Sublinear Spectral Clustering Oracle with Little Memory

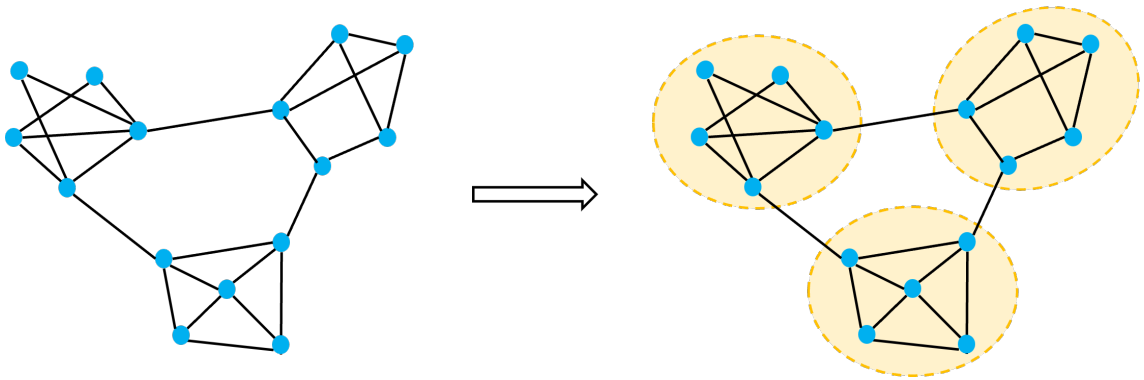
Ranran Shen[1] Xiaoyi Zhu[2] Pan Peng[1] Zengfeng Huang[2,3]

[1]University of Science and Technology of China (USTC)
[2]Fudan University
[3]Shanghai Innovation Institute

March 30, 2026

Graph Clustering

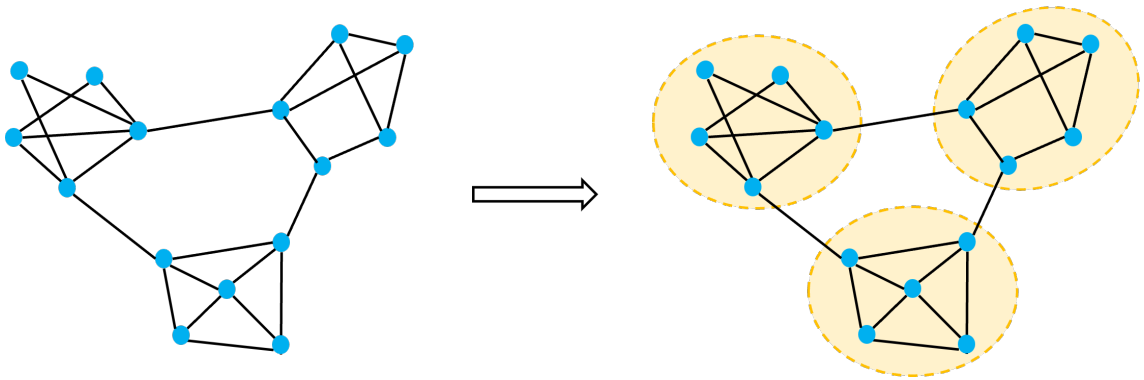
- **Input:** $G = (V, E)$ and k $(k \geq 2)$
- **Goal:** partition V into k disjoint clusters $C_1, \ldots, C_k$, such that each cluster exhibits
  - tight connections inside
  - loose connections outside



Example $(k = 3)$

Graph Clustering

- **Input:** $G = (V, E)$ and k $(k \geq 2)$
- **Goal:** partition V into k disjoint clusters $C_1, \ldots, C_k$, such that each cluster exhibits
  - tight connections inside
  - loose connections outside

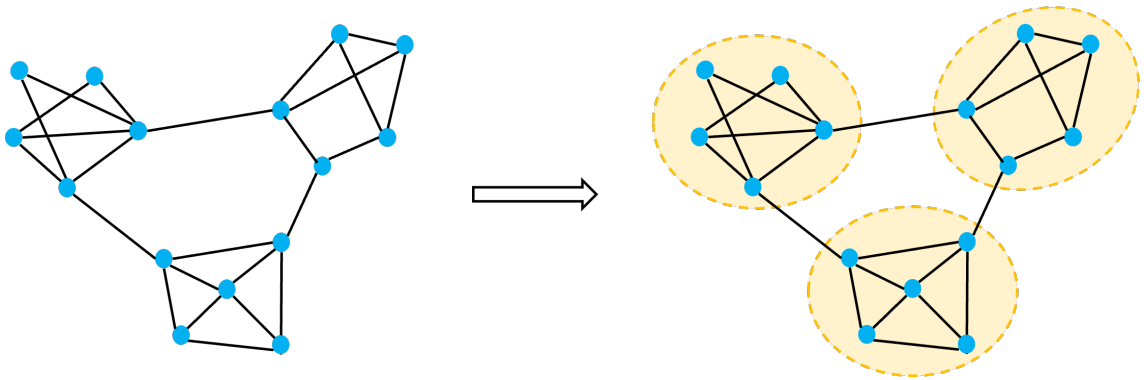


Example $(k = 3)$

Global algorithms: $\mathrm{poly}(n)$ time and space $\Rightarrow$ expensive.

Graph Clustering

- **Input:** $G = (V, E)$ and k $(k \geq 2)$
- **Goal:** partition V into k disjoint clusters $C_1, \ldots, C_k$, such that each cluster exhibits
  - tight connections inside
  - loose connections outside



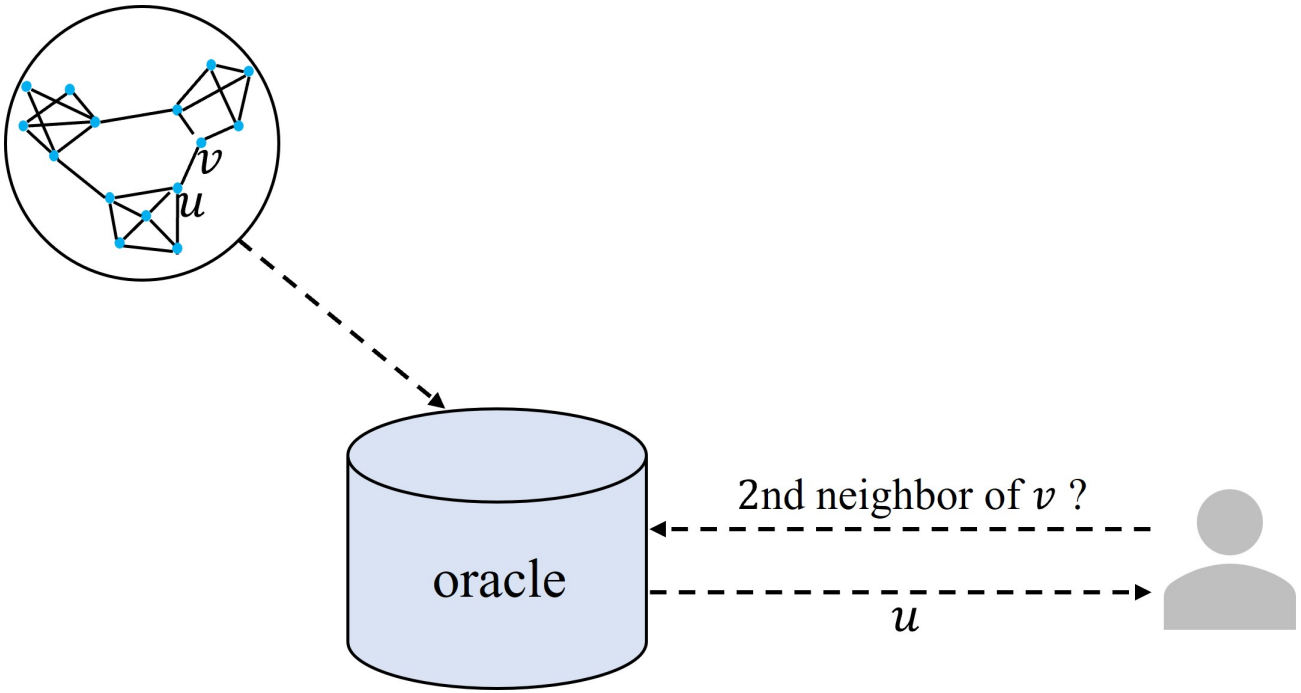
Example $(k = 3)$

Global algorithms: $\mathrm{poly}(n)$ time and space $\Rightarrow$ expensive.

We focus on sublinear spectral clustering oracles.

Query Access

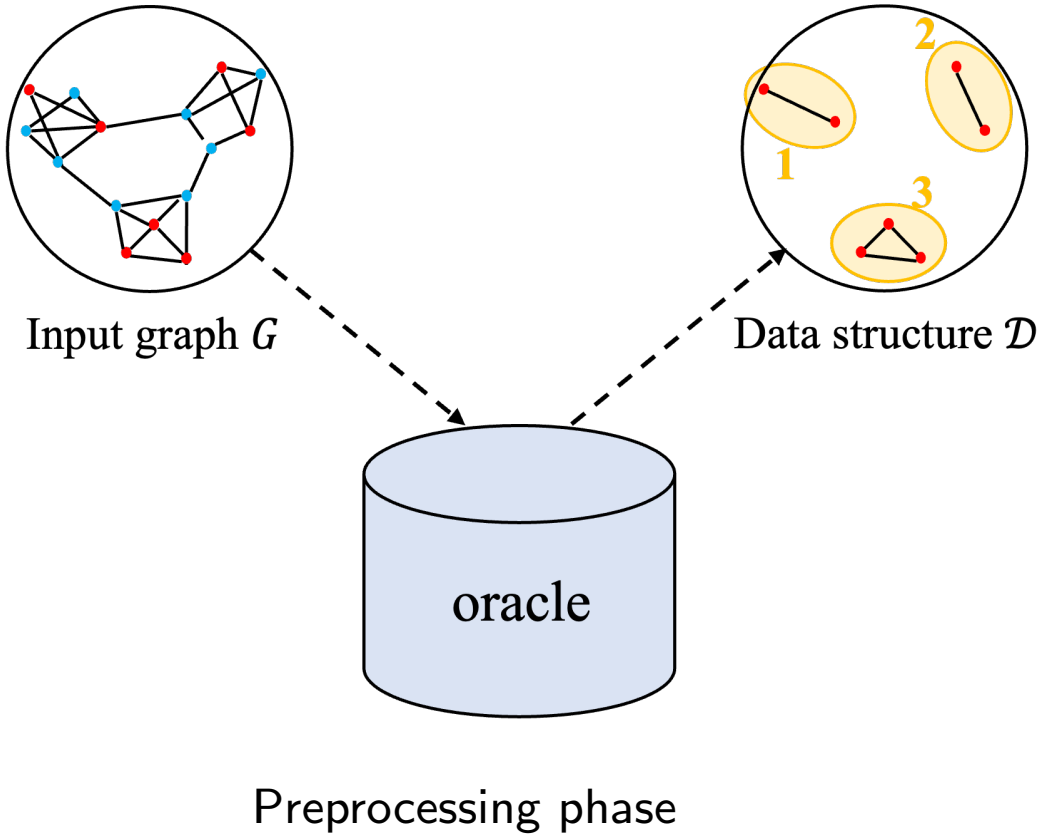
- Adjacency list of G: query i-th neighbor of any vertex u in $O(1)$ time



Neighbor query

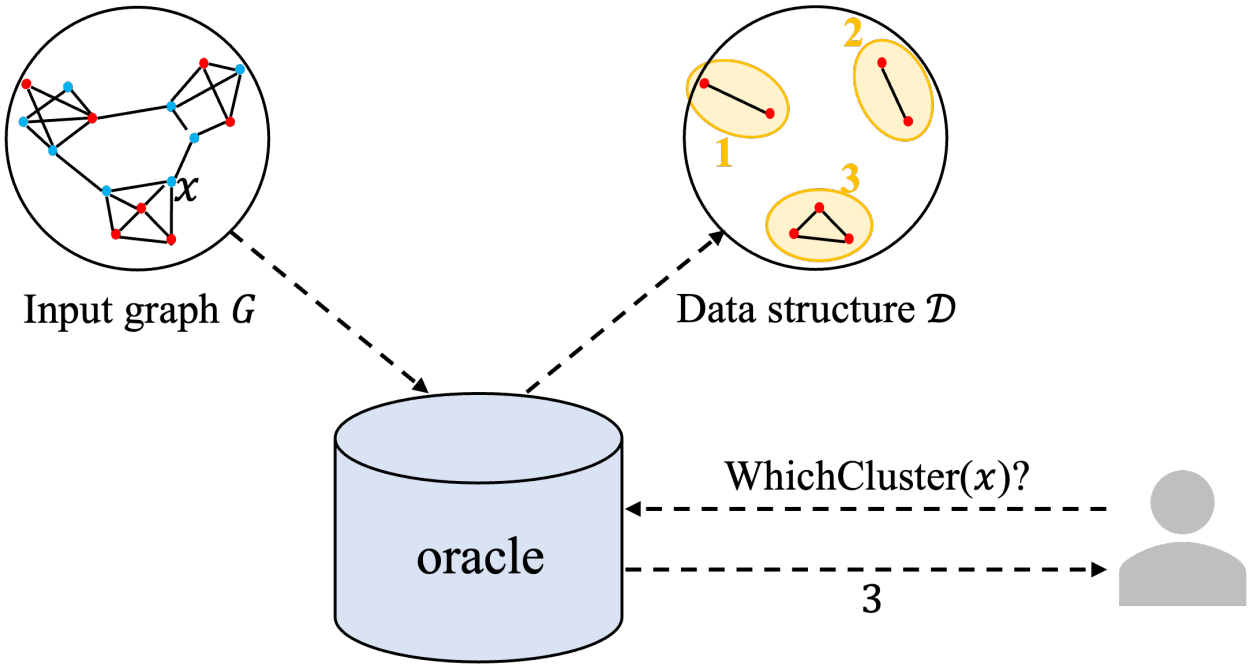
Two Phases (both in sublinear time and space)

- Preprocessing phase
  - build a data structure $\mathcal{D}$



Two Phases (both in sublinear time and space)

- Preprocessing phase
  - build a data structure $\mathcal{D}$
- Query phase
  - use $\mathcal{D}$ to answer $\textsc{WhichCluster}(x)$ queries



Query phase

Requirement

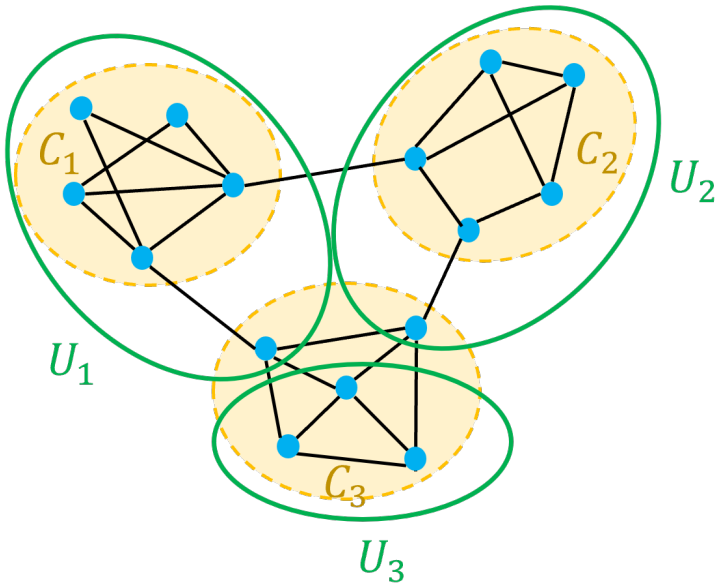
Let $U_i = \{x \in V : \textsc{WhichCluster}(x) = i\}, 1 \leq i \leq k$.

Requirement

Let $U_i = \{x \in V : \textsc{WhichCluster}(x) = i\}, 1 \le i \le k$.

The resulting partition $U_1, \ldots, U_k$ should be

- close to the ground-truth clustering



$|U_i \triangle C_i|$ is small

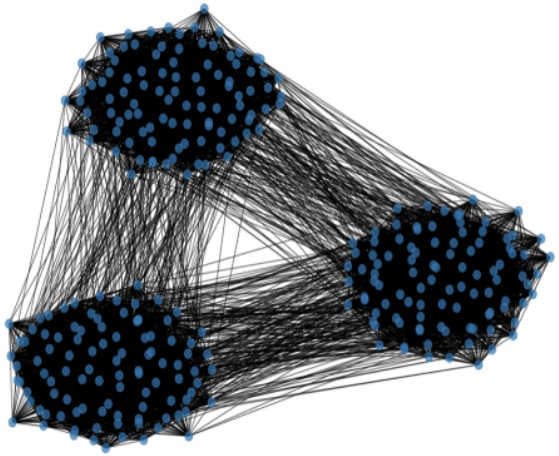
Small misclassification error

More Formally About Input Graph G

- d-bounded graphs: maximum degree $\leq d$

More Formally About Input Graph G

- d-bounded graphs: maximum degree $\leq d$

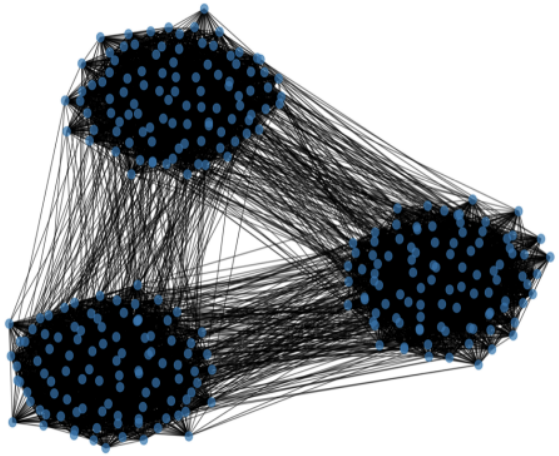


A clusterable graph

- $(k, \varphi, \varepsilon)$-clusterable graphs ($\varepsilon \ll \varphi$)

More Formally About Input Graph G

- d-bounded graphs: maximum degree $\leq d$

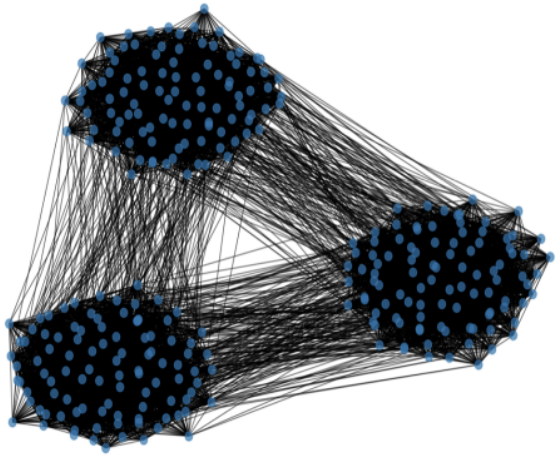


A clusterable graph

- $(k, \varphi, \varepsilon)$-clusterable graphs ($\varepsilon \ll \varphi$)
  - has a k-partition of V, denoted by $C_1, \ldots, C_k$, $\frac{|C_i|}{|C_j|} \in O(1)$

More Formally About Input Graph G

- d-bounded graphs: maximum degree $\leq d$

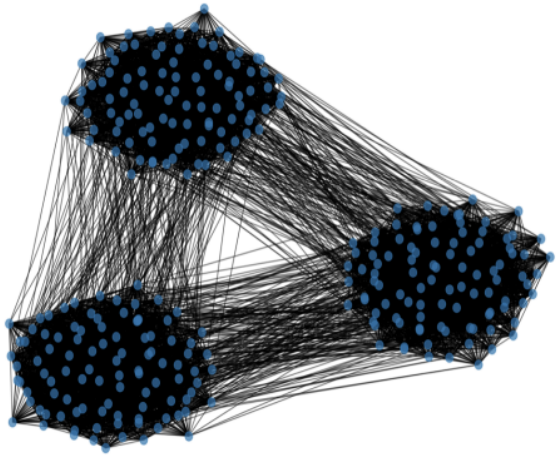


A clusterable graph

- $(k, \varphi, \varepsilon)$-clusterable graphs ($\varepsilon \ll \varphi$)
  - has a k-partition of V, denoted by $C_1, \ldots, C_k$, $\frac{|C_i|}{|C_j|} \in O(1)$
  - tight connections inside: inner conductance $\phi_{\text{in}}(C_i) \geq \varphi$

More Formally About Input Graph G

- d-bounded graphs: maximum degree $\leq d$



A clusterable graph

- $(k, \varphi, \varepsilon)$-clusterable graphs ($\varepsilon \ll \varphi$)
  - has a k-partition of V, denoted by $C_1, \ldots, C_k$, $\frac{|C_i|}{|C_j|} \in O(1)$
  - tight connections inside: inner conductance $\phi_{\text{in}}(C_i) \geq \varphi$
  - loose connections outside: outer conductance $\phi_{\text{out}}(C_i, V) \leq \varepsilon$

Previous Work

	space S for constructing $\mathcal{D}$	query time T
[Pen20]	$\widetilde{O}_\varphi\left(\sqrt{n}\cdot \mathrm{poly}(\frac{k}{\varepsilon})\right)$	$\widetilde{O}_\varphi\left(\sqrt{n}\cdot \mathrm{poly}(\frac{k}{\varepsilon})\right)$
[GKL+21]	$\widetilde{O}_\varphi\left(n^{1-\delta+O(\varepsilon)}\cdot \mathrm{poly}(\frac{k}{\varepsilon})\right)^\dagger$	$\widetilde{O}_\varphi\left(n^{\delta+O(\varepsilon)}\cdot \mathrm{poly}(\frac{k}{\varepsilon})\right)^\dagger$
[SP23]	$\widetilde{O}_\varphi\left(n^{1-\delta+O(\varepsilon)}\cdot \mathrm{poly}(k)\right)^\dagger$	$\widetilde{O}_\varphi\left(n^{\delta+O(\varepsilon)}\cdot \mathrm{poly}(k)\right)^\dagger$

- $\widetilde{O}$ hides all $\mathrm{poly}(\log n)$ factors
- O_φ suppresses dependence on φ
- † $\delta \in (0, \frac{1}{2}]$

The Problem

	space S for constructing $\mathcal{D}$	query time T
[Pen20]	$\widetilde{O}_\varphi\left(\sqrt{n}\cdot \mathrm{poly}(\frac{k}{\varepsilon})\right)$	$\widetilde{O}_\varphi\left(\sqrt{n}\cdot \mathrm{poly}(\frac{k}{\varepsilon})\right)$
[GKL+21]	$\widetilde{O}_\varphi\left(n^{1-\delta+O(\varepsilon)}\cdot \mathrm{poly}(\frac{k}{\varepsilon})\right)^\dagger$	$\widetilde{O}_\varphi\left(n^{\delta+O(\varepsilon)}\cdot \mathrm{poly}(\frac{k}{\varepsilon})\right)^\dagger$
[SP23]	$\widetilde{O}_\varphi\left(n^{1-\delta+O(\varepsilon)}\cdot \mathrm{poly}(k)\right)^\dagger$	$\widetilde{O}_\varphi\left(n^{\delta+O(\varepsilon)}\cdot \mathrm{poly}(k)\right)^\dagger$

- † $\delta \in (0, \frac{1}{2}]$
- Existing oracles:
  - at least $\Omega(\sqrt{n})$ space to construct $\mathcal{D}$

The Problem

	space S for constructing $\mathcal{D}$	query time T
[Pen20]	$\widetilde{O}_\varphi\left(\sqrt{n}\cdot \mathrm{poly}(\frac{k}{\varepsilon})\right)$	$\widetilde{O}_\varphi\left(\sqrt{n}\cdot \mathrm{poly}(\frac{k}{\varepsilon})\right)$
[GKL+21]	$\widetilde{O}_\varphi\left(n^{1-\delta+O(\varepsilon)}\cdot \mathrm{poly}(\frac{k}{\varepsilon})\right)^\dagger$	$\widetilde{O}_\varphi\left(n^{\delta+O(\varepsilon)}\cdot \mathrm{poly}(\frac{k}{\varepsilon})\right)^\dagger$
[SP23]	$\widetilde{O}_\varphi\left(n^{1-\delta+O(\varepsilon)}\cdot \mathrm{poly}(k)\right)^\dagger$	$\widetilde{O}_\varphi\left(n^{\delta+O(\varepsilon)}\cdot \mathrm{poly}(k)\right)^\dagger$

- † $\delta \in (0, \frac{1}{2}]$
- Existing oracles:
  - at least $\Omega(\sqrt{n})$ space to construct $\mathcal{D}$
- Realistic environments:

The Problem

	space S for constructing $\mathcal{D}$	query time T
[Pen20]	$\widetilde{O}_\varphi\left(\sqrt{n}\cdot \mathrm{poly}(\frac{k}{\varepsilon})\right)$	$\widetilde{O}_\varphi\left(\sqrt{n}\cdot \mathrm{poly}(\frac{k}{\varepsilon})\right)$
[GKL+21]	$\widetilde{O}_\varphi\left(n^{1-\delta+O(\varepsilon)}\cdot \mathrm{poly}(\frac{k}{\varepsilon})\right)^\dagger$	$\widetilde{O}_\varphi\left(n^{\delta+O(\varepsilon)}\cdot \mathrm{poly}(\frac{k}{\varepsilon})\right)^\dagger$
[SP23]	$\widetilde{O}_\varphi\left(n^{1-\delta+O(\varepsilon)}\cdot \mathrm{poly}(k)\right)^\dagger$	$\widetilde{O}_\varphi\left(n^{\delta+O(\varepsilon)}\cdot \mathrm{poly}(k)\right)^\dagger$

- † $\delta \in (0, \frac{1}{2}]$
- Existing oracles:
  - at least $\Omega(\sqrt{n})$ space to construct $\mathcal{D}$
- Realistic environments:
  - working memory is limited
  - $\Omega(\sqrt{n})$ space becomes the bottleneck

Central Question

- Can we break the $\Omega(\sqrt{n})$ space barrier?

Central Question

- Can we break the $\Omega(\sqrt{n})$ space barrier?
  - What is the trade-off between space S and query time T?

Main Result

Theorem (informal)

Suppose $\varepsilon \leq h(d, k, \varphi)$ for some function h. Let $G = (V, E)$ be a d-regular $(k, \varphi, \varepsilon)$-clusterable graph with clusters $C_1, \ldots, C_k$. There exists a sublinear spectral clustering oracle that:

- space $S = \widetilde{O}_{\varphi,k}\left(n^{O(\varepsilon)} \cdot M\right)$ bits, where $1 \leq M < \sqrt{n}$
- query time $T = \widetilde{O}_{\varphi,k}\left(n^{1+O(\varepsilon)}/M\right)$.

Main Result

Theorem (informal)

Suppose $\varepsilon \leq h(d, k, \varphi)$ for some function h. Let $G = (V, E)$ be a d-regular $(k, \varphi, \varepsilon)$-clusterable graph with clusters $C_1, \ldots, C_k$. There exists a sublinear spectral clustering oracle that:

- space $S = \widetilde{O}_{\varphi,k}\left(n^{O(\varepsilon)} \cdot M\right)$ bits, where $1 \leq M < \sqrt{n}$
- query time $T = \widetilde{O}_{\varphi,k}\left(n^{1+O(\varepsilon)}/M\right)$.

- break the $\Omega(\sqrt{n})$ space barrier:
  - S can be far below $\sqrt{n}$

Main Result

Theorem (informal)

Suppose $\varepsilon \leq h(d, k, \varphi)$ for some function h. Let $G = (V, E)$ be a d-regular $(k, \varphi, \varepsilon)$-clusterable graph with clusters $C_1, \ldots, C_k$. There exists a sublinear spectral clustering oracle that:

- space $S = \widetilde{O}_{\varphi,k}\left(n^{O(\varepsilon)} \cdot M\right)$ bits, where $1 \leq M < \sqrt{n}$
- query time $T = \widetilde{O}_{\varphi,k}\left(n^{1+O(\varepsilon)}/M\right)$.

- break the $\Omega(\sqrt{n})$ space barrier:
  - S can be far below $\sqrt{n}$
- space-time trade-off:
  - $S \cdot T \approx \widetilde{O}_{\varphi,k}(n^{1+O(\varepsilon)})$

Key Primitive: Dot Product Estimation

Key Idea

Sublinear spectral clustering oracles rely on estimating

$$\langle f_x, f_y \rangle = f_x^T f_y,$$

where f_x is the spectral embedding of vertex x.

Key Primitive: Dot Product Estimation

Key Idea

Sublinear spectral clustering oracles rely on estimating

$$\langle f_x, f_y \rangle = f_x^T f_y,$$

where f_x is the spectral embedding of vertex x.

- estamate $\langle f_x, f_y \rangle$ with
  - space S
  - time T
- clustering oracle uses
  - $\widetilde{O}(\mathrm{poly}(k) \cdot S)$ space
  - $\widetilde{O}(\mathrm{poly}(k) \cdot T)$ query time

Our Technique

Previous Approach

Reduce $\langle f_x, f_y \rangle$ to to computing collision probability $\langle M^t \mathbf{1}_x, M^t \mathbf{1}_y \rangle$ using

- run $\Omega(\sqrt{n})$ random walks $\Rightarrow \widetilde{\Omega}(\sqrt{n})$ time
- store all $\Omega(\sqrt{n})$ endpoints $\Rightarrow \Omega(\sqrt{n})$ space

Remark:

- M: random walk transition matrix

Our Technique

Previous Approach

Reduce $\langle f_x, f_y \rangle$ to to computing collision probability $\langle M^t \mathbf{1}_x, M^t \mathbf{1}_y \rangle$ using

- run $\Omega(\sqrt{n})$ random walks $\Rightarrow \widetilde{\Omega}(\sqrt{n})$ time
- store all $\Omega(\sqrt{n})$ endpoints $\Rightarrow \Omega(\sqrt{n})$ space

Remark:

- M: random walk transition matrix

Our Idea: Batch-Based Random Walks

- divide R walks into $\frac{R}{M}$ batches
- each batch stores only M endpoints
- average over batches

Our Technique

Previous Approach

Reduce $\langle f_x, f_y \rangle$ to to computing collision probability $\langle M^t \mathbf{1}_x, M^t \mathbf{1}_y \rangle$ using

- run $\Omega(\sqrt{n})$ random walks $\Rightarrow \widetilde{\Omega}(\sqrt{n})$ time
- store all $\Omega(\sqrt{n})$ endpoints $\Rightarrow \Omega(\sqrt{n})$ space

Remark:

- M: random walk transition matrix

Our Idea: Batch-Based Random Walks

- divide R walks into $\frac{R}{M}$ batches
- each batch stores only M endpoints
  - reuse M space across batches
- average over batches

Our Technique

Previous Approach

Reduce $\langle f_x, f_y \rangle$ to to computing collision probability $\langle M^t \mathbf{1}_x, M^t \mathbf{1}_y \rangle$ using

- run $\Omega(\sqrt{n})$ random walks $\Rightarrow \widetilde{\Omega}(\sqrt{n})$ time
- store all $\Omega(\sqrt{n})$ endpoints $\Rightarrow \Omega(\sqrt{n})$ space

Remark:

- M: random walk transition matrix

Our Idea: Batch-Based Random Walks

- divide R walks into $\frac{R}{M}$ batches
- each batch stores only M endpoints
  - reuse M space across batches
  - space-time trade-off: $M < \sqrt{n} \Rightarrow M \cdot R \approx n$
- average over batches

Additional Results

- 1-cluster vs. 2-cluster problem
  - upper bound: batch-based approach
  - lower bound:
    - ⋆ upper bound matches the lower bound up to $\mathrm{poly}(\log n)$ factors
    - ⋆ the space-time trade-off of our clustering oracle is essentially tight (at least for approaches based on collision probability estimation)
- experiments on SBM graphs validating the theory

Thanks!

References:

[Pen20] Peng P. Robust clustering oracle and local reconstructor of cluster structure of graphs. SODA 2020.

[GKL+21] Gluch G, Kapralov M, Lattanzi S, Mousavifar A and Sohler C. Spectral clustering oracles in sublinear time. SODA 2021.

[SP23] Shen R, Peng P. A sublinear-time spectral clustering oracle with improved preprocessing time. NeurIPS 2023.