Online Algorithms for Spectral Sparsification of Hypergraphs

Kam Chuen (Alex) Tung
PhD Candidate, University of Waterloo
15 January 2025 @ U Waterloo

Joint work with Tasuku Soma (ISM Japan) and Yuichi Yoshida (NII Japan)
Slides mostly based on Soma and Yoshida

Table of Content

Table of Content

Graph Sparsification

- Let G = (V, E) be a graph
- Objective: reduce graph size to speed up downstream algorithms

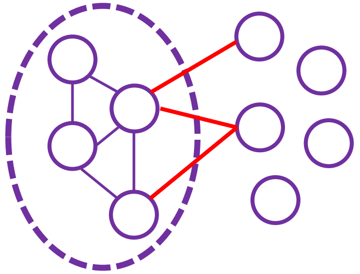
- What to preserve?
  - Shortest paths -> Spanners [Peleg, Schäffer '89]
  - Cuts -> Cut sparsifiers [Karger '93], [Benczur, Karger '96]
  - Spectrum -> Spectral sparsifiers [Spielman, Teng '11]
  - Many other possibilities…

Cut Sparsification of Graphs

- $G = (V, E, w)$ weighted graph
- $\tilde{G} = (V, E, \widetilde{w})$ reweighted subgraph, edge weight $\widetilde{w}_e \geq 0$

$\tilde{G}$ is an ϵ-cut sparsifier of G if:

- $(1-\epsilon)cut_G(S) \leq cut_{\tilde{G}}(S) \leq (1+\epsilon)cut_G(S)$ for all $S \subseteq V$
- The number of edges (with positive $\widetilde{w}_e$) is small



$cut_G(S) \coloneqq \sum_{e \in \delta(S)} w_e$

Spectral Sparsification of Graphs [Spielman, Teng '11]

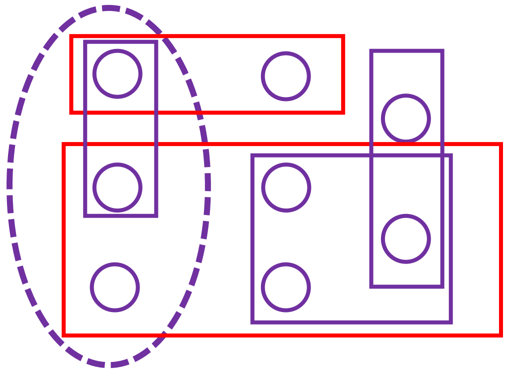
- L: Laplacian of G, $\tilde{L}$: Laplacian of $\tilde{G}$
- Quadratic form: $x^T Lx = \sum_{uv \in E} w_{uv} \big(x(u) - x(v)\big)^2$
- Want $(1-\epsilon)x^T Lx \leq x^T \tilde{L} x \leq (1+\epsilon) x^T Lx$ for all $x \in \mathbb{R}^V$
- Spectral sparsifier is cut sparsifier: when $x = \chi_S$, $x^T Lx = cut_G(S)$

Spectral Sparsification of Graphs: Methods

- Decomposition-based: $O\left(\frac{n \operatorname{polylog} n}{\epsilon^2}\right)$ edges [Spielman, Teng '11]
- Sampling-based: $O\left(\frac{n \log n}{\epsilon^2}\right)$ edges [Spielman, Srivastava '08]
- Potential function-based: $O\left(\frac{n}{\epsilon^2}\right)$ edges [Batson, Spielman, Srivastava '09]

- Applications: Laplacian solvers, flow/cut algorithms, clustering, sampling spanning trees...

Spectral Sparsification of Hypergraphs



- Hypergraph cut: $cut_H(S) := \sum_{0<|e\cap S|<|e|} w_e$
- Hypergraph energy/"quadratic form" [Soma, Yoshida '19]

$$Q_H(x) := \sum_{e\in E} w_e \cdot \max_{u,v\in e} \left(x(u) - x(v)\right)^2$$

- Want $(1-\epsilon)Q_H(x) \leq Q_{\tilde{H}}(x) \leq (1+\epsilon)Q_H(x)$ for all $x \in \mathbb{R}^V$
- Spectral sparsifier is cut sparsifier: when $x = \chi_S$, $Q_H(x) = cut_H(S)$

$n := |V|, r := \max_{e \in E} |e|$

Hypergraph Sparsification: Results

Reference	#hyperedges	Cut/ Spectral?	Method
[Kogan, Krauthgamer '15]	$O\left(\frac{n(r+\log n)}{\epsilon^2}\right)$	Cut	Sampling
[Soma, Yoshida '19]	$O\left(\frac{n^3 \log n}{\epsilon^2}\right)$	Spectral	Sampling
[Bansal, Svensson, Trevisan '19]	$O\left(\frac{nr^3 \log n}{\epsilon^2}\right)$	Spectral	Sampling
[Chen, Khanna, Nagda '20]	$O\left(\frac{n \log n}{\epsilon^2}\right)$	Cut	Sampling
[Kapralov, Krauthgamer, Tardos, Yoshida '21]	$\tilde{O}\left(\frac{nr}{\epsilon^{O(1)}}\right)$	Spectral	Decomposition
[Kapralov, Krauthgamer, Tardos, Yoshida '22]	$O\left(\frac{n \log^3 n}{\epsilon^4}\right)$	Spectral	Sampling
[Lee '23, Jambulapati, Liu, Sidford '23]	$O\left(\frac{n \log n \log r}{\epsilon^2}\right)$	Spectral	Sampling

Memory Issue

- All these algorithms are offline
- For hypergraphs, m $:= |E|$ can be as large as 2^n
- May be expensive simply to store the entire hypergraph!

GOAL: Sparsifier without using $\Omega(m)$ memory

Online Setting

- Hyperedges e_i (weight w_i) arrive one by one
- Upon arrival, the algorithm decides **immediately** whether to include e_i, and the new weight $\widetilde{w}_i$ if applicable

- Task: find a spectral sparsifier using poly(n) working memory

Our Result

<u>Definition</u> $\tilde{H}$: (ϵ, δ)-spectral sparsifier of H iff

$$(1-\epsilon)Q_H(x) - \delta\|x\|_2^2 \le Q_{\tilde{H}}(x) \le (1+\epsilon)Q_H(x) + \delta\|x\|_2^2$$

<u>Theorem</u> [Soma, T., Yoshida '24] There is an online algorithm that computes an (ϵ, δ)-spectral sparsifier with $O\left(\epsilon^{-2} n \log n \log r \log \frac{\epsilon W}{\delta n}\right)$ hyperedges w.h.p., using $O(n^2)$ space. (Here $W := \max_e w_e$)

- r-uniform and unweighted: $O(\epsilon^{-2} n r \log^2 n \log r)$ hyperedges
- Sampling-based algorithm

Table of Content

- Introduction
- Past Work
- Our Algorithm
- Analysis
- Summary

Effective Resistance Sampling [Spielman, Srivastava '08]

- Sample graph edge $e = uv$ proportional to effective resistance r_e
- $r_e \coloneqq \left\|L^{-1/2}(\chi_u - \chi_v)\right\|^2 = (\chi_u - \chi_v)^T L^{-1} (\chi_u - \chi_v)$
- Sample edge e with probability $p_e \coloneqq \min(1, C \cdot w_e r_e)$
- New weight $\widetilde{w}_e \coloneqq w_e / p_e$ if sampled

Oversampling rate

Edge Laplacian

- Unbiased: $\mathbb{E}[\widetilde{L}] = \mathbb{E}[\sum_{e \in E} \widetilde{w}_e L_e] = \sum_{e \in E} w_e L_e = L$
- Matrix Chernoff $\Rightarrow$ ϵ-spectral sparsifier w.h.p. when $C = \Theta\left(\frac{\log n}{\epsilon^2}\right)$
- Expected #edges = $O(\sum_{e \in E} p_e) = O(C \cdot \sum_{e \in E} w_e r_e) = O\left(\frac{n \log n}{\epsilon^2}\right)$

Sum of $w_e r_e$ is $O(n)$

Importance (aka Leverage Score) Sampling

- Effective resistance as "importance" of edge $e = uv$:

$$r_e := \left\| L^{-\frac{1}{2}}(\chi_u - \chi_v) \right\|^2 = \max_{x \in \mathbb{R}^V} \frac{x^T L_e x}{x^T L x}$$

("Maximum contribution of edge energy to the overall energy")

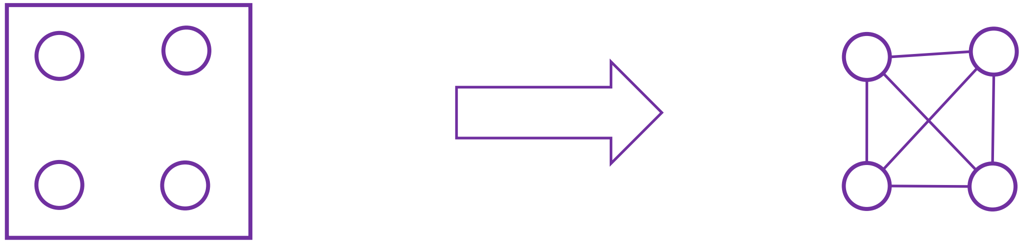
- Importance of a hyperedge e can be similarly defined as

$$r_e := \max_{x \in \mathbb{R}^V} \frac{Q_e(x)}{Q_H(x)}$$

$\max_{u,v \in e} \left(x(u) - x(v) \right)^2$

Clique-Graph Reweighting [CKN ’20, KKTY ’22, Lee ’23]

- $G = (V, F)$, each hyperedge replaced with clique



<u>Lemma</u> [KKTY ’22, Lee ’23] There exists edge weights $c_{e,u,v} \geq 0$ on F s.t.:
- $\sum_{u,v \in e} c_{e,u,v} = w_e$ for all $e \in E$
- If $c_{e,u,v} > 0$, then $r_{u,v} = \max_{u',v' \in e} r_{u',v'}$, where $r_{u,v}$ is the effective resistance between u and v in the c-weighted graph.

Computing $c_{e,u,v}$

<u>Lemma</u> [KKTY '22, Lee '23] There exist edge weights $c_{e,u,v} \geq 0$ on F s.t.:

- $\sum_{u,v \in e} c_{e,u,v} = w_e$ for all $e \in E$
- If $c_{e,u,v} > 0$, then $r_{u,v} = \max_{u',v' \in e} r_{u',v'}$, where $r_{u,v}$ is the effective

resistance between u and v in the c-weighted clique-graph.

- The weights $c_{e,u,v}$ can be found via convex optimization:

$$\begin{aligned} \max \quad & \log\det\left(\sum_{e \in E} \sum_{u,v \in e} c_{e,u,v} L_{u,v} + J\right) \\ \text{s.t.} \quad & \sum_{u,v \in e} c_{e,u,v} = w_e \qquad \forall e \in E \\ & c_{e,u,v} \geq 0 \qquad \forall e \in E, u, v \in e \end{aligned}$$

- The second condition in lemma follows from KKT condition

Sampling Algorithm [Lee '23]

- Given hypergraph $H = (V, E, w)$
- Compute $c_{e,u,v}$ for the clique graph $G = (V, F)$
- Sample hyperedge $e \in E$ w.p. $p_e = \min(1, C \cdot w_e \max_{u,v \in e} r_{u,v})$, weight $\widetilde{w}_e = w_e / p_e$ if sampled

- $\max_{u,v \in e} r_{u,v}$ is a computable **overestimate** of the importance of e
- Talagrand's generic chaining $\Longrightarrow$ Sampled hypergraph $\widetilde{H}$ is an ϵ-spectral sparsifier of H w.h.p. when $C = \Theta\left(\frac{\log n \log r}{\epsilon^2}\right)$

Bound on number of hyperedges [Lee '23]

- The expected number of hyperedges is $O(C \cdot \sum_{e \in E} w_e \max_{u,v \in e} r_{u,v})$

Property 1 of $c_{e,u,v}$

$$w_e \max_{u,v \in e} r_{u,v} = \sum_{u,v \in e} c_{e,u,v} \max_{u',v' \in e} r_{u',v'}$$
$$= \sum_{u,v \in e} c_{e,u,v} r_{u,v}$$

Property 2 of $c_{e,u,v}$

- This is $O\left(C \cdot \sum_{e \in E} \sum_{u,v \in e} c_{e,u,v} r_{u,v}\right) = O(C \cdot n) = O\left(\frac{n \log n \log r}{\epsilon^2}\right)$

Sum of $w_{u,v} r_{u,v}$ is $O(n)$

Table of Content

Our online algorithm

$\widetilde{H}_0 \coloneqq (V, \emptyset)$, $L_0 \coloneqq O_n$ (zero matrix)

for $i = 1,2,\dots,T$:

e_i arrives with weight w_i

Solve the convex optimization problem for $c_{i,u,v}$:

$$\max_{c_i \in \Delta_{e_i}} \log\det\left(L_{i-1} + \sum_{u,v\in e_i} w_i c_{i,u,v} L_{u,v} + \eta I_n\right)$$

Ridge regularizer for warm start

$L_i \leftarrow L_{i-1} + \sum_{u,v\in e_i} w_i c_{i,u,v} L_{u,v}$

Clique-graph reweighting

Ridged effective resistance

Set $p_i \coloneqq \min\left(1, C \cdot w_i \max_{u,v\in e_i} \left\| (L_i + \eta I)^{-1/2} (\chi_u - \chi_v) \right\|_2^2\right)$

Add e_i with weight w_i / p_i to $\widetilde{H}_{i-1}$ with probability p_i to obtain $\widetilde{H}_i$

Note that this is independent sampling!!

Table of Content

Outline of Analysis

Need to show two things:

- The sampled hypergraph $\widetilde{H}_T$ is an (ϵ, δ)-spectral sparsifier of H_T
- The number of hyperedges in $\widetilde{H}_T$ is small

$\widetilde{H}_T$ is an (ϵ, δ)-spectral sparsifier of H_T

- Let $\eta := \delta/\epsilon$. Define η-ridged energy as $Q_H^\eta(x) := Q_H(x) + \eta\|x\|^2$
- Let $Z := \sup_{x:\, Q_H^\eta(x) \leq 1} \left| Q_H^\eta(x) - Q_{\widetilde{H}}^\eta(x) \right|$
  - Note pointwise concentration
- Use Talagrand's generic chaining to bound $\mathbb{E}_{\widetilde{H}}[e^{\lambda Z}]$

  [Jambulapati, Lee, Liu, Sidford '23]

  Oversampling rate

- For $C = \Theta(\epsilon^{-2} \log n \log r)$, the exponential MGF bound implies that $\Pr[Z \leq \epsilon] \geq 1 - 1/n$
- Finally, $Z \leq \epsilon$ implies that

$$(1-\epsilon)Q_H(x) - \delta\|x\|^2 \leq Q_{\widetilde{\mathrm{H}}}(x) \leq (1+\epsilon)Q_H(x) + \delta\|x\|^2$$

Bound on the number of hyperedges

- Adaptation of [Cohen, Musco, Pachocki '16]
- Define $\Phi_i := \log\det(L_i + \eta I_n) = \log\det(L_i^\eta)$

<u>Lemma</u> [STY '24] $\Phi_i - \Phi_{i-1} \geq \frac{p_i \log 2}{C}$

- $\mathbb{E}\left[\left|E\left(\widetilde{H}\right)\right|\right] = \sum_i p_i \leq \frac{C(\Phi_T - \Phi_0)}{\log 2}$, where $C = \Theta(\epsilon^{-2} \log n \log r)$
- $\Phi_T - \Phi_0 = \log\left(\frac{\det(L_T + \eta I_n)}{\det(\eta I_n)}\right) = \log\det(I + \eta^{-1} L_T) \leq n \log\left(1 + \frac{tr(L_T)}{\eta \cdot n}\right)$ (AM-GM)
- Plug in $\eta = \delta/\epsilon$ and use $tr(L_T) \leq O(W)$. Conclude that

$$\mathbb{E}\left[\left|E\left(\widetilde{H}\right)\right|\right] \leq O\left(\epsilon^{-2} n \log n \log r \log\left(1 + \frac{\epsilon W}{\delta n}\right)\right)$$

Working memory

- The algorithm maintains a clique-graph reweighting
- $O(n^2)$ working memory

Table of Content

Summary and Open Questions

- First online algorithm for spectral hypergraph sparsification
  - $O(\epsilon^{-2} n \log n \log r \log \frac{\epsilon W}{\delta n})$ hyperedges
  - $\Omega(\epsilon^{-2} n \log \frac{\epsilon W}{\delta n})$ edges needed even for graphs [Cohen, Musco, Pachocki '16]

- Reduce the space complexity from $O(n^2)$ to $\tilde{O}(nr)$?
- Fully dynamic setting?
- Potential function-based algorithms?
- How to certify hypergraph cut/spectral sparsifier?

The End

- Thank you! Any questions?