Approximate Max Flow in Near-linear Time

December 11, 2025

1 Recall some terminology

- Let $G = (V, E)$ be an undirected graph with edge capacities $c \in \mathbb{R}^E_{\geq 0}$ (with n vertices and m edges).
- A **flow** $f : V \times V \to \mathbb{R}$ satisfies $f(u, v) = -f(v, u)$ and $f(u, v) = 0$ for $\{u, v\} \notin E$.
  - The notation $f(u, v) > 0$ means that **mass** is routed in the direction from u to v.
  - The **congestion** of f is $\max_{\{u,v\}\in E} \frac{|f(u,v)|}{c(u,v)}$. If the congestion is at most 1, we say that f is **feasible** or **respects capacities.**
  - The **net flow going out of** u is $f_{out}(u) = \sum_{v\in V} f(u, v)$.
- Let $\mathsf{d} : V \to \mathbb{R}$ be a **demand function**.
  - We say that flow f **satisfies demand** d if $\mathsf{d}(u) = f_{out}(u)$ for all $u \in V$.
  - For any $S \subseteq V$, let $\mathsf{d}(S) = \sum_{v\in S} \mathsf{d}(v)$ be the **total demand** on S.
  - Assume $\mathsf{d}(V) = 0$.

Fact 1.1. *$|\mathsf{d}(S)| \leq \epsilon \cdot \delta(S)$ for all $S \subseteq V$ iff there is a flow with congestion ϵ satisfying d.*

- We say d is **feasible** if $|\mathsf{d}(S)| \leq \delta(S)$ for all $S \subseteq V$ (i.e. there is a feasible flow satisfying the demand).

2 The Approximate Max Flow Problem

- **The $(1 + \epsilon)$-approximate max flow problem**:
  - given a capacitated graph $G = (V, E, c)$ and a demand d,
  - output either
    - a cut S where $\delta(S) < \mathsf{d}(S)$
    - a flow $f \in \mathbb{R}^E$ with congestion $(1 + \epsilon)$ satisfying d

Exercise 2.1 (Routing through maximum spanning tree)**.** Show how to solve the $O(m)$-approximate max flow problem in $O(m \log n)$ time using maximum spanning trees.

- One of the key idea for solving this problem is to change the problem a bit...

2.1 "Almost Route" demand instead

- For a flow f and a demand function d, define **excess** or **residual demand** $\mathsf{d}^f(v) = \mathsf{d}(v) - f_{out}(v)$ for every $v \in V$.
- We say that f **ϵ-satisfies** d if $|\mathsf{d}^f(S)| \leq \epsilon\delta(S)$ for all $S \subseteq V$
  - That is, there exists f_{aug} with congestion ϵ where $f + f_{aug}$satisfies d.
- **The ϵ-almost-route problem**: output either
  - a cut S where $\delta(S) < |\mathsf{d}(S)|$
  - a feasible flow $f \in \mathbb{R}^E$ that ϵ-satisfies d.

Lemma 2.2. *We can solve the approximate max flow problem by solving the almost-route problem $O(\log n)$ time plus $O(m)$ additional time.*

Proof. Given graph G and demand d_0, call almost-route on (G, d_0).

- If we get a cut S where $\delta(S) < |\mathsf{d}_0(S)|$, done.
- If we get a feasible flow f_0 ϵ-satisfying d_0, we will show how to construct f satisfying d_0 with congestion $(1 + O(\epsilon))$.
  - Let $\mathsf{d}_1 \leftarrow \mathsf{d}_0^{f_0}$ be the residual demand. Call almost-route on $(\epsilon \cdot G, \mathsf{d}_1)$.
  - Can we get a cut? No.
    * We knew that $|\mathsf{d}_1(S)| \leq \epsilon \cdot \delta(S)$ for all $S \subseteq V$ (as f_0 ϵ-satisfies d_0).
  - We must get a feasible flow f_1 on $\epsilon \cdot G$ that ϵ-satisfies d_1.
    * f_1 has congestion ϵ on G.
    * $|\mathsf{d}_1^{f_1}(S)| \leq \epsilon^2\delta(S)$ for all $S \subseteq V$.
  - Let $\mathsf{d}_2 \leftarrow \mathsf{d}_1^{f_1}$. Call almost-route on $(\epsilon^2 \cdot G, \mathsf{d}_2)$ and get a feasible flow f_2 on $\epsilon^2 \cdot G$ that ϵ-satisfies d_2.
    * f_2 has congestion ϵ^2 on G.
    * $|\mathsf{d}_2^{f_2}(S)| \leq \epsilon^3\delta(S)$ for all $S \subseteq V$.
  - Repeat until we call almost-route on $(\epsilon^L \cdot G, \mathsf{d}_L)$ we get f_L where $L = O(\log m)$.

Lemma 2.3. *Consider $f = f_0 + f_1 + \cdots + f_L$. We have*

1. *f has congestion at most $1 + \epsilon + \cdots + \epsilon^L = 1 + O(\epsilon)$, and*
2. *f (ϵ^{L+1})-satisfies d_0.*

Proof. For the first statement, this is because each f_i has congestion at most ϵ^i on G.

For the second statement, for any S, we have

$$\begin{aligned}
\mathsf{d}_0^f(S) &= \mathsf{d}_0(S) - f_0(S) - f_1(S) - \cdots - f_L(S) \\
&= \mathsf{d}_1(S) - f_1(S) - \cdots - f_L(S) \\
&= \mathsf{d}_L(S) - f_L(S) \\
&= \mathsf{d}_L^{f_L}(S)
\end{aligned}$$

but $|\mathsf{d}_L^{f_L}(S)| \leq \epsilon^{L+1}\delta(S)$ □

□

- The residual demand d_0^f can be satisfied with a flow with congestion ϵ^{L+1}.
- We can find a flow f_{final} satisfying d_0^f with congestion $O(m) \cdot \epsilon^{L+1} \leq 1/\text{poly}(m)$.
  - How? Route through maximum spanning tree, see Exercise 2.1.
- We just return $f + f_{final}$ which has $1 + O(\epsilon)$ congestion and exactly satisfies d_0

3 First Ingredient: Congestion Approximator

- Motivation:
  - How can we guarantee that f ϵ-satisfies d?
  - There are 2^n many constraints $|\mathsf{d}^f(S)| \leq \epsilon\delta(S)$ for all $S \subseteq V$.

Definition 3.1 (Congestion Approximator)**.** An **congestion approximator of G with quality** q is a family of cuts $\mathcal{C}$ such that, for any demand vector $\mathsf{d} \in \mathbb{R}^V$ we have that if $|\mathsf{d}(S)| \leq \delta(S)$ for all $S \in \mathcal{C}$, then $|\mathsf{d}(S)| \leq q \cdot \delta(S)$ for all $S \subseteq V$.

Exercise 3.2. Let T be a tree. Let $\mathcal{C}$ contains all $n-1$ cuts defined by each tree edge. Show that $\mathcal{C}$ is a congestion approximator of T with quality 1.

Exercise 3.3. Let G be a ϕ-expander. Let $\mathcal{C} = \{\{v\}\}_{v \in V}$ contains all n singleton cuts. Show that $\mathcal{C}$ is a congestion approximator of T with quality $1/\phi$.

3.1 Construction via Tree Flow Sparsifier

Lemma 3.4. *Let T be a tree flow sparsifier of G with quality q. For every $u \in V(T)$, let $S_u \subseteq V$ denote the set of leaves in the subtree rooted at u. Let*

$$\mathcal{C}_T = \{S_u \mid u \in V(T)\}.$$

Then $\mathcal{C}_T$ is a congestion approximator of G with quality q.

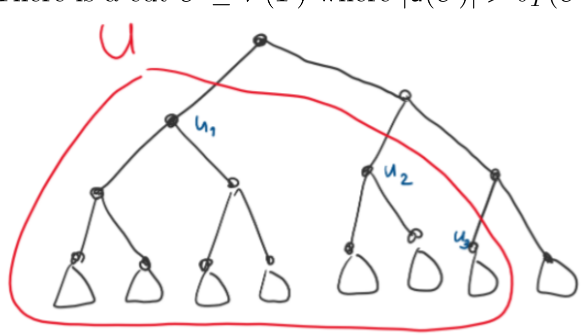
Proof. Given a demand $\mathsf{d} \in \mathbb{R}^{V(G)}$ on $V(G)$, we will prove two things

1. If $|\mathsf{d}(S)| \leq \delta(S)$ for all $S \in \mathcal{C}_T$, then d is feasible in T.
2. If d is feasible in T, then d can be routed with congestion q in G (and so $|\mathsf{d}(S)| \leq q \cdot \delta(S)$ for all $S \subseteq V$).

□

For the first step, suppose for contradiction that there is d is infeasible.

- There is a cut $U \subseteq V(T)$ where $|\mathsf{d}(U)| > \delta_T(U)$.



- But then there must exists $u \in V(T)$ where $|\mathsf{d}(S_u)| > c_T(u, parant(u)) \geq \delta(S_u)$

For the second step, suppose d is feasible in T.

- Let f_T be the feasible flow in T that routes d.
- Let F_T be a set of flow-paths in the decomposition of f_T. Think of F_T as a multi-commodity flow.
- Let D be the multi-commodity demand that F_T routes. In particular, D is routable in T.
- But then D is routable in G with congestion q, because T has is a flow sparsifier with quality q.
- So d is routable in G with congestion q.
  - For more details, let F_G be a multi-commodity flow that routes D in G.
  - Let f_G be obtained from F_G by treating F_G as a single-commodity flow (i.e. allow flow cancellation).
  - We have that f_G routes d in G.

Using the state-of-the-art algorithm for tree-flow sparsifiers, we have:

Corollary 3.5. *A congestion approximator $\mathcal{C}$ of G with quality* $\mathrm{polylog}(n)$ *and $|\mathcal{C}| \leq 2n$ can be constructed in $\tilde{O}(m)$ time.*[1] *Moreover, each vertex is contained in at most $O(\log n)$ sets from $\mathcal{C}$.*

We can also use the expander hierarchy from the previous class. But the quality would be $n^{o(1)}$ and the construction time would be $m^{1+o(1)}$.

3.2 Reduce the problem further...

- So we now reduce our problem to the following:
- Given a congestion approximator $\mathcal{C}$ with quality q and $|\mathcal{C}| = O(n)$, output either
  - a cut S where $\delta(S) < |\mathsf{d}(S)|$
  - a feasible flow $f \in \mathbb{R}^E$ such that for all $S \in \mathcal{C}$

  $$|\mathsf{d}^f(S)| \leq \frac{\epsilon}{q}\delta(S)$$

  because this implies that f ϵ-satisfies d.
- In other words, either
  - find a violating cut, or
  - make sure that the total excess on every set $S \in \mathcal{C}$ is small (instead of all sets $S \subseteq V$).

[1] https://arxiv.org/pdf/1411.7631.pdf

4 Second Ingredient: Multiplicative Weights Update

Here, we state how the multiplicative weight update (MWU) framework can be used for solving *general* LPs.

- Given a following linear program (LP):

$$\begin{aligned} (LP) \quad & \text{Find } x \text{ s.t.} \\ & Ax \geq b \\ & x \in \Delta \end{aligned}$$

  where $A \in \mathbb{R}^{n \times m}$, $b \in \mathbb{R}^n$, $x \in \mathbb{R}^m$, Δ is convex (e.g. $x \geq 0$).

- Define:
  - $Ax \geq b$ as a set of "hard" constraints
  - $x \in \Delta$ as an "easy" constraint

- Our goal is to find $\overline{x}$ that approximately satisfies the LP, i.e.,

$$\begin{aligned} A\overline{x} &\geq b - \epsilon\mathbf{1} \\ \overline{x} &\in \Delta \end{aligned}$$

- The MWU allows us to find $\overline{x}$ if we can solve the problem "average" problem for a few rounds.

- More precisely,
  - In each round $t = 1, 2, \ldots$, the framework will fix some $p_t \in \mathbb{R}^n_{\geq 0}$ (i.e. "weights" of hard constraints).
    - * p_1 is initially the all-one vector.
  - We need to solve the following "average" LP:

$$\begin{aligned} & \text{Find } x \text{ s.t.} \\ & p_t^\top Ax \geq p_t^\top b \\ & x \in \Delta \end{aligned}$$

    Note that, the hard constraint are not average into just one constraint!
  - Suppose we can compute a feasible solution x_t of the average problem (i.e. $p_t^\top Ax_t \geq p_t^\top b$ and $x_t \in \Delta$) such that x_t has **width** at most ρ, i.e.

$$|A_i x_t - b_i| \leq \rho \forall i \in [n]$$

    That is, x_t does not violate or over-satisfy each constraint by more than ρ.
  - Given x_t, the MWU framework will adaptively update p_{t+1} based on x_t as follows:

$$(p_{t+1})_i = (p_t)_i \cdot \exp(\frac{\epsilon}{\rho} \cdot (b_i - A_i x_t))$$

    and proceed to the next round.

  * Intuition: If the i-th constraint is violated a lot by x_t ($A_i x_t \ll b_i$), then the weight $(p_{t+1})_i$ increases a lot. So in the next round, the i-th constraint should be satisfied.
  * From the way we update weights, this give the name *multiplicative weight update* framework.

- After $T = O(\rho^2 \log n/\epsilon^2)$ rounds, the MWU framework guarantees that $\overline{x} = (\sum_{t=1}^{T} x_t)/T$ satisfies

$$\begin{aligned} A\overline{x} &\geq b - \epsilon\mathbf{1} \\ \overline{x} &\in \Delta \end{aligned}$$

- It is quite an amazing framework: reducing a worst-case problem to an average problem.

5 The Algorithm

Recall our goal: either find

- a violating cut S where $\delta(S) < |\mathsf{d}(S)|$, or
- a feasible f where $|\mathsf{d}^f(S)| \leq \epsilon/q \cdot \delta(S)$ for all $S \in \mathcal{C}$.

To formulate this goal as an LP (so that we can apply MWU), let's define some notations.

- Let $\epsilon' = \epsilon/q$.
- Let $B \in \{0, -1, 1\}^{V \times E}$ be the incidence matrix of G.
  - Note that, for every v, we have $f_{out}(v) = (Bf)_v$
- For any $S \subseteq V$, let $1_S \in \mathbb{R}^V$ be the indicator vector of S.
- We have

$$\begin{aligned} |\mathsf{d}^f(S)| \leq \epsilon' \cdot \delta(S) & \quad\iff \\ 1_S^\top(\mathsf{d} - Bf) \leq \epsilon' \cdot \delta(S) \text{ and } 1_S^\top(\mathsf{d} - Bf) \geq -\epsilon' \cdot \delta(S) & \quad\iff \\ \frac{1}{\delta(S)}\mathbf{1}_S^\top Bf \geq \frac{1}{\delta(S)}\mathbf{1}_S^\top \mathsf{d} - \epsilon' \text{ and } -\frac{1}{\delta(S)}1_S^\top Bf \geq -\frac{1}{\delta(S)}\mathbf{1}_S^\top \mathsf{d} - \epsilon' & \end{aligned}$$

We can rewrite our goal: either find

- a violating cut S where $\delta(S) < \mathsf{d}(S)$, or
- a flow f where
  1. for all e, $|f_e|/c_e \leq 1$
  2. for all $S \in \mathcal{C}$, $\frac{1}{\delta(S)}\mathbf{1}_S^\top Bf \geq \frac{1}{\delta(S)}\mathbf{1}_S^\top \mathsf{d} - \epsilon'$ and $-\frac{1}{\delta(S)}1_S^\top Bf \geq -\frac{1}{\delta(S)}\mathbf{1}_S^\top \mathsf{d} - \epsilon'$ for all $S \in \mathcal{C}$.

So now, let's write an LP

$$
\begin{aligned}
&\text{Find } f \text{ s.t.} && \\
&\frac{1}{\delta(S)}\mathbf{1}_S^\top B f \geq \frac{1}{\delta(S)}\mathbf{1}_S^\top \mathsf{d} && \text{for all } S \in \mathcal{C} \qquad (1)\\
&\frac{-1}{\delta(S)}\mathbf{1}_S^\top B f \geq \frac{-1}{\delta(S)}\mathbf{1}_S^\top \mathsf{d} && \text{for all } S \in \mathcal{C} \qquad (2)\\
&|f_e|/c_e \leq 1 && \text{for all } e \in E \qquad (3)
\end{aligned}
$$

We want to find a violating cut S or find f that satisfies the above LP upto ϵ' additive factor. We are now ready to apply MWU:

- 1 and 2 are "hard" constraints.
- 3 are "easy" constraints.

In round t of MWU, the framework sets the weights $\{p_{S,\circ}\}_{S\in\mathcal{C},\circ\in\{+,-\}}$ of each hard constraint where $p_{S,\circ} \geq 0$.

- Given the weights, we have need to solve the following average LP

$$
\begin{aligned}
&\text{Find } f \text{ s.t.} \\
&\sum_{S\in\mathcal{C}} \frac{p_{S,+}}{\delta(S)}\mathbf{1}_S^\top B f - \sum_{S\in\mathcal{C}} \frac{p_{S,-}}{\delta(S)}\mathbf{1}_S^\top B f \geq \sum_{S\in\mathcal{C}} \frac{p_{S,+}}{\delta(S)}\mathbf{1}_S^\top \mathsf{d} - \sum_{S\in\mathcal{C}} \frac{p_{S,-}}{\delta(S)}\mathbf{1}_S^\top \mathsf{d} \\
&|f_e|/c_e \leq 1 \qquad \text{for all } e \in E
\end{aligned}
$$

- Define potentials on vertices as

$$\phi = \sum_{S\in\mathcal{C}} \frac{p_{S,+}}{\delta(S)}\mathbf{1}_S - \sum_{S\in\mathcal{C}} \frac{p_{S,-}}{\delta(S)}\mathbf{1}_S \in \mathbb{R}^V.$$

- We need to solve

$$
\begin{aligned}
&\text{Find } f \text{ s.t.} \\
&\phi^\top B f \geq \phi^\top \mathsf{d} \\
&|f_e|/c_e \leq 1 \qquad \text{for all } e \in E
\end{aligned}
$$

  - To understand ϕ more intuitively, observe that

$$\phi_v = \sum_{S\ni v} \frac{1}{\delta(S)}(p_{S,+} - p_{S,-})$$

  - Each v is contained in $O(\log n)$ sets from $\mathcal{C}$, so ϕ can be computed $O(n \log n)$ time.
    * Actually, we can compute ϕ in $O(n)$ time when $\mathcal{C}$ is defined from a tree flow sparsifier. **(Exercise)**

- To solve this, it suffices to find f^t that maximizes $\phi^\top B f^t$ where $\max_e |f_e^t|/c_e \leq 1$. (Then, check if $\phi^\top B f^t \geq \phi^\top \mathsf{d}$).

- For any f, we can expand the expression as

$$\phi^\top B f = \sum_{e=(u,v)} (\phi_v - \phi_u) f_{(u,v)}$$

- To maximize each term, just set $f^t_{(u,v)} = c_{(u,v)} \mathrm{sgn}(\phi_v - \phi_u)$.
- We can find the optimal f^t where $\phi^\top B f^t = \sum_{e=(u,v)} c_e |\phi_u - \phi_v|$.

- There are two cases
  - If $\phi^\top B f^t < \phi^\top d$, we claim that we can find a violating cut. Done (to be proved below).
  - If $\phi^\top B f^t \geq \phi^\top d$, then the flow f^t satisfies the average problem in round t. We can feed f^t into the MWU framework and proceed to the next round.
  - How many round? This depends on the width of f^t. (How much f^t violates or over-satisfies the original constraints?)
  - **Claim**: the width of f^t is at most 2.
    * We want to prove that for all $S \in \mathcal{C}$

$$|\frac{1}{\delta(S)} \mathbf{1}_S^\top B f - \frac{1}{\delta(S)} \mathbf{1}_S^\top \mathsf{d}| \leq 2.$$

    * We have $\mathbf{1}_S^\top B f = \sum_{e \in E(S, V-S)} f_e \leq \sum_{e \in E(S,V-S)} c_e = \delta(S)$. So $|\frac{1}{\delta(S)} \mathbf{1}_S^\top B f| \leq 1$.
    * Also, we have $|\frac{1}{\delta(S)} \mathbf{1}_S^\top \mathsf{d}| = \frac{\mathsf{d}(S)}{\delta(S)} \leq 1$ because we can actually assume that $d_S \leq \delta(S)$ for each $S \in \mathcal{C}$ (this can be easily checked fast from the very beginning), otherwise we obtain a violating cut.
  - So there are $T = O(\log n / \epsilon'^2) = O(\frac{q^2}{\epsilon^2} \log n) = \tilde{O}(1/\epsilon^2)$ rounds.
  - After T rounds, we get $\overline{f} = \frac{f_1 + \cdots + f_T}{T}$ where

$$\begin{aligned}
\frac{1}{\delta(S)} \mathbf{1}_S^\top B \overline{f} &\geq \frac{1}{\delta(S)} \mathbf{1}_S^\top \mathsf{d} - \epsilon' && \text{for all } S \in \mathcal{C} \\
\frac{-1}{\delta(S)} \mathbf{1}_S^\top B \overline{f} &\geq \frac{-1}{\delta(S)} \mathbf{1}_S^\top \mathsf{d} - \epsilon' && \text{for all } S \in \mathcal{C} \\
|\overline{f}_e| / c_e &\leq 1 && \text{for all } e \in E
\end{aligned}$$

  - So $\overline{f}$ is a feasible flow that ϵ-satisfies d. Done.

- It remains to prove that $\phi^\top B f^t < \phi^\top d$, then there is a violating cut.

Lemma 5.1. *If there is $\phi \in \mathbb{R}^V$ where $\sum_{e=(u,v)} c_e |\phi_u - \phi_v| < \phi^\top \mathsf{d}$, then we can find a threshold cut $S_\tau = \{u \mid \phi_u < \tau\}$ such that $\delta(S_\tau) < \mathsf{d}(S_\tau)$ in $O(n)$ time.*

Proof. Rearrange the indices so that $\phi_1 \leq \phi_2 \leq \cdots \leq \phi_n$. □

- We can translate $\phi_i \leftarrow \phi_i - \phi_1$ so $\phi_1 = 0$. Why?
  - $\sum_{e=(u,v)} c_e |\phi_u - \phi_v|$ is translation invariant.
  - $\phi^\top \mathsf{d}$ is changed to $\phi^\top \mathsf{d} - \phi_0 \mathsf{d}(V) = \phi^\top \mathsf{d}$.

- We can assume $\phi_n = 1$ by scaling.
- Now, choose a random threshold $\tau \in [0, 1]$. Let $S_\tau = \{u \mid \phi_u \geq \tau\}$.

$$\begin{aligned}\mathbb{E}_\tau[\delta(S_\tau)] &= \sum_{e=(u,v)\in E} c_e \Pr[\phi_u \leq \tau < \phi_v] \\ &= \sum_{e=(u,v)} c_e |\phi_u - \phi_v|\end{aligned}$$

  and

$$\begin{aligned}\mathbb{E}_\tau[\mathsf{d}(S_\tau)] &= \sum_{v\in V} \phi_v \mathsf{d}(v) \\ &= \phi^\top \mathsf{d}\end{aligned}$$

- So we have

$$\mathbb{E}_\tau[\delta(S_\tau)] < \mathbb{E}_\tau[\mathsf{d}(S_\tau)]$$

  and so there exists τ where $\delta(S_\tau) < \mathsf{d}(S_\tau)$.

- Let's conclude the algorithm
  - In each round, given $\{p_{S,\circ}\}_{S\in\mathcal{C},\circ\in\{+,-\}}$, compute ϕ in $O(n)$ time.
  - for each edge $(u, v) \in E$, send flow at full capacity from v to u if $\phi_v > \phi_u$, and vice versa. Total time: $O(m)$.
  - There are $O(\frac{q^2}{\epsilon^2} \log n)$ rounds. So total time is $\tilde{O}(m/\epsilon^2)$.
- Output: either
  - a violating cut S where $\delta(S) < |\mathsf{d}(S)|$, or
  - a feasible f where $|\mathsf{d}^f(S)| \leq \epsilon/q \cdot \delta(S)$ for all $S \in \mathcal{C}$.
    - This implies that f ϵ-satisfies d.

6 Intuition: What happen in Expanders?

- Try to interpret this algorithm on ϕ-expanders.
- Here, the congestion approximator is very simple: all singleton cuts give a congestion approximator with quality $\frac{1}{\phi}$.
  - Why? See 3.3.
- When all the cuts from the congestion approximator are singletons, this allows us to interpret the algorithm more easily.
- From how MWU works, observe that $p^{t+1}_{v,+} = p^t_{v,+} \cdot \exp(\frac{\epsilon}{\rho} \cdot \frac{\mathsf{d}^{f^t}(v)}{\deg(v)})$ and $p^{t+1}_{v,-} = p^t_{v,-} \cdot \exp(\frac{\epsilon}{\rho} \cdot \frac{-\mathsf{d}^{f^t}(v)}{\deg(v)})$. So

$$\begin{aligned}p^t_{v,+} &= \exp(\frac{\epsilon}{\rho} \cdot \sum_{t=1}^{t} \frac{\mathsf{d}^{f^i}(v)}{\deg(v)}) \\ p^t_{v,-} &= \exp(\frac{\epsilon}{\rho} \cdot \sum_{t=1}^{t} \frac{-\mathsf{d}^{f^i}(v)}{\deg(v)})\end{aligned}$$

- Define $\overline{f}^t = \frac{f_1+\cdots+f_t}{t}$ as the average flow so far up to time t. Recall that our algorithm returns $\overline{f} = \overline{f}^T = \frac{f_1+\cdots+f_T}{T}$ where $T = O(\frac{\rho^2}{\epsilon^2}\log n)$.
  - By definition, we have $\sum_{i=1}^t \mathsf{d}^{f^i}(v) = t \cdot \mathsf{d}^{\overline{f}^t}(v)$.
  - $\mathsf{d}^{\overline{f}^t}(v)$ has a natural interpretation: it is the average excess at vertex v up to time t.
  - Our goal is that the average excess should be close to 0 by time T. That is, $\mathsf{d}^{\overline{f}^T}(v) \approx 0$.
- Now, the potential of v at round t is

$$\phi_v^t = \frac{p_{v,+}^t - p_{v,-}^t}{\deg(v)} = \frac{\exp(\frac{\epsilon}{\rho}\cdot t\cdot \frac{\mathsf{d}^{\overline{f}^t}(v)}{\deg(v)}) - \exp(\frac{\epsilon}{\rho}\cdot t\cdot \frac{-\mathsf{d}^{\overline{f}^t}(v)}{\deg(v)})}{\deg(v)}$$

- Up to scaling (also in the exponent)

$$\phi_v^t \sim e^{\mathsf{d}^{\overline{f}^t}(v)} - e^{-\mathsf{d}^{\overline{f}^t}(v)}$$

- Interpretation:
  - When $\mathsf{d}^{\overline{f}^t}(v) > 0$, then ϕ_v^t goes up positively exponentially fast.
  - When $\mathsf{d}^{\overline{f}^t}(v) < 0$, then ϕ_v^t goes down negatively exponentially fast.
- This make senses. Why?
  - Recall our we construct the flow in each round. For every edge (u,v), if $\phi_u > \phi_v$, then send flow at full capacity from u to v. Otherwise, do the opposite direction.
  - So if v has excess ($\mathsf{d}^{\overline{f}^t}(v) > 0$), then ϕ_v is big. So the algorithm would likely send flow out of v in the next round.
  - So if v has deficit ($\mathsf{d}^{\overline{f}^t}(v) < 0$), then ϕ_v is small. So the algorithm would likely send flow into v in the next round.
- At the end, we expect $\mathsf{d}^{\overline{f}}(v) \approx 0$ for all v.
  - More precisely, $|\mathsf{d}^{\overline{f}}(v)| \le \epsilon \deg(v)$ for all $v \in V$.
  - This implies $|\mathsf{d}^{\overline{f}}(S)| \le \frac{\epsilon}{\phi}\delta(S)$ for all $S \subseteq V$. So $\overline{f}$ $\frac{\epsilon}{\phi}$-satisfies d.
- This intuitively is similar to push-relabel algorithms.
  - We maintain levels and flow from higher vertices to lower vertices.

7 History and State of the art

- Nice book by David Williamson: http://www.networkflowalgs.com/
- 60-70's
  - Ford-Fulkerson
  - $O(mn^2)$ Blocking flow (Dinic)

- $O(m\min\{m^{1/2}, n^{2/3}\})$ time in unit capacity (Even-Tarjan, Karzanov)

- 80's
  - $O(mn\log n)$ Blocking flow + dynamic tree (Sleator Tarjan)
  - $O(mn\log n)$ Push relabel (Goldberg Tarjan)

- 90's
  - $\tilde{O}(m\min\{m^{1/2}, n^{2/3}\})$ in general graph (Goldbeg Rao)

- In sparse graphs, $n^{1.5}$ is the best since 70's... but there was a breakthrough in approximation algorithms in

- Approximation in edge-capacitated graphs
  - $\tilde{O}(mn^{1/3})$ Electrical flow + MWU + Arc Boosting (Christiano et al '10 [2])
  - $\tilde{O}(m)$ Congestion approximator + MWU (Sherman'13[3], KLOS'13[4], Peng'16[5])

- Approximation in vertex-capacitated graphs
  - $O(m^{1+o(1)})$ Dynamic shortest path + MWU (Bernstein Gutenberg **S**'21[6])

- Exact (based on Interior Point Method + Dynamic algorithms)
  - $\tilde{O}(m+n^{1.5})$ (BLLSSSW'21[7])
  - $O(m^{4/3+o(1)})$ unit-capacity [Kathuria'20] [Lui Sidford'20] https://www.youtube.com/watch?v=VF3EbC

- My belief:
  - Exact max flow in $\tilde{O}(m)$ time (in 5-10 years).
  - Maybe in your PhD thesis.

[2] https://arxiv.org/abs/1010.2921
[3] https://arxiv.org/pdf/1304.2338.pdf
[4] https://arxiv.org/pdf/1304.2077.pdf
[5] https://arxiv.org/abs/1411.7631
[6] https://arxiv.org/pdf/2101.07149.pdf
[7] https://arxiv.org/pdf/2101.05719.pdf