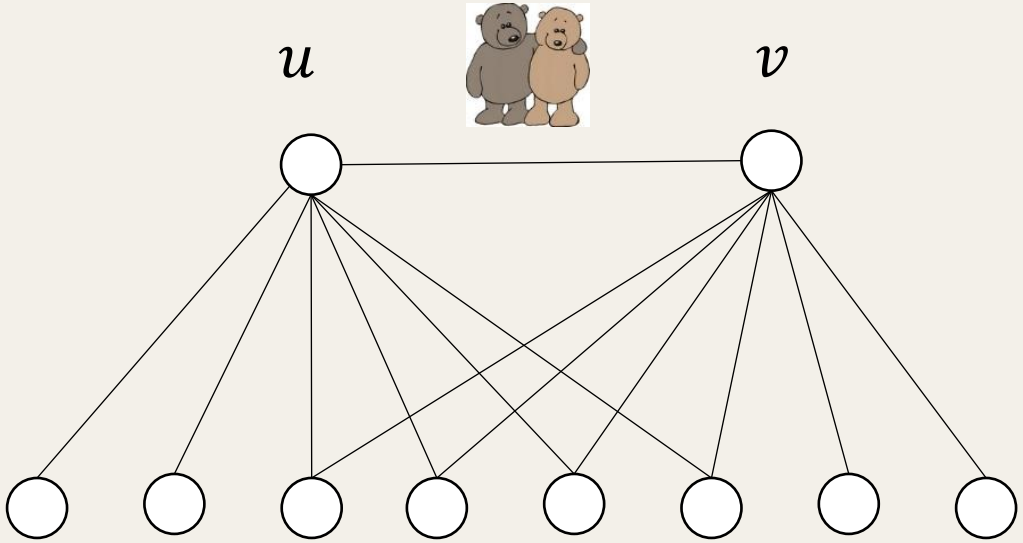
# Neighborhood Similarity: A New Application and Technique

Hsin-Hao Su
Boston College

**Workshop on Foundations of Distributed and Parallel Graph Algorithms 2026**

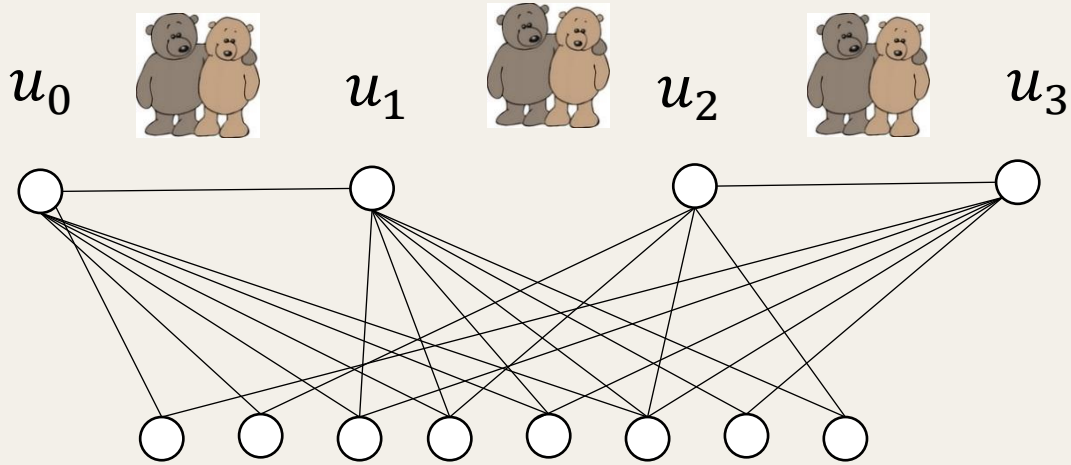
# Neighborhood Similarity

- Vertices *u* and *v* are *friends* if they have many common neighbors



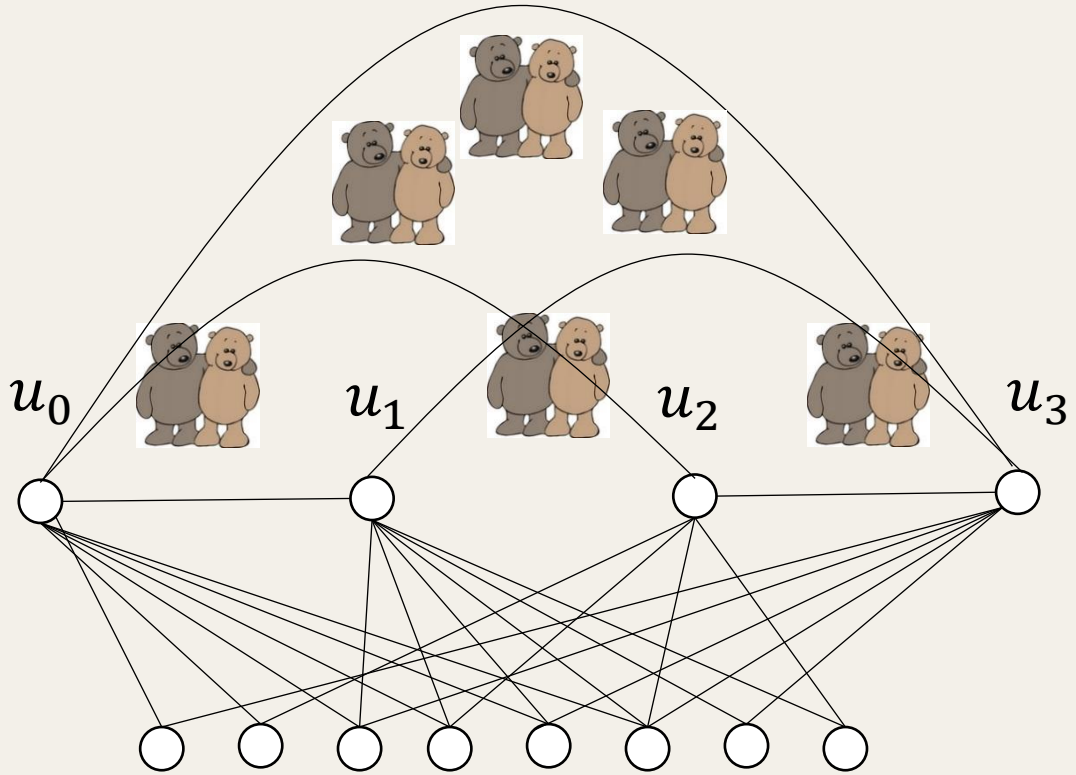
# Neighborhood Similarity

+ Transitivity of friends (on vertices with many friends).



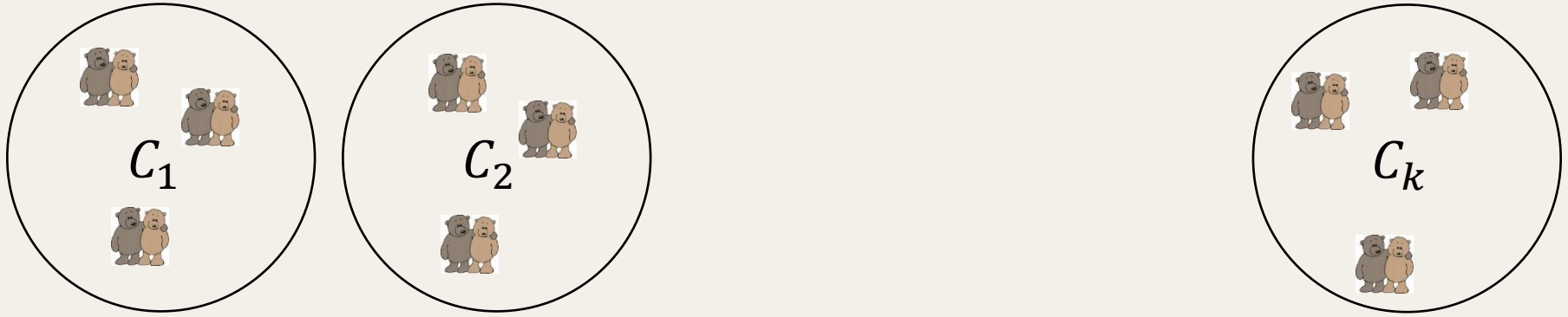
# Neighborhood Similarity

- Transitivity of friends (on vertices with many friends).



# Neighborhood Similarity

- This naturally defines a decomposition:
  - *Almost-clique decomposition, sparse-dense decomposition, preclustering (in correlation clustering contexts).*




Vertices with few friends

# Neighborhood Similarity

- Bounds on the Chromatic number: [Reed '98]
- Coloring in distributed, parallel, and semi-streaming settings: [Harris-Schneider-Su '18, Chang-Li-Pettie '20, Halldorsson-Kuhn-Maus-Nolin '20, Halldorsson-Kuhn-Maus-Tonoyan '21 '22, Parter-Su '18, Czumaj-Davies-Parter '20, Alon-Assadi '20, Assadi-Chen-Khanna '19, Chang-Fischer-Ghaffari-Uitto-Zheng '19, …, Flin-Halldorsson-Jakob-Maus '26]

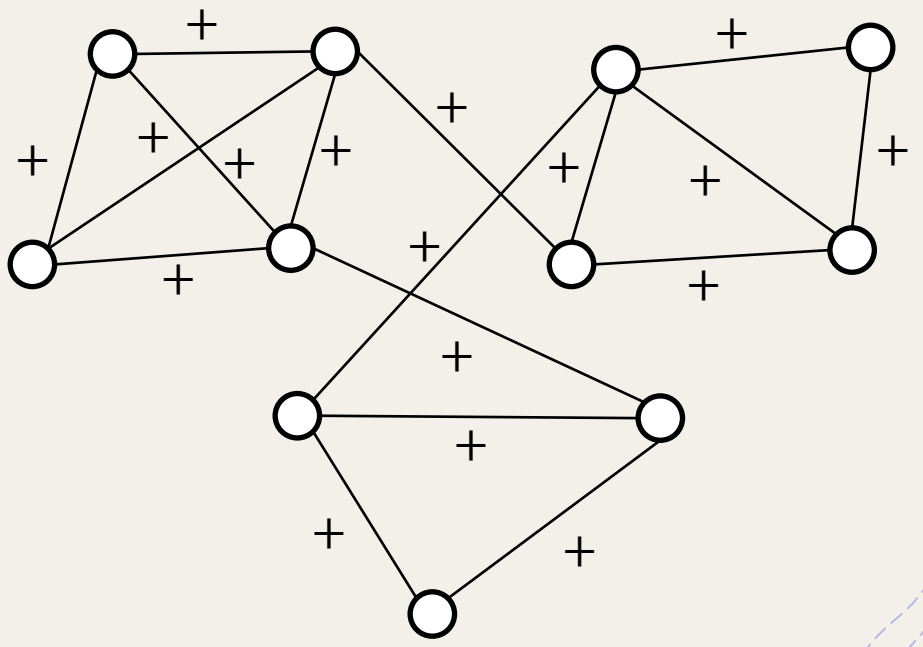
- $l_1$-correlation clustering: [Cohen-Addad-Lattanzi-Mitrovic-Norouzi-Frad-Parotsidis-Tarnawski '21, Assadi-Wang '22, Cohen-Addad-Lee-Li-Newman '23, Cao-Cohen-Addad-Lee-Li-Newman-Vogl '24, Cao-Cohen-Addad-Lee-Li-Lolck-Newman-Thorup, Vogl-Yan-Zhang '25]

# In This Talk

- New application:
  - 3-approximation for $l_\infty$-correlation clustering
- New technique:
  - How to compute neighborhood similarity efficiently.
- Based on result: “Min-Max Correlation Clustering via Neighborhood Similarity”, ESA 2025, with Nairen Cao and Steven Roche

# Correlation Clustering

- [Bansal, Blum, Chawla 04’]
- $G = (V, E^+ \cup E^-)$, with $E^+ \cup E^- = \binom{V}{2}$

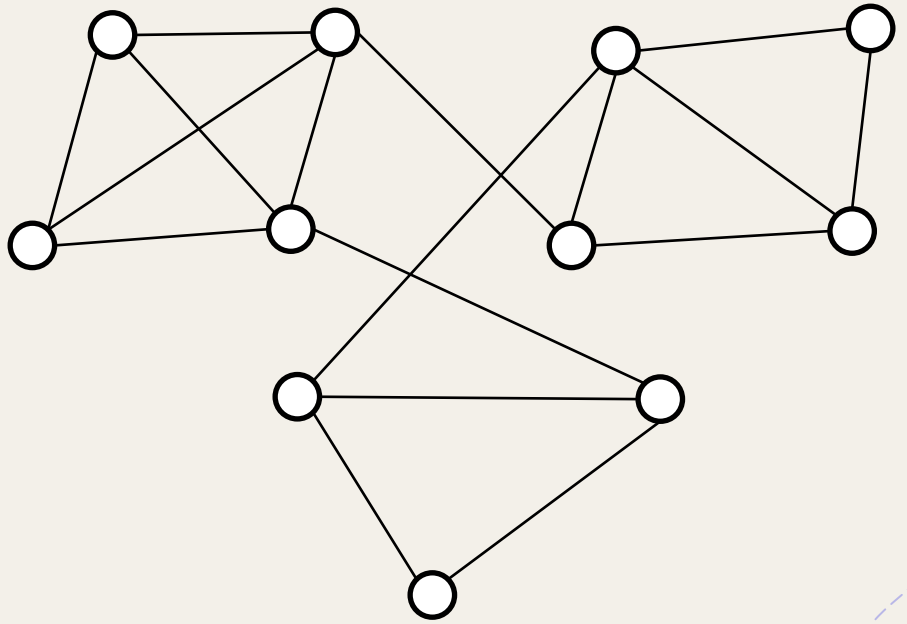


Solid edges are positive edges

Negative edges are not shown

# Correlation Clustering

- [Bansal, Blum, Chawla 04’]
- $G = (V, E^+ \cup E^-)$, with $E^+ \cup E^- = \binom{V}{2}$

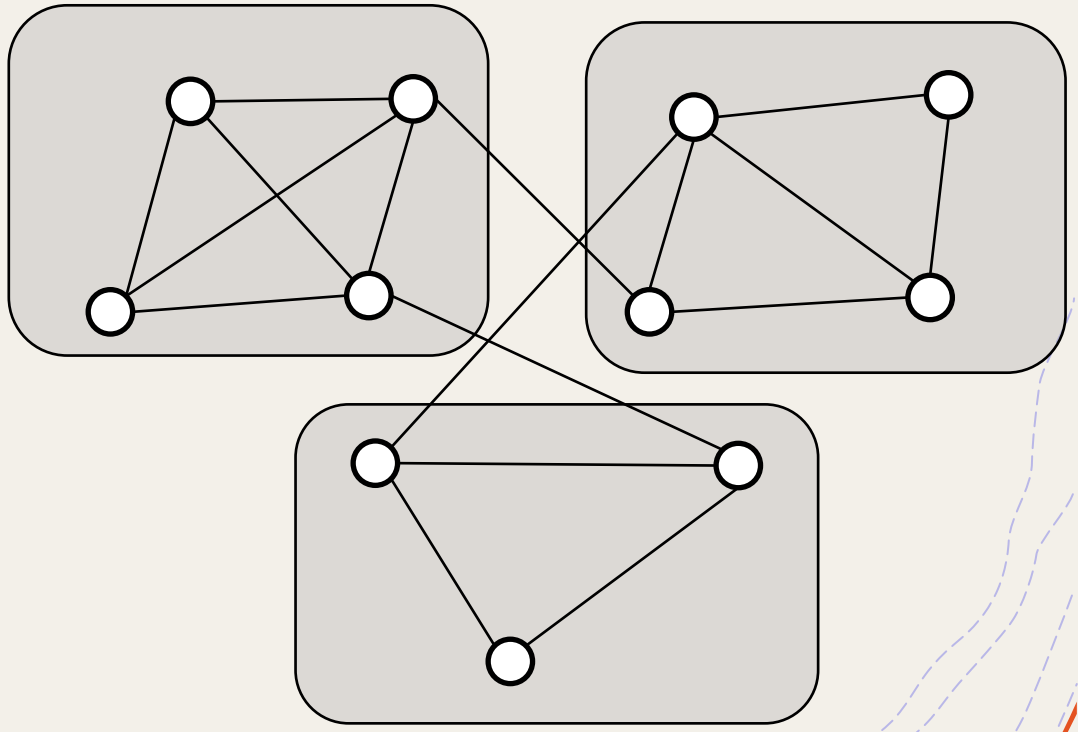


Solid edges are positive edges

Negative edges are not shown

# Correlation Clustering

- [Bansal, Blum, Chawla 04']
- $G = (V, E^+ \cup E^-)$, with $E^+ \cup E^- = \binom{V}{2}$
- $(u, v)$ is in a disagreement if:
  - $(u, v) \in E^-$ and they are in the same cluster
  - $(u, v) \in E^+$ and they are in different clusters

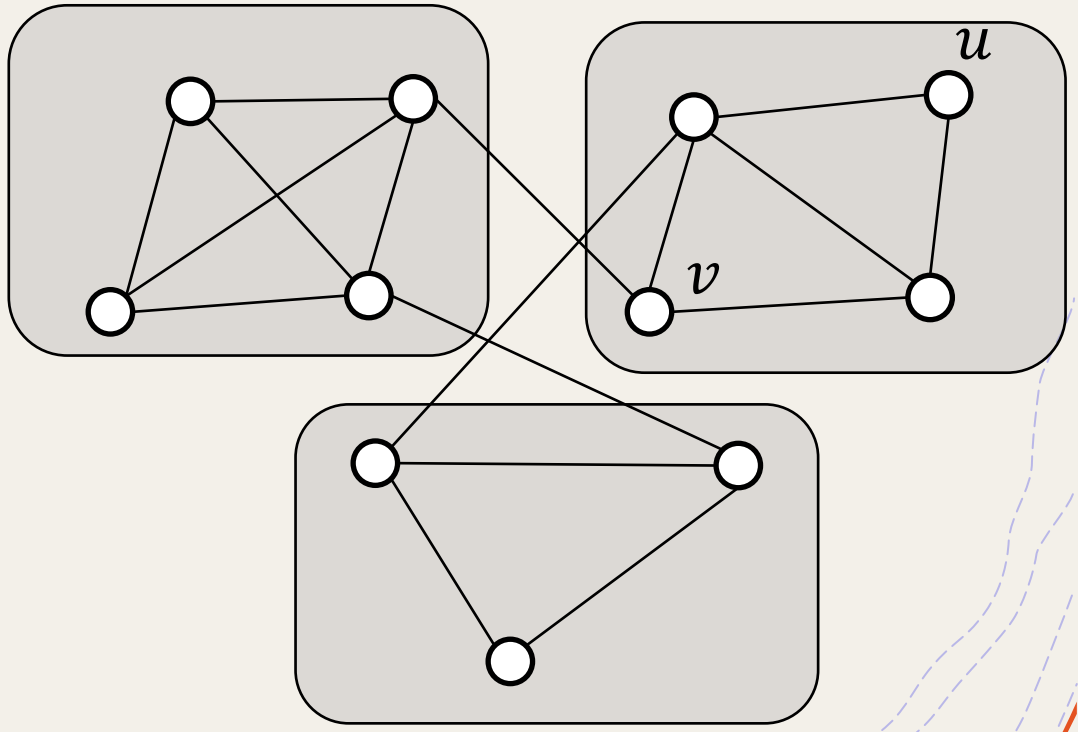


Solid edges are positive edges

Negative edges are not shown

# Correlation Clustering

- [Bansal, Blum, Chawla 04']
- $G = (V, E^+ \cup E^-)$, with $E^+ \cup E^- = \binom{V}{2}$
- $(u, v)$ is in a disagreement if:
  - $(u, v) \in E^-$ and they are in the same cluster
  - $(u, v) \in E^+$ and they are in different clusters

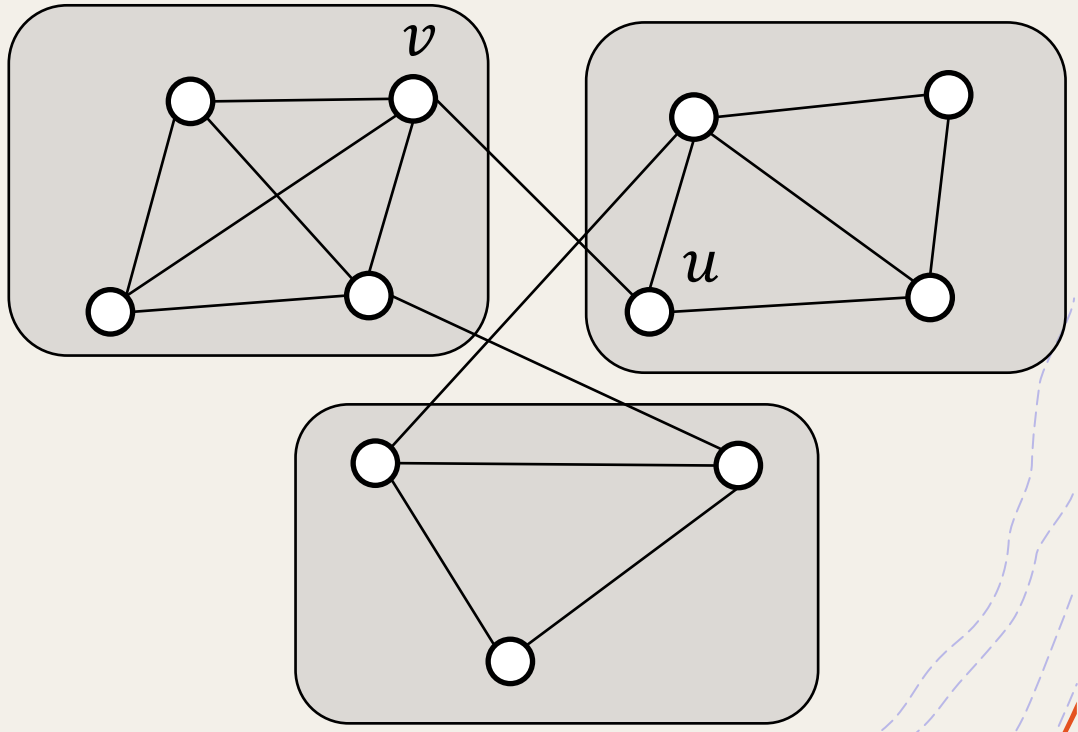



Solid edges are positive edges

Negative edges are not shown

# Correlation Clustering

- [Bansal, Blum, Chawla 04']
- $G = (V, E^+ \cup E^-)$, with $E^+ \cup E^- = \binom{V}{2}$
- $(u, v)$ is in a disagreement if:
  - $(u, v) \in E^-$ and they are in the same cluster
  - $(u, v) \in E^+$ and they are in different clusters




Solid edges are positive edges

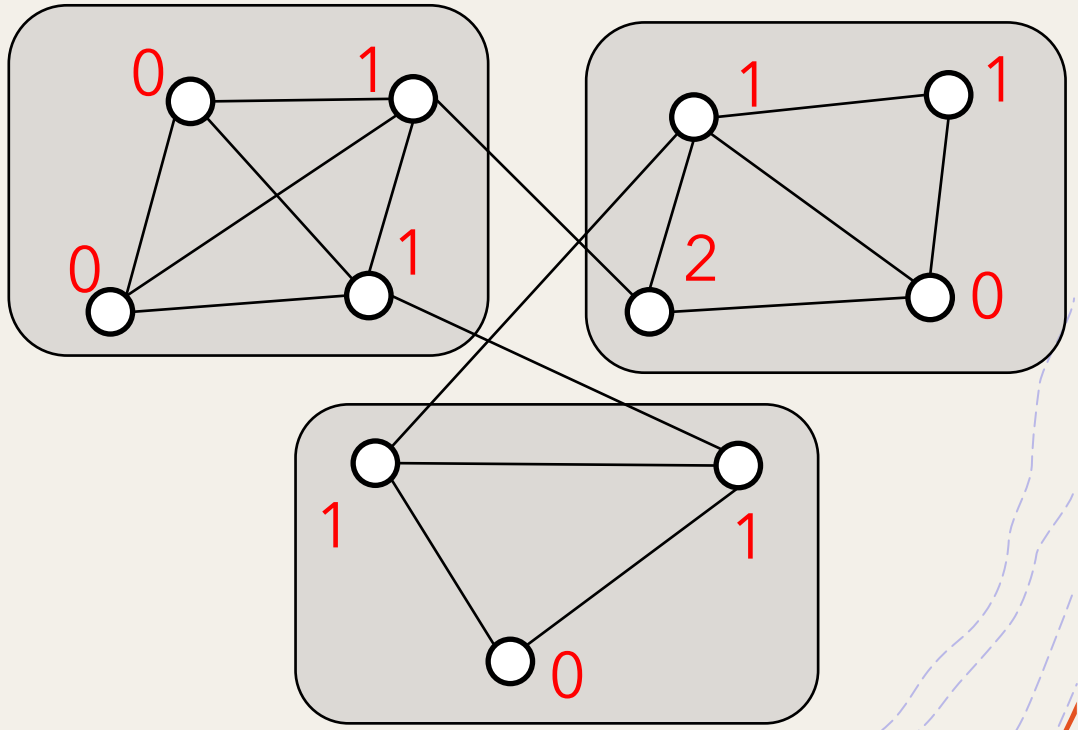
Negative edges are not shown

# Correlation Clustering

- disagreement vector: $(z_{v_1} \dots z_{v_n})$
  - $z_{v_i}$ denotes the number of incident edges in a disagreement to $v_i$

- $\ell_p$-correlation clustering:
  - Find a clustering that minimizes

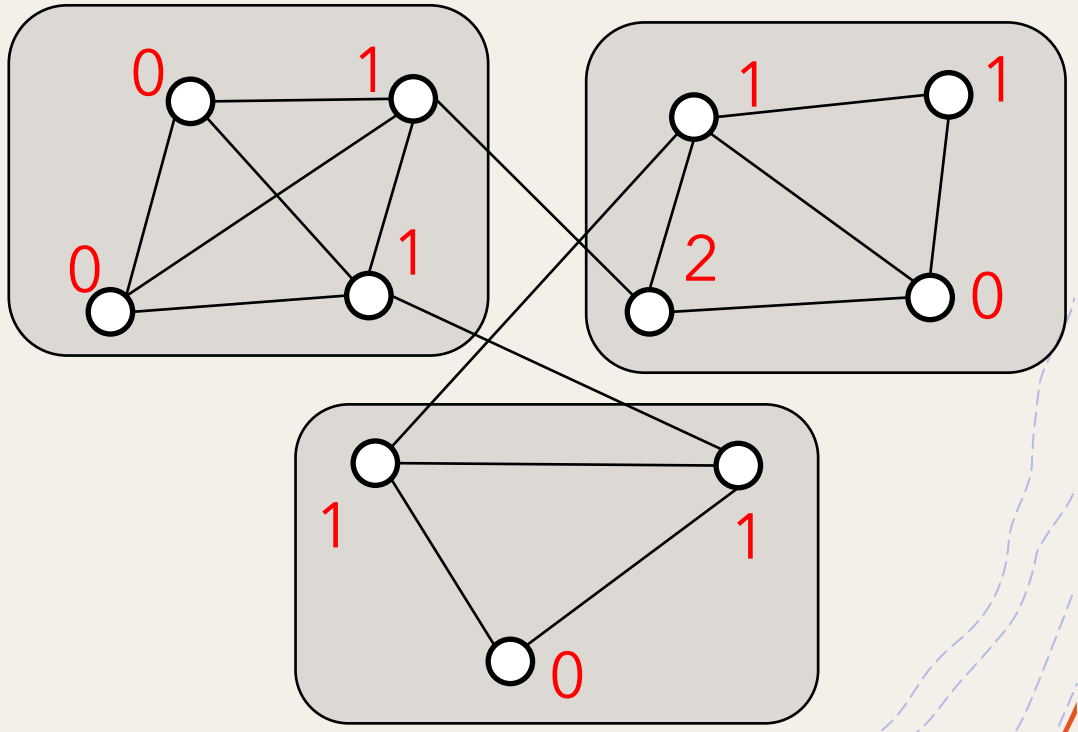
$$\left(\sum_{i=1}^{n} \left|\left|z_{v_i}\right|\right|^p\right)^{1/p}$$



# Correlation Clustering

- $\ell_1$-correlation clustering:
  [Bansal, Blum, Chawla, '04]
  - minimize the total number of disagreement

- $\ell_\infty$-correlation clustering:
  [Puleo and Milenkovic '16]
  - minimize the maximum incident disagreements

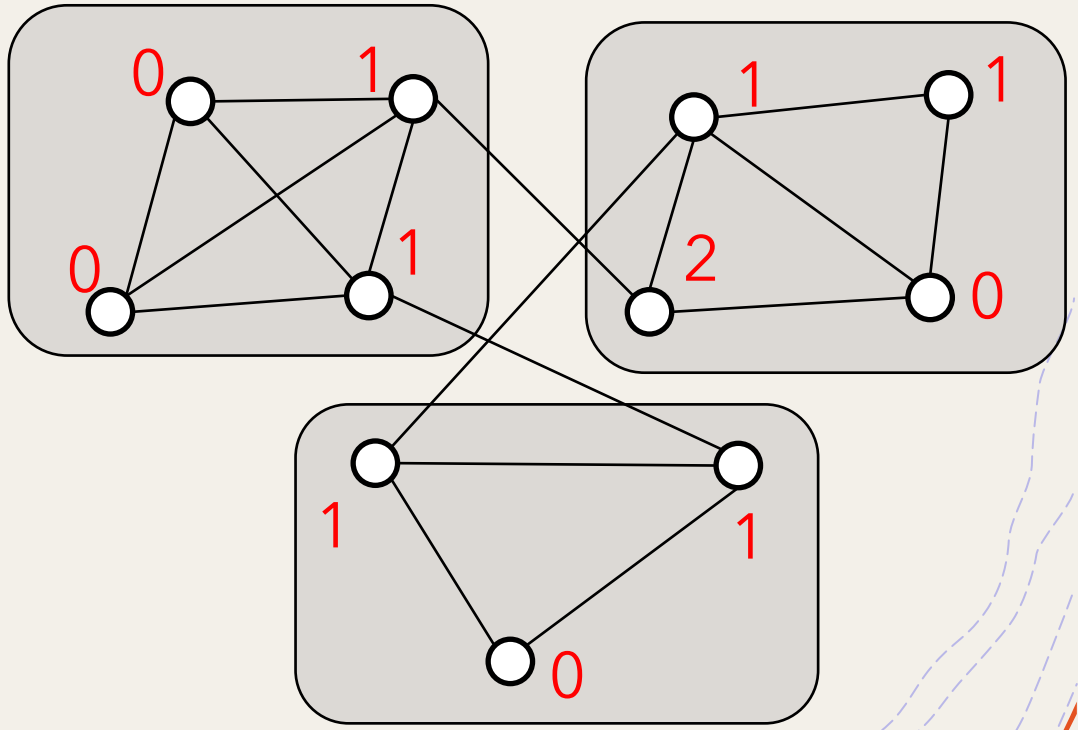


# Correlation Clustering

- $\ell_1$-correlation clustering:
  [Bansal, Blum, Chawla, '04]
  - minimize the total number of disagreement

- $\ell_\infty$-correlation clustering:
  [Puleo and Milenkovic '16]
  - minimize the maximum incident disagreements

    NP-Hard



# Previous results for $\ell_1$

| References | Approx. Ratio | Notes |
|---|---|---|
| [Bansal, Blum, Chawla, '04] | $\geq 149$ | LP-based |
| [Charikar, Guruswami, Wirth '05] | 4 | LP-based |
| [Ailon, Charikar, Newman '08] | 3 | PIVOT algorithm |
| [Ailon, Charikar, Newman '08] | 2.5 | LP-based |
| [Chawla, Makarychev, Schramm, Yaroslavtev '15] | 2.06 | LP-based |
| [Cohen-Addad, Lee, Newman '22] | 1.994 | LP-based |
| [Cohen-Addad, Lee, Li, Newman '23] | 1.73 | LP-based |
| [Cao, Cohen-Addad, Lee, Li, Newman, Vogl '24] | 1.437 | LP-based |
| [Cao, Cohen-Addad, Lee, Li, Lolck, Newman, Thorup, Vogl, Yan, Zhang '25] | 1.437 | Sublinear Time |

# Previous results for $\ell_\infty$

| References | Approx. Ratio | Running Time | Notes |
|---|---|---|---|
| [Puleo and Milenkovic '16] | 48 | higher polynomial | LP-based |
| [Charikar, Gupta, Schwartz '17] | 7 | higher polynomial | LP-based |
| [Kalhan, Makarychev, Zhou '19] | 5 | higher polynomial | LP-based |
| [Davies, Moseley, Newman '23] | 40 | $O(n^2 \log n)$ | combinatorial |
| [Heidrich, Irmai, Andres '24] | 4 | $O(n^2 + n\Delta^2)$ | combinatorial |

# Previous results for $\ell_\infty$

| References | Approx. Ratio | Running Time | Notes |
|---|---|---|---|
| [Puleo and Milenkovic '16] | 48 | higher polynomial | LP-based |
| [Charikar, Gupta, Schwartz '17] | 7 | higher polynomial | LP-based |
| [Kalhan, Makarychev, Zhou '19] | 5 | higher polynomial | LP-based |
| [Davies, Moseley, Newman '23] | 40 | $O(n^2 \log n)$ | combinatorial |
| [Heidrich, Irmai, Andres '24] | 4 | $O(n^2 + n\Delta^2)$ | combinatorial |
| Our result | $3 + \epsilon$ | $\tilde{O}(m/\epsilon^2)$ | combinatorial |

$$m = |\mathrm{E}^+|$$

# Our Results for Other Models

- Massively Parallel Computation (MPC) Model
  - An $O(1)$-round $(3+\epsilon)$-approximation algorithm using $O(n^{\delta})$ memory per machine for any constants $0 < \delta < 1$.
  - Previously, the only known MPC result for this problem is a 63.3-approximation algorithm for all $\ell_p$ by [Cao, Ye, and Li '25].

- Semi-Streaming Model
  - A single-pass algorithm $(3+\epsilon)$-approximation algorithm that uses $O(n \log n / \epsilon^2)$ space.

# Technical Overview

1. Technical Overview: Approximation Ratio

2. Technical Overview: Running Time

# Overview: Approximation Ratio

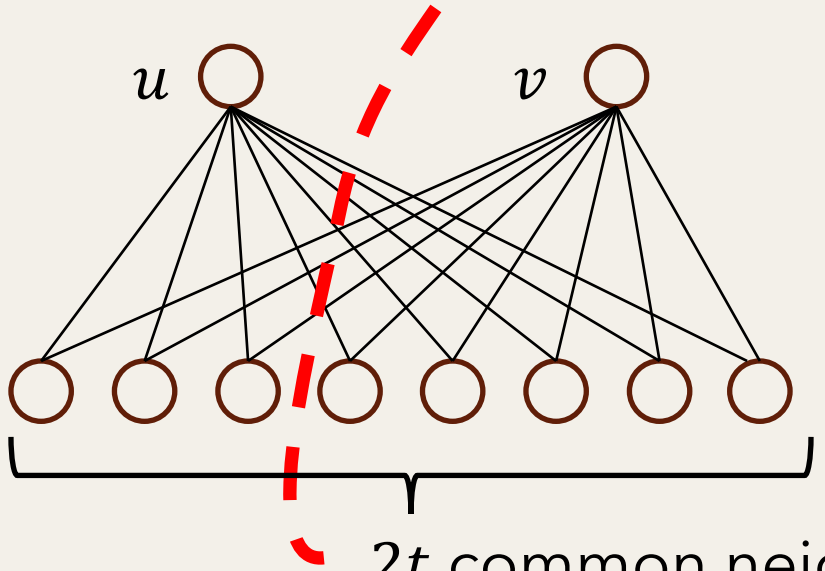
+ [Heidrich, Irmai, Andres ‘24]:

1. If $|N[u] \cap N[v]| > 2OPT$
   + $u$ and $v$ must be together in the optimal solution

# Overview: Approximation Ratio

+ [Heidrich, Irmai, Andres '24]:

1. If $|N[u] \cap N[v]| > 2OPT$
   + $u$ and $v$ must be together in the optimal solution




Either u or v has disagreement $\geq t$

# Overview: Approximation Ratio

- [Heidrich, Irmai, Andres ‘24]:

1. If $|N[u] \cap N[v]| > 2OPT$
   - $u$ and $v$ must be together in the optimal solution

2. If $|N[u] \oplus N[v]| > 2OPT$
   - $u$ and $v$ must be apart in the optimal solution

# Overview: Approximation Ratio

- [Heidrich, Irmai, Andres ‘24]:

1. If $|N[u] \cap N[v]| > 2OPT$
   - $u$ and $v$ must be together in the optimal solution

2. If $|N[u] \oplus N[v]| > 2OPT$
   - $u$ and $v$ must be apart in the optimal solution

- They used this to develop a 4-approximation algorithm

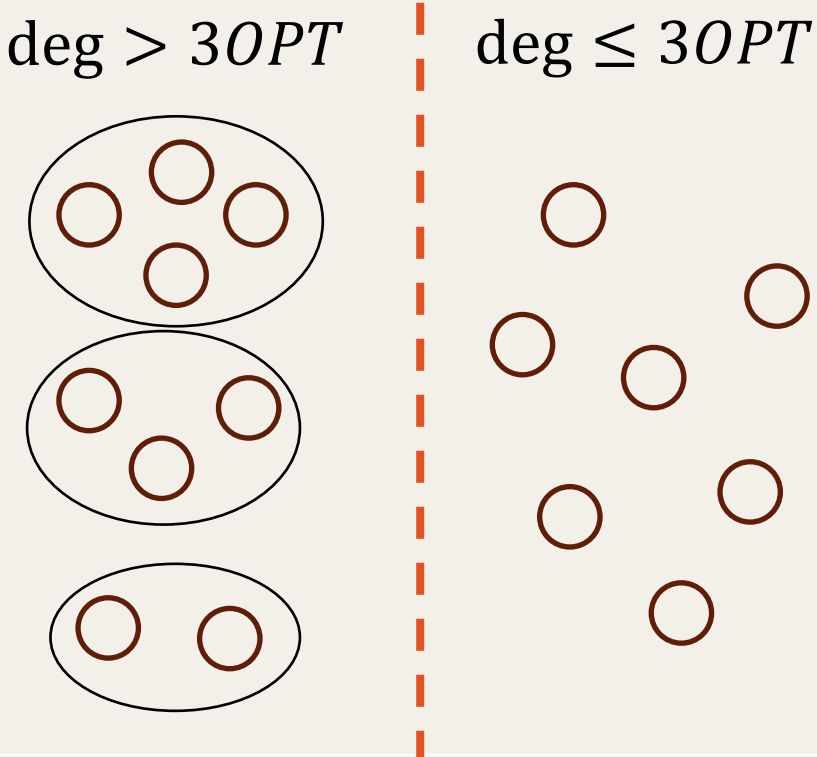
# Overview: Approximation Ratio

- To get a 3-approximation algorithm, note that if **both** u and v have *high-degree* (deg > 3OPT) then either:
  - $|N[u] \cap N[v]| > 2OPT$
  - $|N[u] \oplus N[v]| > 2OPT$

# Overview: Approximation Ratio

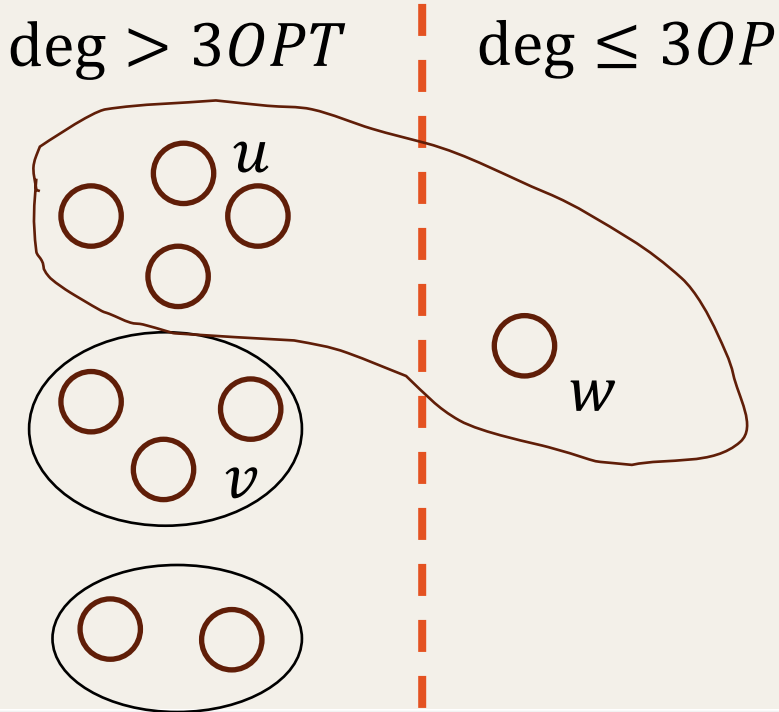
- We can use this to cluster high-degree vertices.

- It remains to determine where to put the low-degree vertices



# Overview: Approximation Ratio

- Key structural result: If a low-degree vertex w and a high-degree vertex u are in the same cluster in an optimal solution
  - $|N[w] \oplus N[u]| \leq 2OPT$
  - No high-degree vertex v from other cluster may have $|N[v] \oplus N[w]| \leq 2OPT$



# Overview: Approximation Ratio

- Key structural result: If a low-degree vertex w and a high-degree vertex u are in the same cluster in an optimal solution
  - $|N[w] \oplus N[u]| \leq 2OPT$
  - No high-degree vertex v from other cluster may have $|N[v] \oplus N[w]| \leq 2OPT$

- If we add all low-degree vertices $w$ with $|N[w] \oplus N[u]| \leq 2OPT$ to the cluster of $u$ and put remaining low-degree vertices as singleton clusters,
  - The min-max objective is guaranteed to be at most $3OPT$

# Overview: Approximation Ratio

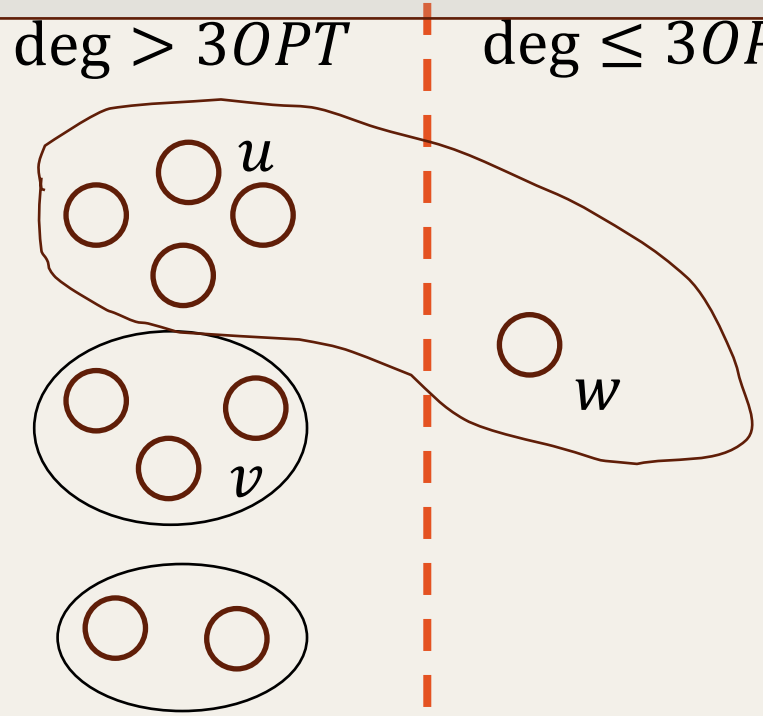
- Key structural result: If a low-degree vertex w and a high-degree vertex u are in the same cluster in an optimal solution
  - $|N[w] \oplus N[u]| \leq 2OPT$
  - No high-degree vertex v from other cluster may have $|N[v] \oplus N[w]| \leq 2OPT$

- If we add all low-degree vertices $w$ with $|N[w] \oplus N[u]| \leq 2OPT$ to the cluster of $u$ and put remaining low-degree vertices as singleton clusters,
  - The min-max objective is guaranteed to be at most $3OPT$

- This gives a polynomial-time 3-approximation algorithm

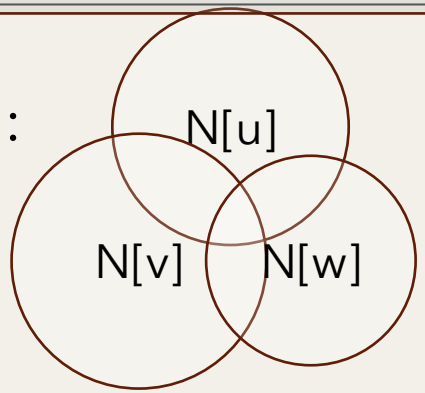
# Overview: Approximation Ratio

- Key structural result: If a low-degree vertex w and a high-degree vertex u are in the same cluster in an optimal solution
  - $|N[w] \oplus N[u]| \leq 2OPT$
  - No high-degree vertex v from other cluster may have $|N[v] \oplus N[w]| \leq 2OPT$




Proof Tools:

1. Venn Diagram:




2. Triangle Inequality:

$$|A \oplus C| \leq |A \oplus B| + |B \oplus C|$$

# Overview: Running Time

+ Remaining Challenge: How to do it in **nearly-linear time or nearly-linear MPC total memory?**

1. Need a fast way to test $|N[u] \oplus N[v]| \leq 2OPT$

2. At most $\tilde{O}(m)$ tests are allowed

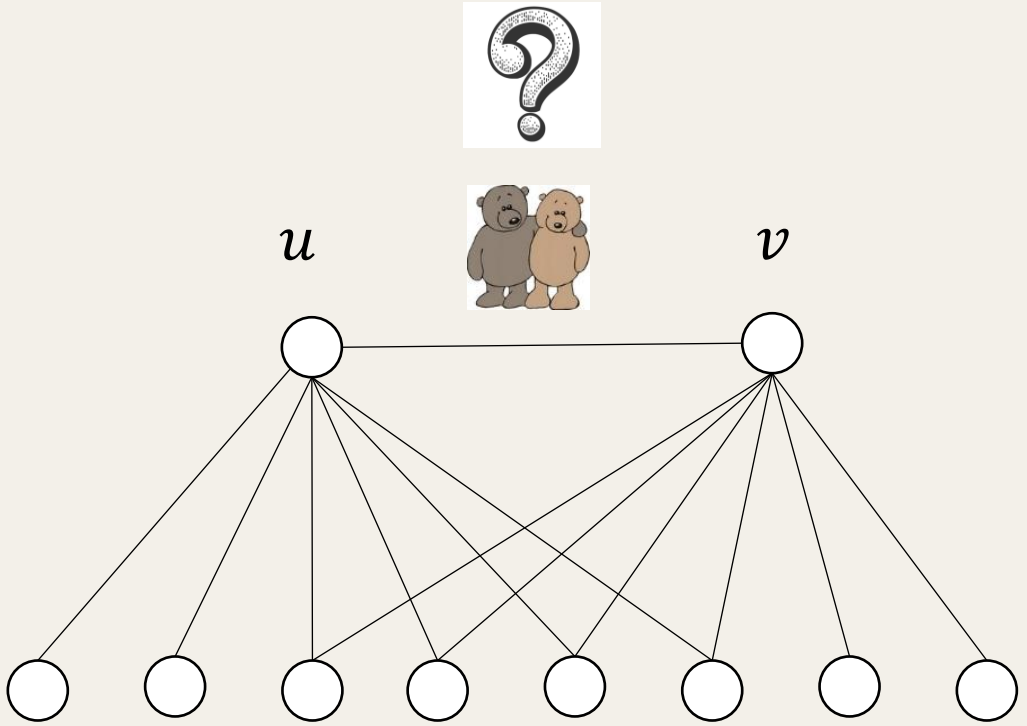
# Overview: Running Time

+ Remaining Challenge: How to do it in **nearly-linear time or nearly-linear MPC total memory?**

1. Need a fast way to test $|N[u] \oplus N[v]| \leq 2OPT$

# The Technique

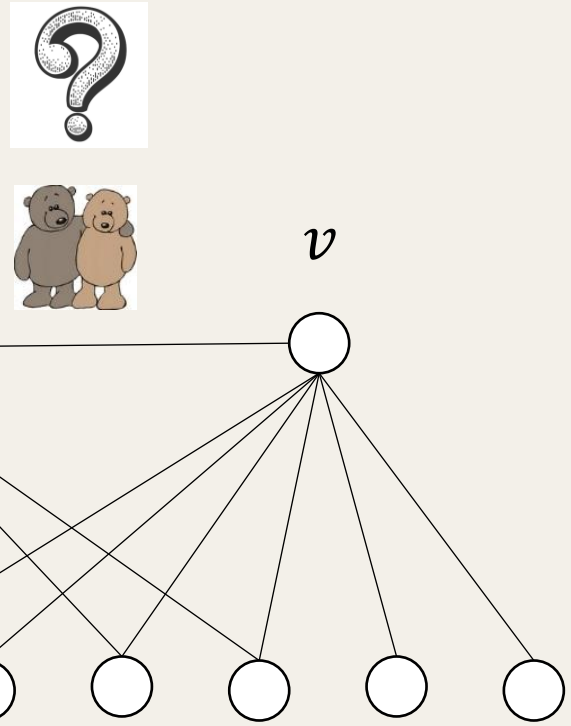
- Need a fast way to test $|N[u] \oplus N[v]| \leq 2OPT$



# The Technique

- Need a fast way to test $|N[u] \oplus N[v]| \leq 2OPT$

  Sampling may not work when $|N[u] \oplus N[v]|$ and *OPT* are small.



# The Technique

- Need a fast way to test $|N[u] \oplus N[v]| \leq 2OPT$
- Random projection (Johnson-Lindenstrauss Transform)

# The Technique

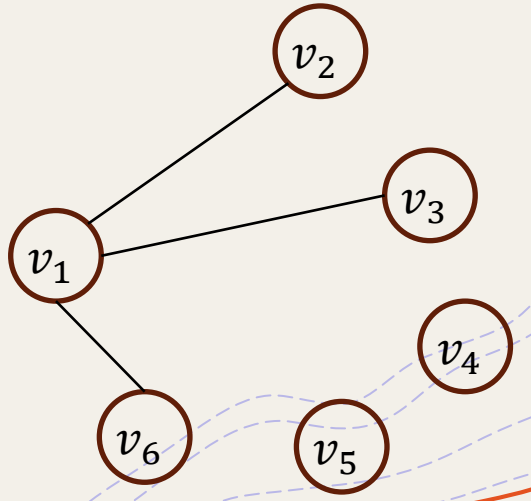
- Need a fast way to test $|N[u] \oplus N[v]| \leq 2OPT$
- Random projection (Johnson-Lindenstrauss Transform)
  - Let A be a $\frac{C \log n}{\epsilon^2} \times n$ matrix with each entry drawn from {-1,1} randomly

$$A = \begin{pmatrix} \pm 1 & \cdots & \pm 1 \\ \vdots & \ddots & \vdots \\ \pm 1 & \cdots & \pm 1 \end{pmatrix}$$

# The Technique

- Need a fast way to test $|N[u] \oplus N[v]| \leq 2OPT$
- Random projection (Johnson-Lindenstrauss Transform)
  - Let A be a $\frac{C \log n}{\epsilon^2} \times n$ matrix with each entry drawn from {-1,1} randomly
  - Every vertex $v$ computes $A \cdot \overrightarrow{N[v]}$ and stores it using $O(\log n / \epsilon^2)$ memory

$$A \cdot \overrightarrow{N[v]} = \begin{pmatrix} \pm 1 & \cdots & \pm 1 \\ \vdots & \ddots & \vdots \\ \pm 1 & \cdots & \pm 1 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$



# The Technique

- Need a fast way to test $|N[u] \oplus N[v]| \leq 2OPT$
- Random projection (Johnson-Lindenstrauss Transform)
  - Let A be a $\frac{C \log n}{\epsilon^2} \times n$ matrix with each entry drawn from {-1,1} randomly
  - Every vertex $v$ computes $A \cdot \overrightarrow{N[v]}$ and stores it using $O(\log n / \epsilon^2)$ memory
  - To determine $|N[u] \oplus N[v]|$, just compute $\left|A \cdot \overrightarrow{N[u]} - A \cdot \overrightarrow{N[v]}\right|_1$ because w.h.p.

$$|N[u] \oplus N[v]| \approx_\epsilon \left|A \cdot \overrightarrow{N[u]} - A \cdot \overrightarrow{N[v]}\right|_1$$

# The Technique

- Need a fast way to test $|N[u] \oplus N[v]| \leq 2OPT$
- Random projection (Johnson-Lindenstrauss Transform)
  - Let A be a $\frac{C \log n}{\epsilon^2} \times n$ matrix with each entry drawn from {-1,1} randomly
  - Every vertex $v$ computes $A \cdot \overrightarrow{N[v]}$ and stores it using $O(\log n / \epsilon^2)$ memory
  - To determine $|N[u] \oplus N[v]|$, just compute $\left|A \cdot \overrightarrow{N[u]} - A \cdot \overrightarrow{N[v]}\right|_1$ because w.h.p.

$$|N[u] \oplus N[v]| \approx_\epsilon \left|A \cdot \overrightarrow{N[u]} - A \cdot \overrightarrow{N[v]}\right|_1$$

  - Tests between k pairs of vertices: $O(k \cdot \log n / \epsilon^2)$

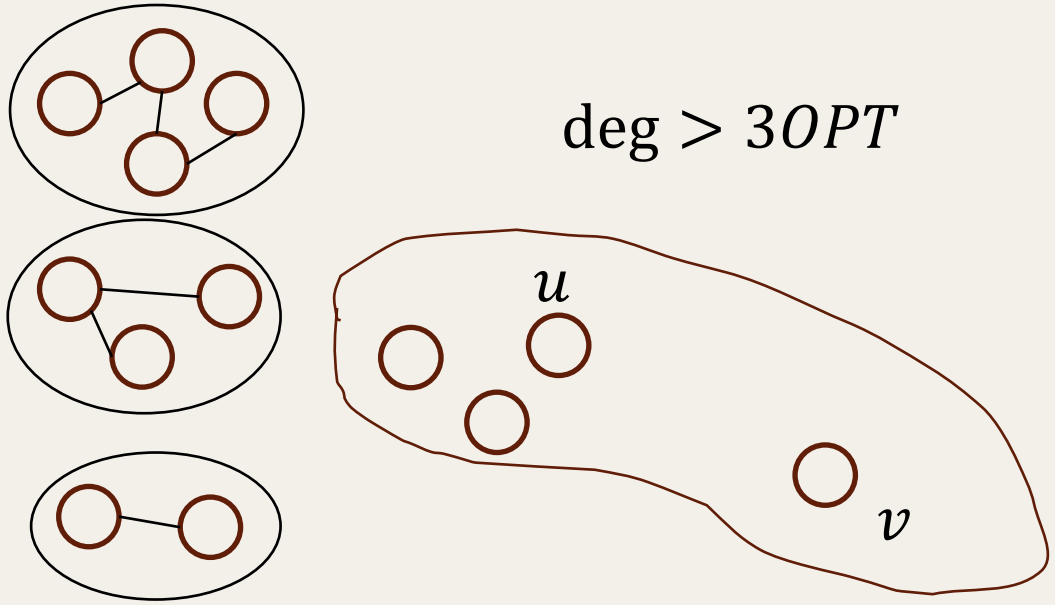
# Overview: Running Time

+ Remaining Challenge: How to do it in **nearly-linear time or nearly-linear MPC total memory?**

2. At most $\tilde{O}(m)$ tests are allowed

# Overview: Running Time

+ Remaining Challenge: How to do it in **nearly-linear time or nearly-linear MPC total memory?**
  2. At most $\tilde{O}(m)$ tests are allowed

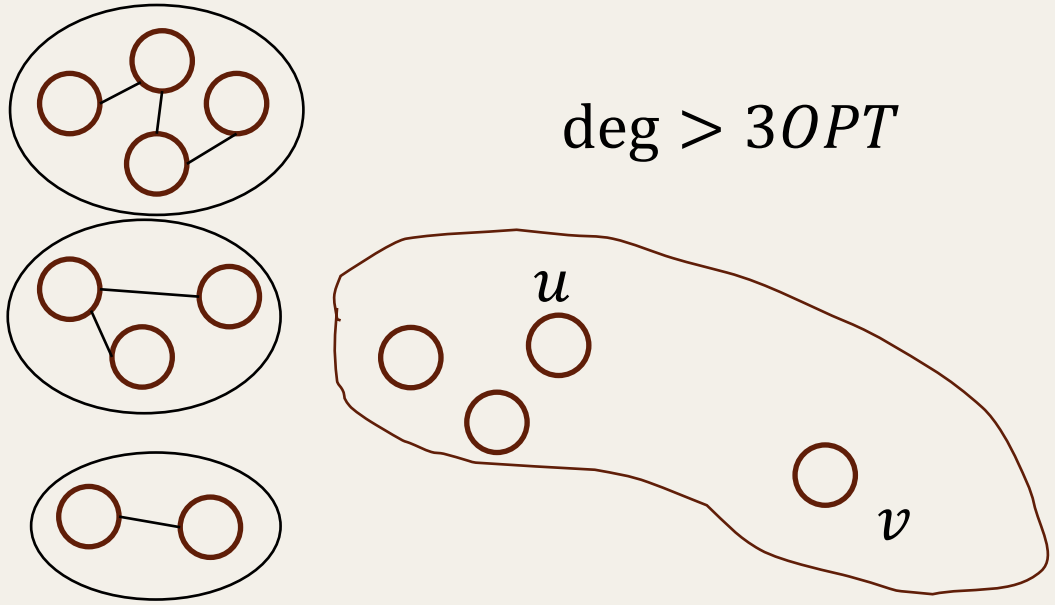


# Overview: Running Time

+ Remaining Challenge: How to do it in **nearly-linear time or nearly-linear MPC total memory?**

2. At most $\tilde{O}(m)$ tests are allowed

It is possible that $(u, v) \notin E^+$, but we need to check if $|N[u] \oplus N[v]| \leq 2OPT$ to see if u and v are in the same cluster

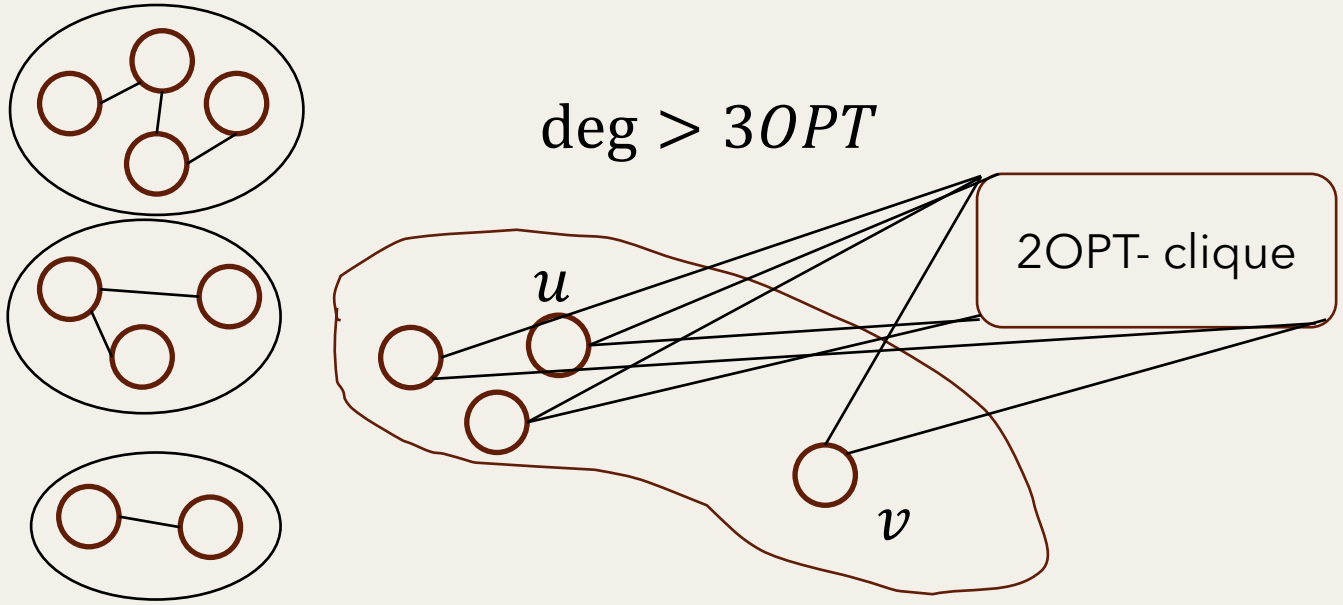


# Overview: Running Time

+ Remaining Challenge: How to do it in **nearly-linear time or nearly-linear MPC total memory?**

2. At most $\tilde{O}(m)$ tests are allowed

It is possible that $(u, v) \notin E^+$, but we need to check if $|N[u] \oplus N[v]| \leq 2OPT$ to see if u and v are in the same cluster

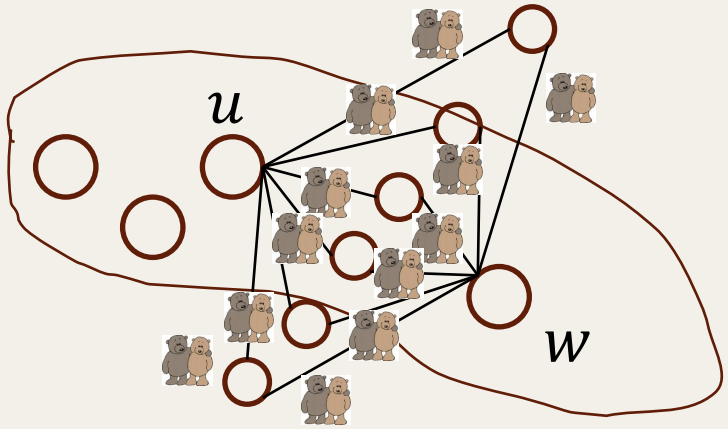


# Overview: Running Time

+ Remaining Challenge: How to do it in **nearly-linear time or nearly-linear MPC total memory?**

2. At most $\tilde{O}(m)$ tests are allowed

Additional property: If $u$ and $w$ have similar neighborhood if and only if $u$ and $w$ shares $\geq \mathrm{OPT} + 1$ "friends" neighbors




friends: neighborhood symmetric difference $\leq 2OPT$

# Overview: Running Time

- (Skipped) Additional challenges in a single-pass semi-streaming settings

# Conclusion

- A $(3+\epsilon)$-approximation algorithm for $\ell_\infty$-correlation clustering
  - Nearly-linear time
  - $O(1)$-round sublinear MPC
  - Single-pass semi-streaming setting

- A convenient tool for computing neighborhood similarity

# Open Problems

- Better than 3-approximation factor for $\ell_\infty$-correlation clustering?

- Efficient algorithms for $\ell_p$-correlation clustering for general $p$
  - Non-LP based: 63.3-approximation [Cao, Ye, and Li '25]
  - LP-based: 5-approximation [Kalhan, Makarychev, Zhou '19]

# Open Problems

- Better than 3-approximation factor for $\ell_\infty$-correlation clustering?

- Efficient algorithms for $\ell_p$-correlation clustering for general $p$
  - Non-LP based: 63.3-approximation [Cao, Ye, and Li '25]
  - LP-based: 5-approximation [Kalhan, Makarychev, Zhou '19]
- Deterministic algorithm for O(1)-approximation for correlation clustering

# Open Problems

- Better than 3-approximation factor for $\ell_\infty$-correlation clustering?

- Efficient algorithms for $\ell_p$-correlation clustering for general $p$
  - Non-LP based: 63.3-approximation [Cao, Ye, and Li '25]
  - LP-based: 5-approximation [Kalhan, Makarychev, Zhou '19]
- Deterministic algorithm for O(1)-approximation for correlation clustering
- A deterministic algorithm for computing almost-clique decomposition.
  - CONGEST: $poly(\log n)$-round? $n^{o(1)}$ rounds?
  - Semi-streaming, MPC?

# Open Problems

- Better than 3-approximation factor for $\ell_\infty$-correlation clustering?

- Efficient algorithms for $\ell_p$-correlation clustering for general $p$
  - Non-LP based: 63.3-approximation [Cao, Ye, and Li '25]
  - LP-based: 5-approximation [Kalhan, Makarychev, Zhou '19]
- Deterministic algorithm for O(1)-approximation for correlation clustering
- A deterministic algorithm for computing almost-clique decomposition.
  - CONGEST: $poly(\log n)$-round? $n^{o(1)}$ rounds?
  - Semi-streaming, MPC?

Thank you!