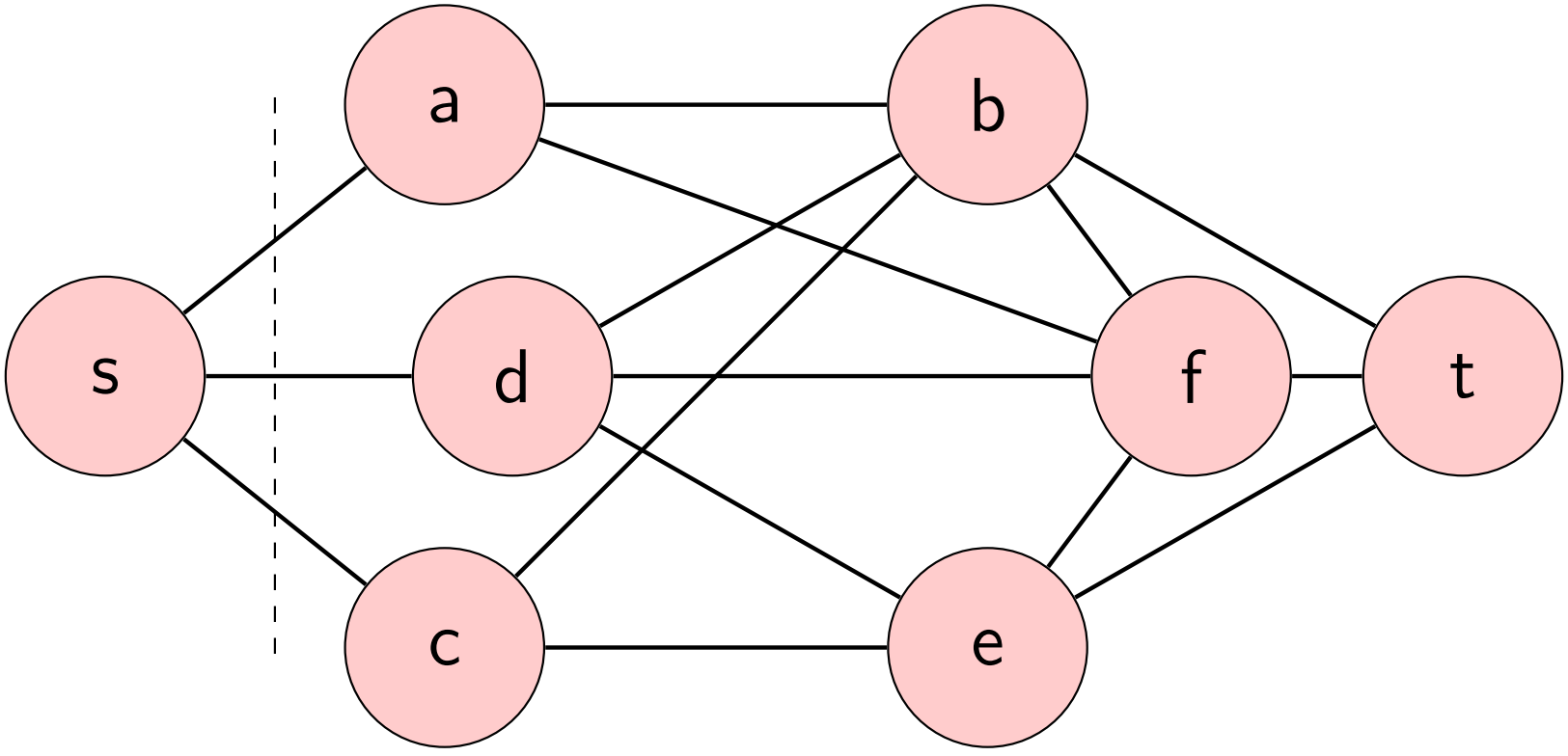
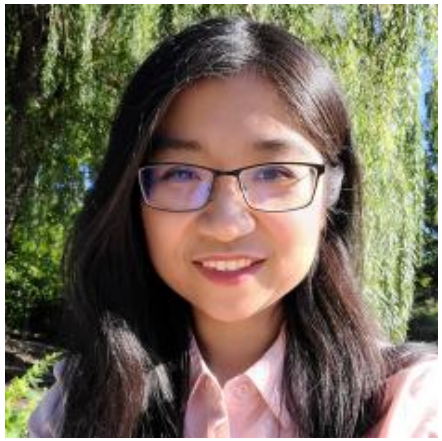


Lipschitz Graph Algorithms via Proximal Gradient Analysis

Q.C. Liu, G. Velegkas, Y. Yoshida, F. Zhou

Yale University, National Institute of Informatics

Collaborators



Quanquan C. Liu
(Yale)

Grigoris Velegkas
(Yale)

Yuichi Yoshida
(NII)

Table of Contents

Table of Contents

Motivation

Graph algorithms are widely used for decision making.
Their outputs should remain stable under small perturbations.

Motivation

Graph algorithms are widely used for decision making.
Their outputs should remain stable under small perturbations.

- "Menu" costs

Motivation

Graph algorithms are widely used for decision making.
Their outputs should remain stable under small perturbations.

- "Menu" costs
- User trust

Motivation

Graph algorithms are widely used for decision making.
Their outputs should remain stable under small perturbations.

- "Menu" costs
- User trust
- Safe decision making

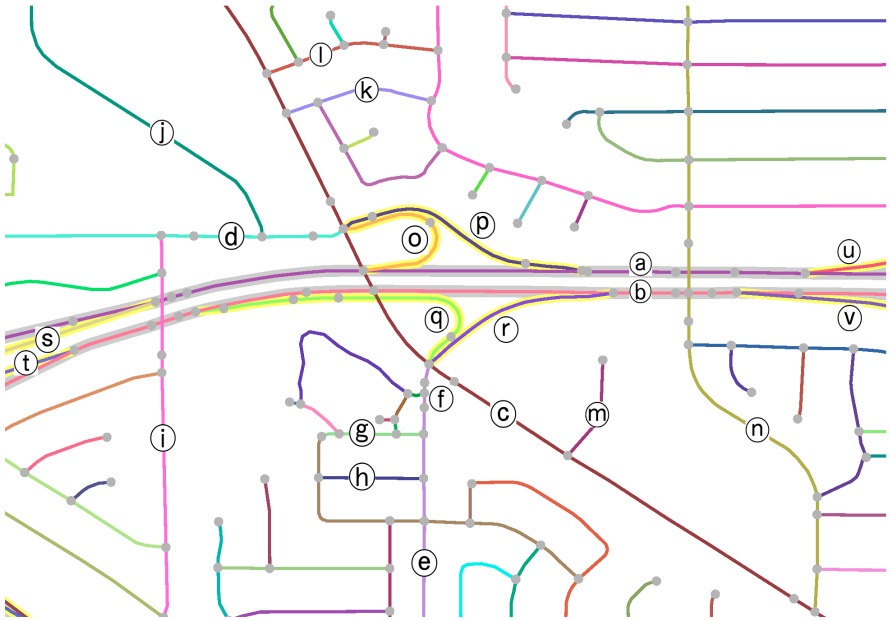
Motivation

Graph algorithms are widely used for decision making.
Their outputs should remain stable under small perturbations.

- "Menu" costs
- User trust
- Safe decision making
- Replicability

Example

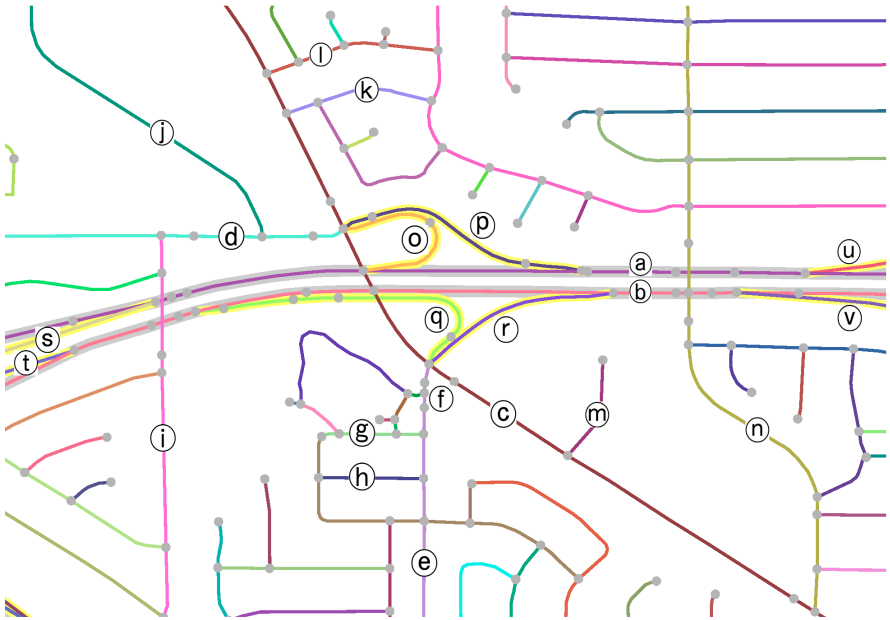
Bottlenecks in a traffic network



Example

Bottlenecks in a traffic network

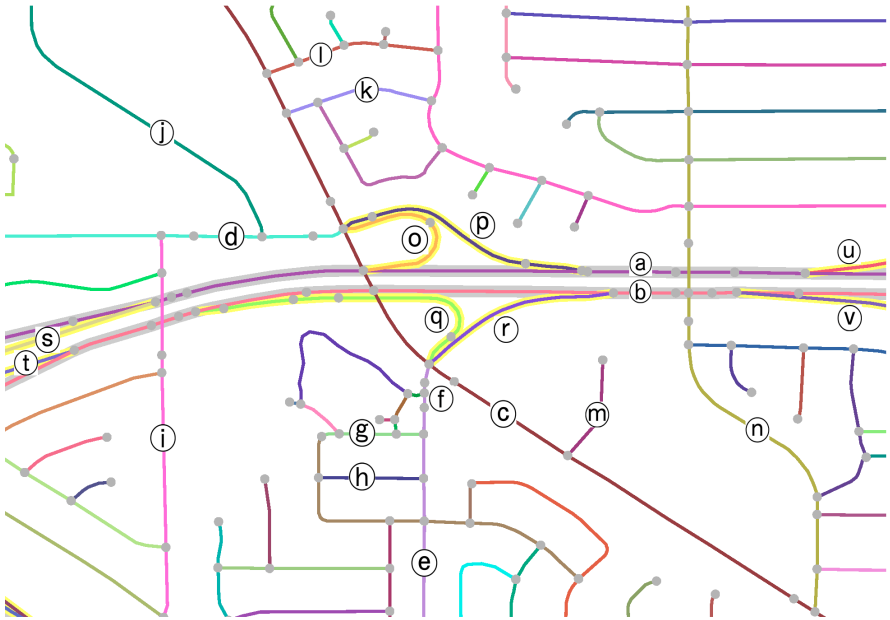
- Intersections as nodes



Example

Bottlenecks in a traffic network

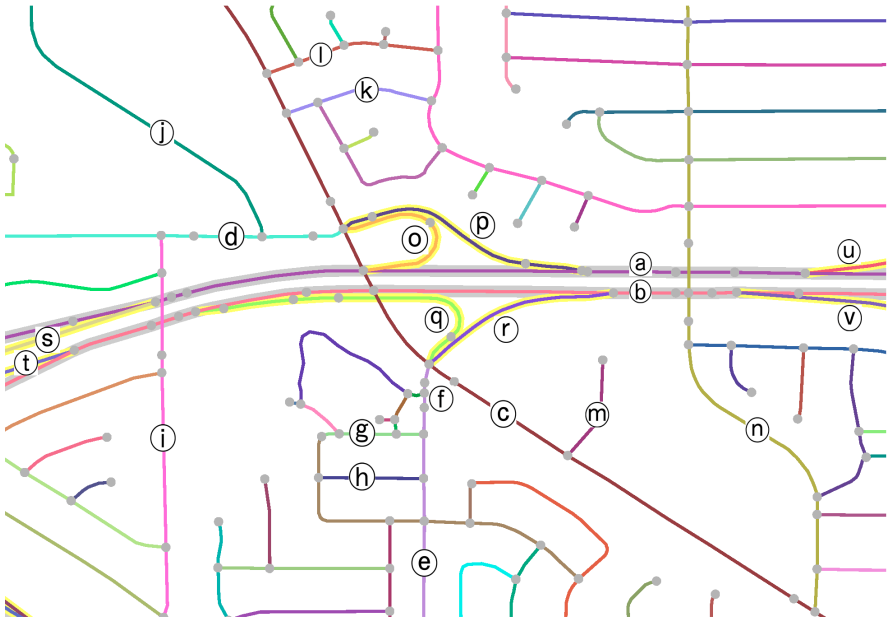
- Intersections as nodes
- Streets as edges



Example

Bottlenecks in a traffic network

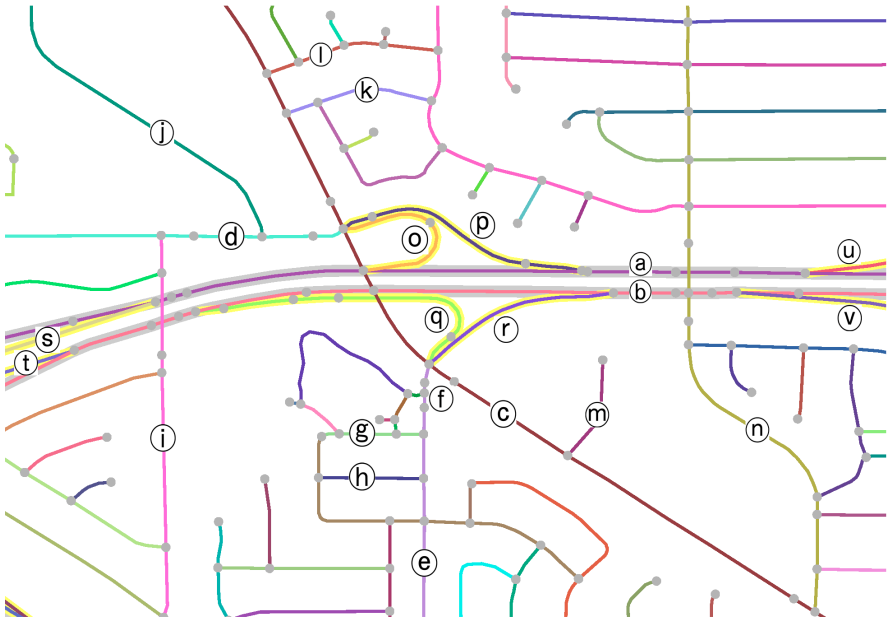
- Intersections as nodes
- Streets as edges
- Maximum amount of traffic as edge capacity



Example

Bottlenecks in a traffic network

- Intersections as nodes
- Streets as edges
- Maximum amount of traffic as edge capacity
- Bottlenecks as min s-t cuts



Definition of Lipschitz Continuity

- $G = (V, E)$ graph

Definition of Lipschitz Continuity

- $G = (V, E)$ graph
- $w : E \to \mathbb{R}_+$ edge weights

Definition of Lipschitz Continuity

- $G = (V, E)$ graph
- $w : E \to \mathbb{R}_+$ edge weights
- $d(A, B) = |A \Delta B| := |(A \cup B) \setminus (A \cap B)| = \|\mathbb{1}_A - \mathbb{1}_B\|_1$

Definition of Lipschitz Continuity

- $G = (V, E)$ graph
- $w : E \to \mathbb{R}_+$ edge weights
- $d(A, B) = |A \Delta B| := |(A \cup B) \setminus (A \cap B)| = \|\mathbb{1}_A - \mathbb{1}_B\|_1$
- $(\pi, \tilde{\pi}) \sim \mathcal{D}$ coupling of random variables $\pi, \tilde{\pi}$

Definition of Lipschitz Continuity

- $G = (V, E)$ graph
- $w : E \to \mathbb{R}_+$ edge weights
- $d(A, B) = |A \Delta B| := |(A \cup B) \setminus (A \cap B)| = \|\mathbb{1}_A - \mathbb{1}_B\|_1$
- $(\pi, \tilde{\pi}) \sim \mathcal{D}$ coupling of random variables $\pi, \tilde{\pi}$

Definition (Pointwise Lipschitz; [KY23a])

The pointwise Lipschitz constant of a randomized algorithm $\mathcal{A}_\pi(G, w)$ is

Definition of Lipschitz Continuity

- $G = (V, E)$ graph
- $w : E \to \mathbb{R}_+$ edge weights
- $d(A, B) = |A \Delta B| := |(A \cup B) \setminus (A \cap B)| = \|\mathbb{1}_A - \mathbb{1}_B\|_1$
- $(\pi, \tilde{\pi}) \sim \mathcal{D}$ coupling of random variables $\pi, \tilde{\pi}$

Definition (Pointwise Lipschitz; [KY23a])

The pointwise Lipschitz constant of a randomized algorithm $\mathcal{A}_\pi(G, w)$ is

$$\limsup_{\tilde{w} \to w} \frac{\mathrm{EMD}_d(\mathcal{A}_{\tilde{\pi}}(G, \tilde{w}), \mathcal{A}_\pi(G, w))}{\|\tilde{w} - w\|_1}$$

Definition of Lipschitz Continuity

- $G = (V, E)$ graph
- $w : E \to \mathbb{R}_+$ edge weights
- $d(A, B) = |A \Delta B| := |(A \cup B) \setminus (A \cap B)| = \|\mathbb{1}_A - \mathbb{1}_B\|_1$
- $(\pi, \tilde{\pi}) \sim \mathcal{D}$ coupling of random variables $\pi, \tilde{\pi}$

Definition (Pointwise Lipschitz; [KY23a])

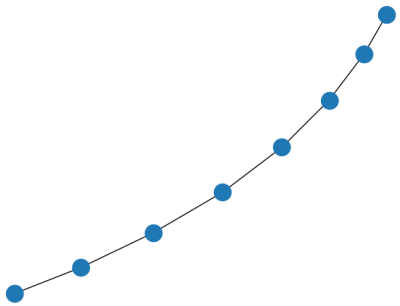
The pointwise Lipschitz constant of a randomized algorithm $\mathcal{A}_\pi(G, w)$ is

$$\limsup_{\tilde{w} \to w} \frac{\mathrm{EMD}_d(\mathcal{A}_{\tilde{\pi}}(G, \tilde{w}), \mathcal{A}_\pi(G, w))}{\|\tilde{w} - w\|_1}$$

$$\limsup_{\tilde{w} \to w} \frac{\inf_{\mathcal{D}} \mathbb{E}_{(\pi, \tilde{\pi}) \sim \mathcal{D}} \left[d(\mathcal{A}_{\tilde{\pi}}(G, \tilde{w}), \mathcal{A}_\pi(G, w)) \right]}{\|\tilde{w} - w\|_1}$$

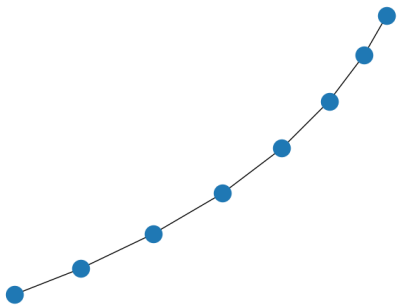
Remarks

- Limit taken over weights but not graphs



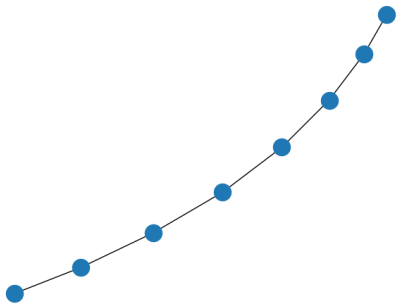
Remarks

- Limit taken over weights but not graphs
  - Streets/intersections stay constant



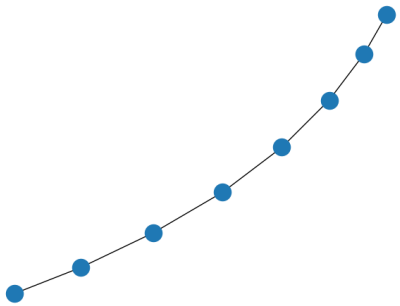
Remarks

- Limit taken over weights but not graphs
  - Streets/intersections stay constant
  - Capacity depends on weather, traffic, etc



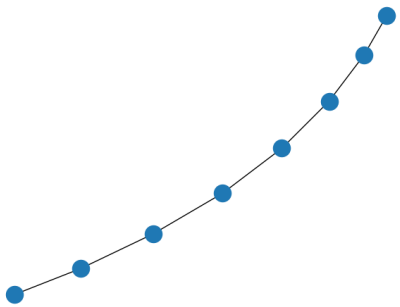
Remarks

- Limit taken over weights but not graphs
  - Streets/intersections stay constant
  - Capacity depends on weather, traffic, etc
- Randomness is necessary!



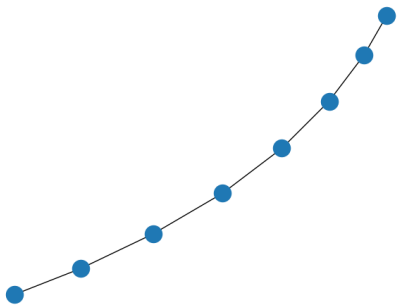
Remarks

- Limit taken over weights but not graphs
  - Streets/intersections stay constant
  - Capacity depends on weather, traffic, etc
- Randomness is necessary!
  - Deterministic algorithm experiences "phase transition"



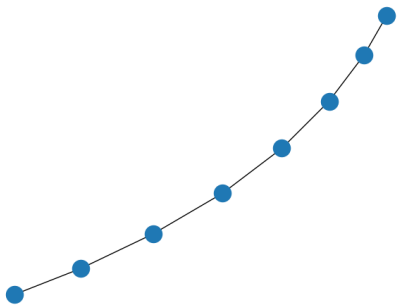
Remarks

- Limit taken over weights but not graphs
  - Streets/intersections stay constant
  - Capacity depends on weather, traffic, etc
- Randomness is necessary!
  - Deterministic algorithm experiences "phase transition"
- Everything works under shared randomness



Remarks

- Limit taken over weights but not graphs
  - Streets/intersections stay constant
  - Capacity depends on weather, traffic, etc
- Randomness is necessary!
  - Deterministic algorithm experiences "phase transition"
- Everything works under shared randomness
  - Can observe very different outputs when run 2x independently



Remarks

- Limit taken over weights but not graphs
  - Streets/intersections stay constant
  - Capacity depends on weather, traffic, etc
- Randomness is necessary!
  - Deterministic algorithm experiences "phase transition"
- Everything works under shared randomness
  - Can observe very different outputs when run 2x independently
  - Require coupling be obtained via shared randomness

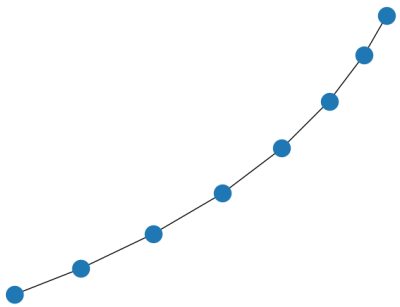


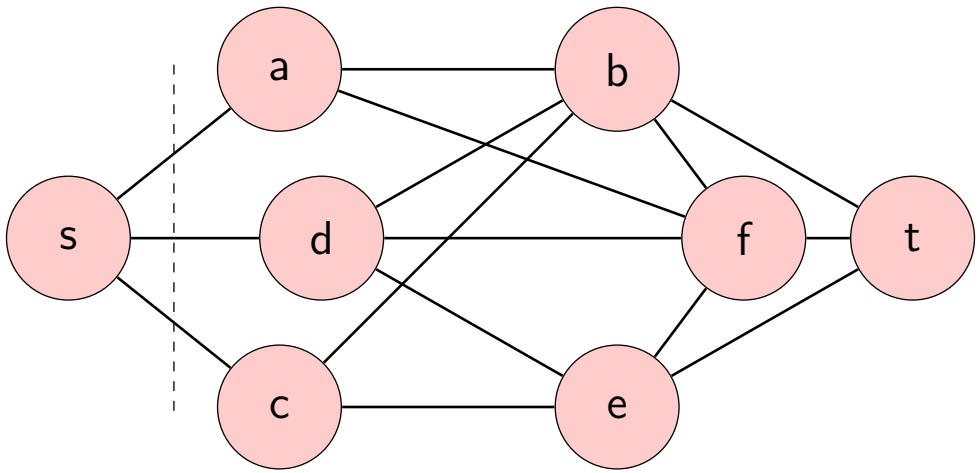
Table of Contents

(Undirected) Minimum s-t Cut

Given:

- Graph $G = (V, E)$

Want:

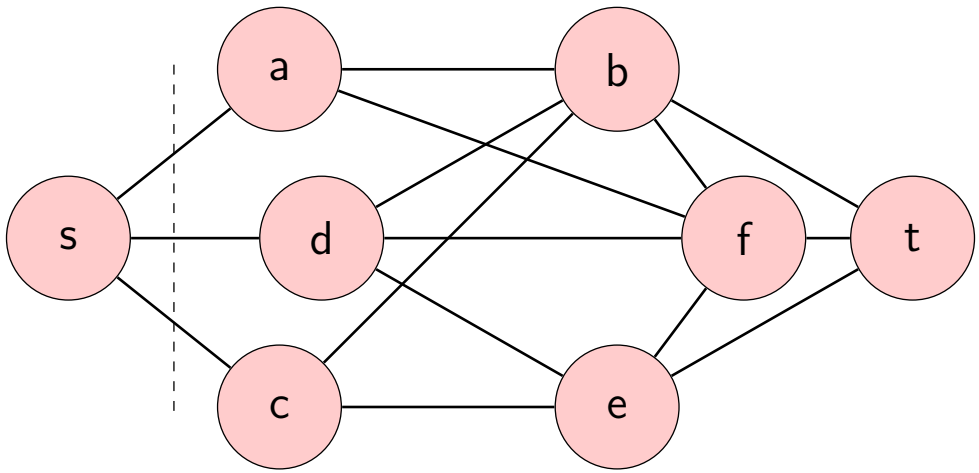


(Undirected) Minimum s-t Cut

Given:

- Graph $G = (V, E)$
- Edge weights $w : E \to \mathbb{R}_+$

Want:

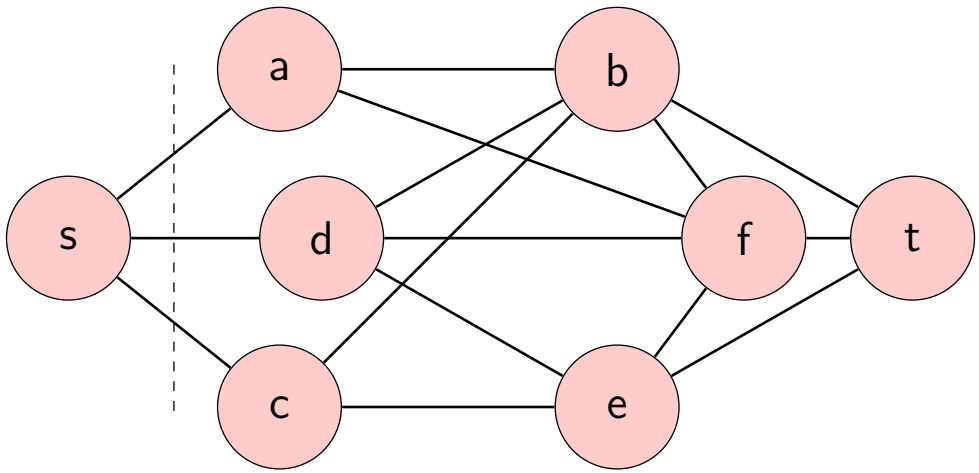


(Undirected) Minimum s-t Cut

Given:

- Graph $G = (V, E)$
- Edge weights $w : E \to \mathbb{R}_+$
- terminals $s, t \in V$

Want:



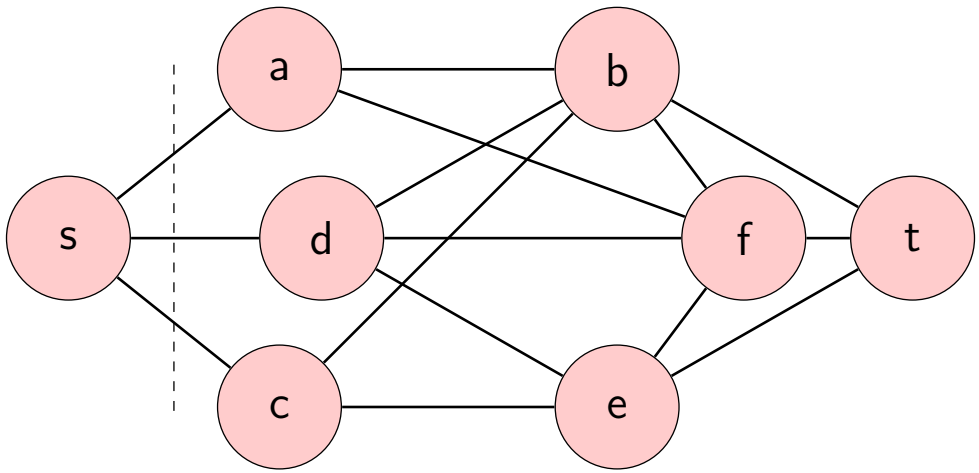
(Undirected) Minimum s-t Cut

Given:

- Graph $G = (V, E)$
- Edge weights $w : E \to \mathbb{R}_+$
- terminals $s, t \in V$

Want:

- Vertex subset $S \subseteq V$



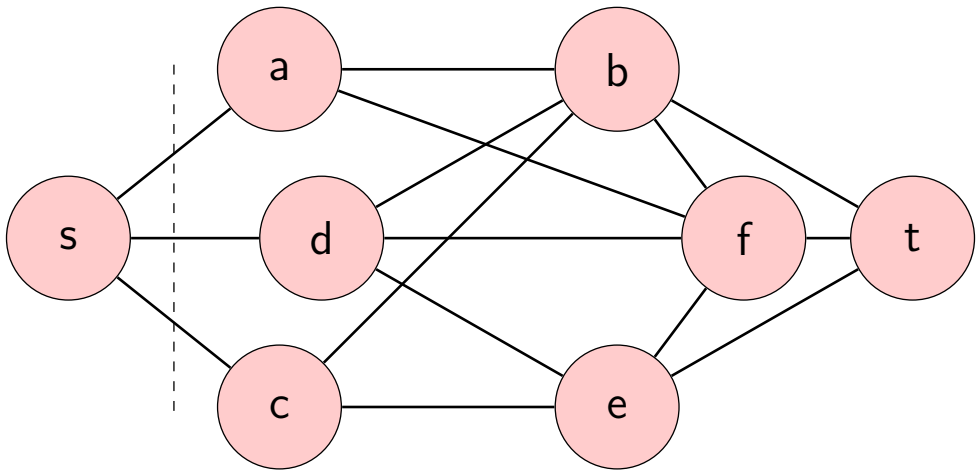
(Undirected) Minimum s-t Cut

Given:

- Graph $G = (V, E)$
- Edge weights $w : E \to \mathbb{R}_+$
- terminals $s, t \in V$

Want:

- Vertex subset $S \subseteq V$
- $s \in S, t \notin S$,



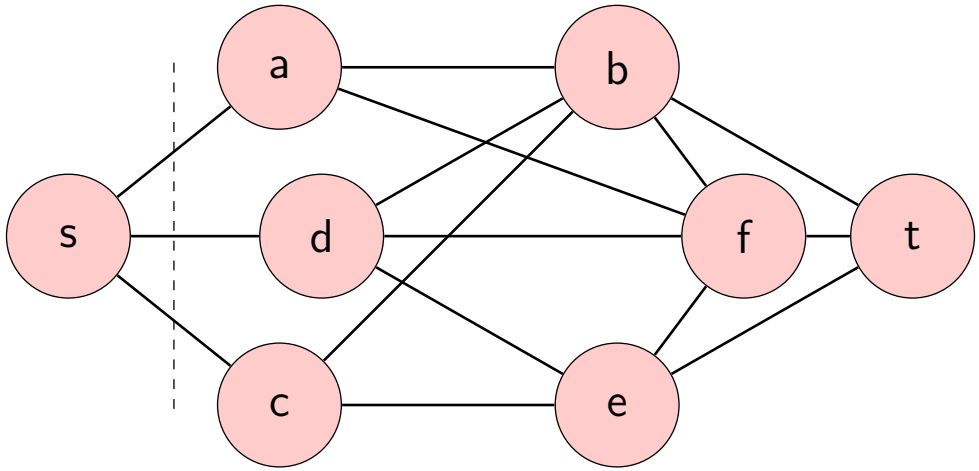
(Undirected) Minimum s-t Cut

Given:

- Graph $G = (V, E)$
- Edge weights $w : E \to \mathbb{R}_+$
- terminals $s, t \in V$

Want:

- Vertex subset $S \subseteq V$
- $s \in S, t \notin S$,
- minimizing $w(\partial S) := \sum_{uv \in E : u \in S, v \notin S} w(uv)$



Pointwise Lipschitz Constant of Output

And: small pointwise Lipschitz constant

$$\limsup_{\tilde{w} \to w} \frac{\inf_{\mathcal{D}} \mathbb{E}_{(\pi,\tilde{\pi}) \sim \mathcal{D}} \left[\left| \tilde{S}_{\tilde{\pi}} \Delta S_{\pi} \right| \right]}{\left\| \tilde{w} - w \right\|_1}$$

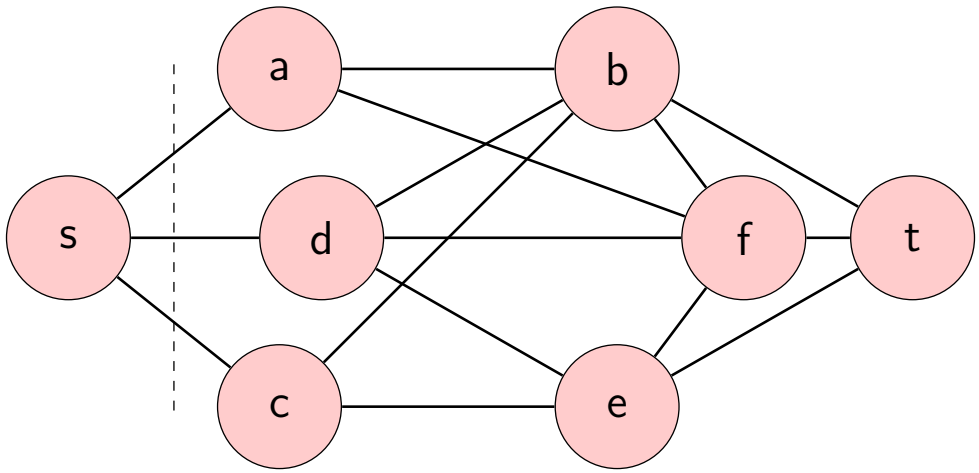


Table of Contents

Recall

- Graph Laplacian $\mathcal{L}(G, w) = D(G, w) - A(G, w) \in \mathbb{R}^{V \times V}$

Recall

- Graph Laplacian $\mathcal{L}(G, w) = D(G, w) - A(G, w) \in \mathbb{R}^{V \times V}$
  - D weighted degree matrix (diagonal)

Recall

- Graph Laplacian $\mathcal{L}(G, w) = D(G, w) - A(G, w) \in \mathbb{R}^{V \times V}$
  - D weighted degree matrix (diagonal)
  - A weighted adjacency matrix

Recall

- Graph Laplacian $\mathcal{L}(G, w) = D(G, w) - A(G, w) \in \mathbb{R}^{V \times V}$
  - D weighted degree matrix (diagonal)
  - A weighted adjacency matrix
- Eigenvalues $0 = \lambda_1(G, w) \leq \lambda_2(G, w) \leq \cdots \leq \lambda_n(G, w)$

Recall

- Graph Laplacian $\mathcal{L}(G, w) = D(G, w) - A(G, w) \in \mathbb{R}^{V \times V}$
  - D weighted degree matrix (diagonal)
  - A weighted adjacency matrix
- Eigenvalues $0 = \lambda_1(G, w) \leq \lambda_2(G, w) \leq \cdots \leq \lambda_n(G, w)$
  - $\lambda_1 = 0$ with eigenvector $\mathbb{1}_n$

Recall

- Graph Laplacian $\mathcal{L}(G, w) = D(G, w) - A(G, w) \in \mathbb{R}^{V \times V}$
  - D weighted degree matrix (diagonal)
  - A weighted adjacency matrix
- Eigenvalues $0 = \lambda_1(G, w) \leq \lambda_2(G, w) \leq \cdots \leq \lambda_n(G, w)$
  - $\lambda_1 = 0$ with eigenvector $\mathbb{1}_n$
  - λ_2 is algebraic connectivity of weighted graph G, w

Recall

- Graph Laplacian $\mathcal{L}(G, w) = D(G, w) - A(G, w) \in \mathbb{R}^{V \times V}$
  - D weighted degree matrix (diagonal)
  - A weighted adjacency matrix
- Eigenvalues $0 = \lambda_1(G, w) \leq \lambda_2(G, w) \leq \cdots \leq \lambda_n(G, w)$
  - $\lambda_1 = 0$ with eigenvector $\mathbb{1}_n$
  - λ_2 is algebraic connectivity of weighted graph G, w
  - $\lambda_2 = 0 \iff G, w$ is disconnected

Recall

- Graph Laplacian $\mathcal{L}(G, w) = D(G, w) - A(G, w) \in \mathbb{R}^{V \times V}$
  - D weighted degree matrix (diagonal)
  - A weighted adjacency matrix
- Eigenvalues $0 = \lambda_1(G, w) \leq \lambda_2(G, w) \leq \cdots \leq \lambda_n(G, w)$
  - $\lambda_1 = 0$ with eigenvector $\mathbb{1}_n$
  - λ_2 is algebraic connectivity of weighted graph G, w
  - $\lambda_2 = 0 \iff G, w$ is disconnected
  - $\lambda_2 = d - O(\sqrt{d})$ for a random d-regular graph

Recall

- Graph Laplacian $\mathcal{L}(G, w) = D(G, w) - A(G, w) \in \mathbb{R}^{V \times V}$
  - D weighted degree matrix (diagonal)
  - A weighted adjacency matrix
- Eigenvalues $0 = \lambda_1(G, w) \leq \lambda_2(G, w) \leq \cdots \leq \lambda_n(G, w)$
  - $\lambda_1 = 0$ with eigenvector $\mathbb{1}_n$
  - λ_2 is algebraic connectivity of weighted graph G, w
  - $\lambda_2 = 0 \iff G, w$ is disconnected
  - $\lambda_2 = d - O(\sqrt{d})$ for a random d-regular graph
  - $\lambda_2 = n - 1$ for a complete graph on n vertices

Our Results

THEOREM (LVYZ '24)

There is a polynomial time algorithm such that with probability 0.99, outputs an s-t cut with

Our Results

THEOREM (LVYZ '24)

There is a polynomial time algorithm such that with probability 0.99, outputs an s-t cut with

- $(1 + O(1/\sqrt{n}))$-multiplicative error

Our Results

THEOREM (LVYZ '24)

There is a polynomial time algorithm such that with probability 0.99, outputs an s-t cut with

- $(1 + O(1/\sqrt{n}))$-multiplicative error
- $O(\lambda_2)$-additive error

Our Results

THEOREM (LVYZ '24)

There is a polynomial time algorithm such that with probability 0.99, outputs an s-t cut with

- $(1 + O(1/\sqrt{n}))$-multiplicative error
- $O(\lambda_2)$-additive error
- pointwise Lipschitz constant $O(n/\lambda_2)$

A Lower Bound

THEOREM (LVYZ '24)

Any algorithm that outputs with probability 0.99 an s-t cut with

A Lower Bound

THEOREM (LVYZ '24)

Any algorithm that outputs with probability 0.99 an s-t cut with

- $O(1)$-multiplicative error

A Lower Bound

THEOREM (LVYZ '24)

Any algorithm that outputs with probability 0.99 an s-t cut with

- $O(1)$-multiplicative error
- and λ_2-additive error

A Lower Bound

THEOREM (LVYZ '24)

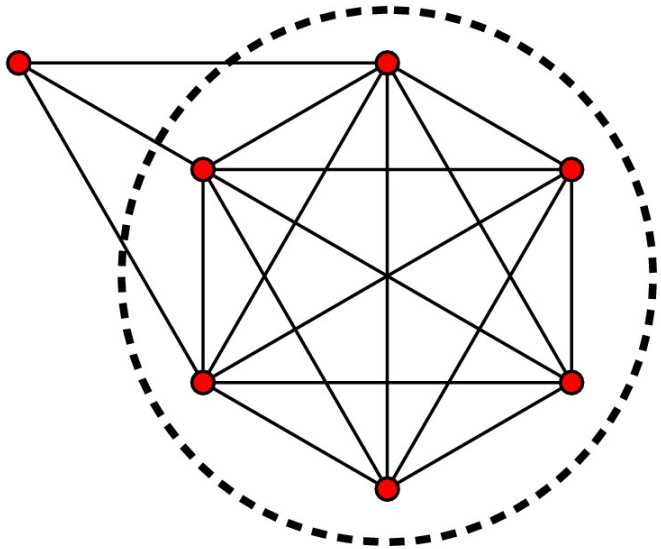
Any algorithm that outputs with probability 0.99 an s-t cut with

- $O(1)$-multiplicative error
- and λ_2-additive error

must have pointwise Lipschitz constant $\Omega(n/\lambda_2)$

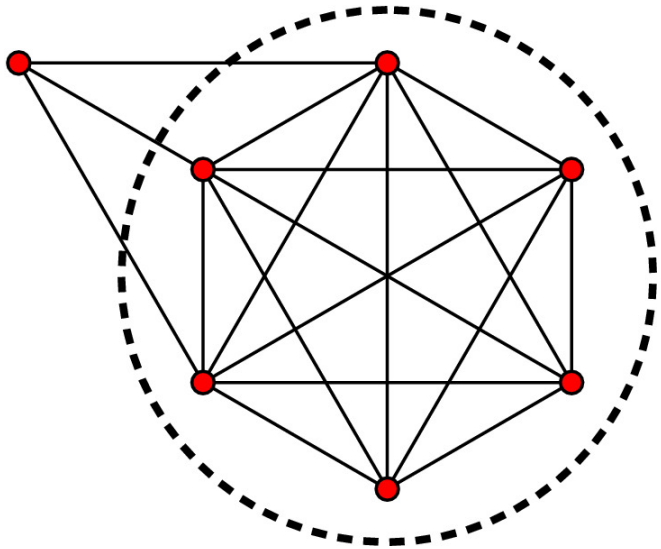
Remarks

- Tight upper/lower bounds!



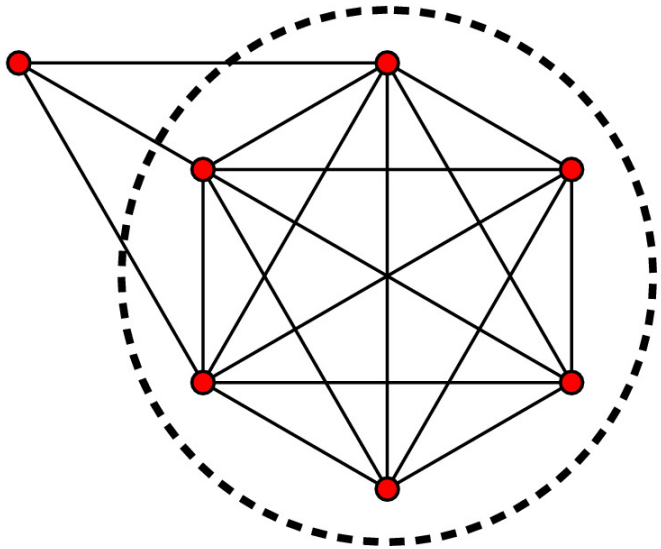
Remarks

- Tight upper/lower bounds!
- Algorithmic techniques very general and extends



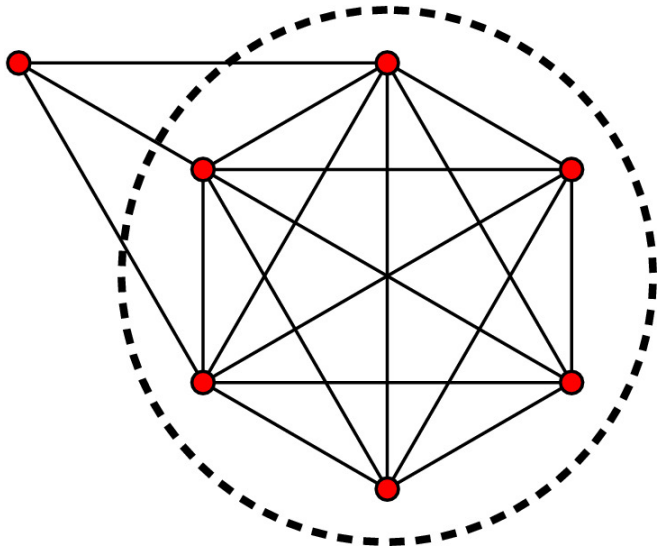
Remarks

- Tight upper/lower bounds!
- Algorithmic techniques very general and extends
  - multiway cut



Remarks

- Tight upper/lower bounds!
- Algorithmic techniques very general and extends
  - multiway cut
  - densest subgraph



Remarks

- Tight upper/lower bounds!
- Algorithmic techniques very general and extends
  - multiway cut
  - densest subgraph
  - matchings, b-matchings

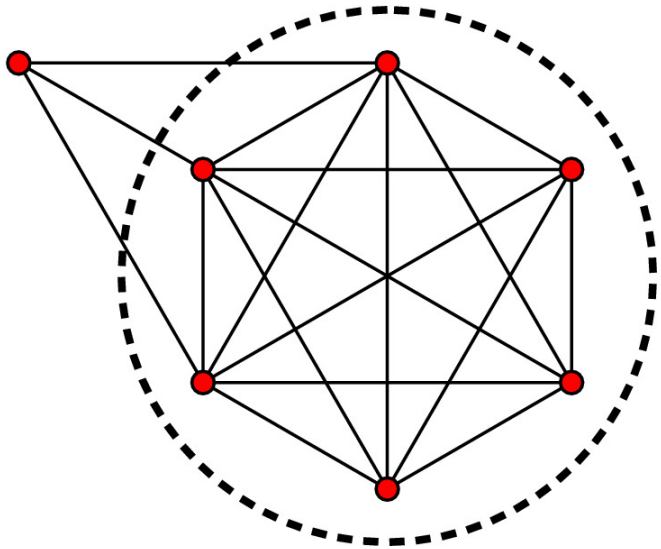


Table of Contents

Lipschitz Continuous Graph Algorithms

Definition introduced by Kumabe & Yoshida [KY23a, FOCS].
Considered weighted metric on output space

Briefly studied unweighted metric (our focus)

Lipschitz Continuous Graph Algorithms

Definition introduced by Kumabe & Yoshida [KY23a, FOCS].
Considered weighted metric on output space

- Minimum spanning tree

Briefly studied unweighted metric (our focus)

Lipschitz Continuous Graph Algorithms

Definition introduced by Kumabe & Yoshida [KY23a, FOCS].
Considered weighted metric on output space

- Minimum spanning tree
- Shortest path

Briefly studied unweighted metric (our focus)

Lipschitz Continuous Graph Algorithms

Definition introduced by Kumabe & Yoshida [KY23a, FOCS].
Considered weighted metric on output space

- Minimum spanning tree
- Shortest path
- Maximum weight matching

Briefly studied unweighted metric (our focus)

Lipschitz Continuous Graph Algorithms

Definition introduced by Kumabe & Yoshida [KY23a, FOCS].
Considered weighted metric on output space

- Minimum spanning tree
- Shortest path
- Maximum weight matching

Briefly studied unweighted metric (our focus)

- Minimum spanning tree

Lipschitz Continuous Graph Algorithms

Definition introduced by Kumabe & Yoshida [KY23a, FOCS].
Considered weighted metric on output space

- Minimum spanning tree
- Shortest path
- Maximum weight matching

Briefly studied unweighted metric (our focus)

- Minimum spanning tree
  - $(1+\varepsilon)$-approximation

Lipschitz Continuous Graph Algorithms

Definition introduced by Kumabe & Yoshida [KY23a, FOCS].
Considered weighted metric on output space

- Minimum spanning tree
- Shortest path
- Maximum weight matching

Briefly studied unweighted metric (our focus)

- Minimum spanning tree
  - $(1+\varepsilon)$-approximation
  - $O(n/\varepsilon \operatorname{OPT})$ pointwise Lipschitz constant

Lipschitz Continuous Graph Algorithms

Definition introduced by Kumabe & Yoshida [KY23a, FOCS].
Considered weighted metric on output space

- Minimum spanning tree
- Shortest path
- Maximum weight matching

Briefly studied unweighted metric (our focus)

- Minimum spanning tree
  - $(1+\varepsilon)$-approximation
  - $O(n/\varepsilon \operatorname{OPT})$ pointwise Lipschitz constant
- Maximum weight bipartite matching

Lipschitz Continuous Graph Algorithms

Definition introduced by Kumabe & Yoshida [KY23a, FOCS].
Considered weighted metric on output space

- Minimum spanning tree
- Shortest path
- Maximum weight matching

Briefly studied unweighted metric (our focus)

- Minimum spanning tree
  - $(1+\varepsilon)$-approximation
  - $O(n/\varepsilon \operatorname{OPT})$ pointwise Lipschitz constant
- Maximum weight bipartite matching
  - $(1/2-\varepsilon)$-approximation

Lipschitz Continuous Graph Algorithms

Definition introduced by Kumabe & Yoshida [KY23a, FOCS].
Considered weighted metric on output space

- Minimum spanning tree
- Shortest path
- Maximum weight matching

Briefly studied unweighted metric (our focus)

- Minimum spanning tree
  - $(1+\varepsilon)$-approximation
  - $O(n/\varepsilon \operatorname{OPT})$ pointwise Lipschitz constant
- Maximum weight bipartite matching
  - $(1/2-\varepsilon)$-approximation
  - $O(n^{3/2} \log m/\varepsilon \operatorname{OPT})$ pointwise Lipschitz constant

Other Notions of Stability in Graph Algorithms

- Worst-case sensitivity [VY21, SODA]

Other Notions of Stability in Graph Algorithms

- Worst-case sensitivity [VY21, SODA]
- Replicability [ILPS22, STOC]

Other Notions of Stability in Graph Algorithms

- Worst-case sensitivity [VY21, SODA]
- Replicability [ILPS22, STOC]
- Differential privacy [DMNS06, TCC]

Other Notions of Stability in Graph Algorithms

- Worst-case sensitivity [VY21, SODA]
- Replicability [ILPS22, STOC]
- Differential privacy [DMNS06, TCC]
- Low-recourse online algorithms [DEI06, APPROX]

Other Notions of Stability in Graph Algorithms

- Worst-case sensitivity [VY21, SODA]
- Replicability [ILPS22, STOC]
- Differential privacy [DMNS06, TCC]
- Low-recourse online algorithms [DEI06, APPROX]
- Perturbation resilience [BL09, ECCC]

Table of Contents

High-Level Framework

1. Solve regularized LP relaxation using any algorithm.

High-Level Framework

1. Solve regularized LP relaxation using any algorithm.
   - Deterministic fractional solutions

High-Level Framework

1. Solve regularized LP relaxation using any algorithm.
   - Deterministic fractional solutions
   - Analyze stability using iterates of proximal gradient method

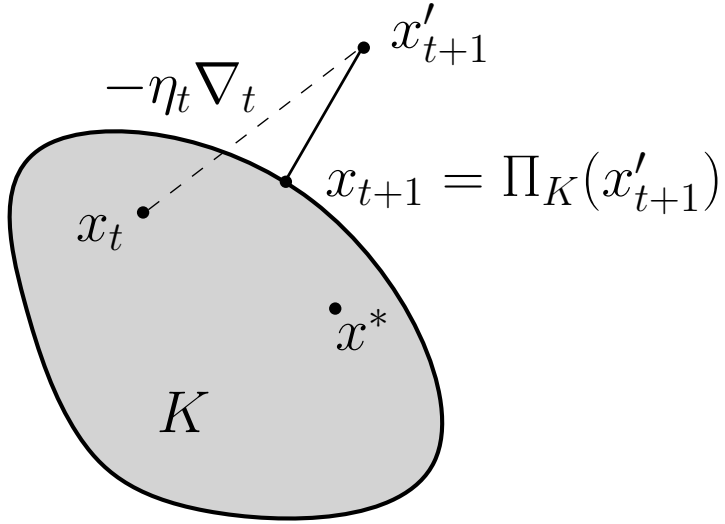
High-Level Framework

1. Solve regularized LP relaxation using any algorithm.
   - Deterministic fractional solutions
   - Analyze stability using iterates of proximal gradient method
2. Round fractional solutions using problem-dependent scheme

Proximal Gradient Analysis

Recall proximal gradient method:

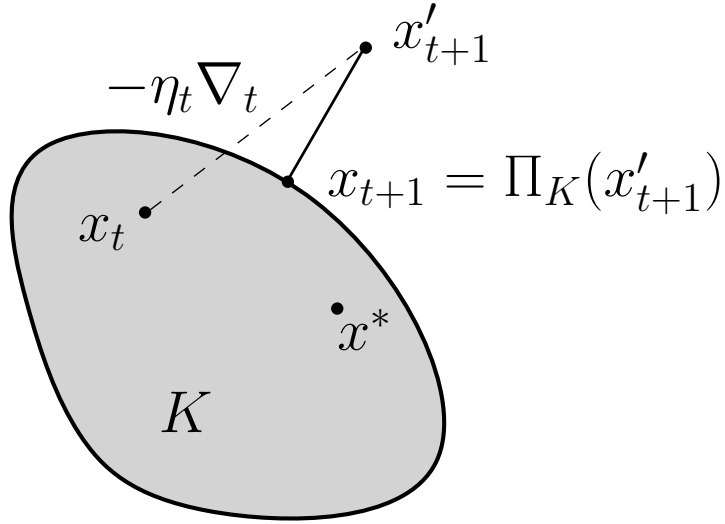
- Want to solve composite $\min_{x \in K} f(x) + g(x)$



Proximal Gradient Analysis

Recall proximal gradient method:

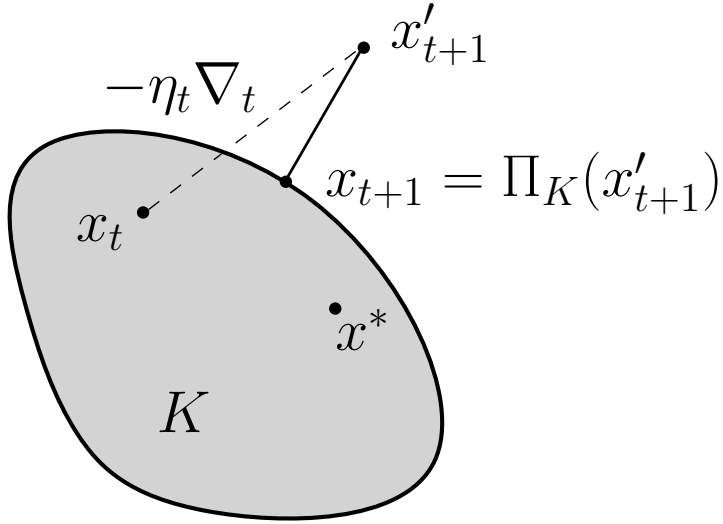
- Want to solve composite $\min_{x \in K} f(x) + g(x)$
- feasible region K is convex



Proximal Gradient Analysis

Recall proximal gradient method:

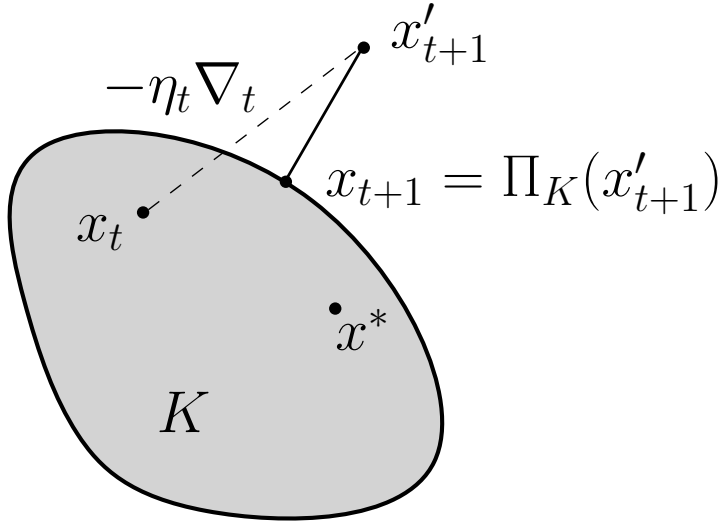
- Want to solve composite $\min_{x \in K} f(x) + g(x)$
- feasible region K is convex
- f is convex



Proximal Gradient Analysis

Recall proximal gradient method:

- Want to solve composite $\min_{x \in K} f(x) + g(x)$
- feasible region K is convex
- f is convex
- g is L-smooth, σ-strongly convex

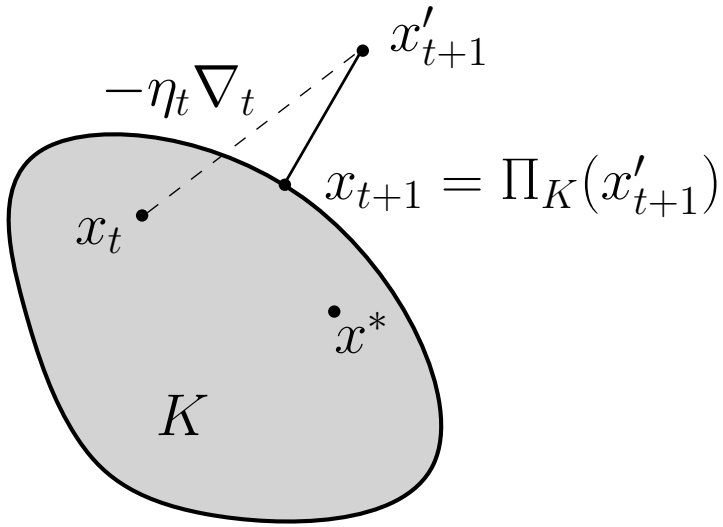


Proximal Gradient Analysis

Iterate

$$x^{t+1} \leftarrow \text{prox}_{L,f}\left(x^t - \frac{1}{L}\nabla g(x^t)\right)$$

$$\text{prox}_{L,f}(y) := \text{argmin}_{z \in K}\, f(z) + \frac{L}{2}\|z - x\|_2^2$$



Proximal Gradient Analysis

- $x^\star = \operatorname{argmin}_{x \in K} f(x) + g(x)$

Proximal Gradient Analysis

- $x^\star = \operatorname{argmin}_{x \in K} f(x) + g(x)$
- $\tilde{x}^\star = \operatorname{argmin}_{x \in K} \tilde{f}(x) + \tilde{g}(x)$

Proximal Gradient Analysis

- $x^\star = \operatorname{argmin}_{x \in K} f(x) + g(x)$
- $\tilde{x}^\star = \operatorname{argmin}_{x \in K} \tilde{f}(x) + \tilde{g}(x)$
- Analyze $\|x^\star - \tilde{x}^\star\|_2 = \lim_{t \to \infty} \|x^t - \tilde{x}^t\|_2$

Proximal Gradient Analysis

- $x^\star = \operatorname{argmin}_{x \in K} f(x) + g(x)$
- $\tilde{x}^\star = \operatorname{argmin}_{x \in K} \tilde{f}(x) + \tilde{g}(x)$
- Analyze $\|x^\star - \tilde{x}^\star\|_2 = \lim_{t \to \infty} \|x^t - \tilde{x}^t\|_2$

THEOREM (INFORMAL; LVYZ '24)

If $\|\nabla g - \nabla \tilde{g}\|_2$ is "small" and $\left\|\operatorname{prox}_{L,f} - \operatorname{prox}_{L,\tilde{f}}\right\|_2$ is "small", then $\|x^\star - \tilde{x}^\star\|_2$ is "small".

Application to Minimum s-t Cut

- $\varepsilon > 0$

minimize
$\sum_{uv \in E} w_{uv} |y_u - y_v|$
$+ \frac{\varepsilon}{2} \sum_{uv \in E} w_{uv} (y_u - y_v)^2$

subject to
$y_s - y_t = 1$
$\langle \mathbb{1}, y \rangle = 0$
$y \in [-1, 1]^V$
$y \in [y_t, y_s]^V$

Application to Minimum s-t Cut

- $\varepsilon > 0$
- $\|w - \tilde{w}\|_1 = \delta > 0$

minimize
$\sum_{uv \in E} w_{uv} |y_u - y_v|$
$+ \frac{\varepsilon}{2} \sum_{uv \in E} w_{uv} (y_u - y_v)^2$

subject to
$y_s - y_t = 1$
$\langle \mathbb{1}, y \rangle = 0$
$y \in [-1, 1]^V$
$y \in [y_t, y_s]^V$

Application to Minimum s-t Cut

- $\varepsilon > 0$
- $\|w - \tilde{w}\|_1 = \delta > 0$
- Optimal fractional solutions $y, \tilde{y}$

minimize

$$\sum_{uv \in E} w_{uv} |y_u - y_v| + \frac{\varepsilon}{2} \sum_{uv \in E} w_{uv} (y_u - y_v)^2$$

subject to

$$y_s - y_t = 1$$
$$\langle \mathbb{1}, y \rangle = 0$$
$$y \in [-1, 1]^V$$
$$y \in [y_t, y_s]^V$$

Application to Minimum s-t Cut

- $\varepsilon > 0$
- $\|w - \tilde{w}\|_1 = \delta > 0$
- Optimal fractional solutions $y, \tilde{y}$

We have $\|y^\star - \tilde{y}^\star\|_2 = O(\delta/\varepsilon\lambda_2)$

minimize
$\sum_{uv \in E} w_{uv}|y_u - y_v|$
$+\frac{\varepsilon}{2}\sum_{uv \in E} w_{uv}(y_u - y_v)^2$

subject to
$y_s - y_t = 1$
$\langle \mathbb{1}, y \rangle = 0$
$y \in [-1, 1]^V$
$y \in [y_t, y_s]^V$

Remarks

- $\|y^\star - \tilde{y}^\star\|_1 = O(\delta\sqrt{n}/\varepsilon\lambda_2)$

minimize
$\sum_{uv\in E} w_{uv}|y_u - y_v|$
$+\frac{\varepsilon}{2} y^\top \mathcal{L}(G, w) y$

subject to
$y_s - y_t = 1$
$\langle \mathbb{1}, y \rangle = 0$
$y \in [-1, 1]^V$
$y \in [y_t, y_s]^V$

Remarks

- $\|y^\star - \tilde{y}^\star\|_1 = O(\delta\sqrt{n}/\varepsilon\lambda_2)$
- $\sum_{uv\in E} w_{uv}|y^\star_u - y^\star_v| \le (1+\varepsilon)\,\mathrm{OPT}$

minimize
$\sum_{uv\in E} w_{uv}|y_u - y_v|$
$+\frac{\varepsilon}{2} y^\top \mathcal{L}(G,w) y$

subject to
$y_s - y_t = 1$
$\langle \mathbb{1}, y\rangle = 0$
$y \in [-1,1]^V$
$y \in [y_t, y_s]^V$

Rounding a Fractional Solution

Threshold cut

1. Draw threshold $\tau \sim U[-1, 1]$

Rounding a Fractional Solution

Threshold cut

1. Draw threshold $\tau \sim U[-1, 1]$
2. Output $S_\tau = \{v : y_v^\star \leq \tau\}$

Rounding a Fractional Solution

Threshold cut

1. Draw threshold $\tau \sim U[-1, 1]$
2. Output $S_\tau = \{v : y^\star_v \leq \tau\}$

- $\mathbb{E}[w(\partial S_\tau)] = \frac{1}{2} \sum_{uv \in E} w_{uv} |y^\star_u - y^\star_v|$

Rounding a Fractional Solution

Threshold cut

1. Draw threshold $\tau \sim U[-1, 1]$
2. Output $S_\tau = \{v : y^\star_v \leq \tau\}$

- $\mathbb{E}[w(\partial S_\tau)] = \frac{1}{2} \sum_{uv \in E} w_{uv} |y^\star_u - y^\star_v|$
- $\mathbb{E}_\tau[|S_\tau \Delta \tilde{S}_\tau|] = O(\|y^\star - \tilde{y}^\star\|_1)$

Rounding a Fractional Solution

Threshold cut

1. Draw threshold $\tau \sim U[-1, 1]$
2. Output $S_\tau = \{v : y^\star_v \leq \tau\}$

- $\mathbb{E}[w(\partial S_\tau)] = \frac{1}{2} \sum_{uv \in E} w_{uv} |y^\star_u - y^\star_v|$
- $\mathbb{E}_\tau[|S_\tau \Delta \tilde{S}_\tau|] = O(\|y^\star - \tilde{y}^\star\|_1)$
- But S_τ is s-t cut w.p. only $\approx 1/2$ ...

Rounding a Fractional Solution

Exponential mechanism + threshold cut

1. Solve LP(i) for each $i \in [k]$

minimize

$$\sum_{uv \in E} w_{uv} |y_u - y_v| + \frac{\varepsilon}{2} y^\top \mathcal{L}(G, w) y$$

subject to

$$y_s - y_t = 1$$
$$\langle \mathbb{1}, y \rangle = 0$$
$$y \in \left[-i/k, 1 - (i-1)/k\right]^V$$
$$y \in [y_t, y_s]^V$$

Rounding a Fractional Solution

Exponential mechanism + threshold cut

1. Solve LP(i) for each $i \in [k]$
2. Sample $i^\star = M_\eta(\mathrm{OPTLP} \in \mathbb{R}^k)$ w.p. $\propto \exp(-\eta\,\mathrm{OPTLP}(i))$ (exponential mechanism M_η)

minimize

$$\sum_{uv \in E} w_{uv}|y_u - y_v| + \tfrac{\varepsilon}{2} y^\top \mathcal{L}(G, w) y$$

subject to

$$y_s - y_t = 1$$
$$\langle \mathbb{1}, y \rangle = 0$$
$$y \in \left[-i/k,\, 1 - (i-1)/k\right]^V$$
$$y \in [y_t, y_s]^V$$

Rounding a Fractional Solution

Exponential mechanism + threshold cut

1. Solve LP(i) for each $i \in [k]$
2. Sample $i^\star = M_\eta(\mathrm{OPTLP} \in \mathbb{R}^k)$ w.p. $\propto \exp(-\eta\,\mathrm{OPTLP}(i))$ (exponential mechanism M_η)
3. Threshold cut S_τ for $\tau \sim U[-{}^{i^\star}\!/_k, 1 - {}^{(i^\star-1)}\!/_k]$

minimize

$$\sum_{uv \in E} w_{uv}|y_u - y_v| + \tfrac{\varepsilon}{2} y^\top \mathcal{L}(G, w) y$$

subject to

$$y_s - y_t = 1$$
$$\langle \mathbb{1}, y \rangle = 0$$
$$y \in [-{}^{i}\!/_k, 1 - {}^{(i-1)}\!/_k]^V$$
$$y \in [y_t, y_s]^V$$

Rounding a Fractional Solution

Exponential mechanism + threshold cut

- $w(\partial S_\tau) \leq (1 + k\varepsilon)\,\mathrm{OPT} + O(\eta^{-1} k \log k)$ w.p. $1 - 1/k$

Rounding a Fractional Solution

Exponential mechanism + threshold cut

- $w(\partial S_\tau) \leq (1 + k\varepsilon)\,\mathrm{OPT} + O(\eta^{-1} k \log k)$ w.p. $1 - 1/k$
- S_τ is s-t cut w.p. $1 - 1/k$

Rounding a Fractional Solution

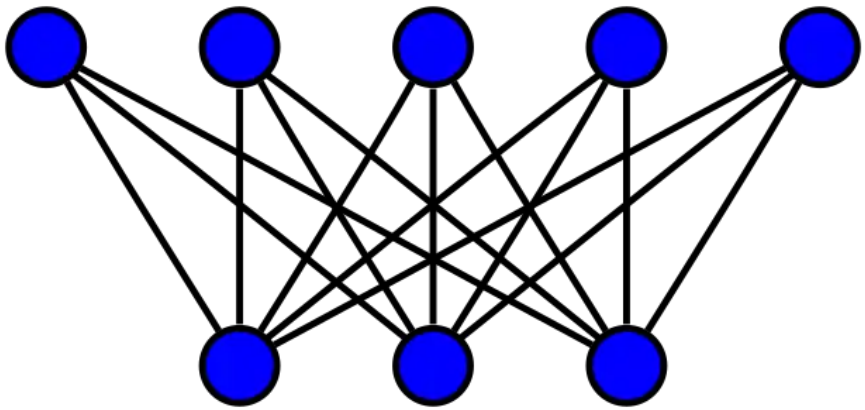
Exponential mechanism + threshold cut

- $w(\partial S_\tau) \le (1 + k\varepsilon)\,\mathrm{OPT} + O(\eta^{-1} k \log k)$ w.p. $1 - 1/k$
- S_τ is s-t cut w.p. $1 - 1/k$
-

$$
\begin{aligned}
&\mathbb{E}_{M,\tilde{M},\tau}[|S_\tau \Delta \tilde{S}_\tau|] \\
&\le (1 - \mathrm{TV}(M, \tilde{M})) \cdot O(\|y^\star - \tilde{y}^\star\|_1) + \mathrm{TV}(M, \tilde{M}) \cdot n \\
&\le O\left(\frac{\delta\sqrt{n}}{\varepsilon\lambda_2}\right) + O(\eta k \delta) \cdot n \\
&= O\left(\frac{n}{\lambda_2}\right) \qquad\qquad (\eta = 1/k\lambda_2, \varepsilon = 1/\sqrt{n})
\end{aligned}
$$

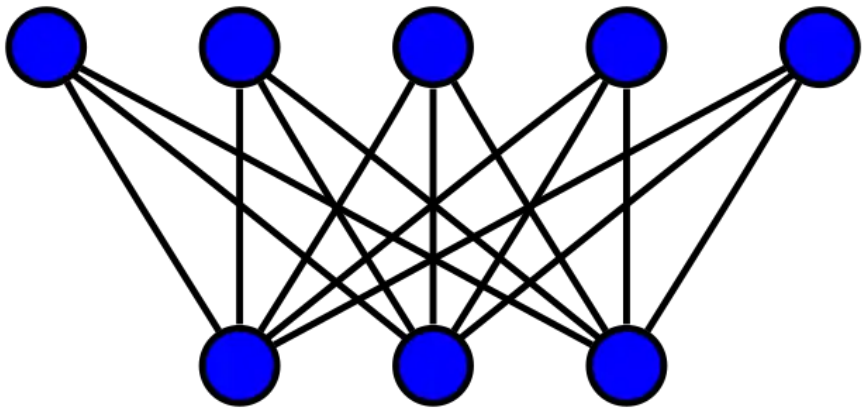
Remarks

- $\lambda_2 \neq \tilde{\lambda}_2$, need to sample from $U[\lambda_2/2, \lambda_2]$



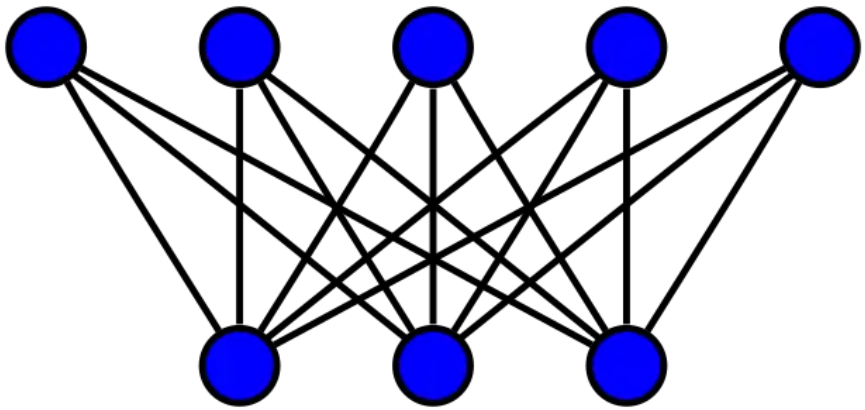
Remarks

- $\lambda_2 \neq \tilde{\lambda}_2$, need to sample from $U[\lambda_2/2, \lambda_2]$
- Alternative rounding algorithm using k-way submodularity yields different guarantees



Remarks

- $\lambda_2 \neq \tilde{\lambda}_2$, need to sample from $U[\lambda_2/2, \lambda_2]$
- Alternative rounding algorithm using k-way submodularity yields different guarantees
- Lower bound attained by a complete bipartite graph



felix.zhou@yale.edu

Thank You!

arxiv.org/abs/2405.08938