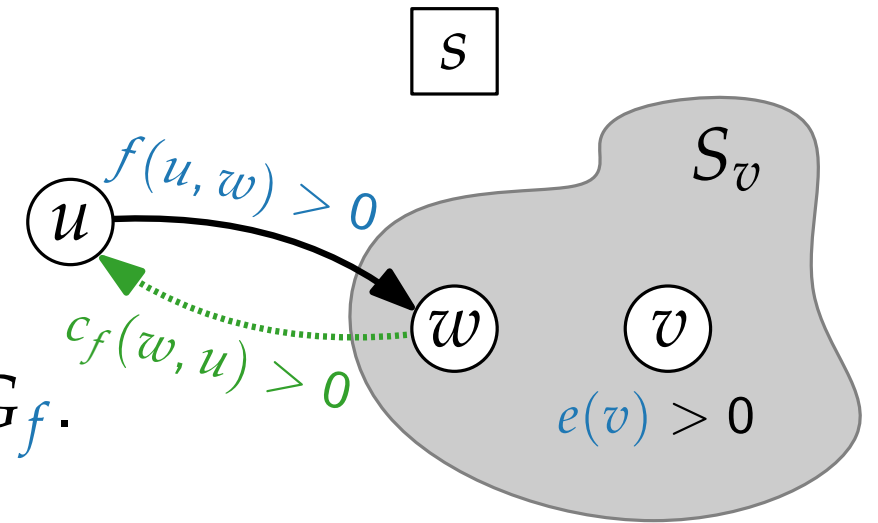
Upper Bounds on the Height and $\#$ RELABEL Operations

Height function:
- $h(s) = |V|$
- $h(t) = 0$
- $h(u) \leq h(v) + 1 \quad \forall (u, v) \in E_f$

Condition: u is overflowing and
$h(u) \leq h(v) \; \forall v \in V$ with $(u, v) \in E_f$.

RELABEL(u):
$h(u) = 1 + \min\{h(v) \colon (u, v) \in E_f\}$

Lemma 7.
During the push–relabel algorithm,
we have $h(v) \leq 2|V| - 1$ for all $v \in V$.

Proof.

- Statement holds after initialization.

- Let v be an overflowing vertex that is relabeled.

- By Lemma 6, there is a path $v = v_0, v_1, \ldots, v_k = s$ in G_f.

- Then $h(v_i) \leq h(v_{i+1}) + 1$ for $0 \leq i \leq k - 1$.

- Since $k \leq |V| - 1$, we have $h(v) \leq h(s) + k \leq 2|V| - 1$.

Corollary 8.
The push–relabel algorithm executes
at most $2|V|^2$ RELABEL operations.
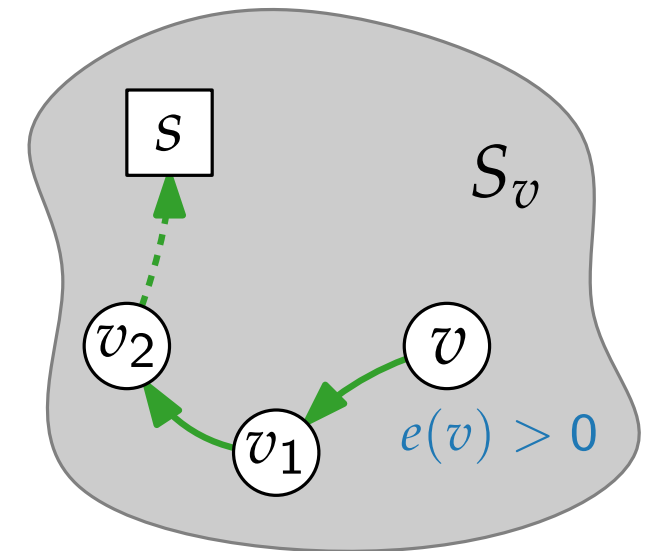
Saturating and Unsaturating PUSH Operations

The operation $\text{PUSH}(u, v)$ is

- **saturating** if afterwards $c_f(u, v) = 0$,

$$u \xrightarrow{\;2/6\;} v \qquad \begin{array}{c} \text{PUSH}(u, v) \\ \diagup \\ \Delta = 4 \end{array} \qquad u \xrightarrow{\;6/6\;} v$$
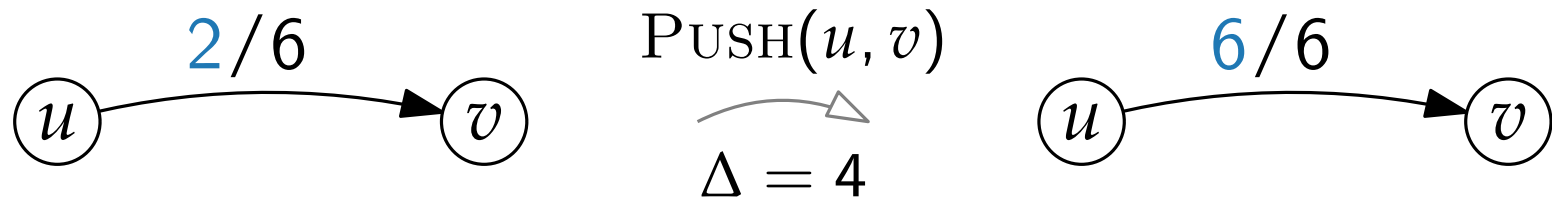
- and **unsaturating** otherwise.

$$u \xrightarrow{\;2/6\;} v \qquad \begin{array}{c} \text{PUSH}(u, v) \\ \diagup \\ \Delta = 2 \end{array} \qquad u \xrightarrow{\;4/6\;} v$$
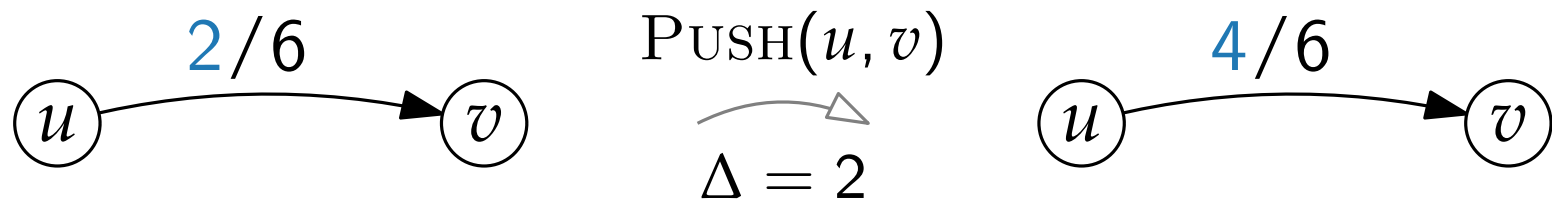
Upper Bound on the Number of Saturating PUSH Operations

> **Lemma 9.**
> The push–relabel algorithm executes
> at most $2|V| \cdot |E|$ saturating PUSH operations.

Condition: u is overflowing,
$\qquad c_f(u, v) > 0$, and $h(u) = h(v) + 1$.

PUSH(u, v):

$\qquad \Delta = \min(e(u), c_f(u, v))$
$\qquad$ **if** $(u, v) \in E$ **then**
$\qquad \qquad | \quad f(u, v) = f(u, v) + \Delta$
$\qquad$ **else**
$\qquad \qquad | \quad f(v, u) = f(v, u) - \Delta$
$\qquad e(u) = e(u) - \Delta$
$\qquad e(v) = e(v) + \Delta$

Proof.

- Consider saturating PUSH(u, v)
  - $h(u) = h(v) + 1$

- For another saturating PUSH(u, v), first PUSH(v, u) necessary
  - $h(v) = h(u) + 1$ necessary

- After another saturating PUSH(u, v),
  both $h(u)$ and $h(v)$ have increased by at least two.

- But by Lemma 6, $h(u) \leq 2|V| - 1$ and $h(v) \leq 2|V| - 1$.

- There are at most $2|V| - 1$ saturated PUSH operations for every edge (u, v).

PUSH(u, v)
$\cdots$
PUSH(v, u)
$\cdots$
PUSH(u, v)
$\cdots$

Upper Bound on the Number of PUSH Ops

Lemma 10.

The push–relabel algorithm executes at most $4|V|^2 \cdot |E|$ unsaturating PUSH operations

Condition: u is overflowing,
$\quad c_f(u,v) > 0$, and $h(u) = h(v) + 1$.

PUSH(u,v):
$\quad \Delta = \min(e(u), c_f(u,v))$
$\quad$ **if** $(u,v) \in E$ **then**
$\quad\quad f(u,v) = f(u,v) + \Delta$
$\quad$ **else**
$\quad\quad f(v,u) = f(v,u) - \Delta$
$\quad e(u) = e(u) - \Delta$
$\quad e(v) = e(v) + \Delta$

Proof.

■ Consider $\mathcal{H} = \displaystyle\sum_{\substack{v \in V \setminus \{s,t\}, \\ v \text{ overflowing}}} h(v)$.

■ After initialization and at the end $\mathcal{H} = 0$.

■ A saturating PUSH increases $\mathcal{H}$ by at most $2|V| - 1$.

■ By Lemma 9, all saturating PUSH operations increase $\mathcal{H}$ by $\leq (2|V| - 1) \cdot 2|V||E|$.

■ By Lemma 7, all RELABEL operations increase $\mathcal{H}$ by $\leq (2|V| - 1) \cdot |V|$.

■ An unsaturating PUSH(u,v) decreases $\mathcal{H}$ by at least 1 since $h(u) - h(v) \geq 1$.

$\Rightarrow$ At most $4|V|^2|E| + 2|V|^2 - 2|V||E| - |V|$ unsaturating PUSH operations.

Termination of the Algorithm

> **Theorem 5.**
> When the push–relabel algorithm terminates, the computed preflow f is a maximum flow.

> **Theorem 11.**
> The push–relabel algorithm terminates after $\mathcal{O}(|V|^2|E|)$ valid PUSH or RELABEL ops.

Proof.
■ Follows by Corollary 8 and Lemmas 9 & 10.

Implementation

The actual running time depends on the selection order of the overflowing vertices:

- **FIFO implementation:**
  Pick overflowing vertex by *first-in-first-out* principle: $\mathcal{O}(|V|^3)$ running time.
  with dynamic trees: $\mathcal{O}(|V||E|\log\frac{|V|^2}{|E|})$.

- **Highest label:**
  For $\textsc{Push}$ select highest overflowing vertex: $\mathcal{O}(|V|^2|E|^{\frac{1}{2}})$.

- **Excess scaling:**
  For $\textsc{Push}(u,v)$ choose edge (u,v) such that u is overflowing, $e(u)$ is *sufficiently high* and $e(v)$ *sufficiently small*: $\mathcal{O}(|E| + |V|^2\log C)$, where $C = \max\limits_{e\in E} c(e)$.

Discussion

- The push–relabel method offers an alternative and faster framework to the augmenting-paths method to develop algorithms that solve the maximum-flow problem.

- In practice, heuristics are used to improve the performance of push–relabel algorithms. Any ideas?

- The algorithm can be extended to solve the minimum-cost flow problem.

- Meanwhile there are even faster (and more complicated) algorithms for the maximum-flow problem known:

Authors	Year	Basic technique	Running time
Ford, Fulkerson	1956	augmenting paths	$\mathcal{O}(|E||V| \max_{e \in E} c(e))$
Edmonds, Karp	1969	augmenting paths	$\mathcal{O}(|E|^2|V|)$
Dinic [version w/ dynamic trees]	1970	augmenting paths	$\mathcal{O}(|E||V| \log |V|)$
Goldberg, Tarjan [v. w/ dynamic trees]	1988	push–relabel	$\mathcal{O}(|E||V| \log(|V|^2/|E|))$
Orlin, King, Rao, Tarjan	2013	push–relabel & more	$\mathcal{O}(|E||V|)$
Chen, Kyng, Liu, Peng, Gutenberg, Sachdeva	2022	interior point method	$\mathcal{O}(|E|^{1+o(1)} \log \max_{e \in E} c(e))$

Literature

Main source:
- [CLRS Ch26] ← Cormen et al. "Introduction to Algorithms"

Original paper:
- [Goldberg, Tarjan '88] A new approach to the maximum-flow problem

Links:
- Animations of the max-flow algorithms by Ford–Fulkerson and Edmonds–Karp:
  https://visualgo.net/en/maxflow