Advanced Algorithms

Nicole Megow (Universität Bremen)

SoSe 2025

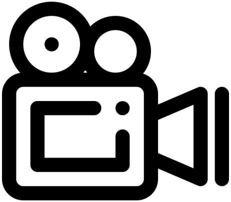
Network Flows: Preflow-Push Algorithms

Lecture 7

Recording of this Lecture

This lecture will be recorded

- Recording only of the lecturers by themselves.
- If there are questions from the audience, please make a clear signal if the microphone shall be muted.
- Our goal is to record the lecture, but it is no guarantee that each lecture will be recorded.



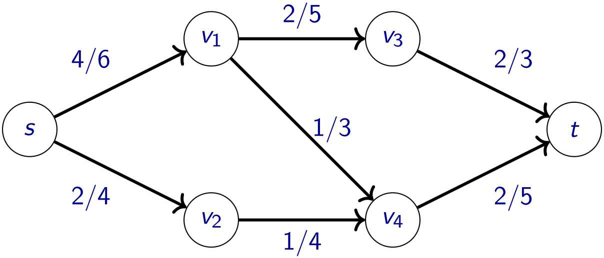
Preflows

Definition (Preflow)

Let $\mathcal{N} = (V, E, c, s, t)$ be an s-t-network. A preflow in $\mathcal{N}$ is a function $f : V \times V \to \mathbb{R}$ such that

1. for all $e \in E$, $0 \leq f(e) \leq c(e)$, and
2. for all $v \in V \setminus \{s\}$, $\sum_{e \in \delta^-(v)} f(e) \geq \sum_{e \in \delta^+(v)} f(e)$.

Excess: $e_f(v) := \sum_{e \in \delta^-(v)} f(e) - \sum_{e \in \delta^+(v)} f(e)$



Height Function and Eligible Edges

Definition (Height function)

Let $\mathcal{N} = (V, E, c, s, t)$ be an s-t-network. A height function is a function $d : V \to \mathbb{N}_0$. A height function d is legal w.r.t. a preflow f in $\mathcal{N}$ if

1. $d(s) = |V|$,
2. $d(t) = 0$, and
3. $d(u) \leq d(v) + 1$ for every edge $(u, v) \in E_f$.

Definition (Eligible edges and active vertices)

An edge $(u, v) \in E_f$ is eligible w.r.t. a preflow f and a height d if

1. $e_f(u) > 0$, and
2. $d(u) = d(v) + 1$.

Moreover, we call a vertex $v \in V \setminus \{s, t\}$ active if $e_f(v) > 0$.

Push and Relabel Operations

Push

Function Push(u, v)**:**

Precondition : $(u, v) \in E_f$, u is active, and (u, v) is eligible

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

$f(u, v) \leftarrow f(u, v) + \min\{e_f(u), c_f(u, v)\}$

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

$f(v, u) \leftarrow f(v, u) - \min\{e_f(u), c_f(u, v)\}$

Relabel

Function Relabel(u)**:**

Precondition : u is active and no edge out of u is eligible

$d(u) \leftarrow d(u) + 1$

The Generic Preflow-Push Algorithm

Init

Function Init($\mathcal{N}$)**:**
  $d(s) = n, d(v) = 0$ for all $v \in V \setminus \{s\}$
  $f := 0$
  for $v \in V$ with $(s, v) \in E$ **do**
    $f(s, v) \leftarrow c(s, v)$

Full Algorithm

Input: An s-t-network $\mathcal{N} = (V, E, c, s, t)$
Init($\mathcal{N}$)
while some Push or Relabel operation is possible **do**
  Carry out such an operation
return f

Correctness

Lemma

After `Init` *and after every* `Push` *or* `Relabel`*, f is a preflow and d is a legal height function with respect to f.*

Lemma

Let f be a preflow such that there exists a legal height function d with respect to f. Then, there exists no s-t-path in $\mathcal{N}_f$.

Lemma

If the algorithm terminates, then f is a maximum flow in $\mathcal{N}$.

Running Time Part 1

Lemma

Let f be a preflow, and let $w \in V$ be an active vertex with respect to f. Then, there exists a path in $\mathcal{N}_f$ from w to s.

Lemma

For all $u \in V$, we always have $d(u) \leq 2n - 1$.

Corollary

The algorithm performs at most $2n^2$ `Relabel` *operations.*

Lemma

There are at most $2nm$ saturating `Push` *operations.*

Running Time Part 2

Lemma

There are at most $6n^2m$ *non-saturating* `Push` *operations.*

Each `Init`, `Push`, and `Relabel` operation takes $O(1)$ time

Theorem

The generic Preflow-Push algorithm computes a maximum flow in an s-t*-network with* n *vertices and* m *edges in* $O(n^2m)$ *time.*

The Max-Height Algorithm

Discharge

```
Function Discharge(u):
    while u is active and at least one edge (u, v) ∈ E_f is eligible do
        Push(u,v)
    if u is active then
        Relabel(u)
```

The Max-Height Algorithm

Input: An s-t-network $\mathcal{N} = (V, E, c, s, t)$

```
Init(N)
while at least one vertex is active do
    u ← an active vertex with height d(u) = max_{v∈A} d(v)
    Discharge(u)
return f
```

Universität Bremen

Analysis Part 1

- $d^* = \max_{v \in A} d(v)$, a^* number of active vertices with height d^*
- $b(v) = |\{u \in V \mid d(u) \leq d(v)\}|$

For $K \geq 1$, we define a potential function $\Phi = \Phi_1 + \Phi_2 + \Phi_3$, where

- $\Phi_1 = Kd^*$,
- $\Phi_2 = \min\{a^*, K\}$, and
- $\Phi_3 = \frac{1}{K} \sum_{v \in V} b(v)$.

Lemma

For every `Push` *operation, we have* $\Delta\Phi_1 + \Delta\Phi_2 \leq 0$. *Moreover, if* d^* *changes due to the* `Push`, *we have* $\Delta\Phi_1 + \Delta\Phi_2 \leq -1$.

Analysis Part 2

- $d^* = \max_{v \in A} d(v)$, a^* number of active vertices with height d^*
- $b(v) = |\{u \in V \mid d(u) \leq d(v)\}|$

For $K \geq 1$, we define a potential function $\Phi = \Phi_1 + \Phi_2 + \Phi_3$, where

- $\Phi_1 = Kd^*$,
- $\Phi_2 = \min\{a^*, K\}$, and
- $\Phi_3 = \frac{1}{K} \sum_{v \in V} b(v)$.

Lemma

We have the following bounds on $\Delta\Phi$:

- *If* `Relabel(`u`)` *is being executed, then $\Delta\Phi \leq K + n/K$.*
- *If* `Push(`u, v`)` *is being executed and saturating, then $\Delta\Phi \leq n/K$.*
- *If* `Push(`u, v`)` *is being executed and non-saturating, then $\Delta\Phi \leq -1$.*

Analysis Part 3

- $d^* = \max_{v \in A} d(v)$, a^* number of active vertices with height d^*
- $b(v) = |\{u \in V \mid d(u) \leq d(v)\}|$

For $K \geq 1$, we define a potential function $\Phi = \Phi_1 + \Phi_2 + \Phi_3$, where

- $\Phi_1 = Kd^*$,
- $\Phi_2 = \min\{a^*, K\}$, and
- $\Phi_3 = \frac{1}{K} \sum_{v \in V} b(v)$.

Lemma

The number of non-saturating `Push` *operations is at most* $O(n^2\sqrt{m})$.

Theorem

The Max-Height algorithm computes a maximum flow in an s-t*-network with* n *vertices and* m *edges in* $O(n^2\sqrt{m})$ *time.*