Algorithms for Network Flows

Lecture 4: Generalized flows II: a strongly polynomial algorithm

Neil Olver

Valparaíso Summer School, 2017

Slides will be available at: `http://nolver.net/home/valparaiso`

Today's lecture

- The first strongly polynomial algorithm was given only quite recently by Végh '14. Unfortunately it's very complicated!

- Here we discuss a much simpler (and faster) algorithm by O.-Végh '16.

Our algorithm	$O((m + n\log n)mn\log(n^2/m))$
Radzik '04	$O((m + n\log n)mn\log B)$
Végh '12	$O(m^2n^3)$

Primal and dual again

$$\begin{aligned} \max \quad & \nabla f_t \\ \text{s.t.} \quad & \nabla f_i \geq b_i \qquad \forall i \neq t \\ & f \geq 0 \end{aligned}$$

$$\begin{aligned} \max \quad & \sum_{j \in V \setminus \{t\}} b_j^{\mu} \\ \text{s.t.} \quad & \gamma_e^{\mu} \leq 1 \qquad \forall e \in E \\ & \mu_t = 1 \\ & \mu_i \in \mathbb{R}_{++} \quad \forall i \in V \end{aligned}$$

Primal and dual again

$$\begin{aligned} \max \quad & \nabla f_t \\ \text{s.t.} \quad & \nabla f_i \geq b_i \qquad \forall i \neq t \\ & f \geq 0 \end{aligned}$$

$$\begin{aligned} \max \quad & \mu_t \sum_{j \in V \setminus \{t\}} b_j^{\mu} \\ \text{s.t.} \quad & \gamma_e^{\mu} \leq 1 \qquad \forall e \in E \\ & \mu_i \in \mathbb{R}_{++} \quad \forall i \in V \end{aligned}$$

Reminder

Given $f \in \mathbb{R}_{+}^{E}$, $\mu \in \mathbb{R}_{++}^{V}$, (f, μ) is called a fitting pair if:

- μ is dual feasible
- $f_e > 0$ implies $\gamma_e^{\mu} = 1$.

If (f, μ) is a fitting pair and $\nabla f_i = b_i$ for all $i \neq t$, then f and μ are both optimal.

- Our algorithm will always maintain a fitting pair.

Goal: find a contractible edge

An edge $e \in E$ is contractible if $\gamma_e^{\mu^*} = 1$ for any dual optimum μ^*.

- Precisely as we saw for min cost flow, if e is contractible, can extend a dual optimum to $G/\{e\}$ to a dual optimum for G.
- So our goal is to produce a contractible arc in strongly polynomial time.

Goal: find a contractible edge

An edge $e \in E$ is contractible if $\gamma_e^{\mu^*} = 1$ for any dual optimum μ^*.

- Precisely as we saw for min cost flow, if e is contractible, can extend a dual optimum to $G/\{e\}$ to a dual optimum for G.
- So our goal is to produce a contractible arc in strongly polynomial time.

$$\mathrm{Ex}(f, \mu) := \sum_{i \neq t} \max\{\nabla f_i^\mu - b_i^\mu, 0\}.$$

Lemma

Suppose f is feasible, and (f, μ) is a fitting pair.
Then if $f^\mu(\hat{e}) > \mathrm{Ex}(f, \mu)$, $\hat{e}$ is contractible.

Plentiful nodes

- Our algorithm will maintain the invariant that $\nabla f_i^\mu < b_i^\mu + 2$, so

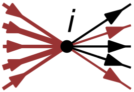
$$\mathrm{Ex}(f, \mu) < 2n.$$

Given a feasible dual μ, we say i is plentiful if

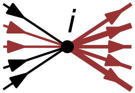
$$|b_i^\mu| \geq 2n^2.$$

Lemma

If (f, μ) is a fitting pair with f feasible, and i is a plentiful node, then some edge adjacent to i is contractible.



$b_i^\mu \geq 2n^2$



$b_i^\mu \leq -2n^2$

Pf: If $b_i^\mu \leq -2n^2$, $\partial f_i^\mu \leq -2n^2 + 2$

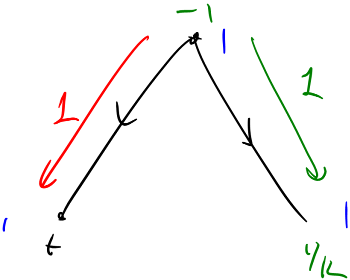
$\exists\, e \in \delta^+(i)$ with $f(e) \geq \frac{2n^2-2}{n-1} > 2n \ldots$

$> \mathrm{Ex}(f,\mu)$

A key new idea

- All previous algorithms maintain (roughly) a fitting pair (f, μ) with f feasible.

- We keep a fitting pair (f, μ)—but do not require f to be feasible!



A key new idea

- All previous algorithms maintain (roughly) a fitting pair (f, μ) with f feasible.
- We keep a fitting pair (f, μ)—but do not require f to be feasible!

Definition

μ is safe if there exists a feasible g s.t. (g, μ) is a fitting pair.

- Instead, we maintain a fitting pair (f, μ) where μ is safe, $\nabla f_i^\mu < b_i^\mu + 2$ for all $i \neq t$.

Lemma

Given such an (f, μ), if node i is plentiful then we can find a contractible arc adjacent to i.

Pf: μ safe $\Rightarrow$ $\exists$ g s.t. $\mathrm{supp}(g) \subseteq \mathrm{tight}(\mu)$

$\uparrow$

$\{e : y_e^{\mu} = 1\}$

$$\nabla g_i^{\mu} \geq b_i^{\mu}$$

Also $\mathrm{supp}(f) \subseteq \mathrm{tight}(\mu)$

$$\nabla f_i^{\mu} \leq b_i^{\mu} + 2$$

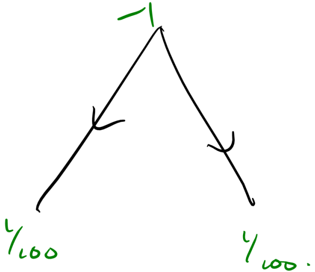
$\exists h$ s.t. $b_i^{\mu} \leq \nabla h_i^{\mu} \leq \nabla b_i^{\mu} + 2$. $\forall i \neq t$.

∎

Integrality vs feasibility

- A big benefit we gain from working with f infeasible is that we will maintain that f^μ is integral.

Feasibility and integrality are not compatible:



Invariants

Our algorithm maintains:

- (f, μ) a fitting pair, with f^μ integral

Invariants

Our algorithm maintains:

- (f, μ) a fitting pair, with f^μ integral
- $\nabla f_i^\mu < b_i^\mu + 2$ for all i (we don't require feasibility)
- $\nabla f_i^\mu > b_1^\mu - 1$ for all $i \in V^-$ $V^- := \{i : b_i < 0\}$

Invariants

Our algorithm maintains:

- (f, μ) a fitting pair, with f^μ integral
- $\nabla f_i^\mu < b_i^\mu + 2$ for all i (we don't require feasibility)
- $\nabla f_i^\mu > b_1^\mu - 1$ for all $i \in V^-$ $V^- := \{i : b_i < 0\}$
- μ is safe

Invariants

Our algorithm maintains:

- (f, μ) a fitting pair, with f^μ integral
- $\nabla f_i^\mu < b_i^\mu + 2$ for all i (we don't require feasibility)
- $\nabla f_i^\mu > b_1^\mu - 1$ for all $i \in V^-$ $V^- := \{i : b_i < 0\}$
- μ is safe

Goal

Adjusting f, μ satisfying above, produce j s.t. $|b_j^\mu| \geq 2n^2$.

Our algorithm

- **Initialization:** Find an initial fitting pair (f, μ), with f feasible.
  - Can do this with cycle cancelling, using strongly polynomial result of Radzik.

- Round f so that f^μ is integral, $-1 < \nabla f_i^\mu - b_i^\mu < 2$ for all $i \neq t$.

$\tilde{\mu} \leftarrow \lambda\mu$ so that $b_i^{\tilde{\mu}} \le \nabla f_i^{\tilde{\mu}} \le b_i^{\tilde{\mu}} + 1$.

Now let $\bar{f}^\mu$ be an integral (regular) flow satisfying $\mathrm{supp}(\bar{f}^\mu) \subseteq \mathrm{tight}(\mu)$

$$\lfloor b_i^{\tilde{\mu}} \rfloor \le \nabla f_i^{\tilde{\mu}} \le \lceil b_i^{\tilde{\mu}} + 1 \rceil$$

$f \leftarrow \bar{f}$, $\mu \leftarrow \tilde{\mu}$

Our algorithm

- **Initialization:** Find an initial fitting pair (f, μ), with f feasible.
  - Can do this with cycle cancelling, using strongly polynomial result of Radzik.

- Round f so that f^μ is integral, $-1 < \nabla f_i^\mu - b_i^\mu < 2$ for all $i \neq t$.

While $|b_j^\mu| < 2n^2$ for all j:

1. Augment f^μ

   (μ won't change)

2. Rescale μ

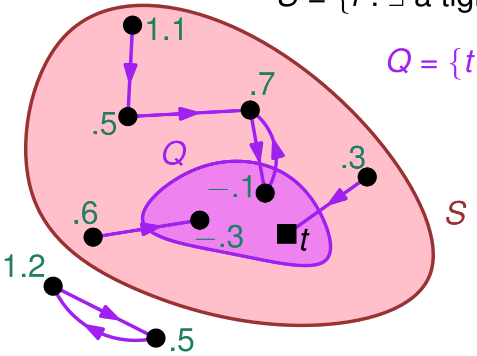
   (f^μ won't change)

Augmentation step

while $\exists i \in S \cap V^-$ with $\nabla f_i^\mu \geq b_i^\mu + 1$ **do**
Send 1 unit of relabelled flow from i to Q

$S = \{i : \exists \text{ a tight } i\text{-}Q\text{-path in } E^f\}$

$Q = \{t\} \cup \{i : \nabla f_i^\mu < b_i^\mu\}$



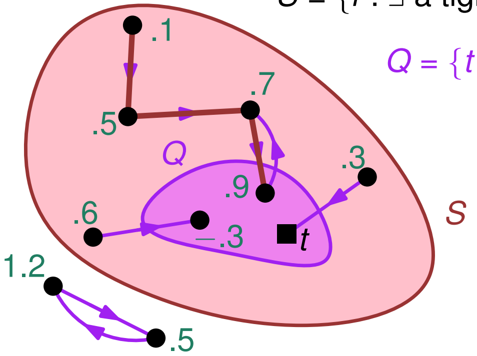
- Only augment on tight arcs, so (f, μ) stays a fitting pair.
- After augmenting, $\nabla f_i^\mu < b_i^\mu + 1$ for all $i \in S$.

Augmentation step

while $\exists i \in S \cap V^-$ with $\nabla f_i^\mu \geq b_i^\mu + 1$ **do**
    Send 1 unit of relabelled flow from i to Q

$S = \{i : \exists \text{ a tight } i\text{-}Q\text{-path in } E^f\}$

$Q = \{t\} \cup \{i : \nabla f_i^\mu < b_i^\mu\}$



- Only augment on tight arcs, so (f, μ) stays a fitting pair.
- After augmenting, $\nabla f_i^\mu < b_i^\mu + 1$ for all $i \in S$.

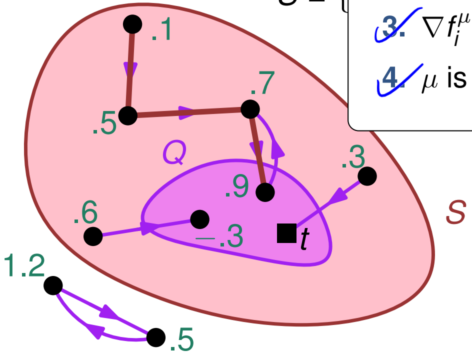
Augmentation step

while $\exists i \in S \cap V^-$ with $\nabla f_i^\mu \geq b$

Send 1 unit of relabelled flow

$S = \{$

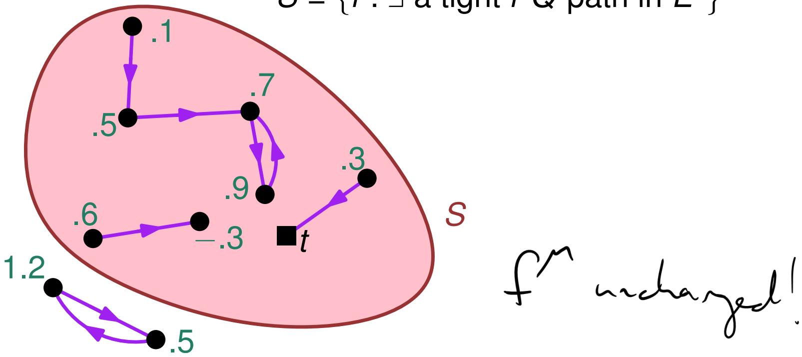
1. (f, μ) a fitting pair, with f^μ integral
2. $\nabla f_i^\mu < b_i^\mu + 2$ for all i
3. $\nabla f_i^\mu > b_1^\mu - 1$ for all $i \in V^-$
4. μ is safe



- Only augment on tight arcs, so (f, μ) stays a fitting pair.
- After augmenting, $\nabla f_i^\mu < b_i^\mu + 1$ for all $i \in S$.

Rescaling step

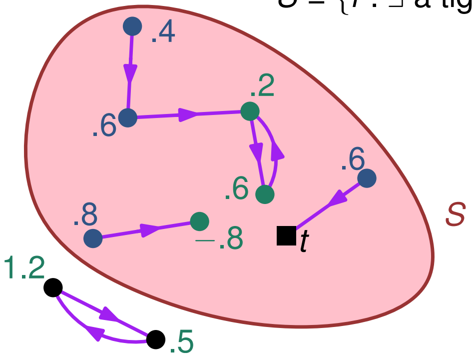
$S = \{i : \exists \text{ a tight } i\text{-}Q\text{-path in } E^f\}$



$$\mu_i' = \begin{cases} \mu_i/\alpha & \text{for } i \in S \\ \mu_i & \text{for } i \notin S \end{cases} \qquad f_e' = \begin{cases} f_e/\alpha & \text{for } e \in E(S) \\ f_e & \text{for } e \notin E(S) \end{cases}$$

Rescaling step

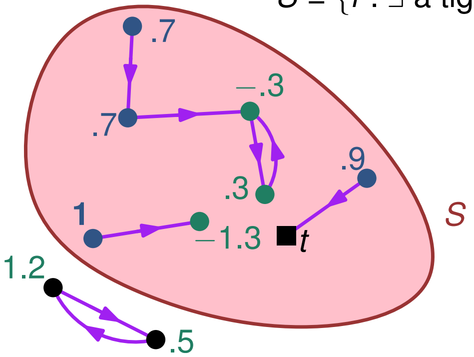
$S = \{i : \exists \text{ a tight } i\text{-}Q\text{-path in } E^f\}$



$$\mu'_i = \begin{cases} \mu_i/\alpha & \text{for } i \in S \\ \mu_i & \text{for } i \notin S \end{cases} \qquad f'_e = \begin{cases} f_e/\alpha & \text{for } e \in E(S) \\ f_e & \text{for } e \notin E(S) \end{cases}$$

Rescaling step

$S = \{i : \exists \text{ a tight } i\text{-}Q\text{-path in } E^f\}$

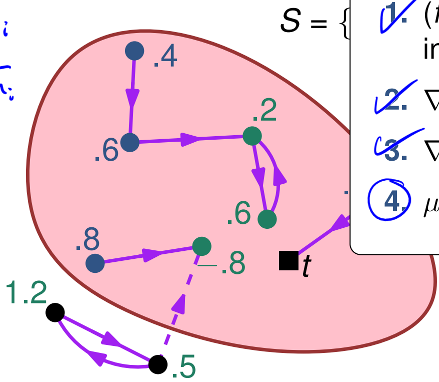


$$\mu'_i = \begin{cases} \mu_i/\alpha & \text{for } i \in S \\ \mu_i & \text{for } i \notin S \end{cases} \qquad f'_e = \begin{cases} f_e/\alpha & \text{for } e \in E(S) \\ f_e & \text{for } e \notin E(S) \end{cases}$$

α chosen maximally s.t. $\gamma_e^\mu \leq 1$ for all $e \in \delta^-(S)$, $\nabla f_i^\mu \leq b_i^\mu + 1$ for all $i \in S$.

Rescaling step

$b^\mu_i = b^i / \mu_i$



1. (f, μ) a fitting pair, with f^μ integral
2. $\nabla f^\mu_i < b^\mu_i + 2$ for all i
3. $\nabla f^\mu_i > b^\mu_i - 1$ for all $i \in V^-$
4. μ is safe

$$\mu'_i = \begin{cases} \mu_i/\alpha & \text{for } i \in S \\ \mu_i & \text{for } i \notin S \end{cases} \qquad f'_e = \begin{cases} f_e/\alpha & \text{for } e \in E(S) \\ f_e & \text{for } e \notin E(S) \end{cases}$$

α chosen maximally s.t. $\gamma^\mu_e \leq 1$ for all $e \in \delta^-(S)$, $\nabla f^\mu_i \leq b^\mu_i + 1$ for all $i \in S$.

Key technical lemma

Safety of μ is preserved in the rescaling step.

$$\mu'_i = \begin{cases} \mu_i/\alpha & i \in S \\ \mu_i & i \notin S \end{cases}$$

R = tight arcs w.r.t. μ

R' = tight arcs w.r.t. μ'

Suppose for a contradiction that μ is safe, but μ' is not.

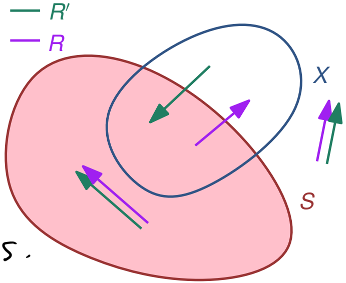
$\therefore$ $\nexists$ flow on R' satisfying demands $b_i^{\mu'}$.

$\therefore$ $\exists X$, $t \notin X$, with $\sum_{i \in X} b_i^{\mu'} > 0$,

$\delta^-_{R'}(X) = \emptyset$.

Claim: $b^\mu(X \setminus S)$

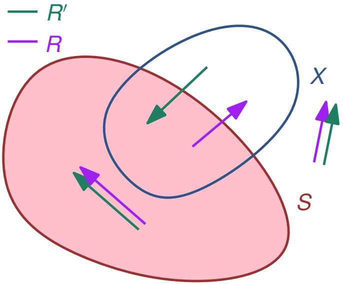
Pf: $\delta^-(S) \cap R = \emptyset$ by def^n of S.



$R = R'$ inside of S

$\therefore\ \delta^-(X \cap S) \cap R = \emptyset$.

$\therefore\ f^\mu(X \cap S) \leq 0$, since μ safe.



$\Rightarrow\ f^\mu(X \setminus S) = f^{\mu'}(X \setminus S) \geq 0$.

<u>Claim</u>: $f(\delta^-(X \setminus S)) = 0$

Pf: $R = R'$ outside of S, so nothing from $V \setminus (X \cup S)$ into $X \setminus S$.

No flow from S into $X \setminus S$.

Bounding the number of augmentations

- Keep track of potentials

$$\Phi := \sum_{i \in V^-} \nabla f_i^\mu - b_i^\mu, \qquad \Psi := -\sum_{i \in V^-} b_i^\mu$$

$\{i : b_i^\mu < 0\}$.

$-n \leq \Phi \leq 2n$.

Rescale: $\Delta \Phi = \Delta \Psi$.

Augmentations: Ψ unchanged,

Helpful augmentation: from V^- to $V \setminus V^-$.

A missing detail

An algorithm is strongly polynomial if:

1. Number of arithmetic operations is polynomial in the number of integers in the input (e.g., size of the graph)
2. Encoding lengths of numbers computed during execution are polynomial in input encoding length

- We need to show that the μ_i values stay under control.
- Dramatically easier for our algorithm than for the algorithm of Végh '14, but still not light entertainment...
- We can exploit the flexibility in some of the rescaling steps to ensure that some μ_i's stay "nice", and for any j, $\mu_j = \gamma(P)\mu_i$ for some "nice" μ_i and path P.

Extra ingredients for a faster running time

With some extra work, the running time of a souped up version of this algorithm is $O((m + n \log n)mn \log(n^2/m))$.

- Strongly polynomial cycle cancelling algorithm of Radzik is too slow. Replace with an execution of our algorithm on an auxilliary instance.
- Algorithm needs to be implemented efficiently so that not too much time is spent updating labels.
- Don't start from scratch after a contraction.
- A refined potential analysis is needed.

Open question

Strongly polynomial algorithm for minimum cost generalized flow?

Open question

Strongly polynomial algorithm for minimum cost generalized flow?

Thank you!

References

N. Olver and L. Végh.
A simpler and faster strongly polynomial algorithm for generalized flow maximization.
arXiv:1611.01778, 2016.

M. Shigeno.
A survey of combinatorial maximum flow algorithms on a network with gains.
Journal of the Operations Research Society of Japan, 47(4):244–264, 2004.

L. Végh.
A strongly polynomial algorithm for generalized flow maximization.
Mathematics of Operations Research.