Optimal Graph Reconstruction by Counting Connected Components in Induced Subgraphs

Conference on Learning Theory (**COLT**) 2025

ENCORE
UC San Diego

Hadley Black | Arya Mazumdar | Barna Saha | Yinzhan Xu




Graph Reconstruction (GR)

Provided *Unknown*

- Given **query access** to simple n-vertex m-edge graph $G(V, E)$, **recover** E exactly.

Early works studied

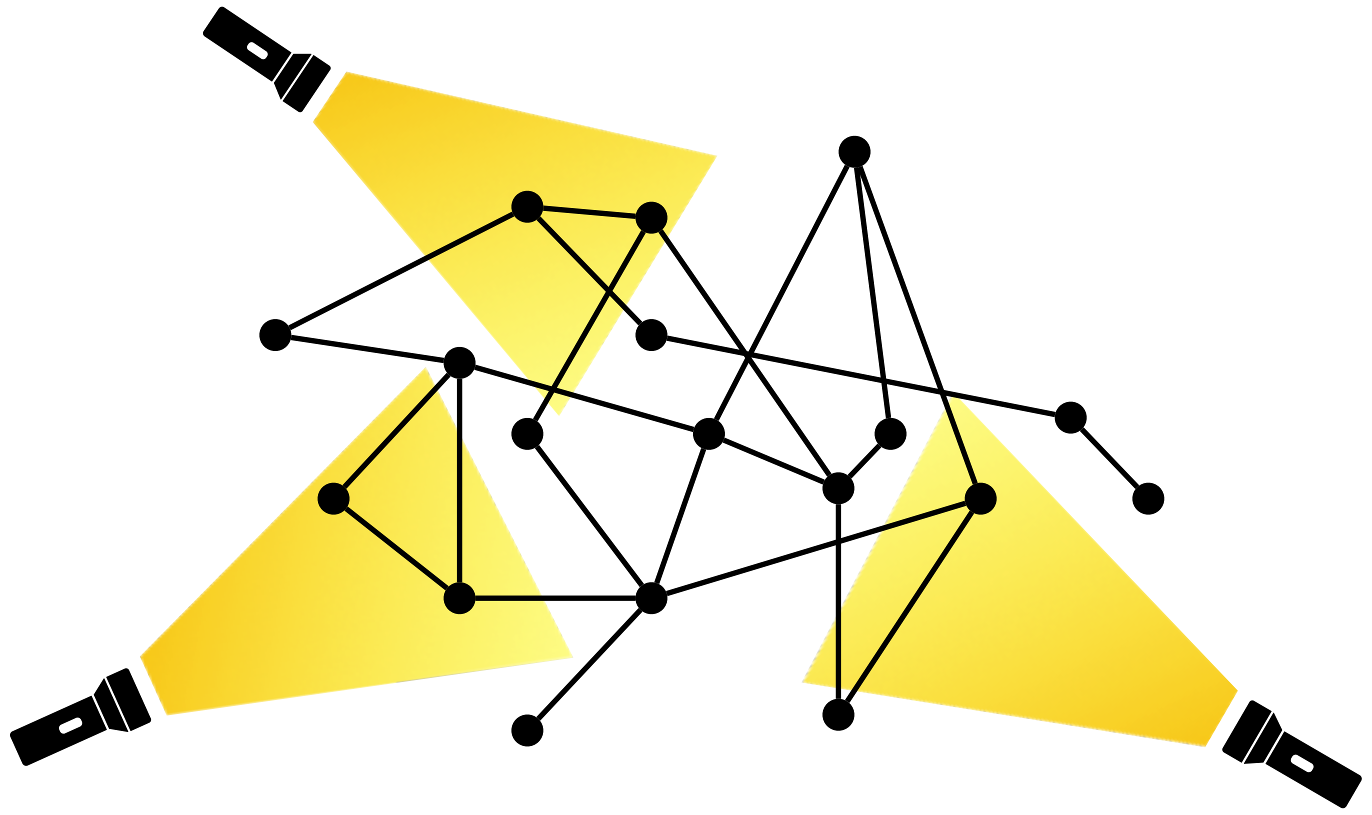
Independent-Set (IS) queries:

Does $G[S]$ contain an edge?

[GK98 ABKRS04, AA05, AC08, AB19]

Question

How many views to reconstruct a graph?



Motivations:

- **Genome mapping:** can be used to model procedures for physical mapping of DNA molecules [GK98, AA05]
- Basic **combinatorial search** question related to coin-weighing, group testing, etc.

GR History

Provided *Unknown*

- Given **query access** to simple n-vertex m-edge graph $G(V, E)$, **recover** E exactly.

Many ways to strengthen IS queries

Independent-Set (IS) queries

Does $G[S]$ contain an edge? $\Theta(m \log n)$

[GK98 ABKRS04, AA05, AC08, AB19]

Additive (ADD) queries

How many edges in $G[S]$?

Grebinski98, GK00, RS07, CK10, Mazzawi10, CJK11, Choi13]

$$\Theta\left(\frac{m \log(n^2/m)}{\log m}\right)$$

Maximal IS queries

Oracle returns a maximal IS in $G[S]$

[KOT25]

More recent

Distance Queries

What is distance from x to y in G?

[KKU95, BEE+06, EHHM06, MZ13, KMZ18, MZ21, RLYW21, BG23]

Classic open question

Connected Component (CC) Queries

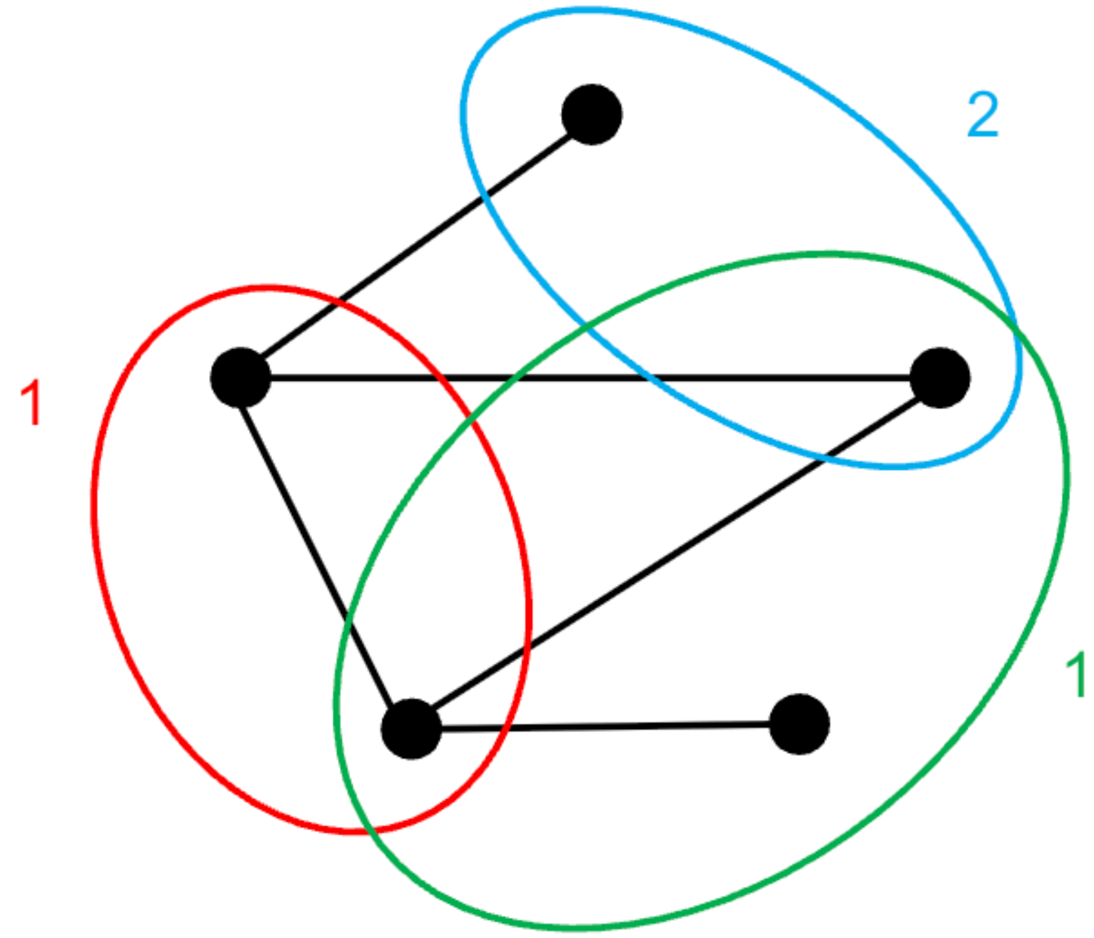
How many CCs in $G[S]$?

This work

Connected Component Queries

- Given **query access** to simple n-vertex m-edge graph $G(V, E)$, **recover** E exactly.

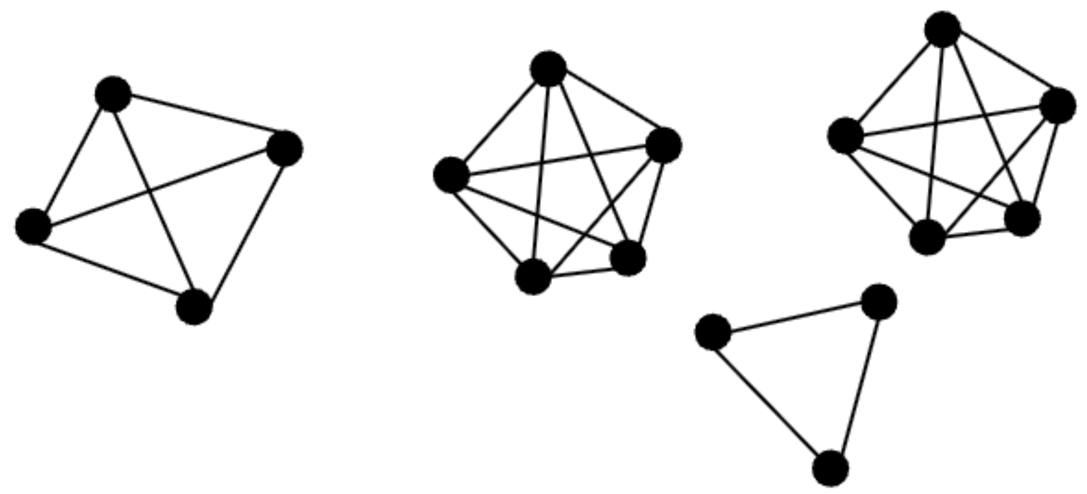
We introduce **<u>CC Queries:</u>** How many CCs in $G[S]$?



Motivations:

- CC count is a natural basic graph parameter
- Another natural way to strengthen IS queries
- CC counts are easy to compute in certain models (e.g., Congested-Clique [GP 16])
- Generalizes partition learning with subset queries

[CL 24, BLMS 24, BMS 25]



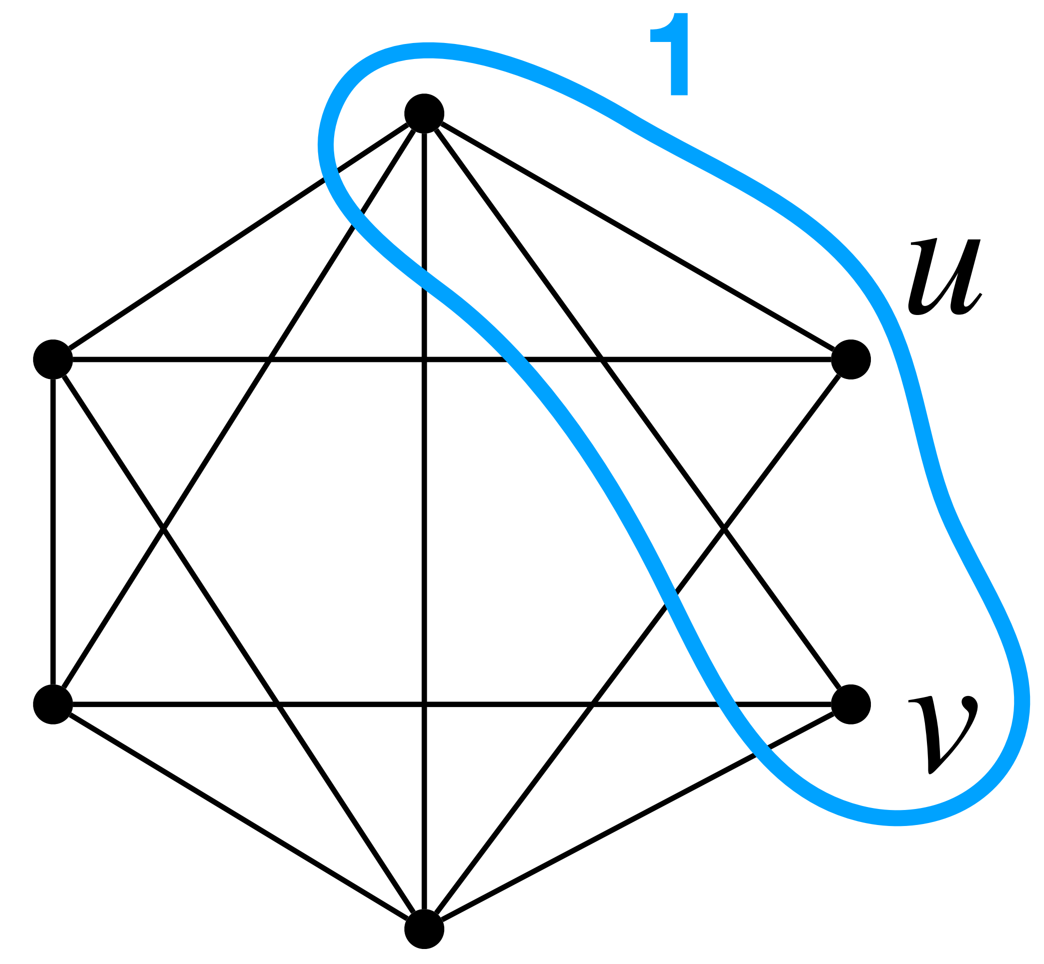
Basic Bounds

- Trivial $O(n^2)$ algorithm: query every pair $(u, v) \in \binom{V}{2}$

- $\Omega(n^2)$ lower bound: $K_n \backslash \{(u, v)\}$ $\implies$ Need to parametrize by m

    - Any query on more than 2 vertices always returns 1 (no information)

    - Querying pairs: finding missing edge is an unstructured search problem of size $\Omega(n^2)$



- $\Omega\left(\frac{m \log n}{\log m}\right)$ lower bound:

$$\binom{n(n-1)/2}{m} = 2^{\Omega(m \log n)} \text{ graphs (for } m \ll n\text{)}$$

CC's in $G[S]$ between $|S| - m$ and $|S|$ $\implies$ $O(\log m)$ bits per query

Results

CC Queries: How many CCs in $G[S]$?

Adaptive algorithm

$$\Theta\left(\frac{m \log n}{\log m}\right)$$

Non-adaptive lower bound

$\Omega(n^2)$ even when $m = O(n)$

Comparison with additive queries

$$\Theta\left(\frac{m \log(n^2/m)}{\log m}\right)$$

Slightly better for very dense graphs

*There is a **non-adaptive** algorithm that attains this bound*

[CK10, BM11, BM15]

Two-round algorithm

$O(m \log n + n \log^2 n)$

1) $O(n \log^2 n)$ queries to **approximate degrees**

2) $O(d(u) \cdot \log n)$ queries to **recover the neighbor of** u

- Using CC queries to simulate a group testing primitive

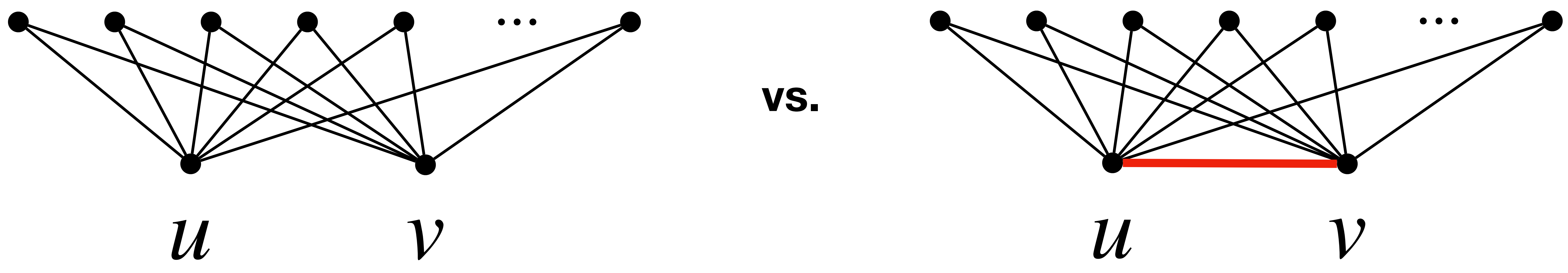
Non-Adaptive Lower Bound

Non-adaptive lower bound

$\Omega(n^2)$ even when $m = O(n)$

- For each $(u, v) \in \binom{V}{2}$, define:

$K_{2,n-2}$ vs. $K_{2,n-2} \cup \{(u, v)\}$



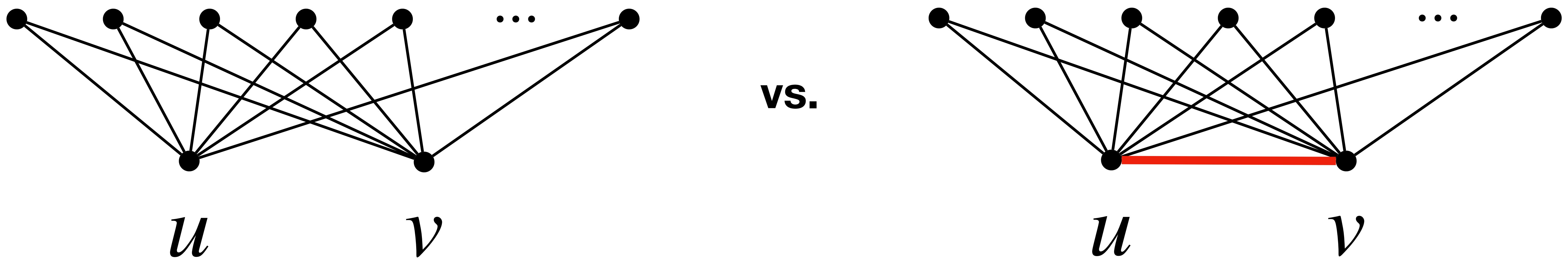
- To distinguish, must query some S containing both u, v

  … but any query larger than 2 containing u, v returns "1 CC" in both cases

  … so only queries of size 2 are useful for a non-adaptive algorithm

$\implies$ need $\Omega(n^2)$ queries to distinguish every such pair of graphs

Why Adaptivity Helps



First, learn structural information about the graph to inform later queries

Observation:
CC's in $G[S]$ < # CC's in $G[S \cup \{u\}]$ iff $N(u) \cap S = \varnothing$

Using this we can easily distinguish high vs. low degree vertices

Then query the edge between the two high-degree vertices

Technique 1: vertices with similar degree

Observation:
If H is a **forest**, then
# edges in $H[S] = |S| -$ # CC's in $H[S]$

Additive queries and CC queries are **equivalent** on forests

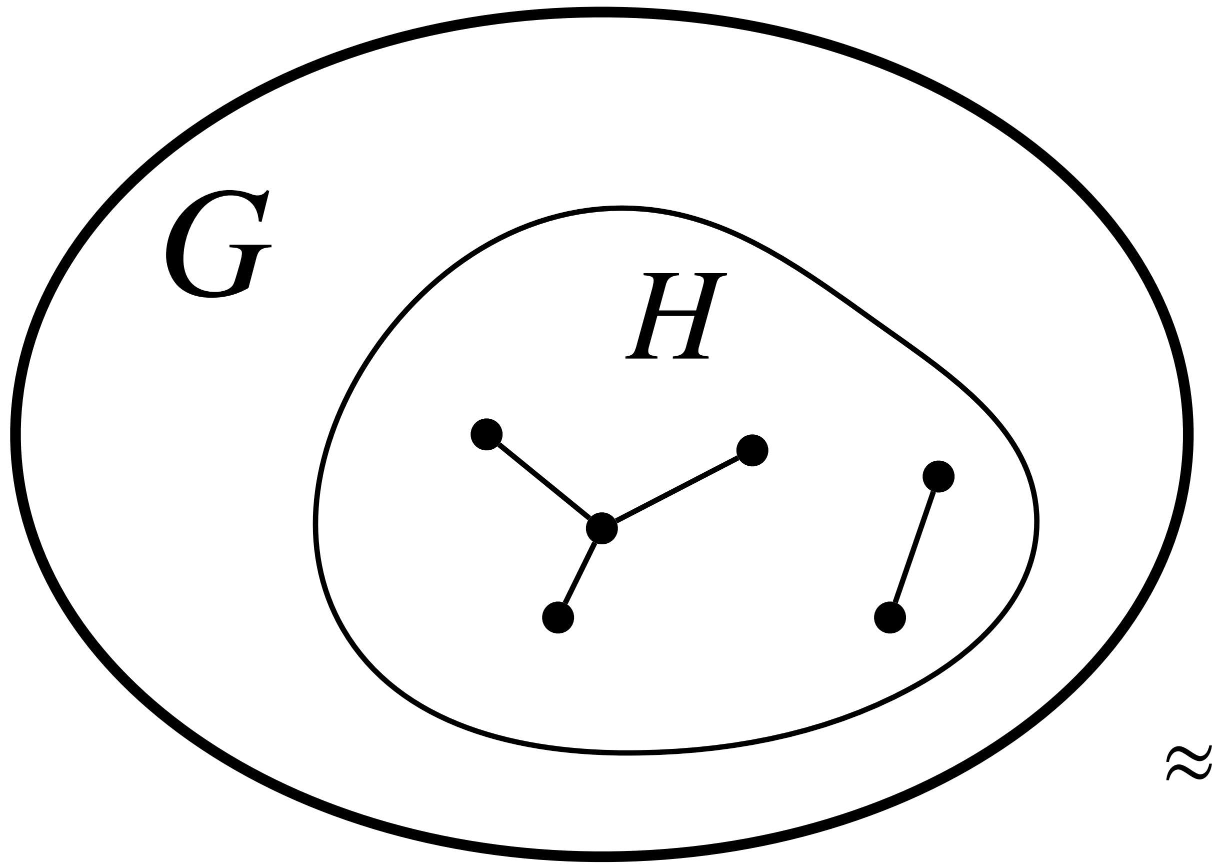
- Assume all vertices have degree $O(D)$ where $D = m/n$

Note: target query complexity is $O(m)$

$\Longrightarrow$ random subgraph H with sample rate $p = O((mD)^{-1/3})$ is a forest with probability $\Omega(1)$

$\Longrightarrow$ simulate ADD-query algorithm in H: $\approx \dfrac{p^2 m \log(pn)}{\log(p^2 m)} \approx p^2 m$

We recover $\approx p^2 m$ edges using $\approx p^2 m$ queries in expectation



$\approx pn$ vertices

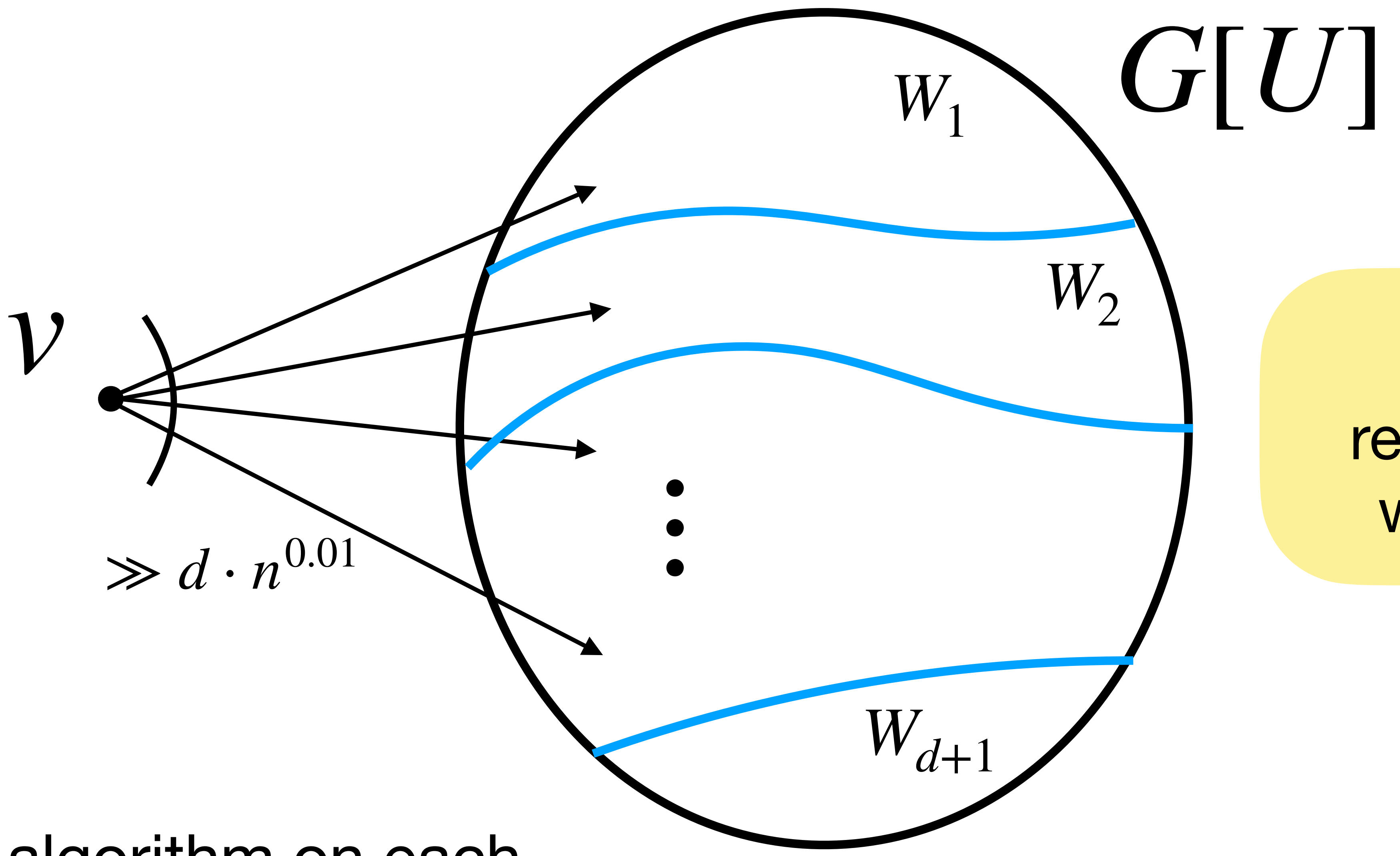
$\approx p^2 m$ edges

Technique 2: vertices with *dissimilar* degree

- Let v be a vertex with degree $D \gg d \cdot n^{0.01}$
- Suppose we have recovered subgraph $G[U]$ with max degree d

$\implies$ Can partition $U = W_1 \sqcup \cdots \sqcup W_{d+1}$ into $d+1$ independent sets

$\implies$ $G[W_i \cup \{v\}]$ is a **forest**: can simulate ADD-query algorithm on each



Goal

recover $G[U \cup \{v\}]$ with $O(D)$ queries

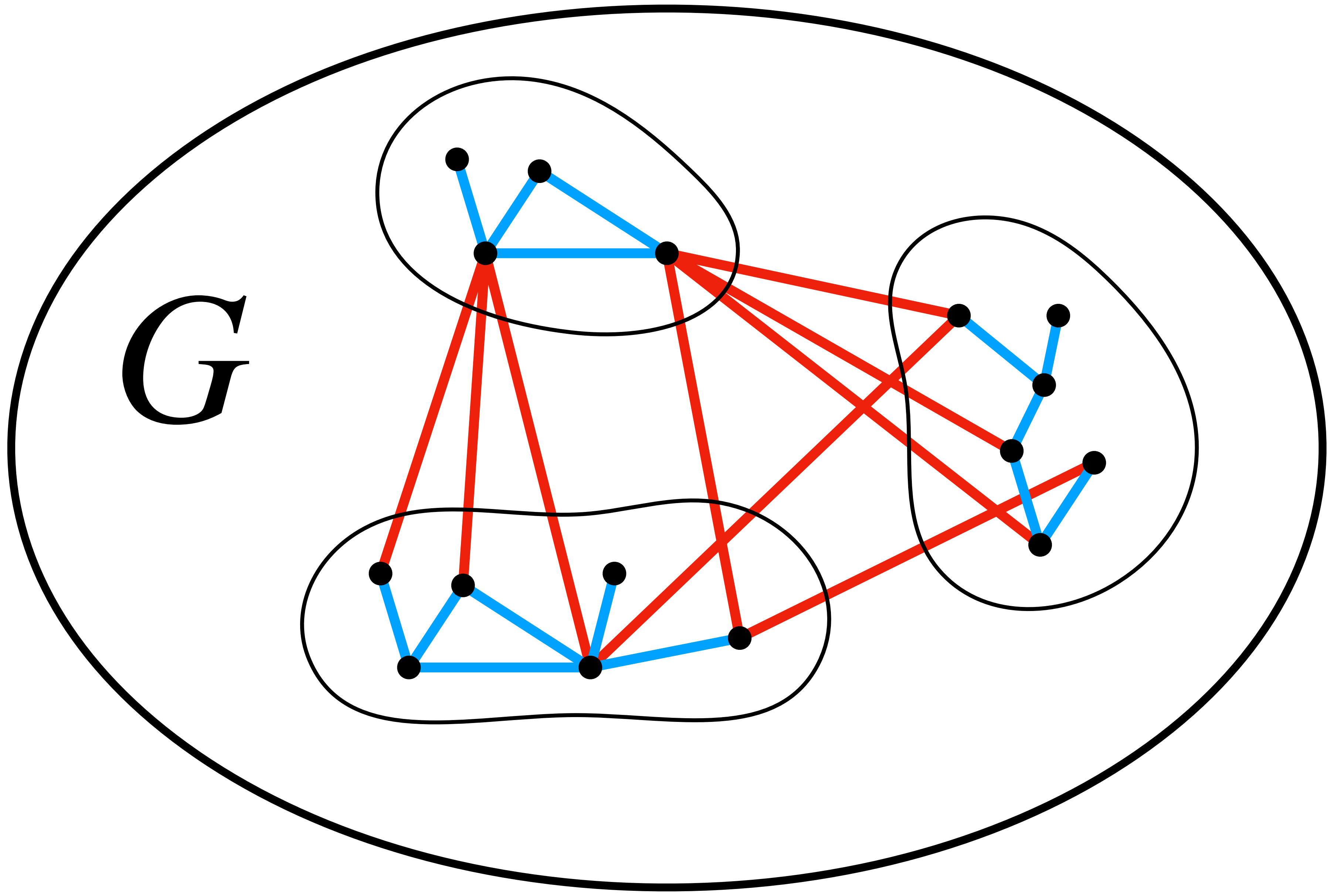
Total query complexity

$$O\left(\log n \cdot \sum_{i=1}^{d+1} \frac{\deg(v, W_i)}{\log \deg(v, W_i)}\right) \leq O\left((d+1)\log n \cdot \frac{D/(d+1)}{\log(D/(d+1))}\right) \leq O\left(\log n \cdot \frac{D}{\log(n^{0.01})}\right) \leq O(D)$$

Jensen's

Adaptive Algorithm

- Carefully choose thresholds which partition vertices by degree $V = V_1, \ldots, V_\ell$
- Use **technique 1** to learn $G[V_i]$ and $G[V_i, V_{i+1}]$ *(similar degree)*
- Use **technique 2** to learn $G[V_i, V_j]$ for $j > i + 1$ *(dissimilar degree)*



Note

This is not the whole story, as we are not provided the degree of vertices

Poses significant other challenges

Conclusion

- We propose a new query model for the classic graph reconstruction problem
    - Obtain tight bounds for adaptive algorithms
    - Show separation from well-studied additive model in terms of adaptivity

<u>Questions</u>

What is the round complexity of GR with CC queries?

Are CC queries interesting for other graph problems?

How many CC queries to count edges? Is this easier than reconstruction?