Advanced Algorithms

Maximum flow problem

Push-relabel algorithm

Jonathan Klawitter · WS20
Flow network

A flow network $G = (V, E)$ is a digraph with
- unique source $s$ and sink $t$,
- no antiparallel edges, and
- a capacity $c(u, v) \geq 0$ for every $(u, v) \in E$. 
A flow in $G$ is a real-value function $f : V \times V \rightarrow \mathbb{R}$ that satisfies

- **flow conservation**, $\sum_{v \in V} f(v, u) = \sum_{v \in V} f(u, v)$ for all $u \in V \setminus \{s, t\}$, and
- **capacity constraint**, $0 \leq f(u, v) \leq c(u, v)$.

The value $|f|$ of a flow $f$ is defined as

$$|f| = \sum_{v \in V} f(s, v) - \sum_{v \in V} f(v, s).$$

Maximum flow problem. Given a flow network $G$ with source $s$ and sink $t$, find a flow of maximum value.

The figure shows a flow network with values on each edge and the total flow $|f| = 23$. The values on the edges represent the flow, with capacities indicated as well. The source is $s$ and the sink is $t$. The flow conservation and capacity constraints are evident from the diagram.
How much may flow change?

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

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

The diagram illustrates the flow values and residual capacities for the edges:

- $c_f(a, b) = 3$
- $c_f(b, a) = 2$
- $c_f(a, c) = 0$
Residual network & augmenting path

The **residual network** $G_f = (V, E_f)$ for a flow network $G$ with 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 $G_f$. ⇒ use to increase $f$

\[
c_f(u, v) = \begin{cases} 
    c(u, v) - f(u, v) & \text{if } (u, v) \in E \\
    f(v, u) & \text{if } (v, u) \in E \\
    0 & \text{otherwise.}
\end{cases}
\]
Ford-Folkerson and Edmons-Karp algorithms

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

1. **foreach** $uv \in E$ **do**
   - $f_{uv} \leftarrow 0$

2. while $G_f$ contains augmenting path $p$ **do**
   - $\Delta \leftarrow \min_{uv \in p} c_f(uv)$
   - **foreach** $uv \in p$ **do**
     - **if** $uv \in E$ **then**
       - $f_{uv} \leftarrow f_{uv} + \Delta$
     - **else**
       - $f_{vu} \leftarrow f_{vu} - \Delta$

3. return $f$

**EdmondsKarp**

- **initialising zero flow**
- **residual capacity of** $p$
- **augmentation along** $p$
- **return max flow**

Ford-Folkerson runs in $O(|E||f^*|)$ and Edmons-Karp in $O(|V||E|^2)$ time.
Max-flow min-cut theorem

**Theorem.**
For a flow $f$ in a flow network $G$, the following conditions are equivalent:

- $f$ is a maximum flow in $G$.
- $G_f$ contains no augmenting paths.
- $|f| = c(S, T)$ for some cut $(S, T)$ of $G$. 

![Diagram](image_url)
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 value 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.
Push-relabel idea
Push-relabel idea
Push-relabel idea

Diagram showing a network with nodes s, a, b, c, d, and t. The edges and their capacities are indicated with values such as 7/9, 2/4, 6/6, 8/8, 0/5, and others.
Preflow, excess flow and height

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

\[
\sum_{v \in V} f(v, u) - \sum_{v \in V} f(u, v) \geq 0.
\]

The excess flow of a vertex $u$ is

\[
e(u) = \sum_{v \in V} f(v, u) - \sum_{v \in V} f(u, v).
\]

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

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

- $h(s) = |V|$,
- $h(t) = 0$, and
- $h(u) \leq h(v) + 1$ for every residual edge $(u, v) \in E_f$. 
**Push operation**

**Push**$(u, v)$

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

**Effect:** Push $\min(e(u), cf(u, v))$ overflow from $u$ to $v$

$$\Delta \leftarrow \min(e(u), cf(u, v))$$

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

$$f(u.v) \leftarrow f(u, v) + \Delta$$

else

$$f(v.u) \leftarrow f(v, u) - \Delta$$

\[ e(u) \leftarrow e(u) - \Delta \]
\[ e(v) \leftarrow e(v) + \Delta \]

**Example.**

\[
\begin{align*}
  e(u) &= 5 & e(v) &= 1 & e(u) &= 1 & e(v) &= 5 \\
  h(u) &= 4 & h(v) &= 3 & h(u) &= 4 & h(v) &= 3
\end{align*}
\]
Relabel operation

Relabel\((u)\)

**Condition:** \( u \) is overflowing and \( h(u) \leq h(v) \) for all \( v \in V \) with \((u, v) \in E_f\)

**Effect:** Increase the height of \( u \)
\[ h(u) \leftarrow 1 + \min\{h(v) : (u, v) \in E_f\} \]

Example.

[Diagram showing a network with nodes labeled \( u, v, x, y, z \) and edges labeled with values.]

- \( h(u) = 4 \) to \( h(u) = 6 \)
- \( h(v) = 1 \) to \( h(v) = 1 \)
- \( h(z) = 6 \) to \( h(z) = 6 \)

[Relabeling \( u \) changes its height from 4 to 6, adjusting the heights of its neighbors accordingly.]
**Push-Relabel algorithm**

**Push-Relabel(G)**

*InitPreflow*(G, s)

while there exists an applicable Push or Relabel operation x do

apply x

**InitPreflow(G, s)**

\[ h(v) \leftarrow 0, \quad e(v) \leftarrow 0 \quad \forall v \in V \]

\[ h(s) \leftarrow |V| \]

\[ f(u, v) \leftarrow 0 \quad \forall (u, v) \in E \]

for each v adjacent to s do

\[
\begin{align*}
    & f(s, v) \leftarrow c(s, v) \\
    & e(v) \leftarrow c(s, v)
\end{align*}
\]

- initialises heights
- pushes max flow over all outgoing edges of s
Correctness

Part 1.
If the algorithm terminates, the preflow is maximum flow.
- If an overflowing vertex exists, the algorithm can continue.
- The algorithm maintains $f$ as a preflow and $h$ as a height function.
- 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 calls of RELABEL.
- Find upper bound for calls of PUSH.
Lemma 1.
If a vertex $u$ is overflowing, either a push or a relabel operation applies to $u$.

Proof.
Assuming $h(u)$ is valid, we have
- $h(u) \leq h(v) + 1$ for all $v$ with $(u, v) \in E_f$.

If no push operation 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:
- $h(s) = |V|$,
- $h(t) = 0$, and
- $h(u) \leq h(v) + 1$ for every residual edge $(u, v) \in E_f$.

Push$(u, v)$

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

$\Delta \leftarrow \min(e(u), c_f(u, v))$

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

$\quad f(u, v) \leftarrow f(u, v) + \Delta$

else

$\quad f(v, u) \leftarrow f(v, u) + \Delta$

$e(u) \leftarrow e(u) - \Delta$

$e(v) \leftarrow e(v) + \Delta$

Relabel$(u)$

\textbf{Condition:} $u$ is overflowing, $h(u) \leq h(v) \forall v \in V$ with $(u, v) \in E_f$

$h(u) \leftarrow 1 + \min\{h(v) : (u, v) \in E_f\}$
Maintaining the preflow

**Lemma 2.**
The push-relabel algorithm maintains a preflow $f$.

**Proof.**
- **InitPreflow** initialises a preflow $f$. ✓
- **Relabel**($u$) doesn’t affect $f$. ✓
- **Push**($u, v$) maintains $f$ as a preflow. ✓

**Height function:**
- $h(s) = |V|$, 
- $h(t) = 0$, and 
- $h(u) \leq h(v) + 1$ for every residual edge $(u, v) \in E_f$.

**Push**($u, v$)

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

$\Delta \leftarrow \min(e(u), c_f(u, v))$

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

$f(u, v) \leftarrow f(u, v) + \Delta$

e($u$) $\leftarrow$ e($u$) $- \Delta$

e($v$) $\leftarrow$ e($v$) $+ \Delta$

else

$f(v, u) \leftarrow f(v, u) + \Delta$

e($u$) $\leftarrow$ e($u$) $- \Delta$

e($v$) $\leftarrow$ e($v$) $+ \Delta$

**Relabel**($u$)

**Condition:** $u$ is overflowing, 

$h(u) \leq h(v)$ $\forall v \in V$ with $(u, v) \in E_f$

$h(u) \leftarrow 1 + \min\{h(v) : (u, v) \in E_f\}$
Maintaining the height function

Lemma 3.
The push-relabel algorithm maintains \( h \) as a height function.

Proof.
- \textsc{InitPreflow} initialises \( h \) as a height function. ✓
- \textsc{Push}(u, v) leaves \( h \) a height function. ✓
  - If \((v, u)\) added to \( E_f \), then \( h(v) = h(u) - 1 < h(u) + 1 \).
  - If \((u, v)\) removed from \( E_f \), then ✓.
- \textsc{Relabel}(u) leaves \( h \) a height function. ✓
  - \((u, v)\) ∈ \( E_f \), then \( h(u) \leq h(v) + 1 \)
  - \((w, u)\) ∈ \( E_f \), then \( h(w) < h(u) + 1 \)

Height function:
- \( h(s) = |V| \),
- \( h(t) = 0 \), and
- \( h(u) \leq h(v) + 1 \) for every residual edge \((u, v)\) ∈ \( E_f \).

\textsc{Push}(u, v)
Condition: \( u \) is overflowing, \( c_f(u, v) > 0 \), and \( h(u) = h(v) + 1 \)
\[ \Delta \leftarrow \min(e(u), c_f(u, v)) \]
if \((u, v)\) ∈ \( E \) then
  \[ f(u,v) \leftarrow f(u,v) + \Delta \]
else
  \[ f(v,u) \leftarrow f(v,u) + \Delta \]
\[ e(u) \leftarrow e(u) - \Delta \]
\[ e(v) \leftarrow e(v) + \Delta \]

\textsc{Relabel}(u)
Condition: \( u \) is overflowing,
\[ h(u) \leq h(v) \forall v \in V \text{ with } (u, v) \in E_f \]
\[ h(u) \leftarrow 1 + \min\{h(v) : (u, v) \in E_f \} \]
Reachability of the sink

**Lemma 4.**
During the push-relabel algorithm, there is no path from $s$ to $t$ in $G_f$.

**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 then and since $k \leq |V| - 1$, follows $h(s) < |V|$. 

**Height function:**
- $h(s) = |V|$,  
- $h(t) = 0$, and  
- $h(u) \leq h(v) + 1$ for every residual edge $(u, v) \in E_f$.
Partial correctness of the algorithm

**Theorem 5.**
If the push-relabel algorithm terminates, the computed preflow $f$ is a maximum flow.

**Proof.**
- By Lemma 1, the algorithm stops, when there is no overflowing vertex.
- By Lemma 2, $f$ is a preflow.
  $\Rightarrow f$ is a flow.
- By Lemma 3, $h$ is a height function.
- So by Lemma 4, there is no st-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.
■ If an overflowing exists, the algorithm can continue.
■ The algorithm maintains $f$ as a preflow and $h$ as a height function.
■ 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 calls of \texttt{Relabel}.
■ Find upper bound for calls of \texttt{Push}.
Reachability of source in residual graph

**Lemma 6.**
There is a path from every overflowing vertex \( v \) to \( s \) in \( G_f \).

**Proof.**
- \( S_v \) ← vertices reachable from \( v \) in \( G_f \).
- Suppose \( v \not\in S_v \).
- Since \( f \) a preflow and \( s \not\in S_v \), we have \( \sum_{w \in S_v} e(w) \geq 0 \).
- Since \( v \in S_v \), we even have \( \sum_{w \in S_v} e(w) > 0 \).
- There is edge \((u, w)\) with \( u \not\in S_v, w \in S_v \) and \( f(u, w) > 0 \).
- But then \( c_f(w, u) > 0 \), meaning \( u \) is reachable from \( v \).
Upper bound on height and \texttt{RELABEL} operations

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

**Proof.**

- Statement holds after initialisation.
- 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-relable algorithm executes at most $2|V|^2$ \texttt{RELABEL} operations.

**Height function:**
- $h(s) = |V|$, 
- $h(t) = 0$, and 
- $h(u) \leq h(v) + 1$ for every residual edge $(u,v) \in E_f$.

**Relabel($u$)**

\textbf{Condition:} $u$ is overflowing, $h(u) \leq h(v) \ \forall v \in V$ with $(u,v) \in E_f$

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

---

**Diagram:**

- Nodes: $s$, $v_1$, $v_2$, $V$
- Edges: $e(v) > 0$

The diagram illustrates the flow in the graph $G_f$ with nodes $s$, $v_1$, and $v_2$, and an edge $e(v) > 0$. The push-relable algorithm operates on this graph, with the height function and relabel condition applied as described.
Saturating and unsaturating \textbf{Push} operations

The operation \textbf{Push}(u, v) is

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

\[
\begin{align*}
\text{Push}(u, v) & \quad \Delta = 4 \\
\end{align*}
\]

- and **unsaturating** otherwise.

\[
\begin{align*}
\text{Push}(u, v) & \quad \Delta = 2 \\
\end{align*}
\]
Upper bound on saturating \textbf{Push} operations

**Lemma 9.**
The push-relable algorithm executes at most $2|V||E|$ saturating \textbf{Push} operations.

**Proof.**
- Consider saturating \textbf{Push}$(u, v)$
  - $h(u) = h(v) + 1$
- For another saturating \textbf{Push}$(u, v)$, first \textbf{Push}$(v, u)$ necessary
  - $h(v) = h(u) + 1$ necessary
- After another saturating \textbf{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 $2|V| - 1$ sat. \textbf{Push} operations for edge $(u, v)$.

\textbf{Push}$(u, v)$

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

\[ \Delta \leftarrow \min(e(u), c_f(u, v)) \]

\textbf{if} $(u, v) \in E$ \textbf{then}

\[ f(u,v) \leftarrow f(u,v) + \Delta \]

\textbf{else}

\[ f(v,u) \leftarrow f(v,u) + \Delta \]

\[ e(u) \leftarrow e(u) - \Delta \]

\[ e(v) \leftarrow e(v) + \Delta \]
Upper bound on unsaturating Push operations

**Lemma 10.**
The push-relable algorithm executes at most \(4|V|^2|E|\) unsaturating Push ops.

**Proof.**
- Consider \(H = \sum_{v \in V \setminus \{s,t\}, v \text{ overflowing}} h(v)\).

- After initialisation and at the end \(H = 0\).
- A saturating Push increases \(H\) by at most \(2|V| - 1\).
- By Lemma 8, all saturating Push ops. increases \(H\) by at most \((2|V| - 1) \cdot 2|V||E|\).
- By Lemma 7, all Relabel ops increases \(H\) by at most \((2|V| - 1) \cdot |V|\).
- An unsaturating Push\((u, v)\) decrease \(H\) by at least 1, since \(h(u) - h(v) \geq 1\).
Termination of the algorithm

**Theorem 5.**
If the push-relabel algorithm terminates, the computed preflow $f$ is a maximum flow.

**Theorem 11.**
The push-relabel algorithm terminates after $O(|V|^2|E|)$ valid Push or Relabel ops.

**Proof.**
- Follows by Corollary 8 and Lemma 9+10.
Implementation

The actual running time depends on the selection order of 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 Push select *highest* overflowing vertex: $\mathcal{O}(|V|^2|E|^\frac{1}{2})$

- **Excess scaling:**
  For 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 $C = \max_{(u,v) \in E} c(u, v)$
The push-relabel method offers an alternative framework to the Ford-Fulkerson method to develop algorithms that solve the maximum flow problem.

Push-relabel algorithms are regarded as benchmarks for maximum flow algorithms.

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.
Literature

Main source:
■ [CLRS Ch26] ← Cormen et al. “Introduction to Algorithms”

Original papers:
■ [Goldberg, Tarjan ’88] A new approach to the maximum-flow problem

Links:
■ MaxFlow Ford-Folkerson and Edmonds-Karp animations
  https://visualgo.net/en/maxflow