Massively Parallel Algorithms for Relaxations of MIS

Shreyas Pai
IIT Madras

Algorithms for Massive Graphs 2024

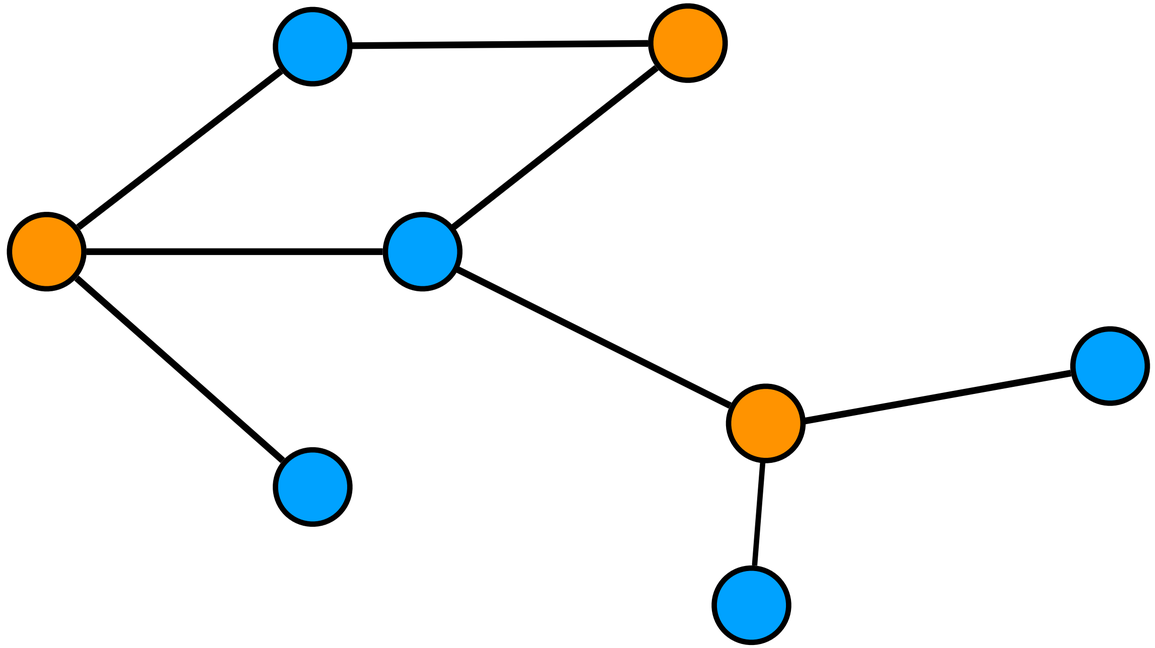
Maximal Independent Set

Maximal Independent Set

- An **independent set** I such that every node is **within** 1 **hop** from some node in I.

Maximal Independent Set

- An **independent set** I such that every node is **within** 1 **hop** from some node in I.

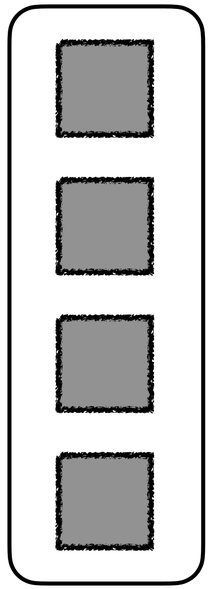


Massively Parallel Computing

Massively Parallel Computing

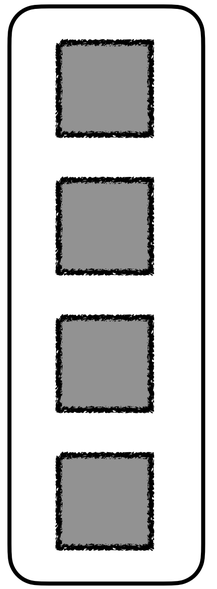
Massively Parallel Computing

- Input: n-node graph distributed across a set of machines.



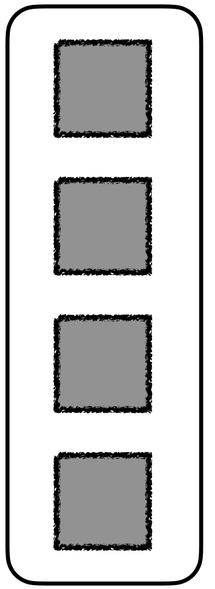
Massively Parallel Computing

- Input: n-node graph distributed across a set of machines.
- Each machine has **local memory** of $M = \tilde{O}(n)$ words.



Massively Parallel Computing

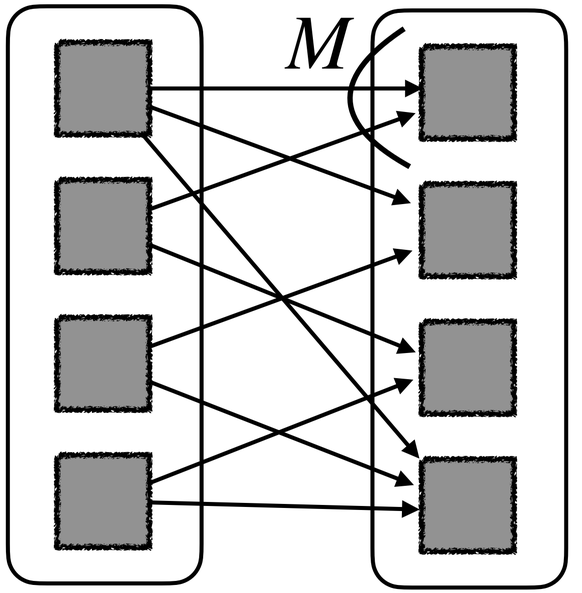
- Input: n-node graph distributed across a set of machines.
- Each machine has **local memory** of $M = \tilde{O}(n)$ words.
- In each round each machine can:



Round t

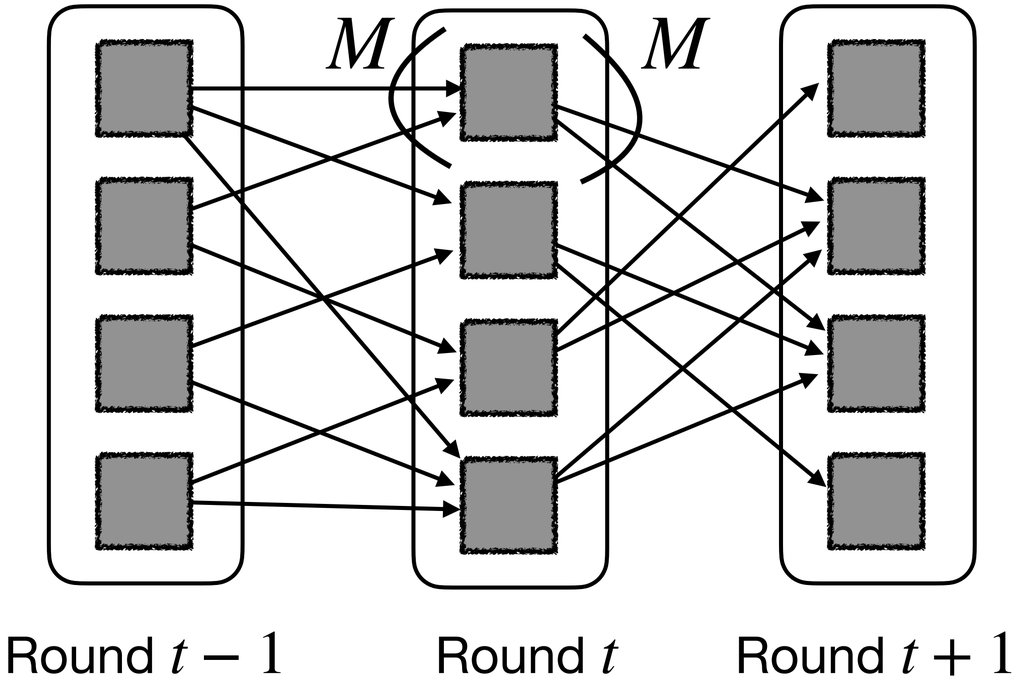
Massively Parallel Computing

- Input: n-node graph distributed across a set of machines.
- Each machine has **local memory** of $M = \tilde{O}(n)$ words.
- In each round each machine can:
  - Receive at most M words.



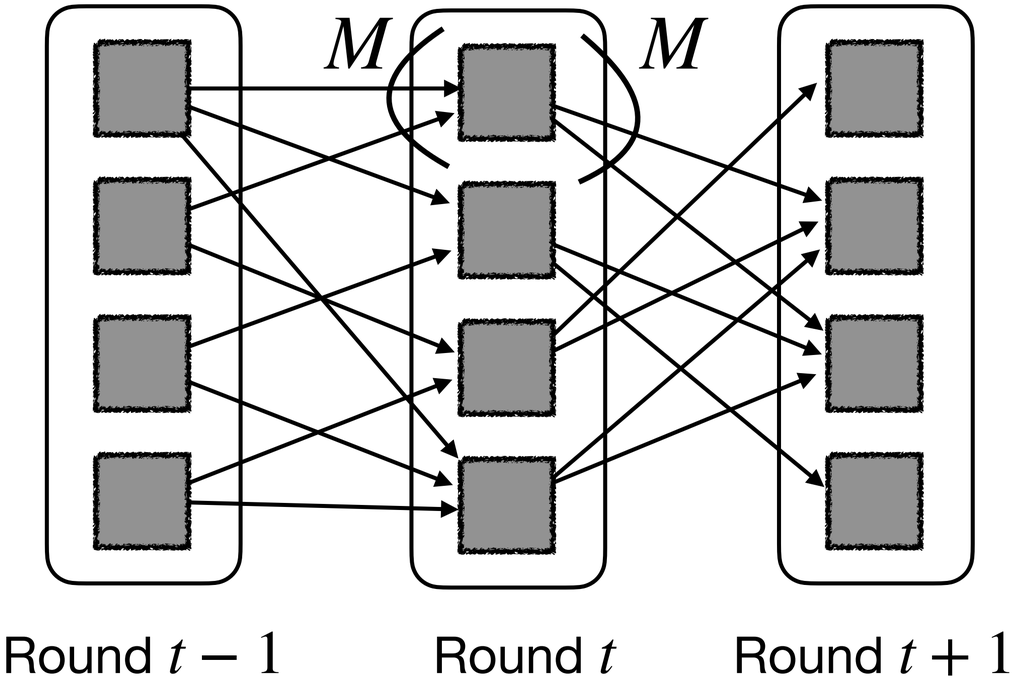
Massively Parallel Computing

- Input: n-node graph distributed across a set of machines.
- Each machine has **local memory** of $M = \tilde{O}(n)$ words.
- In each round each machine can:
  - Receive at most M words.
  - Send at most M words.



Massively Parallel Computing

- Input: n-node graph distributed across a set of machines.
- Each machine has **local memory** of $M = \tilde{O}(n)$ words.
- In each round each machine can:
  - Receive at most M words.
  - Send at most M words.
- Process input in $\tilde{O}(n)$ size chunks at a time.



MIS in MPC

MIS in MPC

- $O(\log \log \Delta)$ rounds randomized algorithm [GGK+18].

[GGK+18] Ghaffari, Gouleakis, Konrad, Mitrovi´c, and Rubinfeld. PODC 2018

MIS in MPC

- $O(\log \log \Delta)$ rounds randomized algorithm [GGK+18].
  - No progress after this.

[GGK+18] Ghaffari, Gouleakis, Konrad, Mitrovi´c, and Rubinfeld. PODC 2018

MIS in MPC

- $O(\log \log \Delta)$ rounds randomized algorithm [GGK+18].
  - No progress after this.
  - Not sure if this is optimal…

[GGK+18] Ghaffari, Gouleakis, Konrad, Mitrovi´c, and Rubinfeld. PODC 2018

MIS in MPC

- $O(\log \log \Delta)$ rounds randomized algorithm [GGK+18].
  - No progress after this.
  - Not sure if this is optimal…
- This talk: can we simplify MIS slightly and get optimal algorithms?

[GGK+18] Ghaffari, Gouleakis, Konrad, Mitrovi´c, and Rubinfeld. PODC 2018

MIS in MPC

- $O(\log \log \Delta)$ rounds randomized algorithm [GGK+18].
  - No progress after this.
  - Not sure if this is optimal…
- This talk: can we simplify MIS slightly and get optimal algorithms?
  - **2-Ruling Sets**: Cambus, Kuhn, P., and Uitto DISC 2023.

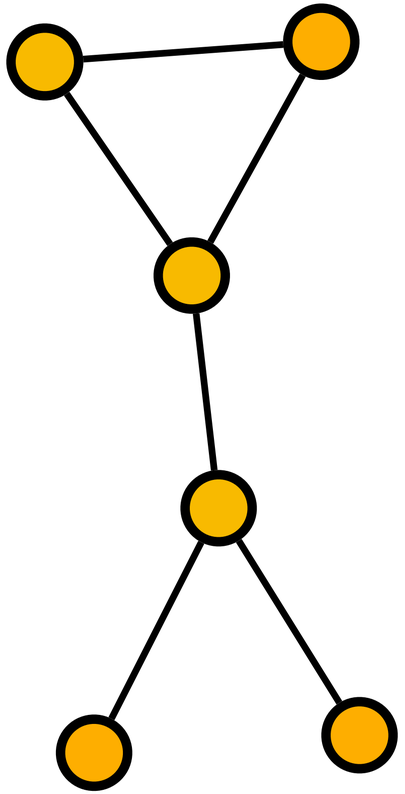
[GGK+18] Ghaffari, Gouleakis, Konrad, Mitrovi´c, and Rubinfeld. PODC 2018

MIS in MPC

- $O(\log \log \Delta)$ rounds randomized algorithm [GGK+18].
  - No progress after this.
  - Not sure if this is optimal…
- This talk: can we simplify MIS slightly and get optimal algorithms?
  - **2-Ruling Sets**: Cambus, Kuhn, P., and Uitto DISC 2023.
  - **Correlation Clustering**: Cambus, Kuhn, Lindy, P., and Uitto SODA 2024.

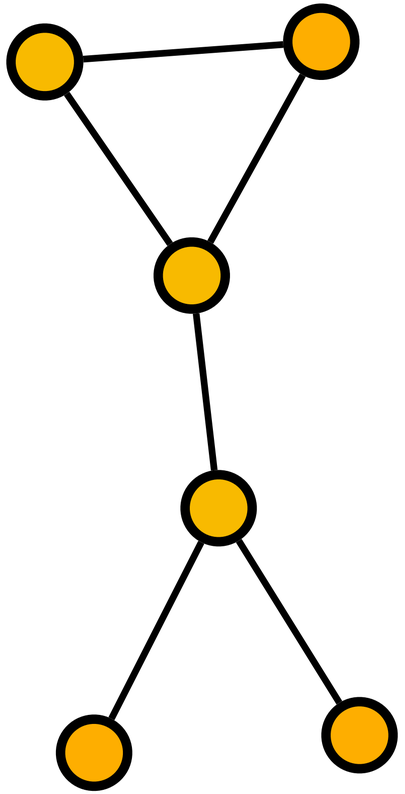
[GGK+18] Ghaffari, Gouleakis, Konrad, Mitrovi´c, and Rubinfeld. PODC 2018

Streaming Model



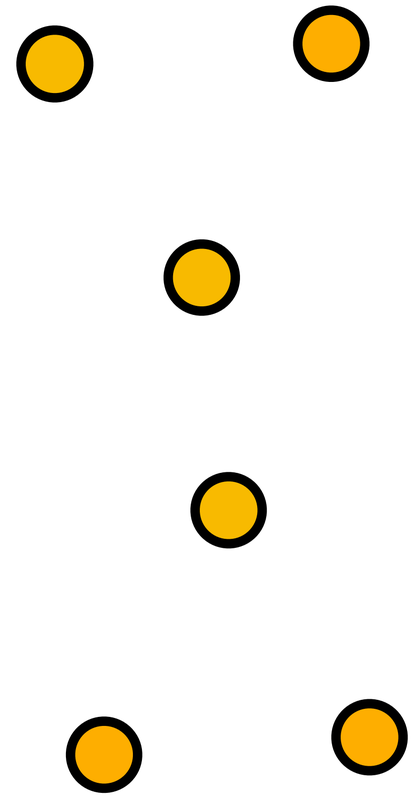
Streaming Model

- Input graph is fixed before the algorithm begins.



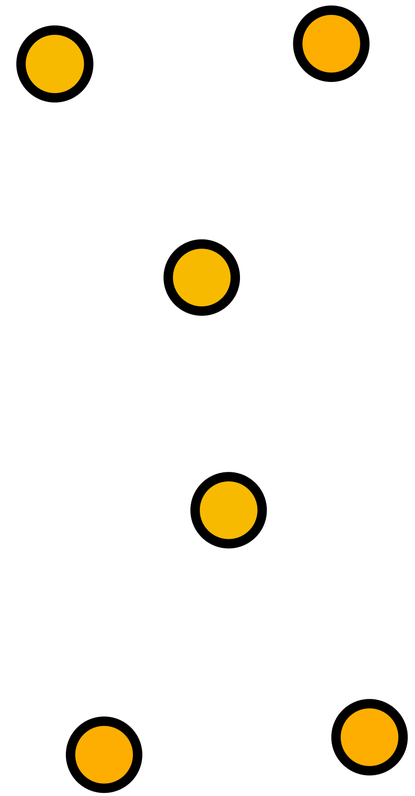
Streaming Model

- Input graph is fixed before the algorithm begins.
- Edges are unknown to the algorithm.



Streaming Model

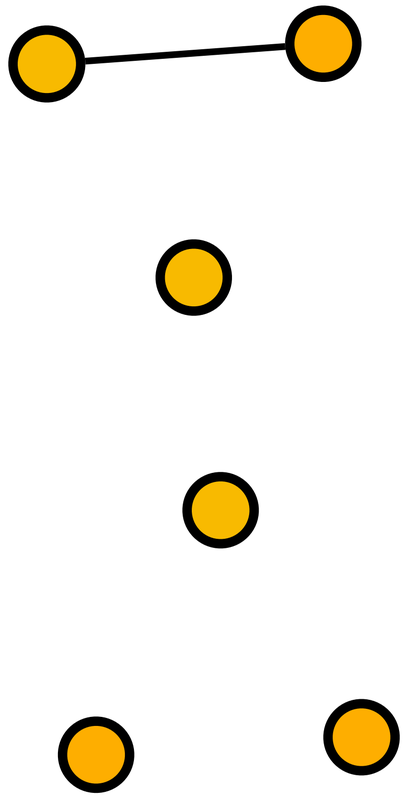
- Input graph is fixed before the algorithm begins.
- Edges are unknown to the algorithm.
- Revealed as a stream of edge insertions.



Streaming Model

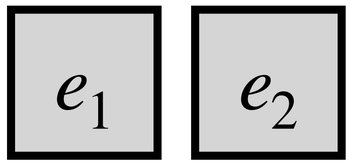
- Input graph is fixed before the algorithm begins.
- Edges are unknown to the algorithm.
- Revealed as a stream of edge insertions.

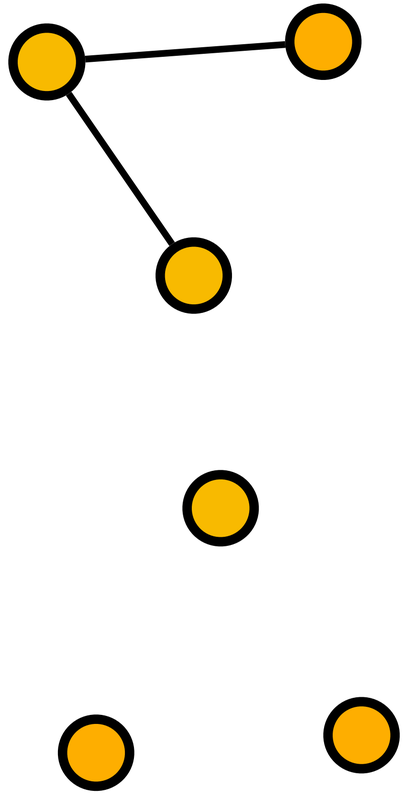
e_1



Streaming Model

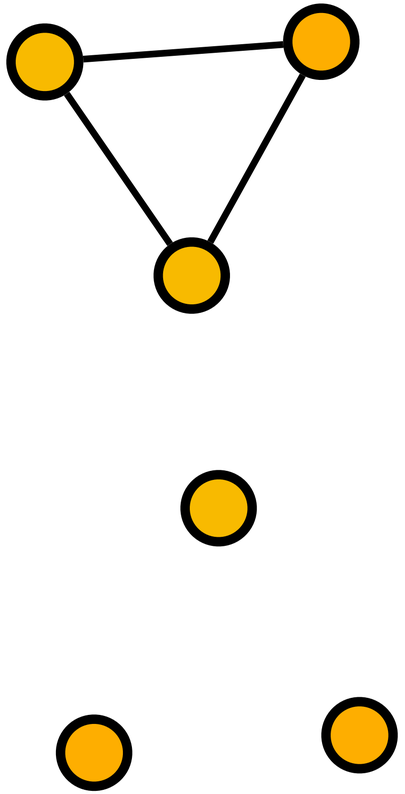
- Input graph is fixed before the algorithm begins.
- Edges are unknown to the algorithm.
- Revealed as a stream of edge insertions.





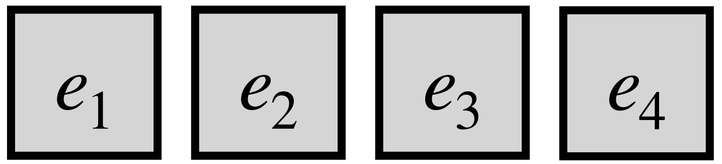
Streaming Model

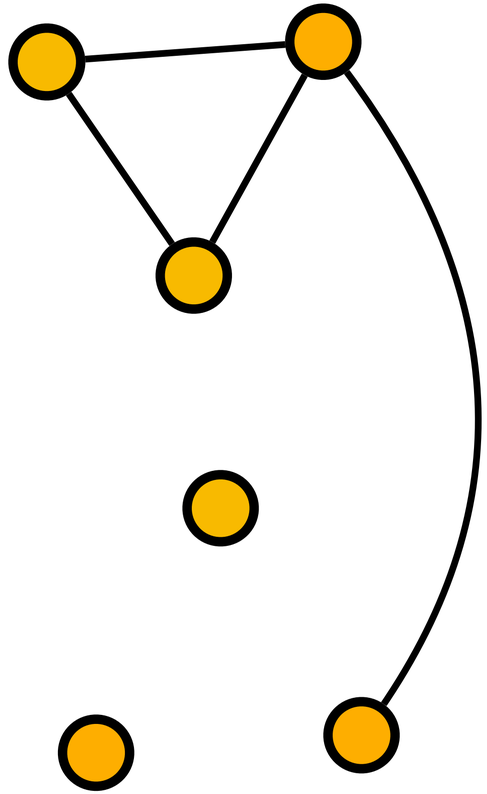
- Input graph is fixed before the algorithm begins.
- Edges are unknown to the algorithm.
- Revealed as a stream of edge insertions.



Streaming Model

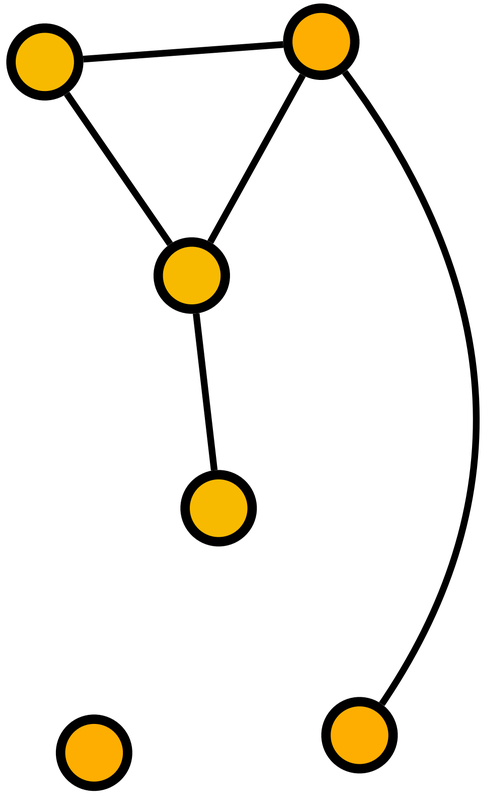
- Input graph is fixed before the algorithm begins.
- Edges are unknown to the algorithm.
- Revealed as a stream of edge insertions.





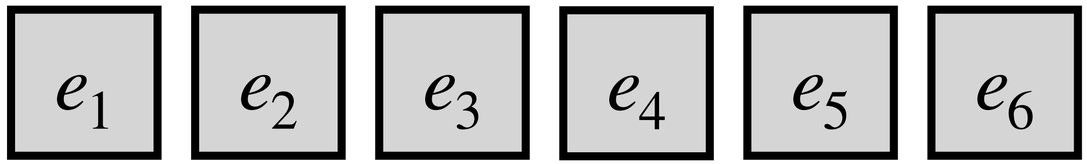
Streaming Model

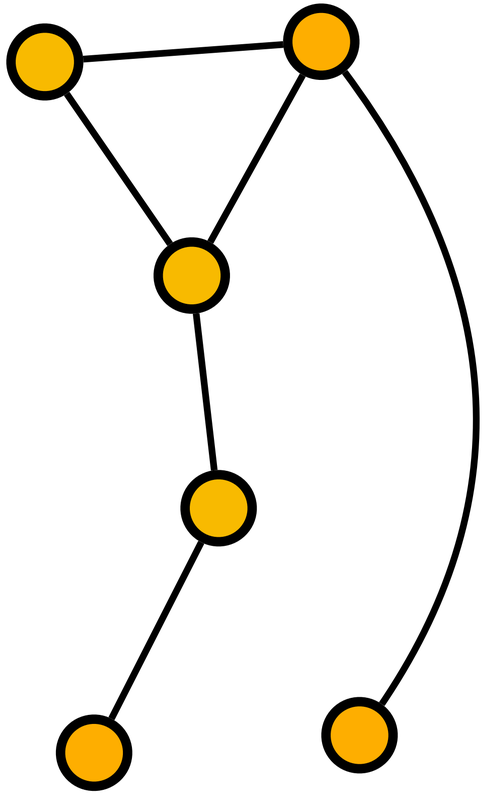
- Input graph is fixed before the algorithm begins.
- Edges are unknown to the algorithm.
- Revealed as a stream of edge insertions.



Streaming Model

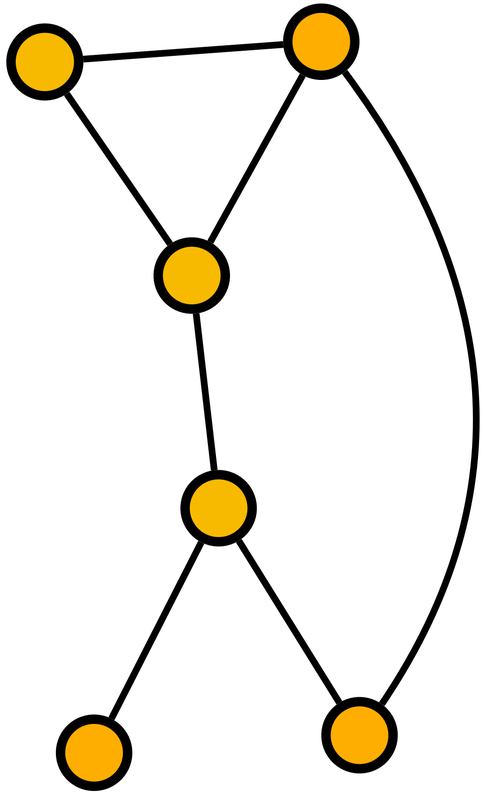
- Input graph is fixed before the algorithm begins.
- Edges are unknown to the algorithm.
- Revealed as a stream of edge insertions.





Streaming Model

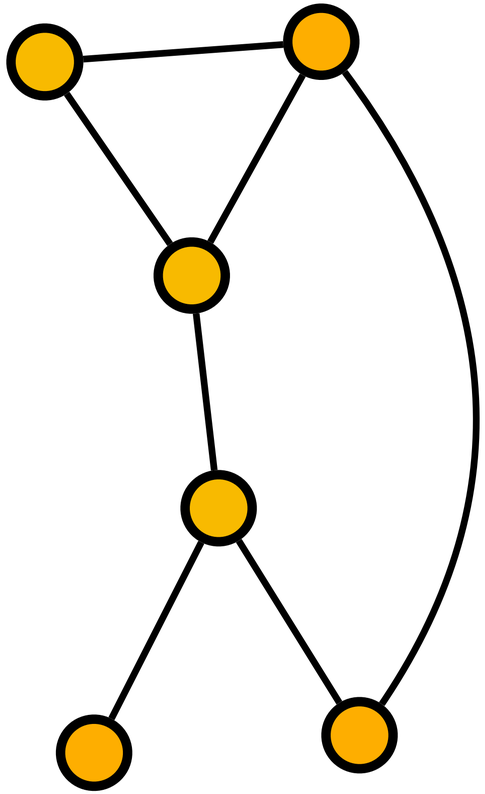
- Input graph is fixed before the algorithm begins.
- Edges are unknown to the algorithm.
- Revealed as a stream of edge insertions.



Streaming Model

- Input graph is fixed before the algorithm begins.
- Edges are unknown to the algorithm.
- Revealed as a stream of edge insertions.

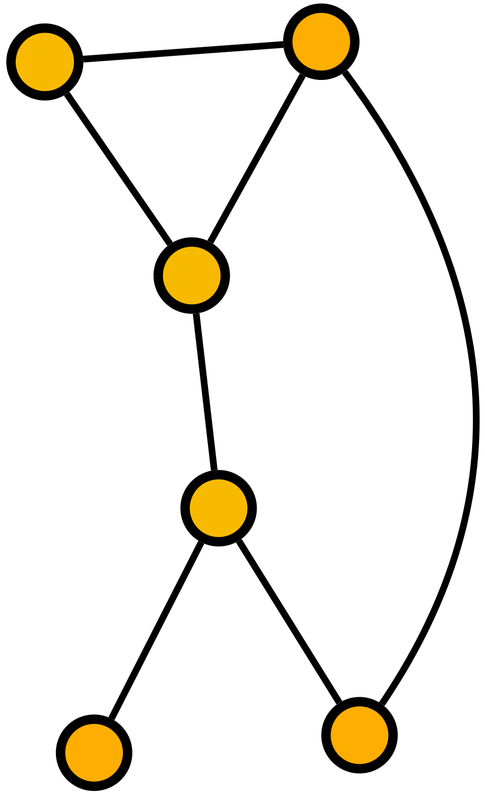
- Measure of efficiency: space



Streaming Model

- Input graph is fixed before the algorithm begins.
- Edges are unknown to the algorithm.
- Revealed as a stream of edge insertions.

- Measure of efficiency: space
    - Happy if we use at most $\tilde{O}(n)$ space.



MIS in Streaming

MIS in Streaming

- Can solve in $O(\log \log n)$ passes. [ACG+15]

[ACG+15] Ahn, Cormode, Guha, McGregor, and Wirth., ICML 2015

MIS in Streaming

- Can solve in $O(\log \log n)$ passes. [ACG+15]
- Typically interested in single pass or $O(1)$ pass algorithms.

[ACG+15] Ahn, Cormode, Guha, McGregor, and Wirth., ICML 2015

MIS in Streaming

- Can solve in $O(\log \log n)$ passes. [ACG+15]
- Typically interested in single pass or $O(1)$ pass algorithms.
- This algorithm is optimal! [AKNS24]

[ACG+15] Ahn, Cormode, Guha, McGregor, and Wirth., ICML 2015

MIS in Streaming

- Can solve in $O(\log \log n)$ passes. [ACG+15]
- Typically interested in single pass or $O(1)$ pass algorithms.
- This algorithm is optimal! [AKNS24]
    - $O(\log \log n)$ passes is required for $\tilde{O}(n)$ space.

[ACG+15] Ahn, Cormode, Guha, McGregor, and Wirth., ICML 2015

MIS in Streaming

- Can solve in $O(\log \log n)$ passes. [ACG+15]
- Typically interested in single pass or $O(1)$ pass algorithms.
- This algorithm is optimal! [AKNS24]
    - $O(\log \log n)$ passes is required for $\tilde{O}(n)$ space.
- Also in this talk:

[ACG+15] Ahn, Cormode, Guha, McGregor, and Wirth., ICML 2015

MIS in Streaming

- Can solve in $O(\log \log n)$ passes. [ACG+15]
- Typically interested in single pass or $O(1)$ pass algorithms.
- This algorithm is optimal! [AKNS24]
  - $O(\log \log n)$ passes is required for $\tilde{O}(n)$ space.
- Also in this talk:
  - Can we get $\tilde{O}(n)$ space with fewer passes for the relaxed problems?

[ACG+15] Ahn, Cormode, Guha, McGregor, and Wirth., ICML 2015
[AKNS24] Assadi, Konrad, Naidu, Sundaresan, STOC 2024

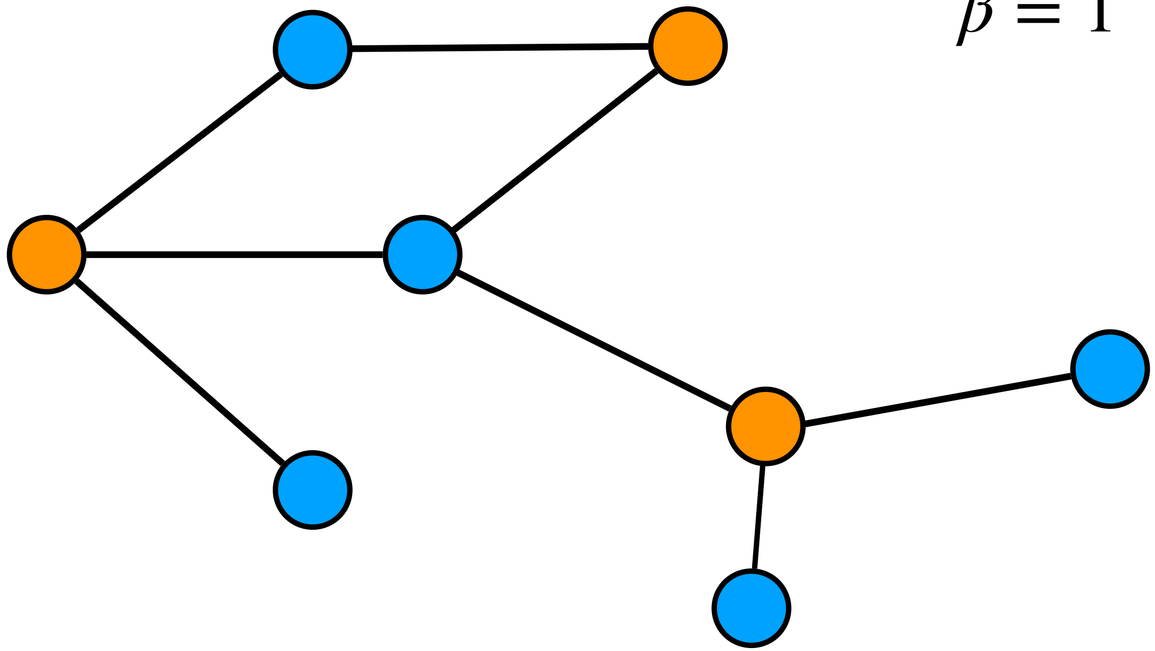
Relaxation 1: Ruling Sets

Relaxation 1: Ruling Sets

- A $(2,\beta)$-ruling set is an **independent set I** such that every node is **within β hops** from some node in I (for integer $\beta \geq 1$).

Relaxation 1: Ruling Sets

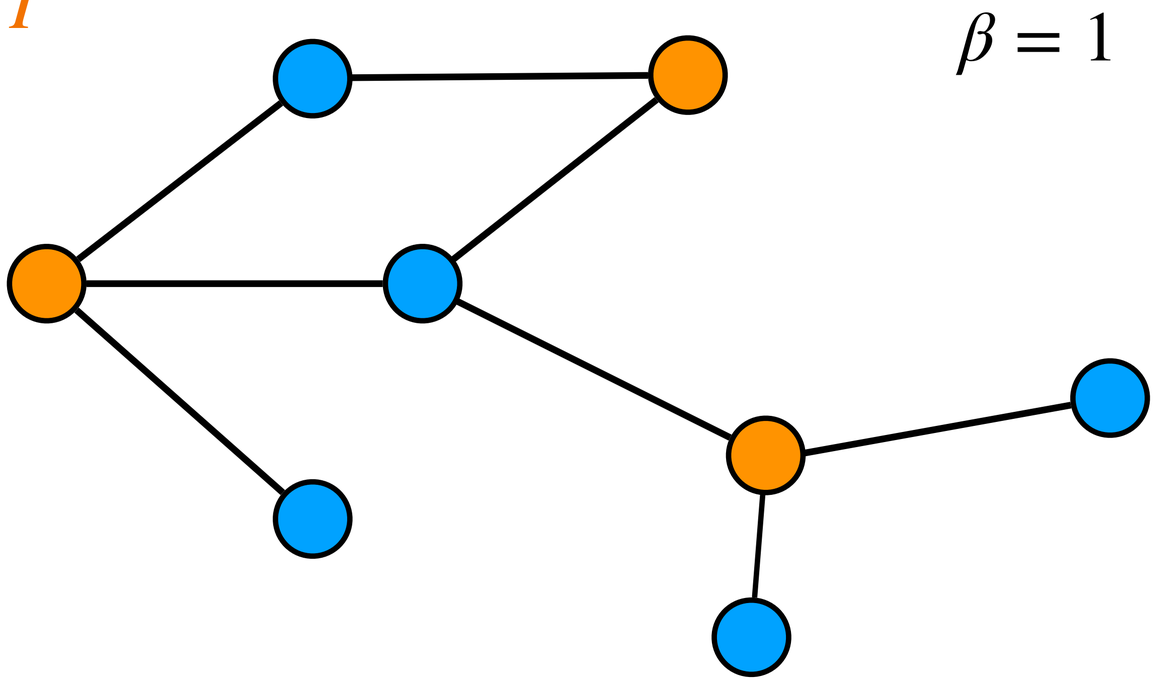
- A $(2,\beta)$-ruling set is an **independent set I** such that every node is **within β hops** from some node in I (for integer $\beta \geq 1$).



$\beta = 1$

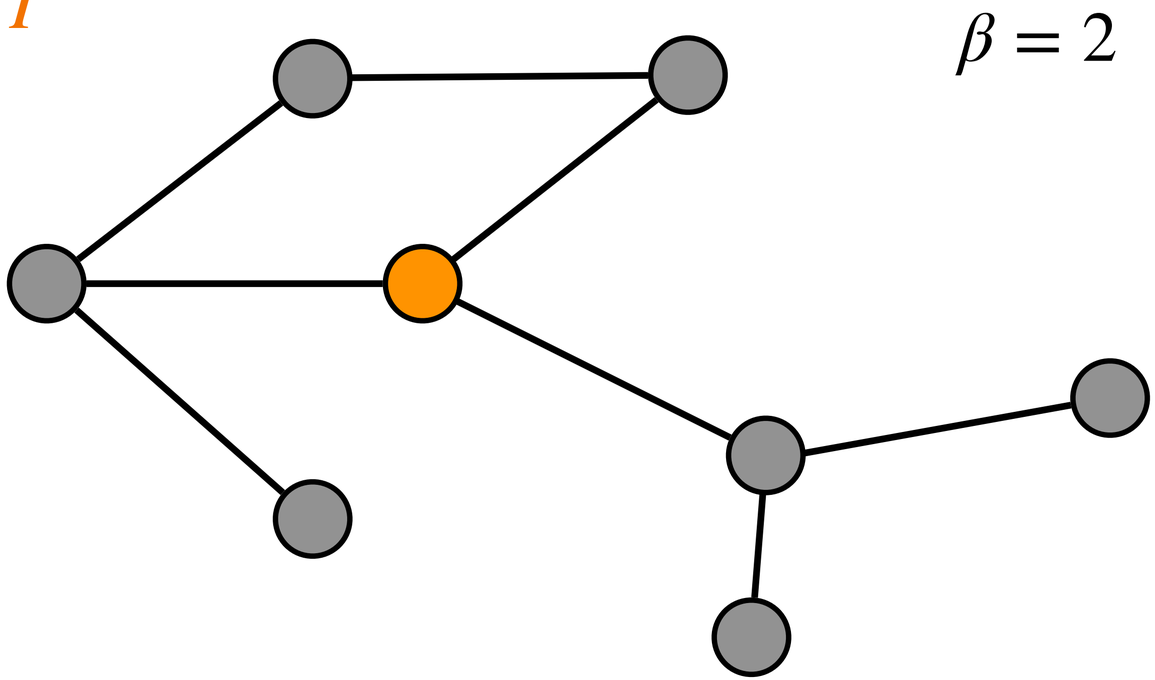
Relaxation 1: Ruling Sets

- A $(2,\beta)$-ruling set is an **independent set I** such that every node is **within β hops** from some node in I (for integer $\beta \geq 1$).
- $\beta = 1$: Maximal Independent Set (MIS).



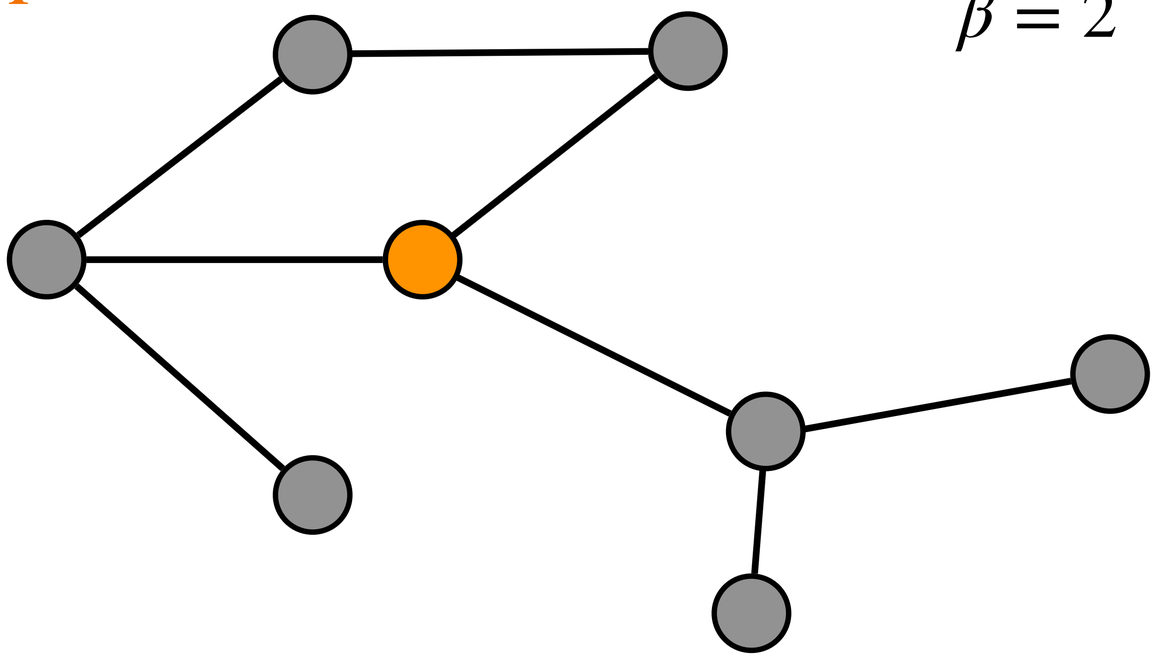
Relaxation 1: Ruling Sets

- A $(2,\beta)$-ruling set is an **independent set I** such that every node is **within β hops** from some node in I (for integer $\beta \geq 1$).
- $\beta = 1$: Maximal Independent Set (MIS).



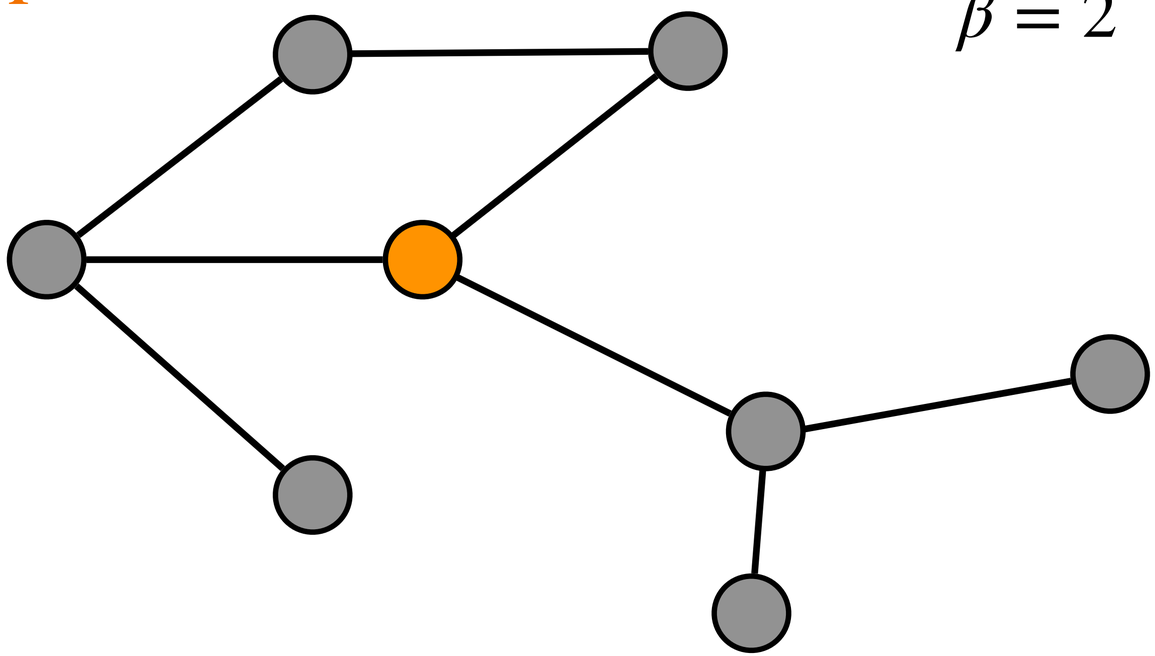
Relaxation 1: Ruling Sets

- A $(2,\beta)$-ruling set is an **independent set I** such that every node is **within β hops** from some node in I (for integer $\beta \geq 1$).
- $\beta = 1$: Maximal Independent Set (MIS).
- Easier to compute as β becomes larger.



Relaxation 1: Ruling Sets

- A $(2,\beta)$-ruling set is an **independent set I** such that every node is **within β hops** from some node in I (for integer $\beta \geq 1$).
- $\beta = 1$: Maximal Independent Set (MIS).
- Easier to compute as β becomes larger.
  - 2-RS is the “hardest” relaxation.



Why Care About Ruling Sets?

Why Care About Ruling Sets?

- Challenging to design fast distributed and parallel algorithms for ruling sets.

Why Care About Ruling Sets?

- Challenging to design fast distributed and parallel algorithms for ruling sets.
- Applications to clustering problems like metric facility location.

Why Care About Ruling Sets?

- Challenging to design fast distributed and parallel algorithms for ruling sets.
- Applications to clustering problems like metric facility location.
    - Create a conflict graph on the set of facilities.

Why Care About Ruling Sets?

- Challenging to design fast distributed and parallel algorithms for ruling sets.
- Applications to clustering problems like metric facility location.
    - Create a conflict graph on the set of facilities.
    - Compute a β-ruling set on the conflict graph.

Why Care About Ruling Sets?

- Challenging to design fast distributed and parallel algorithms for ruling sets.
- Applications to clustering problems like metric facility location.
    - Create a conflict graph on the set of facilities.
    - Compute a β-ruling set on the conflict graph.
    - Gives an $O(\beta)$-approximation.

Main Result for Ruling Sets

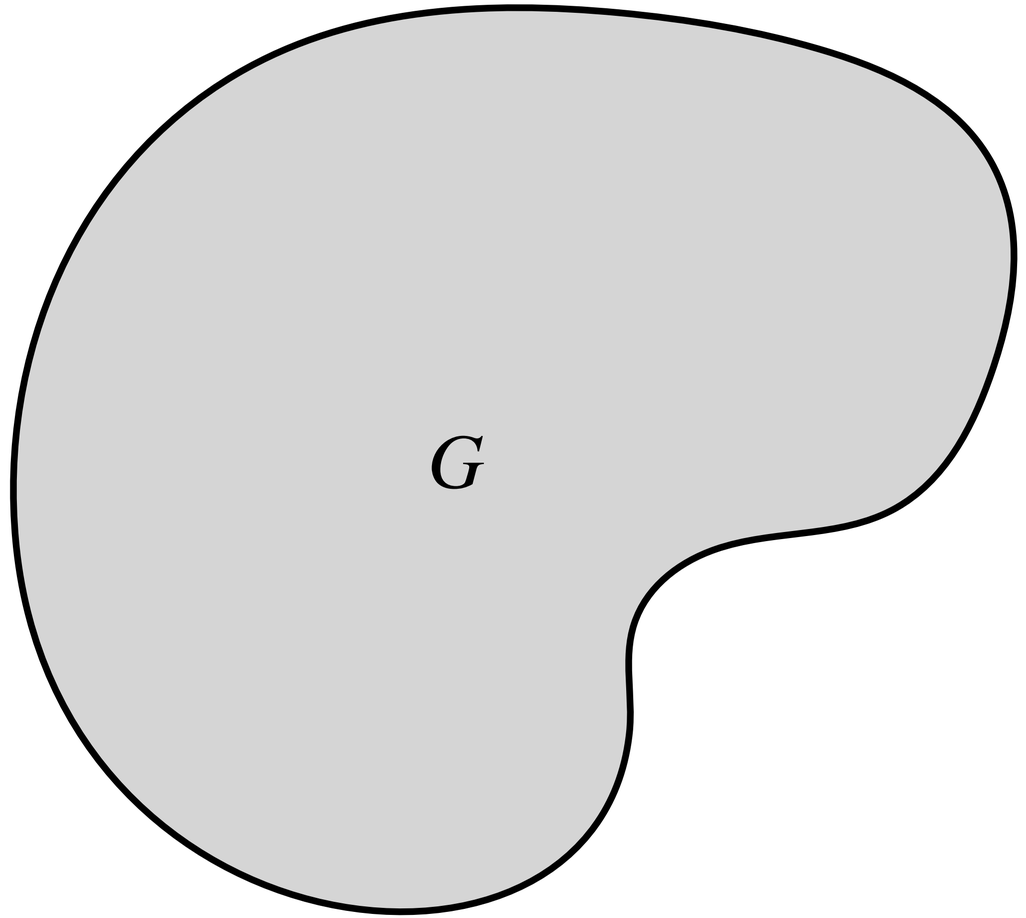
A randomized las-vegas algorithm that computes a 2-ruling set.

Can be implemented in:

- Linear-memory MPC: $O(1)$ rounds whp.
- Congested Clique: $O(1)$ rounds whp.
- Streaming: $O(1)$ passes whp, with $O(n)$ space (insertion-only).

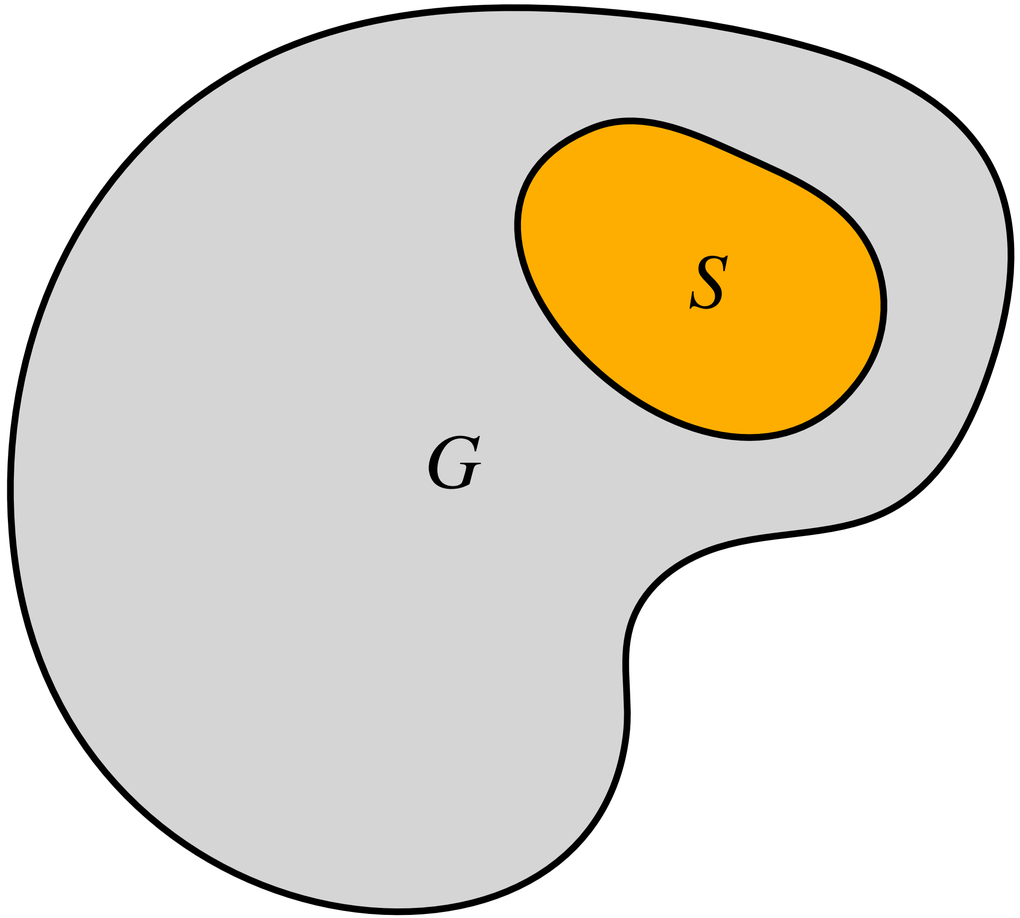
High Level View

High Level View



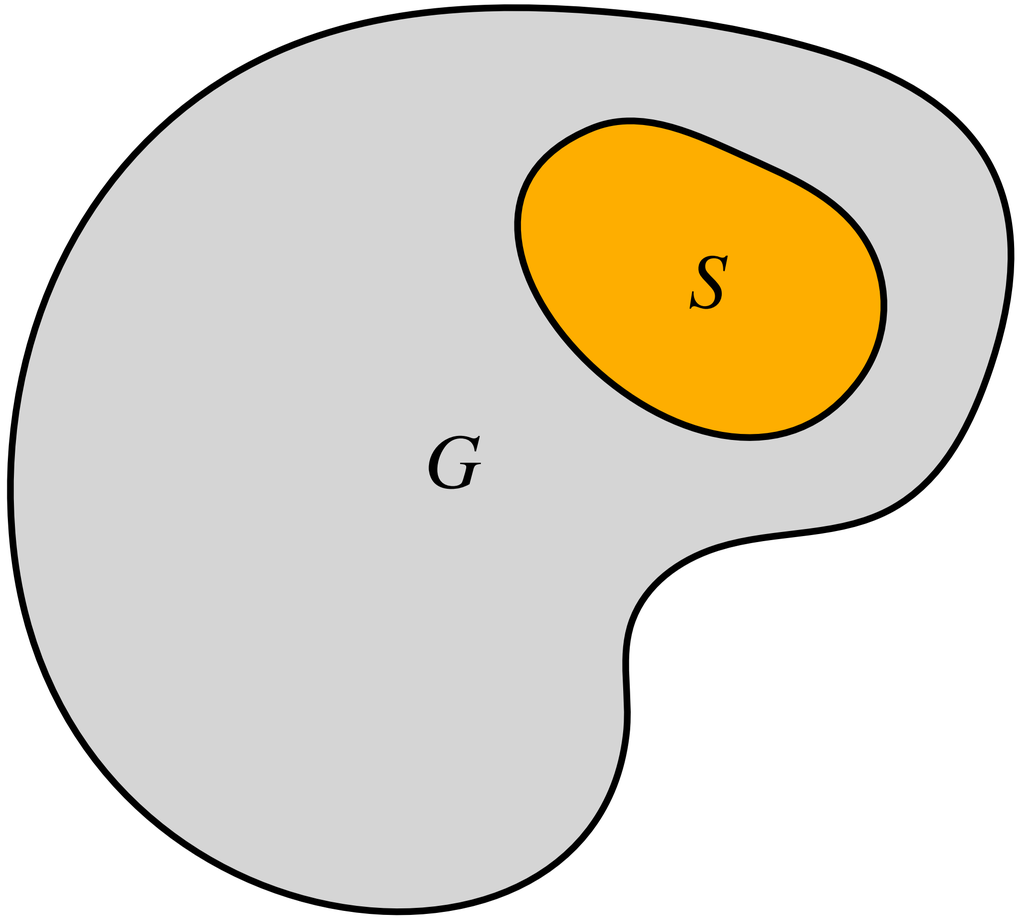
High Level View

- Choose a set S of nodes such that $G[S]$ has $O(n)$ edges.



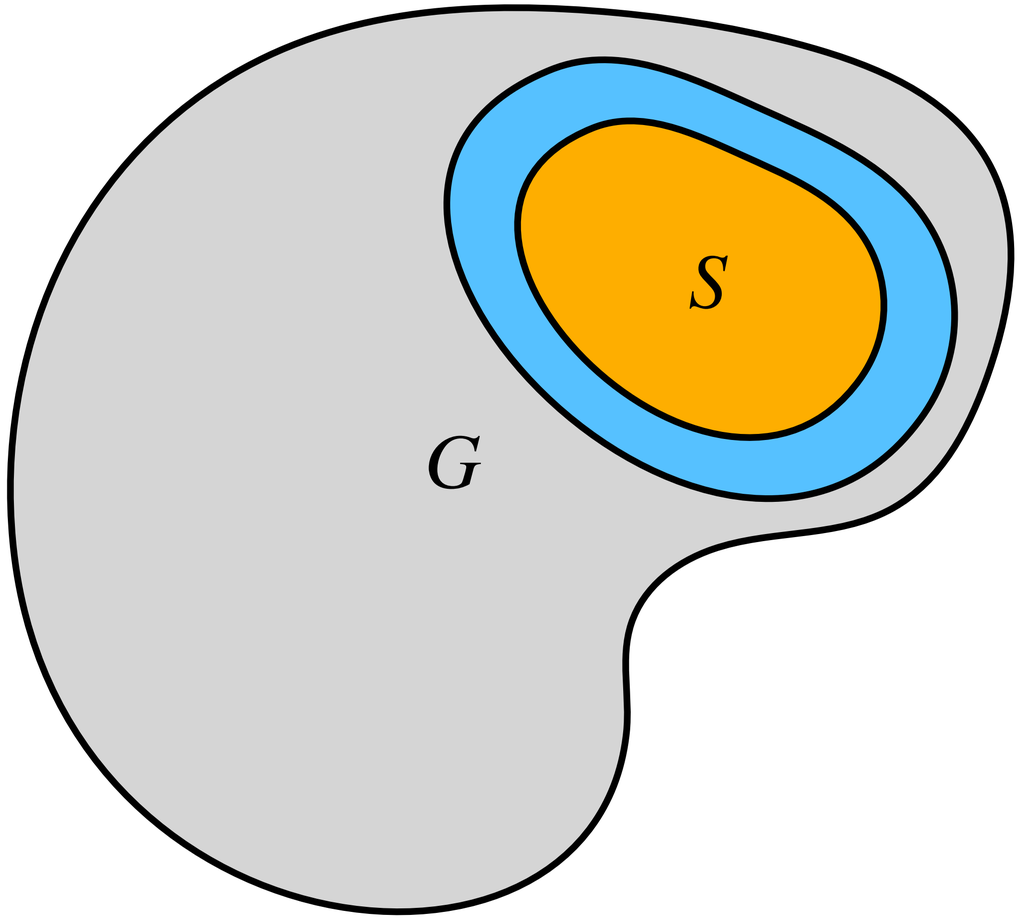
High Level View

- Choose a set S of nodes such that $G[S]$ has $O(n)$ edges.
    - Send $G[S]$ to a single machine which computes an MIS.



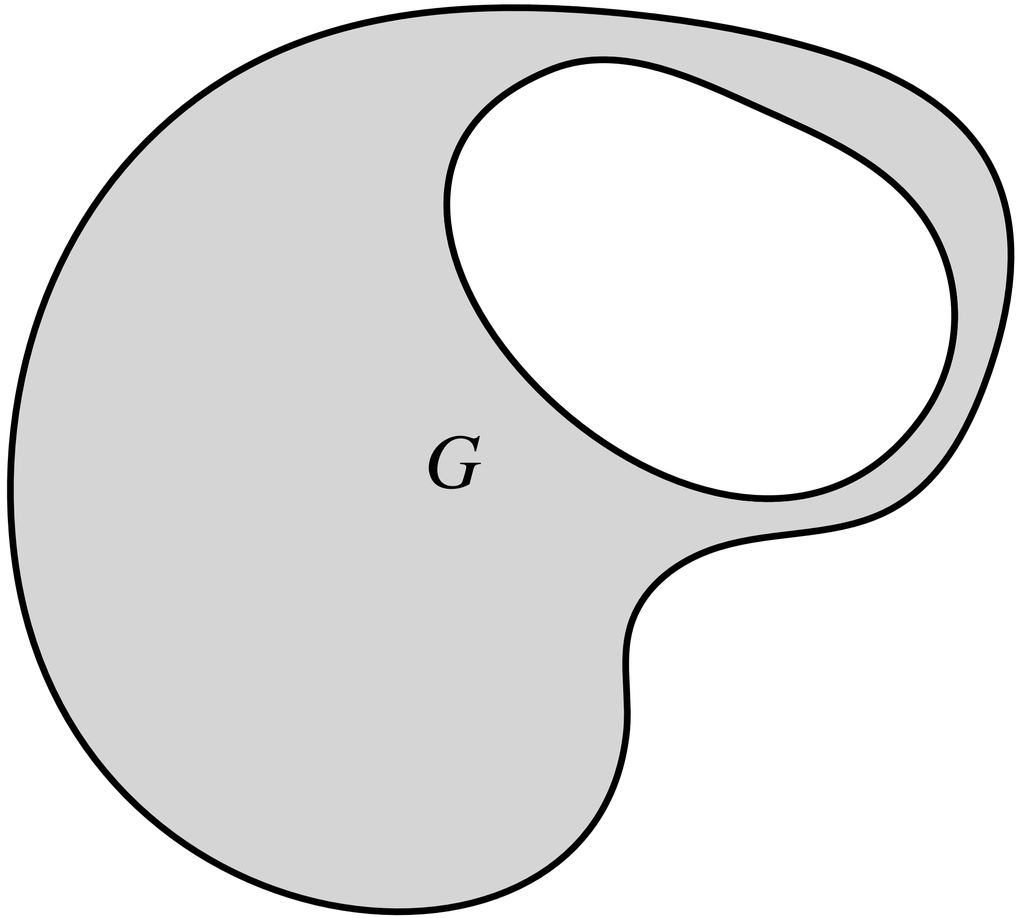
High Level View

- Choose a set S of nodes such that $G[S]$ has $O(n)$ edges.
    - Send $G[S]$ to a single machine which computes an MIS.
    - The MIS is a 2-RS on $S \cup N(S)$.



High Level View

- Choose a set S of nodes such that $G[S]$ has $O(n)$ edges.
    - Send $G[S]$ to a single machine which computes an MIS.
    - The MIS is a 2-RS on $S \cup N(S)$.
- Rinse and repeat until all nodes are covered.
- Repeatedly identify and process **linear sized** subgraphs.



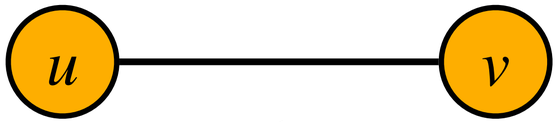
Sampling a Sparse Hitting Set

Sampling a Sparse Hitting Set

Add each vertex u to S with probability $\frac{1}{\sqrt{\deg(u)}}$

Sampling a Sparse Hitting Set

Add each vertex u to S with probability $\frac{1}{\sqrt{\deg(u)}}$

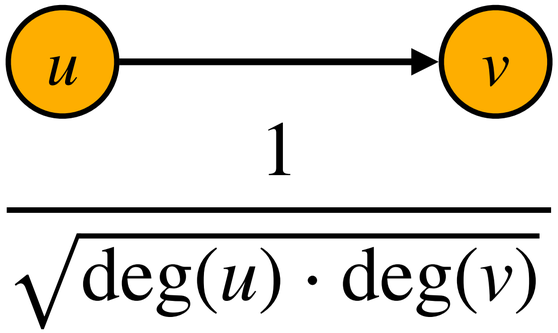


$$\frac{1}{\sqrt{\deg(u) \cdot \deg(v)}}$$

Sampling a Sparse Hitting Set

Add each vertex u to S with probability $\frac{1}{\sqrt{\deg(u)}}$

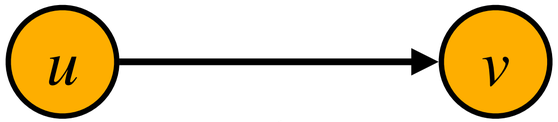
$u \to v$ iff
$\deg(u) \leq \deg(v)$



Sampling a Sparse Hitting Set

Add each vertex u to S with probability $\frac{1}{\sqrt{\deg(u)}}$

$u \to v$ iff
$\deg(u) \leq \deg(v)$

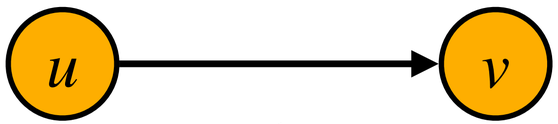


$$\leq \frac{1}{\deg(u)}$$

Sampling a Sparse Hitting Set

Add each vertex u to S with probability $\frac{1}{\sqrt{\deg(u)}}$

$u \to v$ iff
$\deg(u) \leq \deg(v)$



$$\leq \frac{1}{\deg(u)}$$

$G[S]$ has at most n edges in expectation!

Sampling a Sparse Hitting Set

Add each vertex u to S with probability $\frac{1}{\sqrt{\deg(u)}}$

Sampling a Sparse Hitting Set

Add each vertex u to S with probability $\frac{1}{\sqrt{\deg(u)}}$

$G[S]$ has at most n edges in expectation.

Sampling a Sparse Hitting Set

Add each vertex u to S with probability $\frac{1}{\sqrt{\deg(u)}}$

$G[S]$ has at most n edges in expectation.

- Challenge: the random choices for all edges are not independent.

Sampling a Sparse Hitting Set

Add each vertex u to S with probability $\frac{1}{\sqrt{\deg(u)}}$

$G[S]$ has at most n edges in expectation.

- Challenge: the random choices for all edges are not independent.
  - Cannot use standard Chernoff bounds.

Sampling a Sparse Hitting Set

Add each vertex u to S with probability $\frac{1}{\sqrt{\deg(u)}}$

$G[S]$ has at most n edges in expectation.

- Challenge: the random choices for all edges are not independent.
  - Cannot use standard Chernoff bounds.
- We use the **method of bounded differences** to show concentration.

Sampling a Sparse Hitting Set

Add each vertex u to S with probability $\frac{1}{\sqrt{\deg(u)}}$

$G[S]$ has at most n edges in expectation.

- Challenge: the random choices for all edges are not independent.
  - Cannot use standard Chernoff bounds.
- We use the **method of bounded differences** to show concentration.
  - *Technicality*: need $\Delta < n^{1/9}$ (can reduce max degree, no problem!)

Method of Bounded Differences

Method of Bounded Differences

- An n variable function f has Bounded Differences Property if changing the k^{th} coordinate only changes $f\left(\overline{X}\right)$ by at most c_k.

Method of Bounded Differences

- An n variable function f has Bounded Differences Property if changing the k^{th} coordinate only changes $f\left(\overline{X}\right)$ by at most c_k.
- Consider independent random variables $X_1, X_2, \ldots, X_n$. Then for any $t > 0$:

Method of Bounded Differences

- An n variable function f has Bounded Differences Property if changing the k^{th} coordinate only changes $f\left(\overline{X}\right)$ by at most c_k.
- Consider independent random variables $X_1, X_2, \ldots, X_n$. Then for any $t > 0$:

$$\Pr[\,|f(\overline{X}) - \mu\,| \geq t] \leq 2\exp\left(\frac{-t^2}{\sum_{k=1}^{n} c_k^2}\right)$$

Method of Bounded Differences

Method of Bounded Differences

$$\Pr[\,|f(\overline{X}) - \mu\,| \geq t] \leq 2\exp\left(\frac{-t^2}{\sum_{k=1}^{n} c_k^2}\right)$$

Method of Bounded Differences

$$\Pr[\,|f(\overline{X}) - \mu\,| \geq t] \leq 2\exp\left(\frac{-t^2}{\sum_{k=1}^{n} c_k^2}\right)$$

(1) X_u is the indicator random variable for event $u \in S$.

Method of Bounded Differences

$$\Pr[\,|f(\overline{X}) - \mu\,| \geq t] \leq 2\exp\left(\frac{-t^2}{\sum_{k=1}^{n} c_k^2}\right)$$

(1) X_u is the indicator random variable for event $u \in S$.

(2) $f\left(\overline{X}\right) =$ number of edges in $G[S]$.

Method of Bounded Differences

$$\Pr[\,|f(\overline{X}) - \mu\,| \geq t] \leq 2\exp\left(\frac{-t^2}{\sum_{k=1}^{n} c_k^2}\right)$$

(1) X_u is the indicator random variable for event $u \in S$.

(2) $f\left(\overline{X}\right) =$ number of edges in $G[S]$.

(3) $\mu \leq n$ and we can set $t = n$.

Method of Bounded Differences

$$\Pr[\ \ |G[S]| \leq 2n\ ] \leq 2\exp\left(\frac{-t^2}{\sum_{k=1}^{n} c_k^2}\right)$$

(1) X_u is the indicator random variable for event $u \in S$.

(2) $f\left(\overline{X}\right) =$ number of edges in $G[S]$.

(3) $\mu \leq n$ and we can set $t = n$.

Method of Bounded Differences

$$\Pr[\ |G[S]| \leq 2n\ ] \leq 2\exp\left(\frac{-n^2}{\sum_{k=1}^{n} c_k^2}\right)$$

(1) X_u is the indicator random variable for event $u \in S$.

(2) $f\left(\overline{X}\right) =$ number of edges in $G[S]$.

(3) $\mu \leq n$ and we can set $t = n$.

Method of Bounded Differences

$$\Pr[\ |G[S]| \leq 2n\ ] \leq 2\exp\left(\frac{-n^2}{\sum_{k=1}^{n} c_k^2}\right)$$

(1) X_u is the indicator random variable for event $u \in S$.

(2) $f\left(\overline{X}\right) =$ number of edges in $G[S]$.

(3) $\mu \leq n$ and we can set $t = n$.

(4) Since $\Delta \leq n^{1/9}$, we have $c_k \leq n^{1/9}$.

Method of Bounded Differences

$$\Pr[\ |G[S]| \leq 2n\ ] \leq 2\exp\left(\frac{-n^2}{n^{1+2/9}}\right)$$

(1) X_u is the indicator random variable for event $u \in S$.

(2) $f\left(\overline{X}\right) =$ number of edges in $G[S]$.

(3) $\mu \leq n$ and we can set $t = n$.

(4) Since $\Delta \leq n^{1/9}$, we have $c_k \leq n^{1/9}$.

Method of Bounded Differences

$$\Pr[\ |G[S]| \leq 2n\ ] \leq 2\exp\left(\frac{-n^2}{n^{1+2/9}}\right) = 2\exp\left(-n^{1-2/9}\right)$$

(1) X_u is the indicator random variable for event $u \in S$.

(2) $f\left(\overline{X}\right) =$ number of edges in $G[S]$.

(3) $\mu \leq n$ and we can set $t = n$.

(4) Since $\Delta \leq n^{1/9}$, we have $c_k \leq n^{1/9}$.

Method of Bounded Differences

$$\Pr[\ |G[S]| \le 2n\ ] \le 2\exp\left(\frac{-n^2}{n^{1+2/9}}\right) = 2\exp\left(-n^{1-2/9}\right) \ll \frac{1}{\text{poly}(n)}$$

(1) X_u is the indicator random variable for event $u \in S$.

(2) $f\left(\overline{X}\right) =$ number of edges in $G[S]$.

(3) $\mu \le n$ and we can set $t = n$.

(4) Since $\Delta \le n^{1/9}$, we have $c_k \le n^{1/9}$.

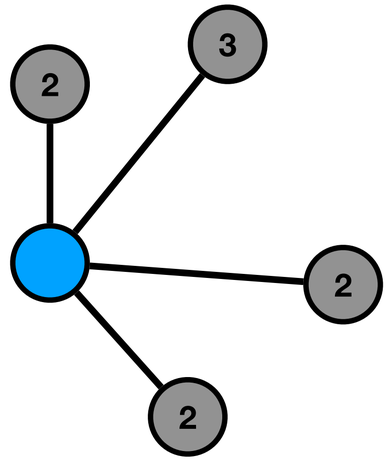
Which Nodes are Hit?

Which Nodes are Hit?

- High degree nodes are less likely to be sampled themselves.

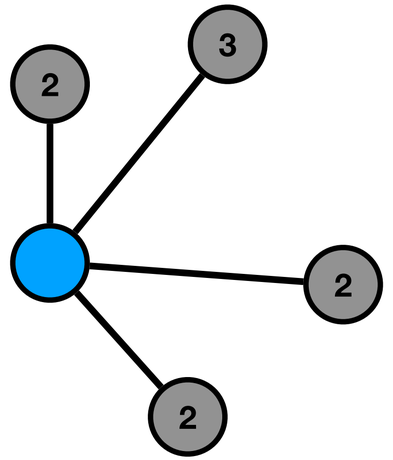
Which Nodes are Hit?

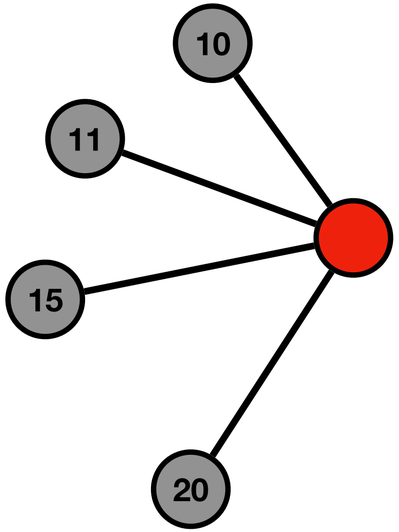
- High degree nodes are less likely to be sampled themselves.
    - **Good:** many low-degree neighbors.



Which Nodes are Hit?

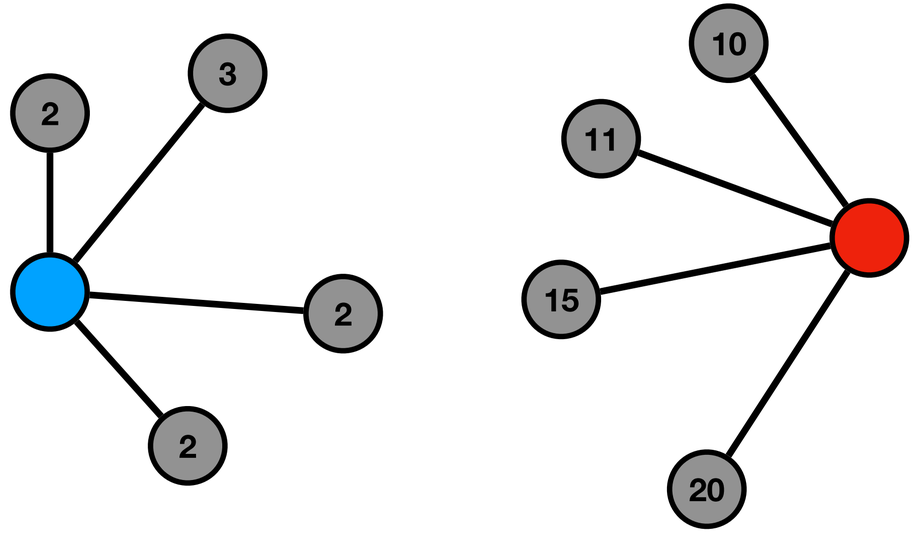
- High degree nodes are less likely to be sampled themselves.
  - **Good:** many low-degree neighbors.
  - **Bad:** many high-degree neighbors.





Which Nodes are Hit?

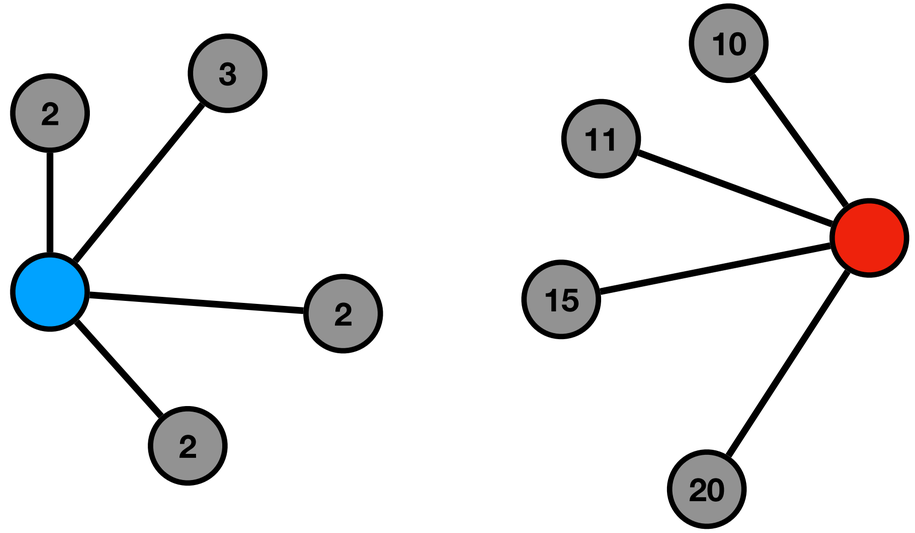
- High degree nodes are less likely to be sampled themselves.
  - **Good:** many low-degree neighbors.
  - **Bad:** many high-degree neighbors.



Node v is **good** if $\sum_{u \in N(v)} \frac{1}{\sqrt{\deg(u)}} > \log \deg(v)$

Which Nodes are Hit?

- High degree nodes are less likely to be sampled themselves.
  - **Good:** many low-degree neighbors.
  - **Bad:** many high-degree neighbors.

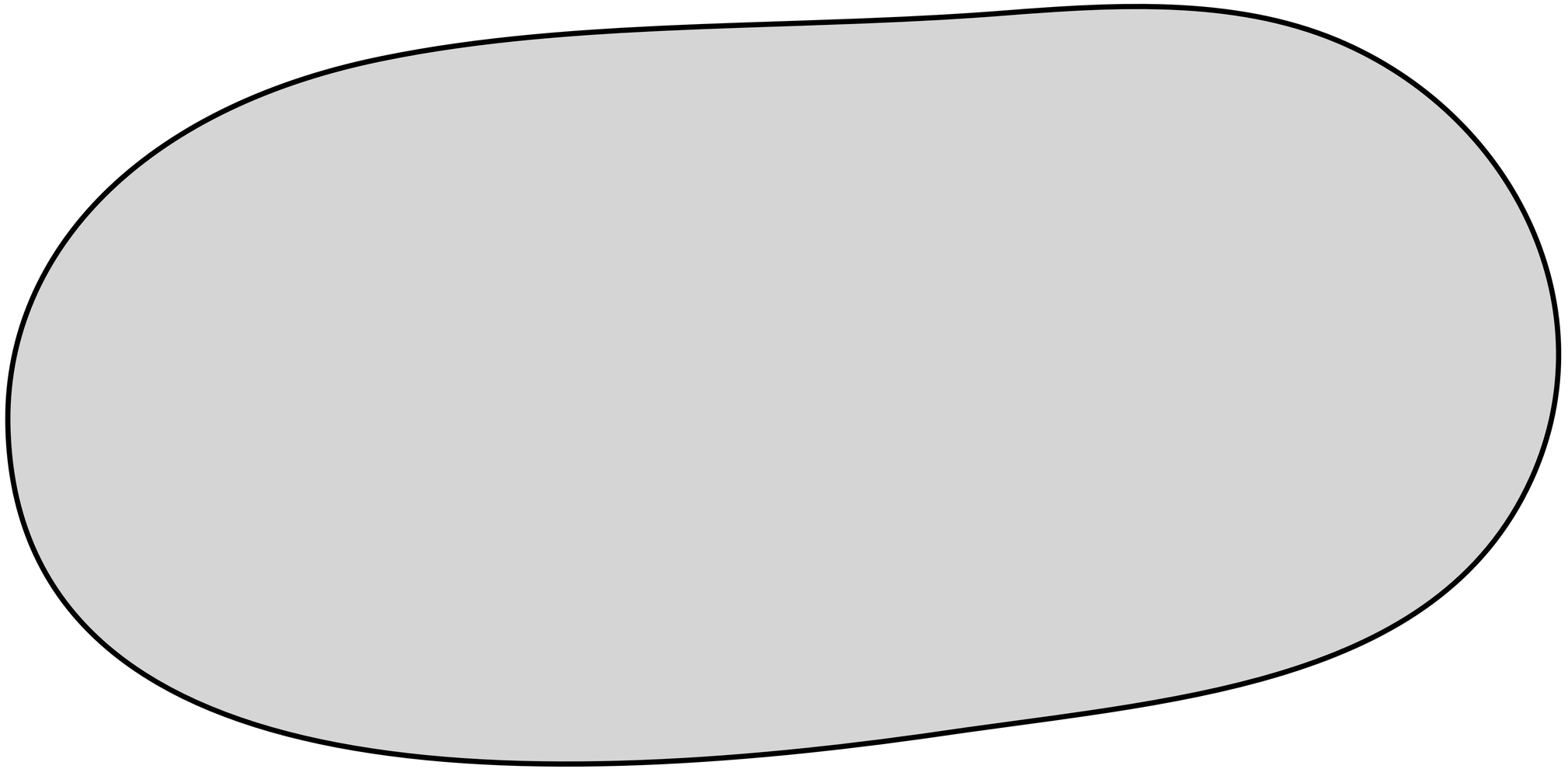


Node v is **good** if $\sum_{u \in N(v)} \frac{1}{\sqrt{\deg(u)}} > \log \deg(v)$

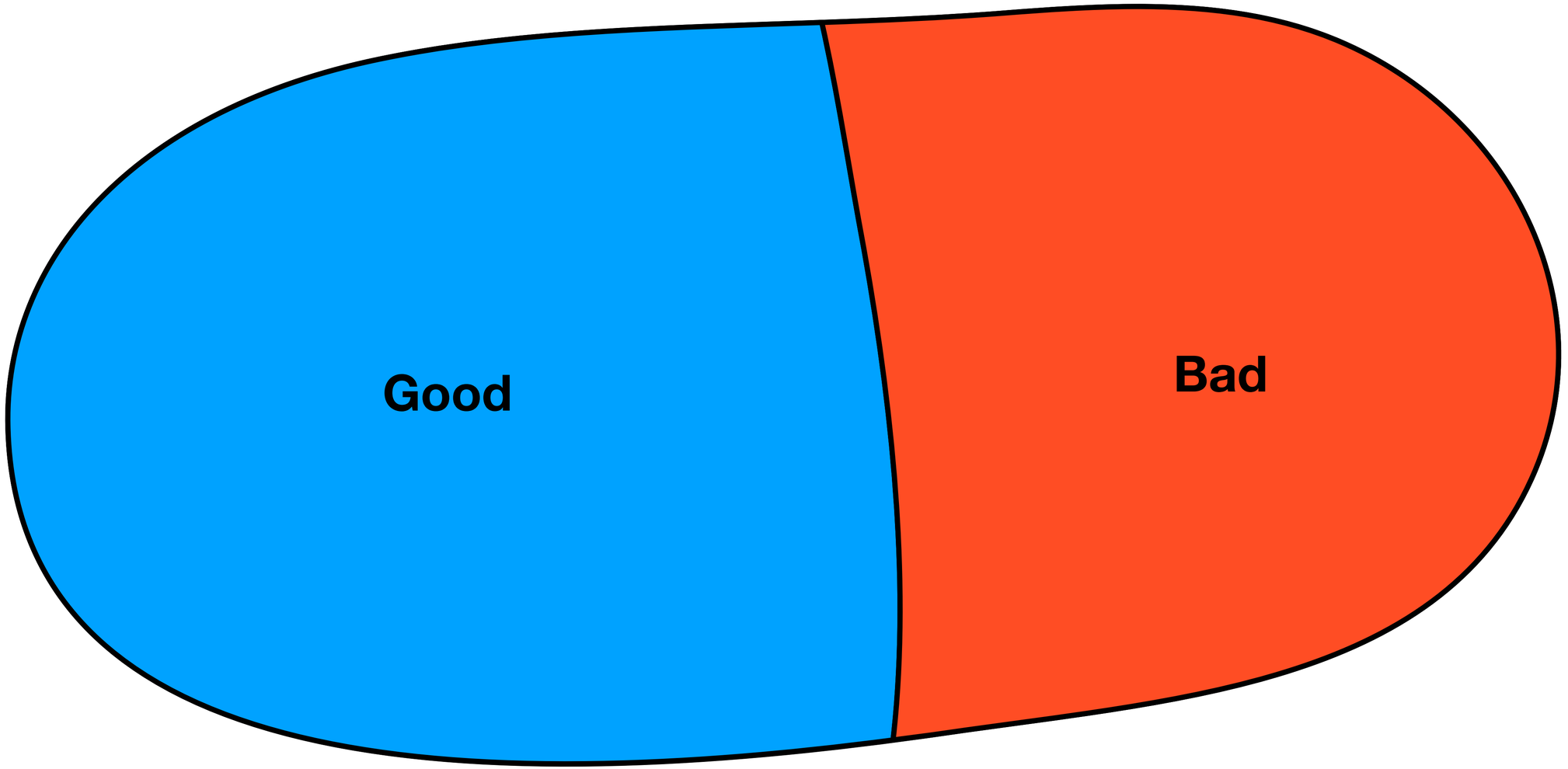
Otherwise v is **bad**

Classification So Far

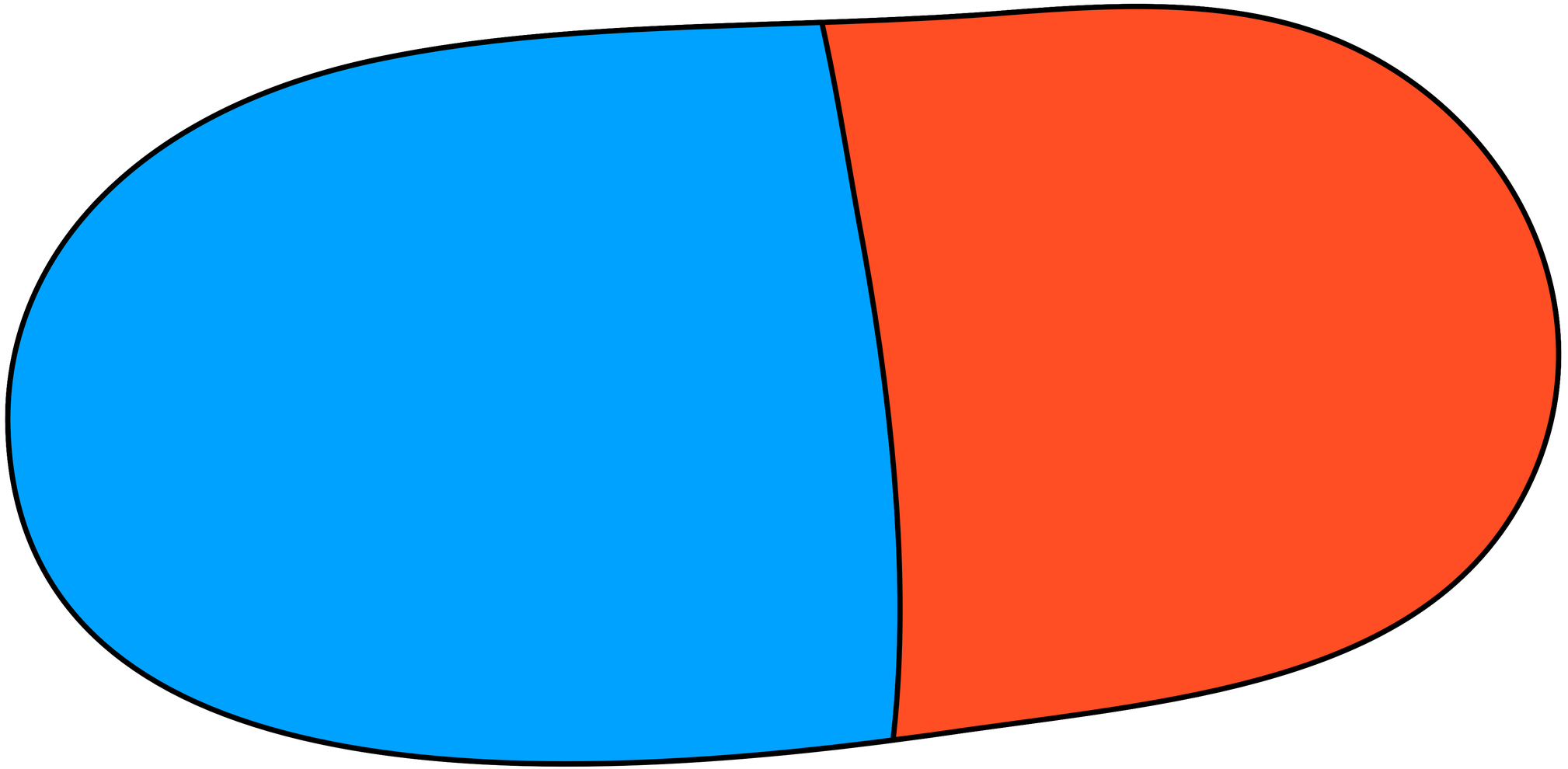
Classification So Far



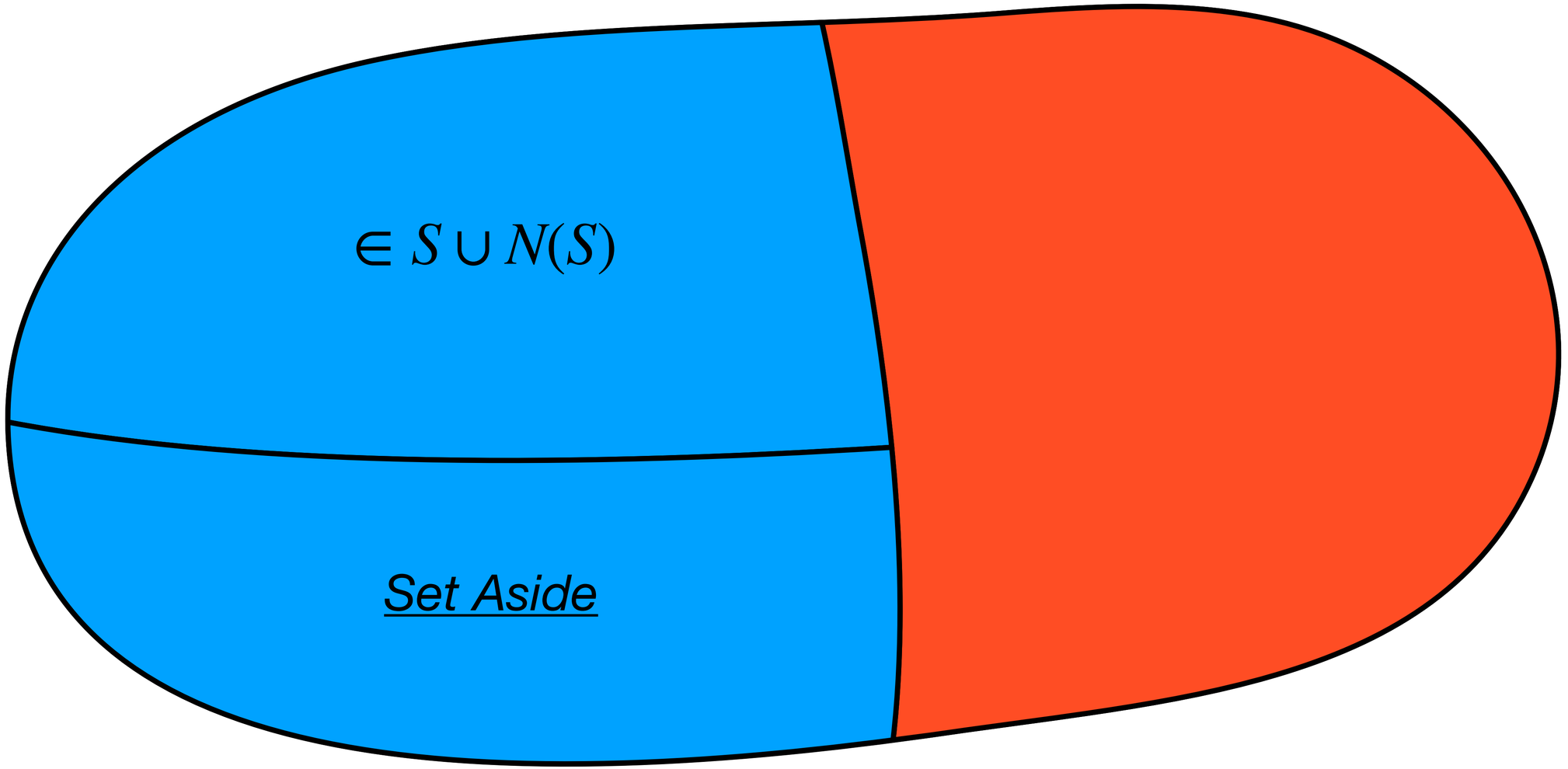
Classification So Far



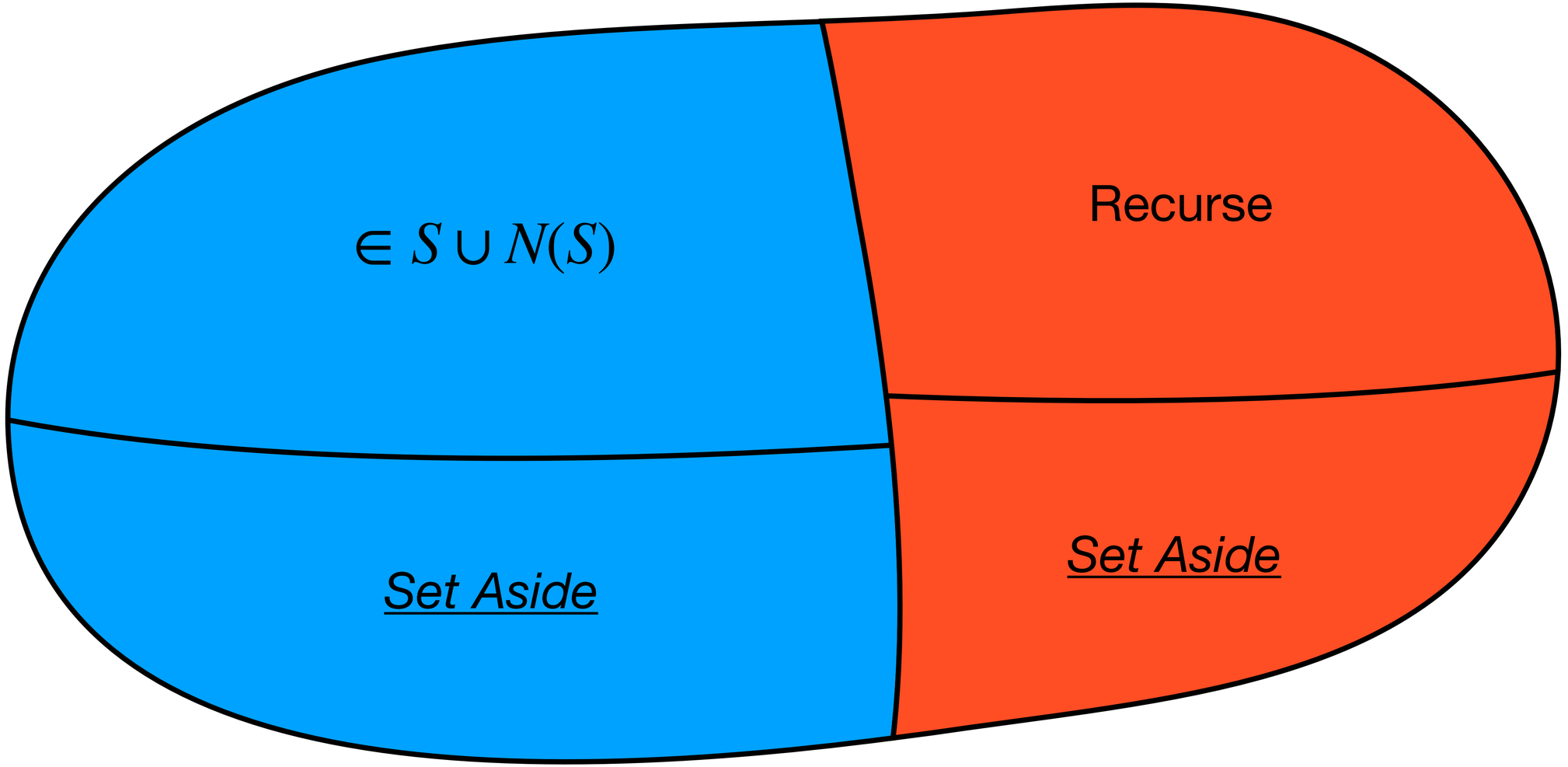
Classification So Far



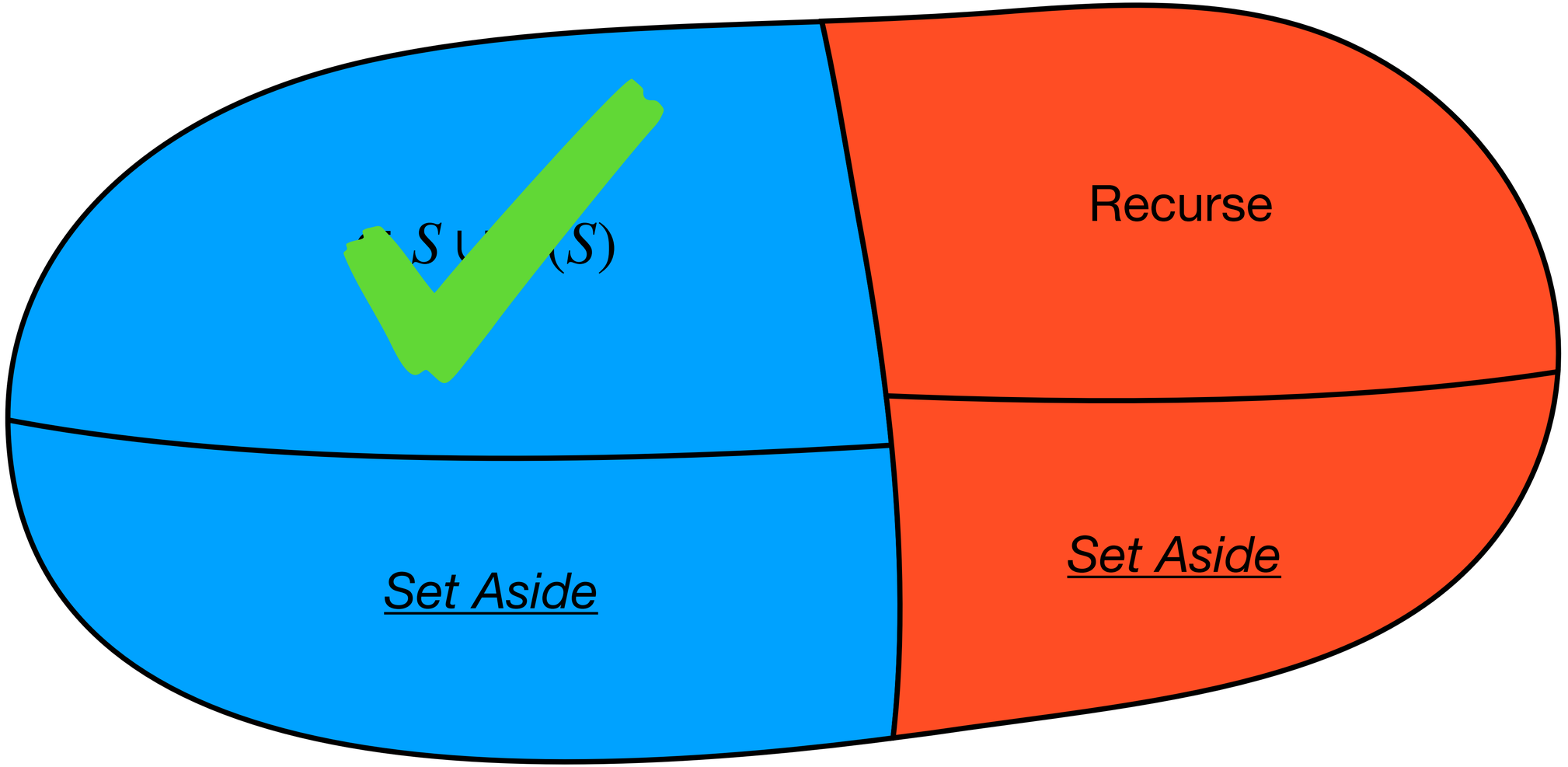
Classification So Far



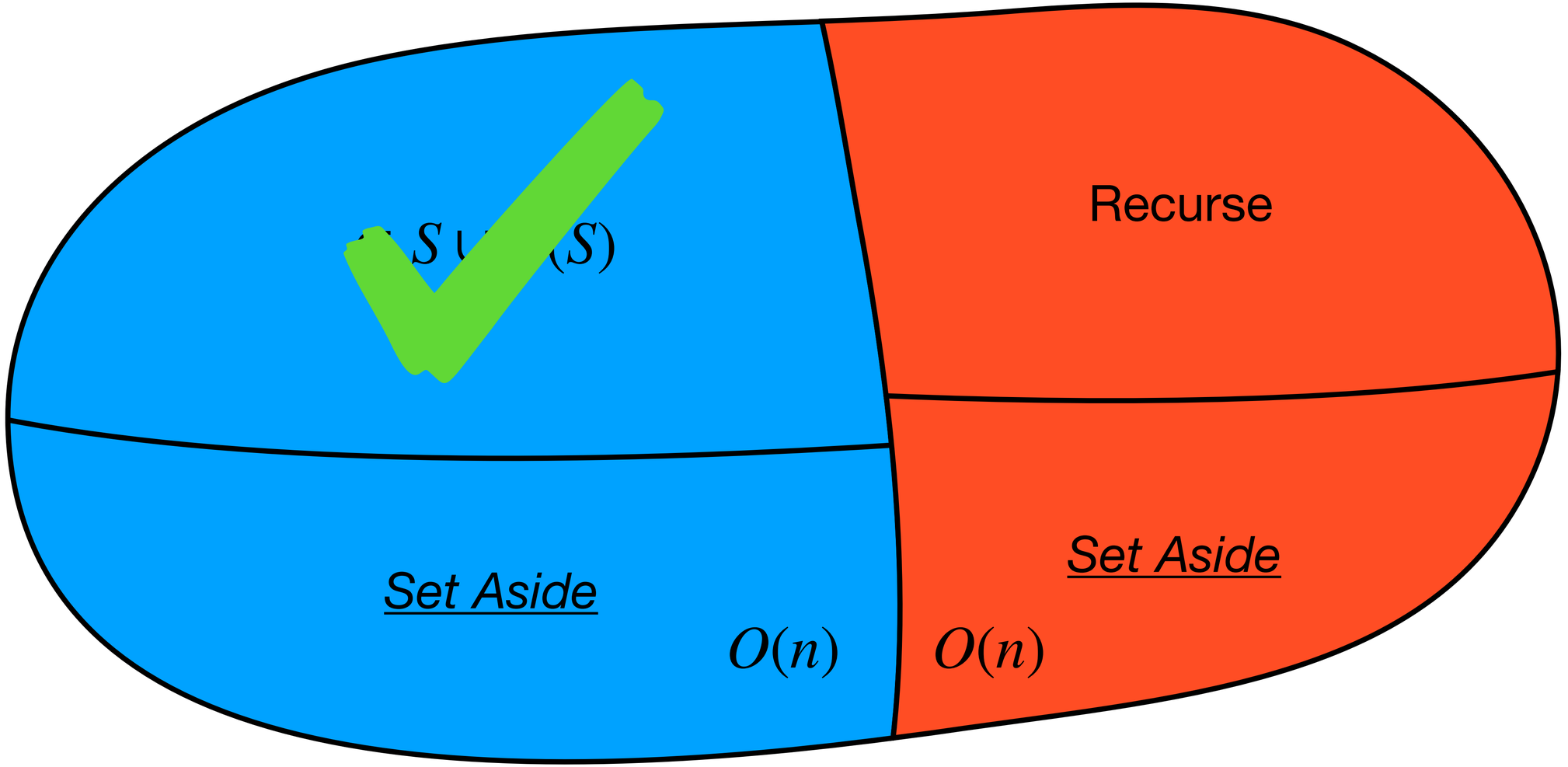
Classification So Far



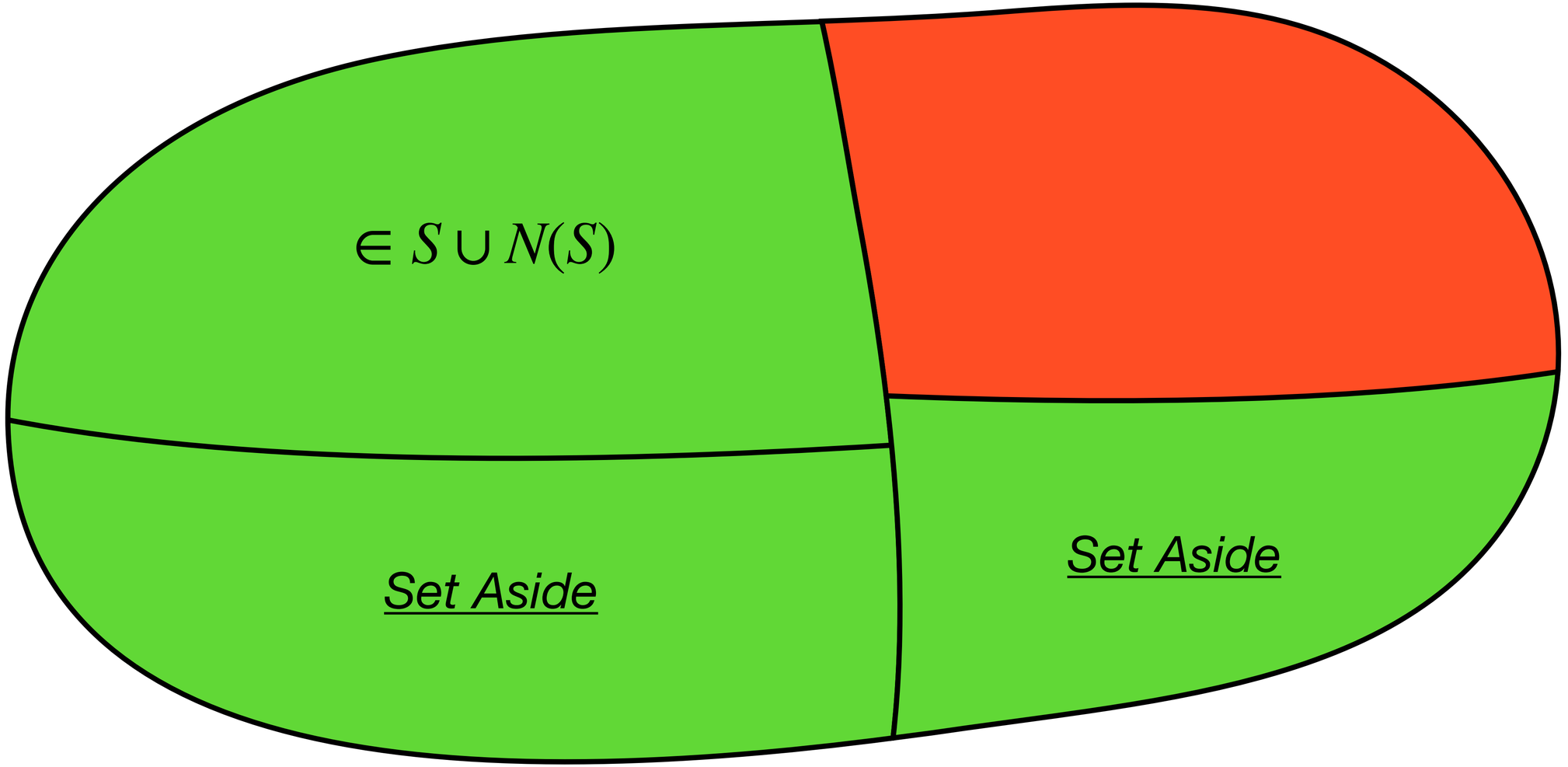
Classification So Far



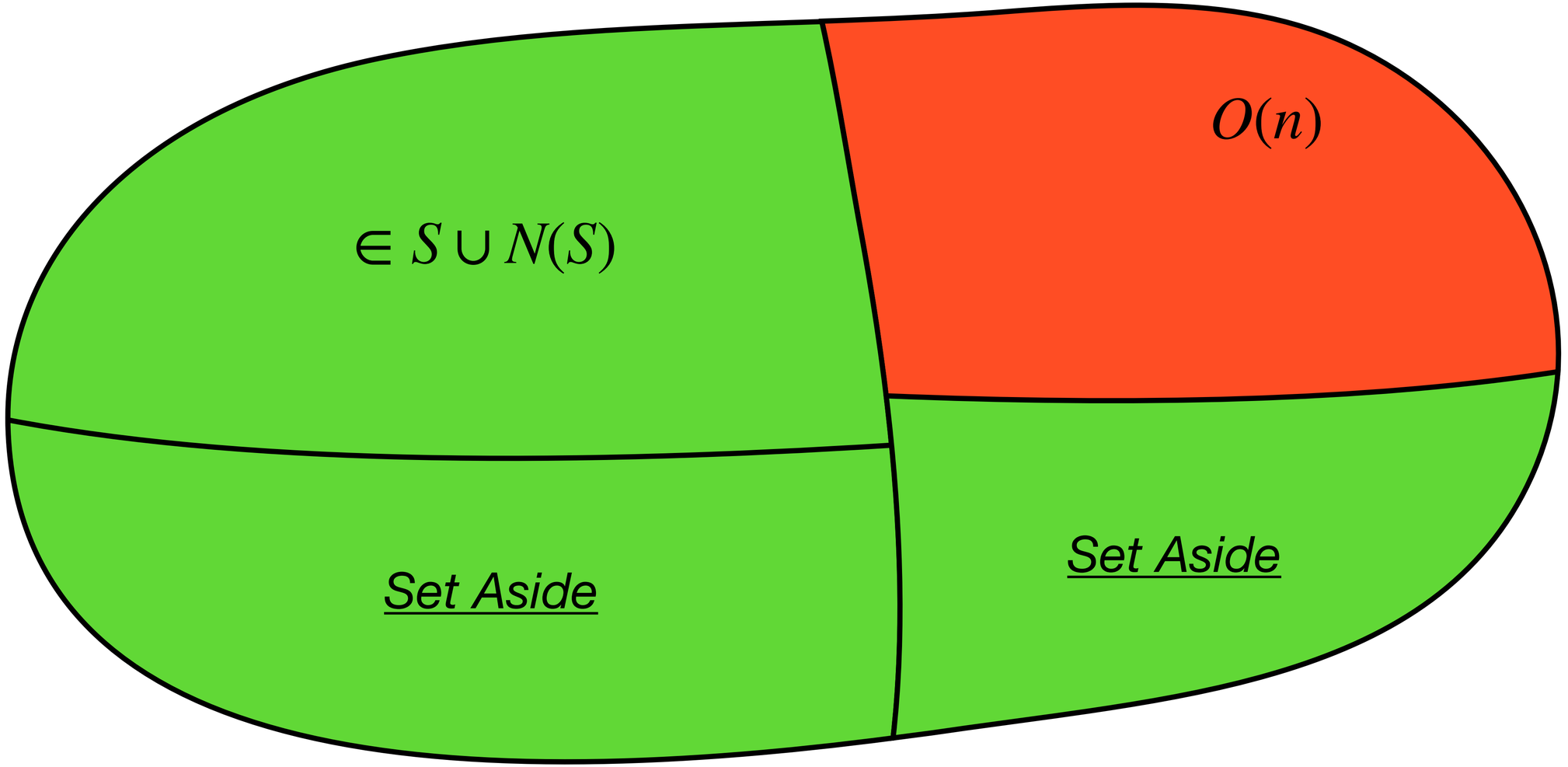
Classification So Far



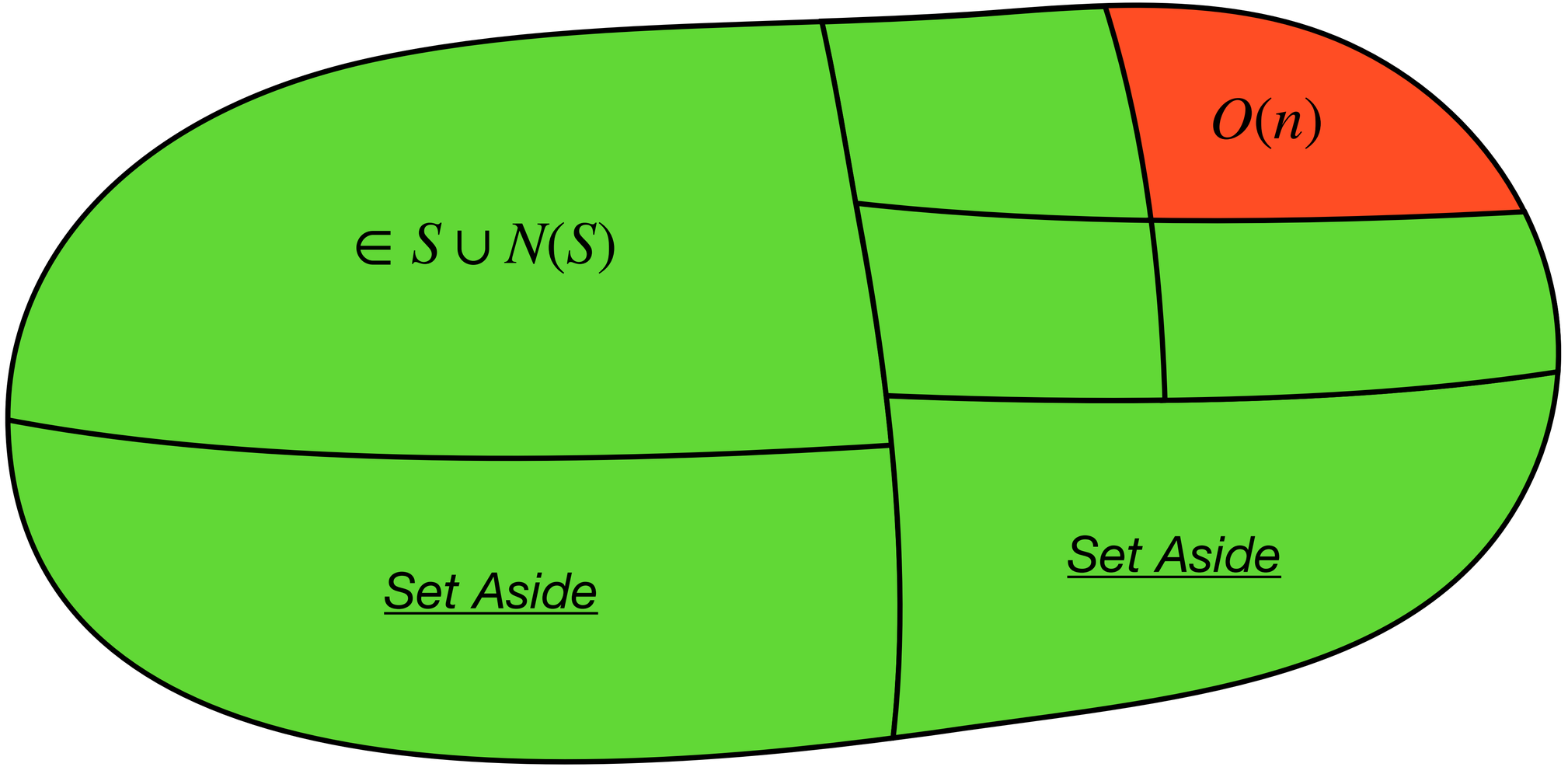
Classification So Far



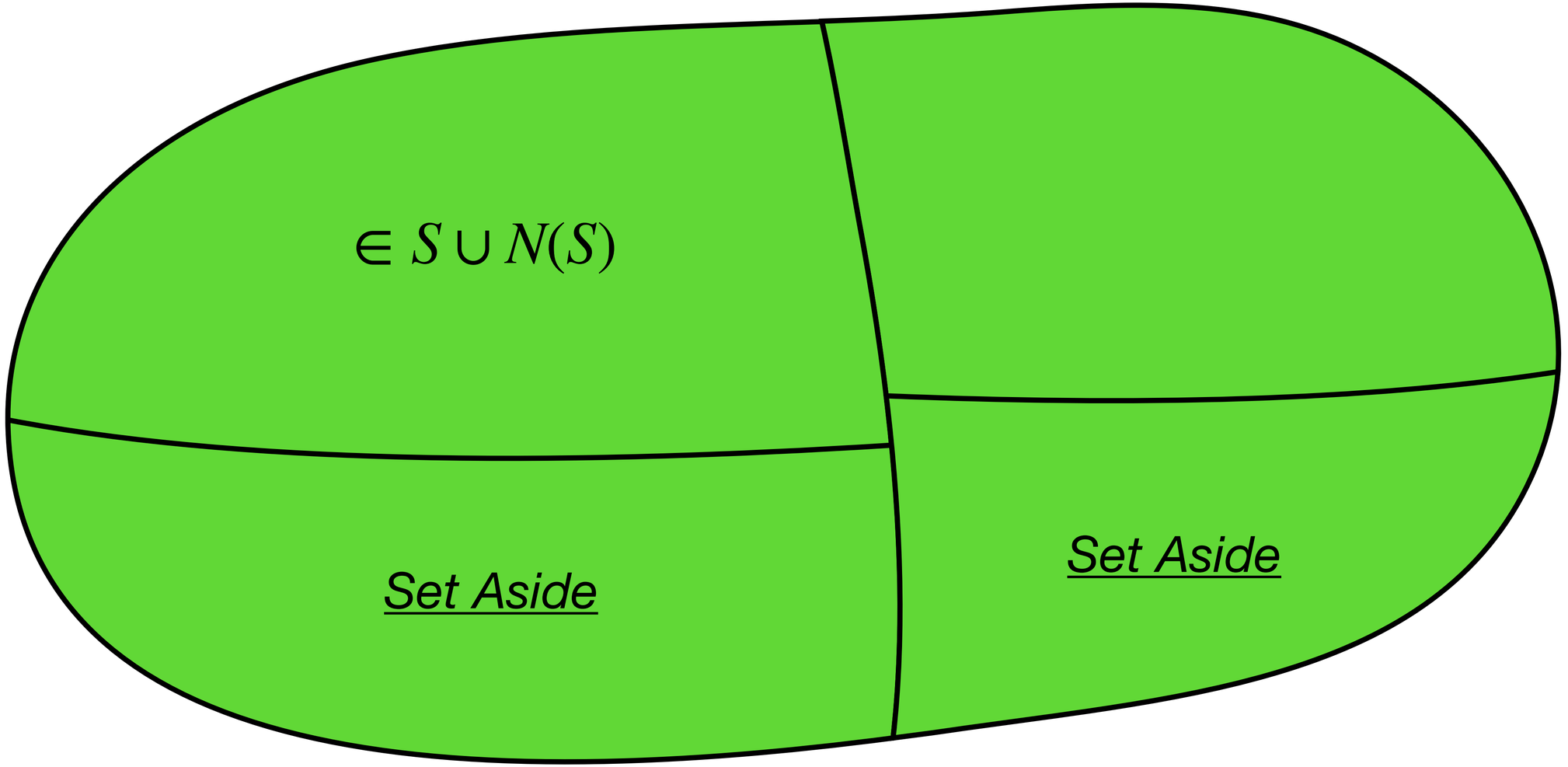
Classification So Far



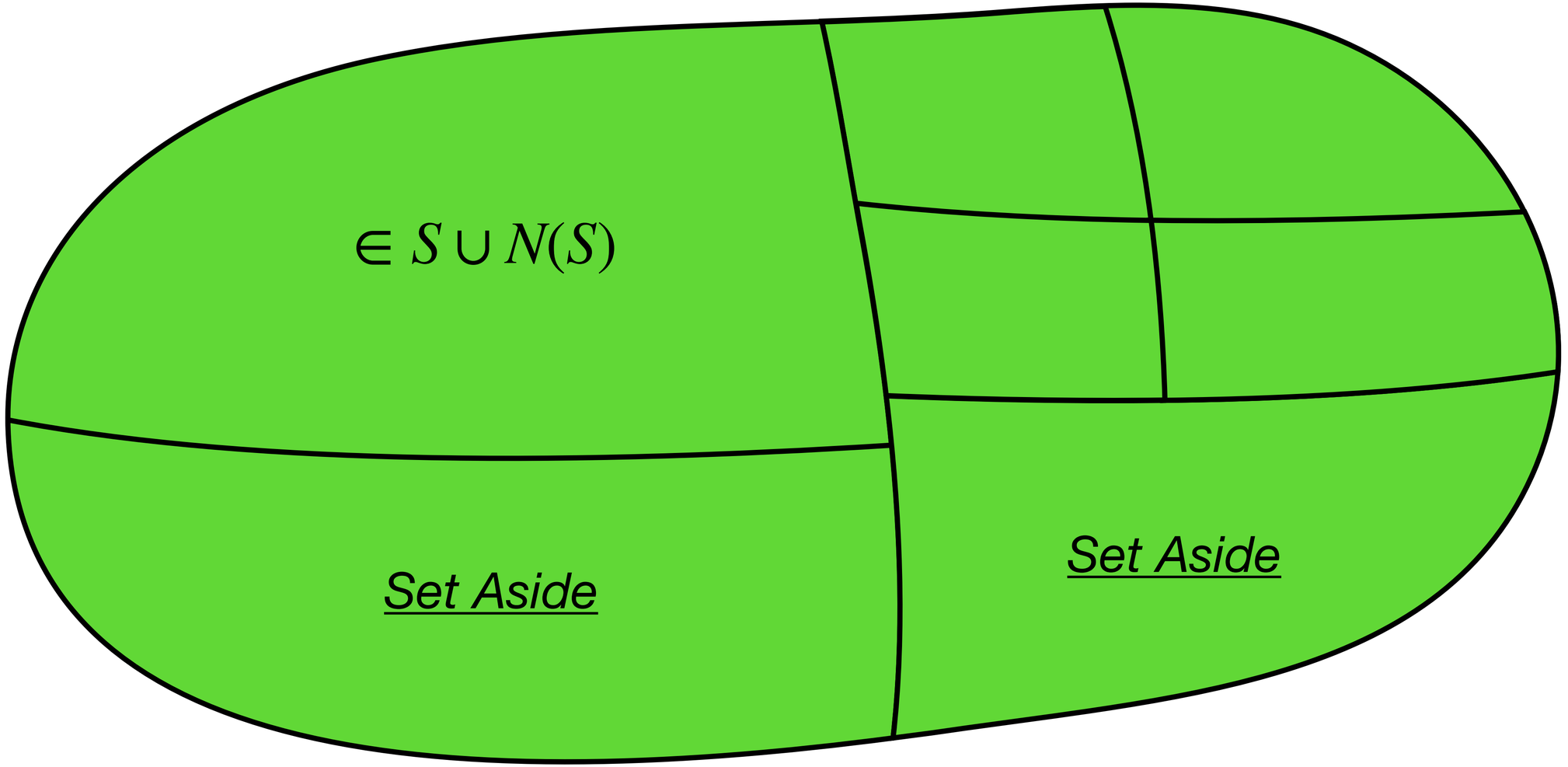
Classification So Far



Final Classification



Final Classification



Ruling Set Summary

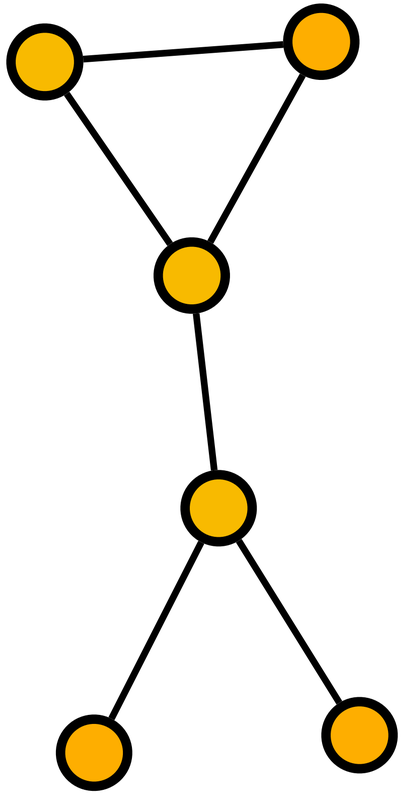
A randomized las-vegas algorithm that computes a 2-ruling set.

Can be implemented in:

- Linear-memory MPC: $O(1)$ rounds whp.
- Congested Clique: $O(1)$ rounds whp.
- Streaming: $O(1)$ passes whp, with $O(n)$ space (insertion-only).

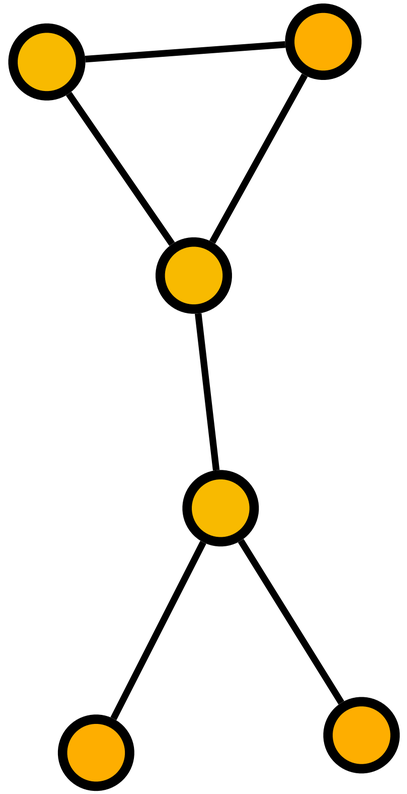
Relaxation 2: Correlation Clustering

Relaxation 2: Correlation Clustering



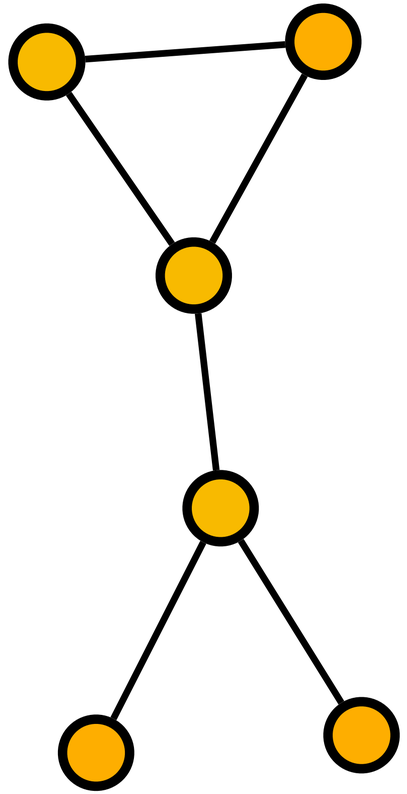
Relaxation 2: Correlation Clustering

- Edges of input graph encode pairwise similarity.



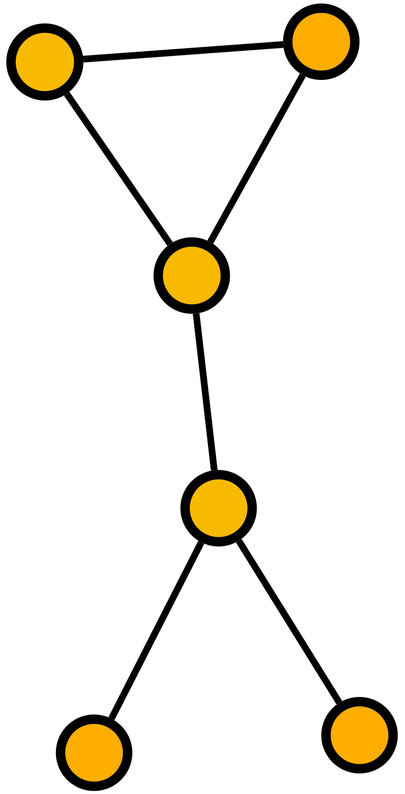
Relaxation 2: Correlation Clustering

- Edges of input graph encode pairwise similarity.
- Non-edges encode pairwise dis-similarity.



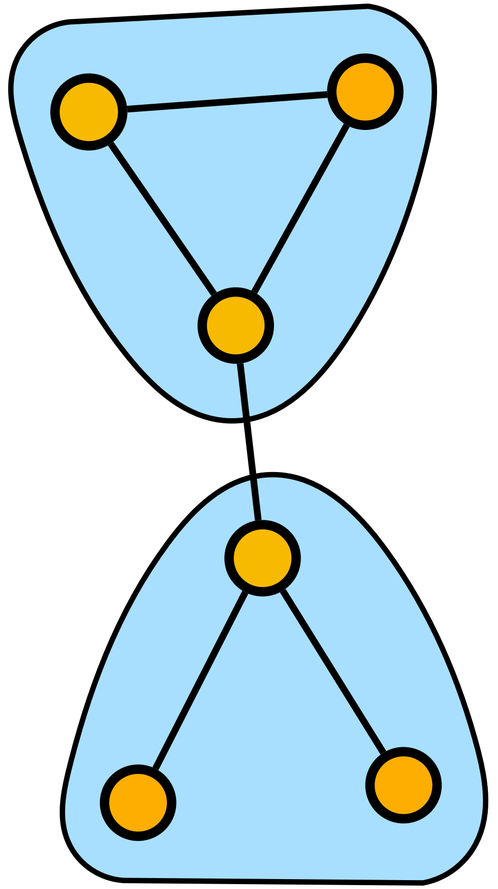
Relaxation 2: Correlation Clustering

- Edges of input graph encode pairwise similarity.
- Non-edges encode pairwise dis-similarity.
- We need to cluster the nodes so that the total **disagreement** is minimized.



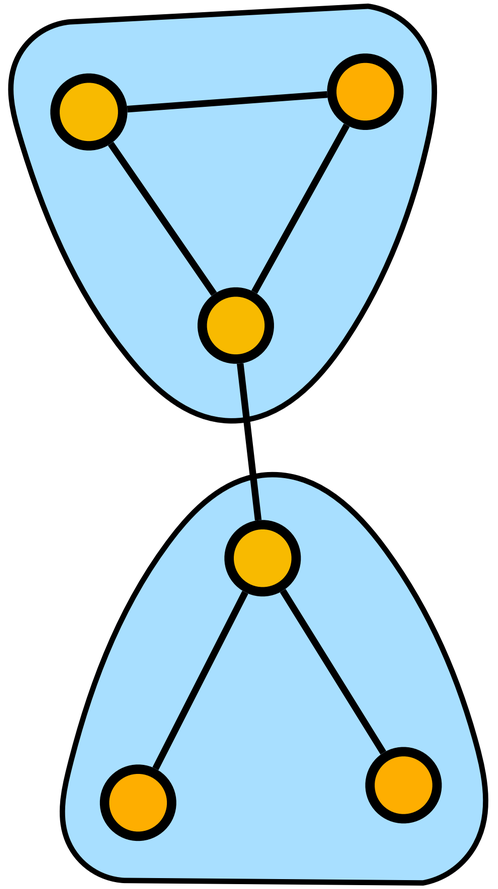
Relaxation 2: Correlation Clustering

- Edges of input graph encode pairwise similarity.
- Non-edges encode pairwise dis-similarity.
- We need to cluster the nodes so that the total **disagreement** is minimized.



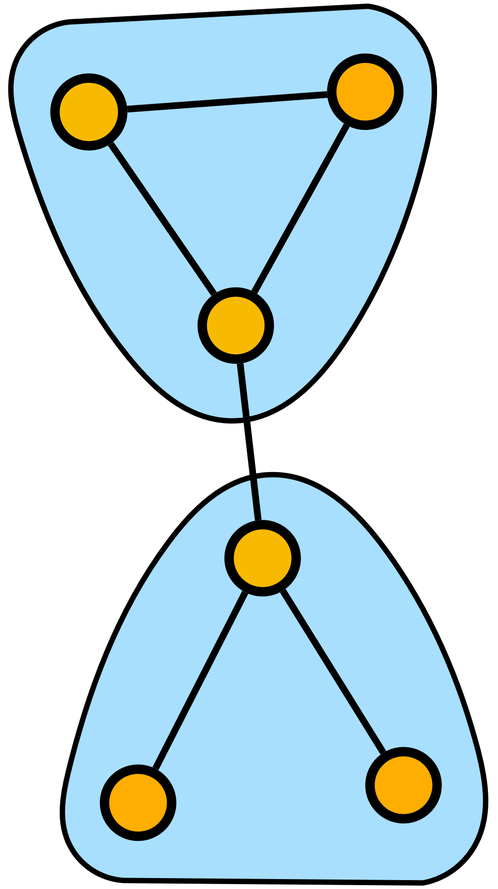
Relaxation 2: Correlation Clustering

- Edges of input graph encode pairwise similarity.
- Non-edges encode pairwise dis-similarity.
- We need to cluster the nodes so that the total **disagreement** is minimized.
    - Number of edges across different clusters.



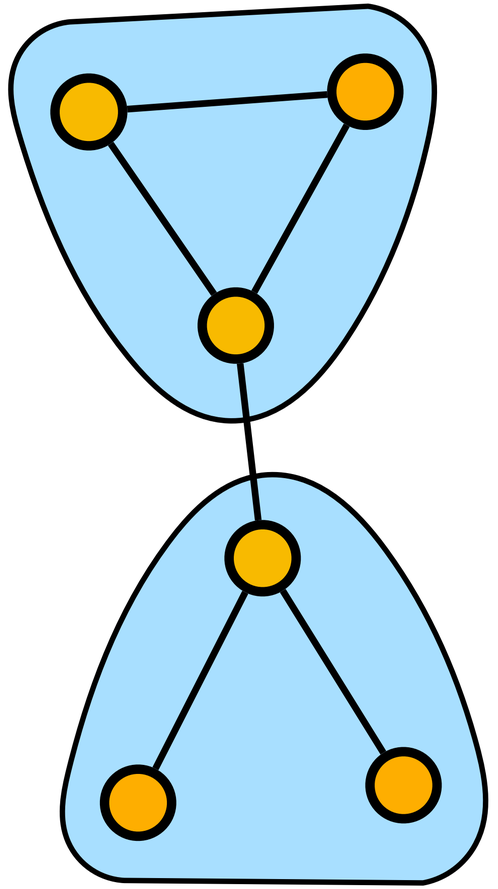
Relaxation 2: Correlation Clustering

- Edges of input graph encode pairwise similarity.
- Non-edges encode pairwise dis-similarity.
- We need to cluster the nodes so that the total **disagreement** is minimized.
    - Number of edges across different clusters.



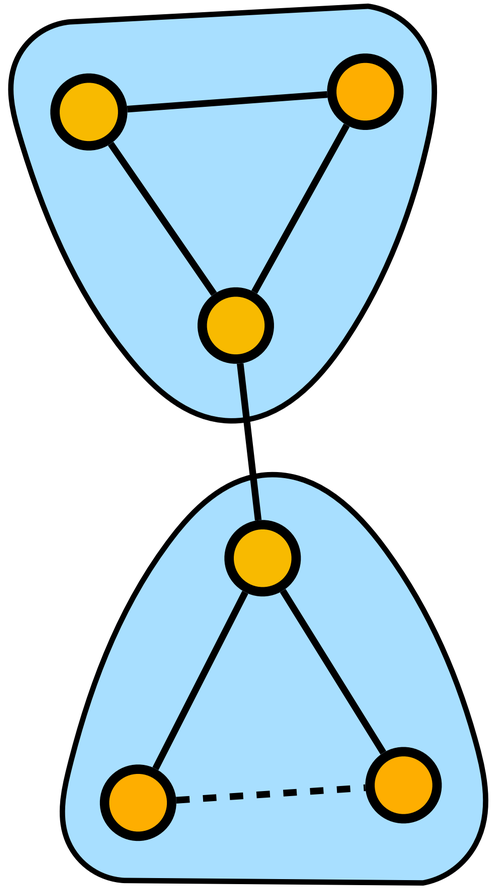
Relaxation 2: Correlation Clustering

- Edges of input graph encode pairwise similarity.
- Non-edges encode pairwise dis-similarity.
- We need to cluster the nodes so that the total **disagreement** is minimized.
    - Number of edges across different clusters.
    - Number of non-edges in the same cluster.



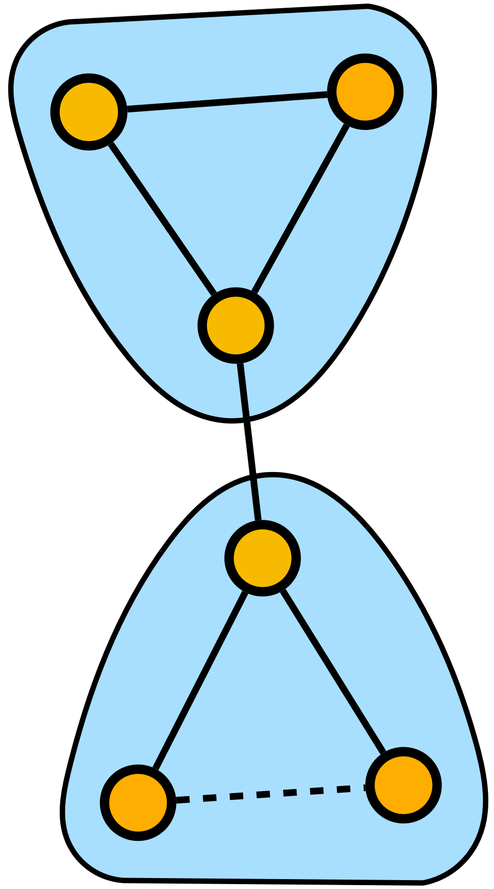
Relaxation 2: Correlation Clustering

- Edges of input graph encode pairwise similarity.
- Non-edges encode pairwise dis-similarity.
- We need to cluster the nodes so that the total **disagreement** is minimized.
    - Number of edges across different clusters.
    - Number of non-edges in the same cluster.

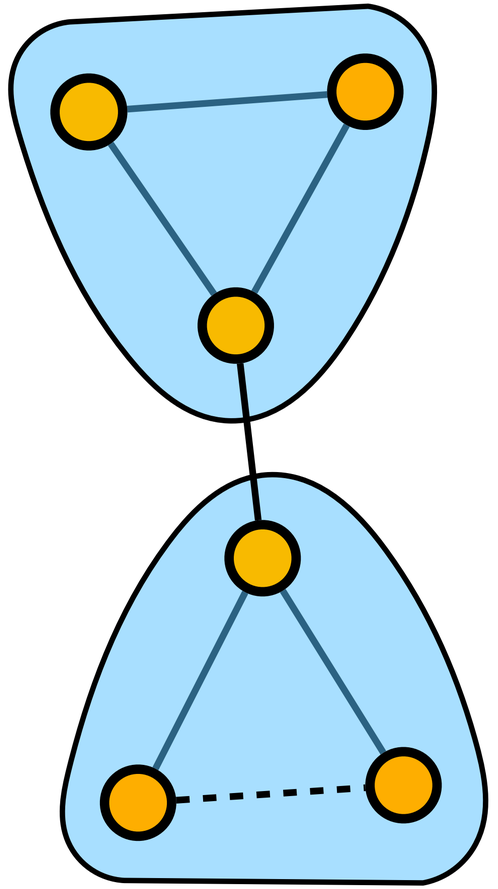


Relaxation 2: Correlation Clustering

- Edges of input graph encode pairwise similarity.
- Non-edges encode pairwise dis-similarity.
- We need to cluster the nodes so that the total **disagreement** is minimized.
    - Number of edges across different clusters.
    - Number of non-edges in the same cluster.
- No constraint on number of clusters.



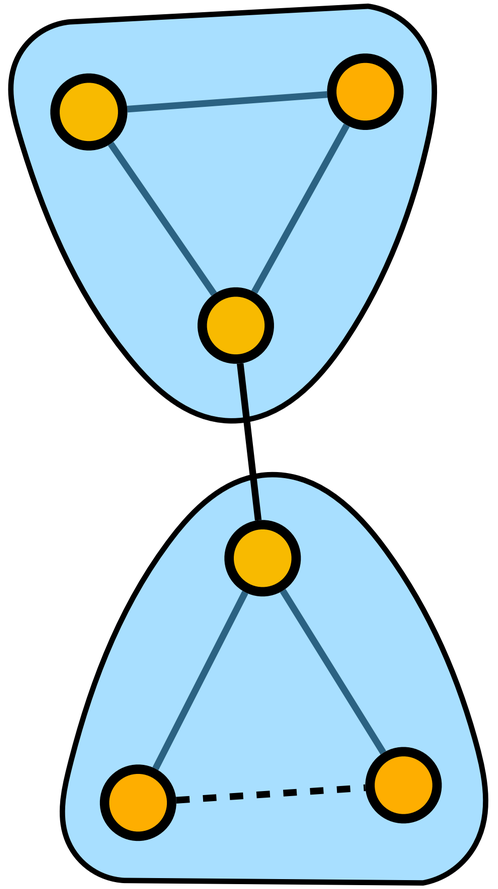
Correlation Clustering



Correlation Clustering

- Natural abstraction for problems such as

  Community and duplicate detection,
  Link prediction,
  Image segmentation.

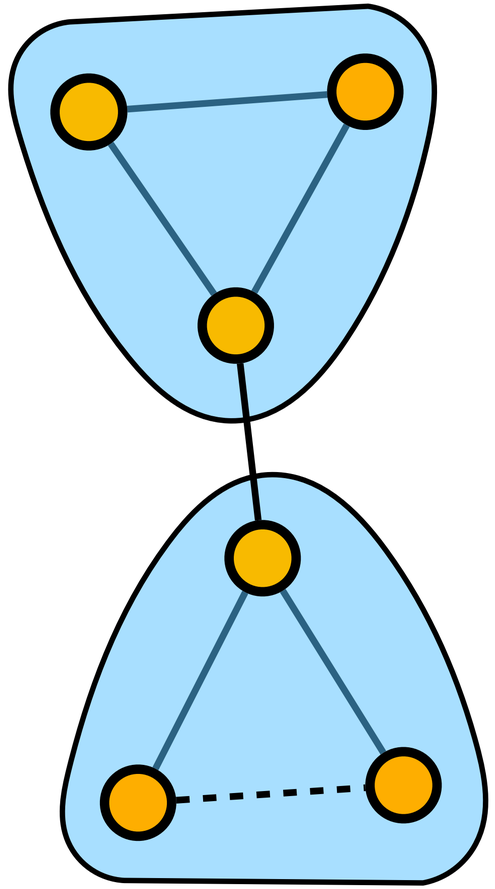


Correlation Clustering

- Natural abstraction for problems such as

  Community and duplicate detection,
  Link prediction,
  Image segmentation.

- APX-hard [CGW05].



[CGW05] Charikar, Guruswami, Wirth Journal of Computer and System Sciences 2005

Correlation Clustering

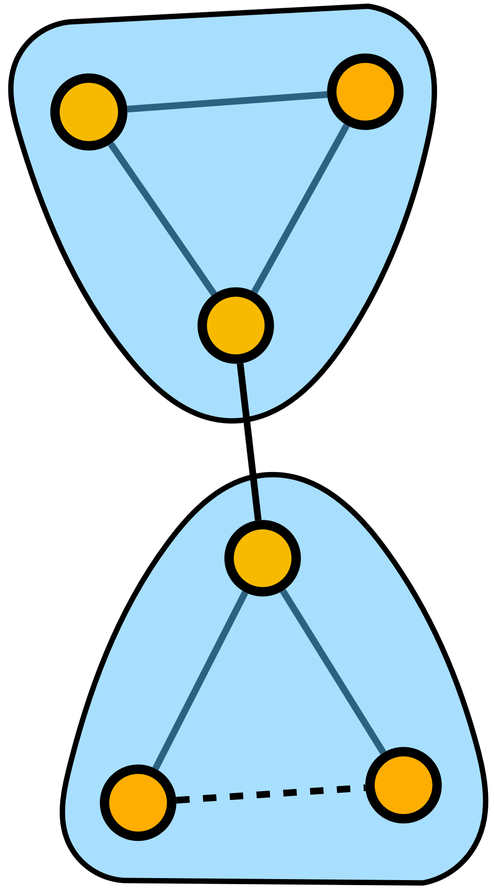
- Natural abstraction for problems such as

  Community and duplicate detection,
  Link prediction,
  Image segmentation.

- APX-hard [CGW05].

- Best known approximation ratio is 1.43 [CCL+24].

  - Rounding a linear programming relaxation.



[CGW05] Charikar, Guruswami, Wirth Journal of Computer and System Sciences 2005
[CCL+24] Cao, Cohen-Addad, Lee, Li, Newman, Vogl STOC 2024

Correlation Clustering

- Natural abstraction for problems such as

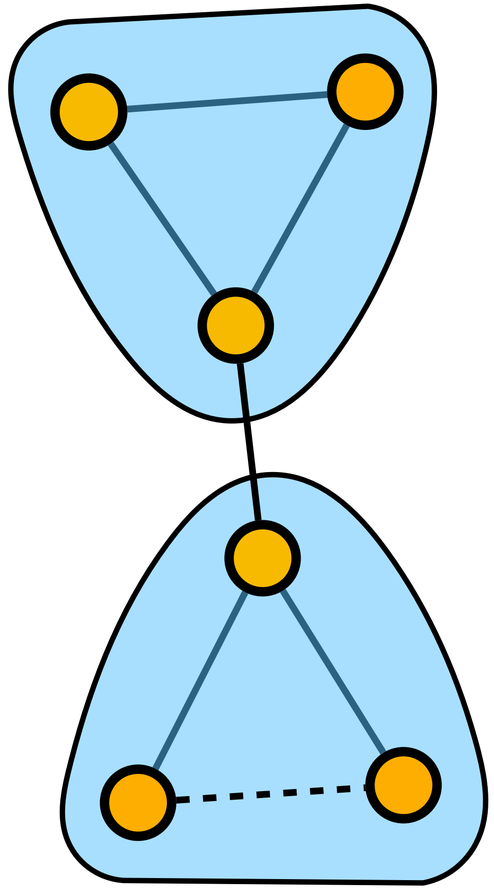
  Community and duplicate detection,
  Link prediction,
  Image segmentation.

- APX-hard [CGW05].

- Best known approximation ratio is 1.43 [CCL+24].

  - Rounding a linear programming relaxation.

- Can achieve < 1.847 approximation in MPC [CLP+24].



[CGW05] Charikar, Guruswami, Wirth Journal of Computer and System Sciences 2005

[CCL+24] Cao, Cohen-Addad, Lee, Li, Newman, Vogl STOC 2024

[CLP+24] Cohen-Addad, Lolck, Pilipczuk, Thorup, Yan, and Zhang. STOC 2024

Pivot Algorithm

Pivot Algorithm

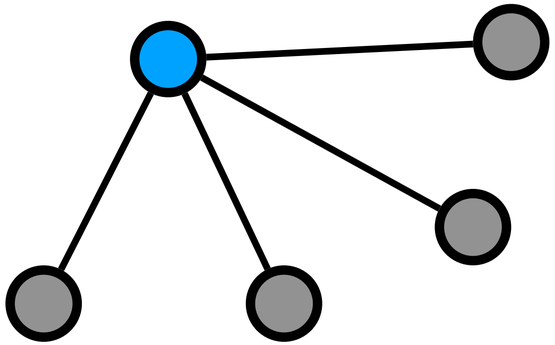
Random Permutation π :

Pivot Algorithm

Random Permutation π :

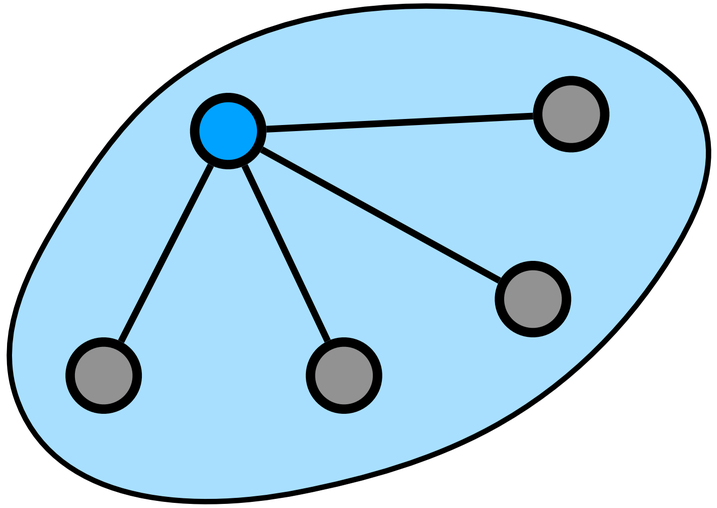
Pivot Algorithm

Random Permutation π :



Pivot Algorithm

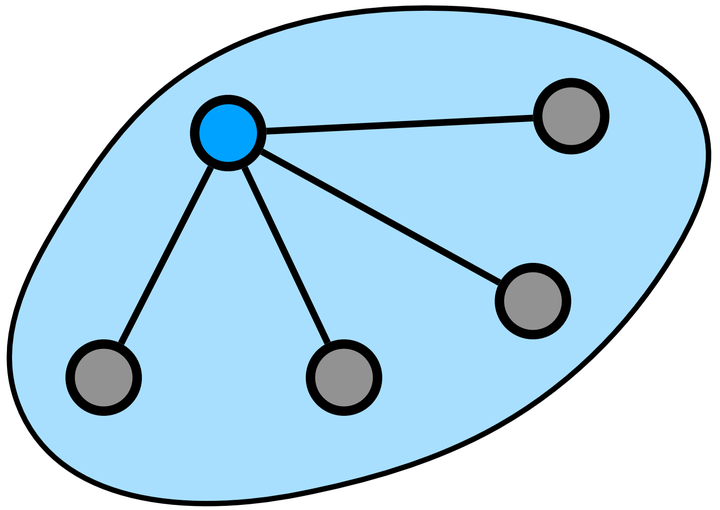
Random Permutation π :



Pivot Algorithm

Random Permutation π :

- Expected cost is a 3-approximation [ACN08].

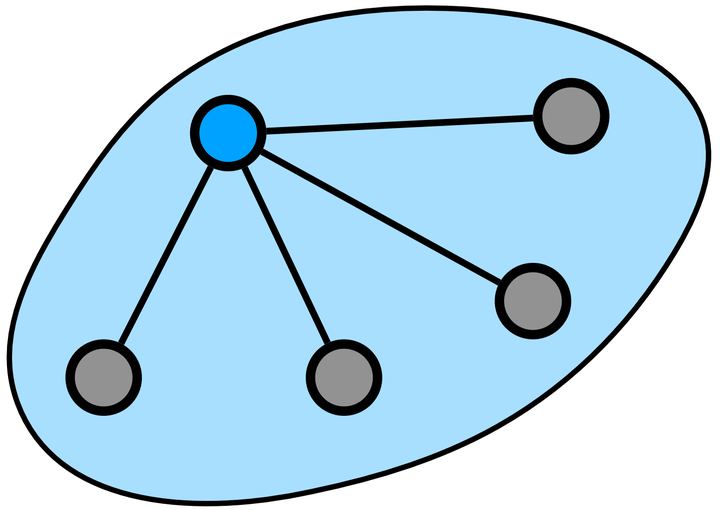


[ACN08] Ailon, Charikar, Newman, JACM 2008

Pivot Algorithm

Random Permutation π :

- Expected cost is a 3-approximation [ACN08].
- The set of pivot nodes forms an MIS.

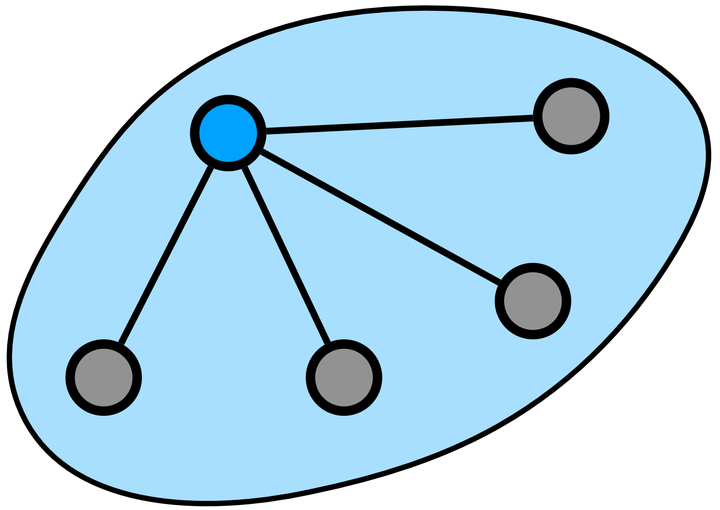


[ACN08] Ailon, Charikar, Newman, JACM 2008

Pivot Algorithm

Random Permutation π :

- Expected cost is a 3-approximation [ACN08].
- The set of pivot nodes forms an MIS.
- How to decide u's cluster?

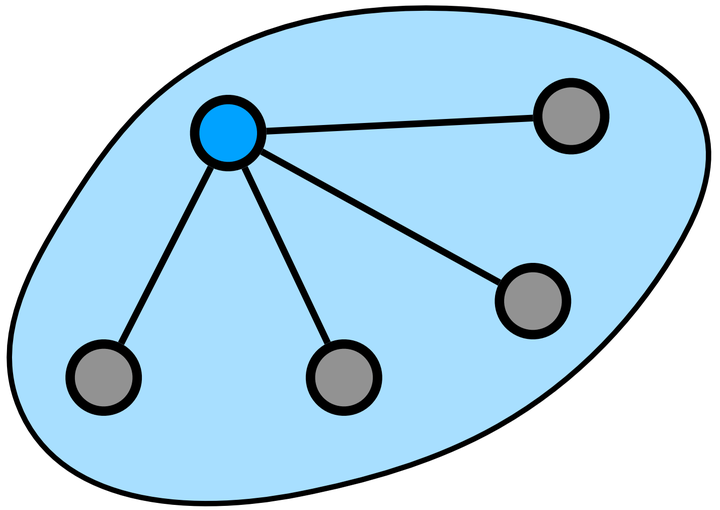


[ACN08] Ailon, Charikar, Newman, JACM 2008

Pivot Algorithm

Random Permutation π :

- Expected cost is a 3-approximation [ACN08].
- The set of pivot nodes forms an MIS.
- How to decide u's cluster?
  - Need to store all preceding neighbors.

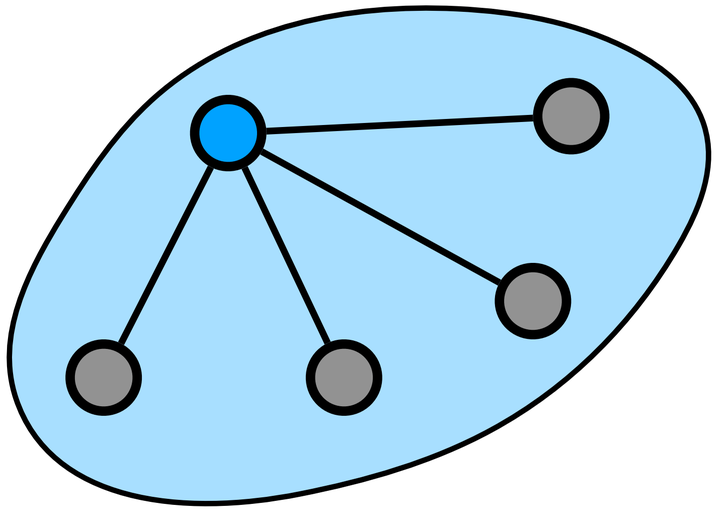


[ACN08] Ailon, Charikar, Newman, JACM 2008

Pivot Algorithm

Random Permutation π :

- Expected cost is a 3-approximation [ACN08].
- The set of pivot nodes forms an MIS.
- How to decide u's cluster?
  - Need to store all preceding neighbors.
- Can we ignore some nodes to save space?



[ACN08] Ailon, Charikar, Newman, JACM 2008

Truncated-Pivot Algorithm

Truncated-Pivot Algorithm

Random Permutation π :

Truncated-Pivot Algorithm

Random Permutation π :

$$\tau_u \approx \frac{n \log n}{\varepsilon \deg(u)}$$

Truncated-Pivot Algorithm

Random Permutation π :

- Node u is **interesting** if its rank is at most τ_u.

$$\tau_u \approx \frac{n \log n}{\varepsilon \deg(u)}$$

Truncated-Pivot Algorithm

Random Permutation π :

- Node u is **interesting** if its rank is at most τ_u.
- Run Pivot only on interesting nodes.

$$\tau_u \approx \frac{n \log n}{\varepsilon \deg(u)}$$

Truncated-Pivot Algorithm

Random Permutation π :

- Node u is **interesting** if its rank is at most τ_u.
- Run Pivot only on interesting nodes.
    - Only need to store incident edges of interesting nodes.

$$\tau_u \approx \frac{n \log n}{\varepsilon \deg(u)}$$

Truncated-Pivot Algorithm

Random Permutation π :

- Node u is **interesting** if its rank is at most τ_u.
- Run Pivot only on interesting nodes.
    - Only need to store incident edges of interesting nodes.
- How do we cluster an uninteresting node u?

$$\tau_u \approx \frac{n \log n}{\varepsilon \deg(u)}$$

Truncated-Pivot Algorithm

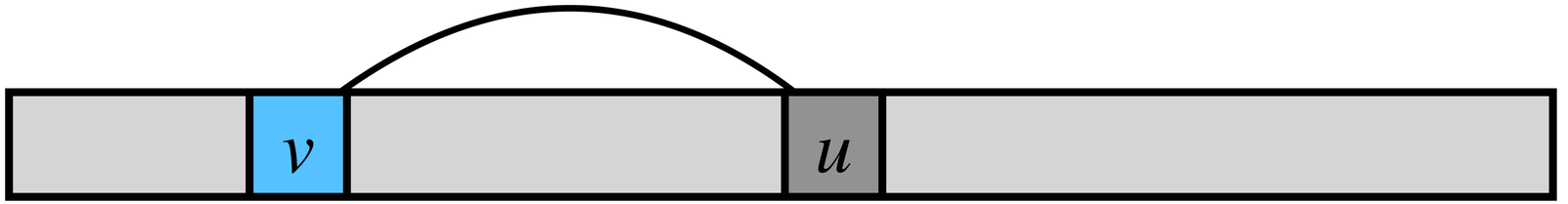
Random Permutation π :

- Node u is **interesting** if its rank is at most τ_u.
- Run Pivot only on interesting nodes.
    - Only need to store incident edges of interesting nodes.
- How do we cluster an uninteresting node u?
    - It either joins a pivot cluster or becomes a singleton.

$$\tau_u \approx \frac{n \log n}{\varepsilon \deg(u)}$$

Truncated-Pivot Algorithm

Random Permutation π :

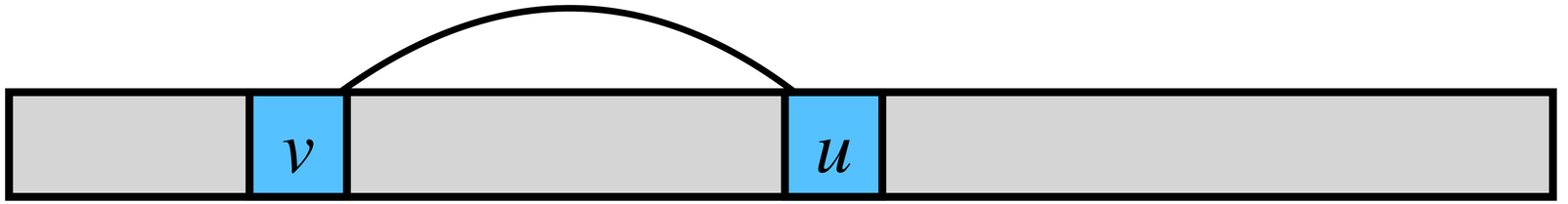


- Node u is **interesting** if its rank is at most τ_u.
- Run Pivot only on interesting nodes.
    - Only need to store incident edges of interesting nodes.
- How do we cluster an uninteresting node u?
    - It either joins a pivot cluster or becomes a singleton.

$$\tau_u \approx \frac{n \log n}{\varepsilon \deg(u)}$$

Truncated-Pivot Algorithm

Random Permutation π :



- Node u is **interesting** if its rank is at most τ_u.
- Run Pivot only on interesting nodes.
    - Only need to store incident edges of interesting nodes.
- How do we cluster an uninteresting node u?
    - It either joins a pivot cluster or becomes a singleton.

$$\tau_u \approx \frac{n \log n}{\varepsilon \deg(u)}$$

Truncated-Pivot Algorithm

Random Permutation π :

- Node u is **interesting** if its rank is at most τ_u.
- Run Pivot only on interesting nodes.
    - Only need to store incident edges of interesting nodes.
- How do we cluster an uninteresting node u?
    - It either joins a pivot cluster or becomes a singleton.

$$\tau_u \approx \frac{n \log n}{\varepsilon \deg(u)}$$

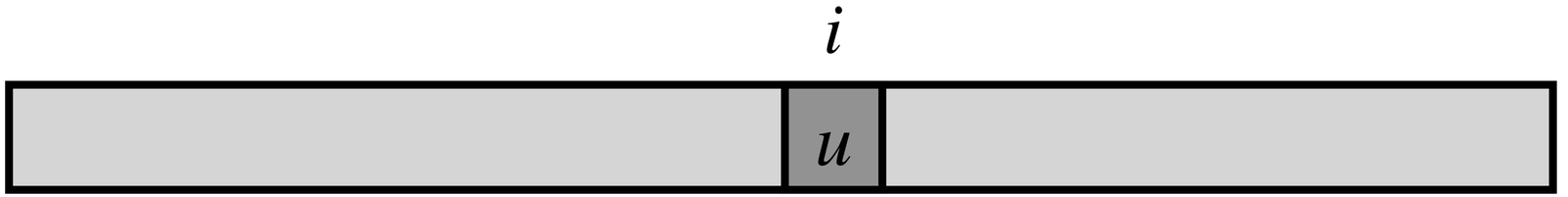
Space Analysis

Random Permutation π :

$$\tau_u \approx \frac{n \log n}{\varepsilon \deg(u)}$$

Space Analysis

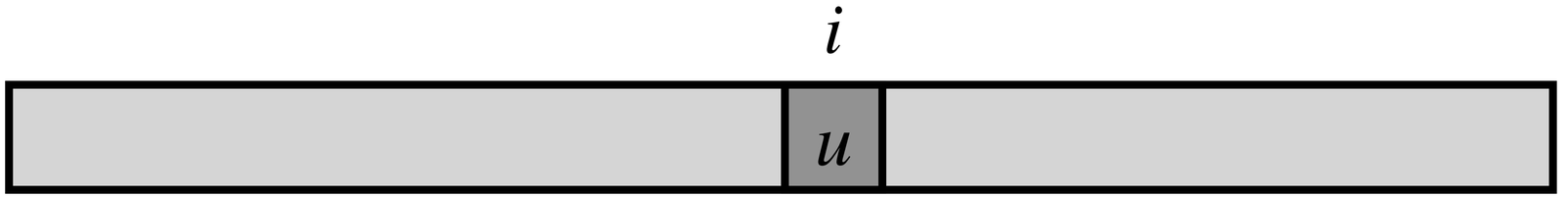
Random Permutation π :



$$\tau_u \approx \frac{n \log n}{\varepsilon \deg(u)}$$

Space Analysis

Random Permutation π :

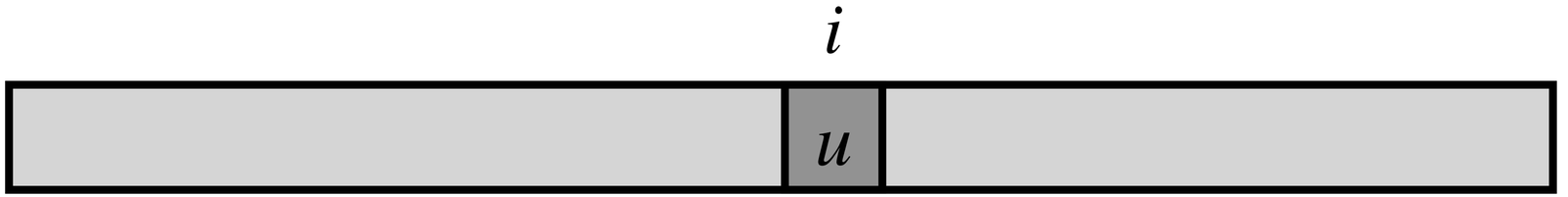


For u to be interesting: $i \leq \dfrac{n \log n}{\varepsilon \deg(u)}$

$$\tau_u \approx \frac{n \log n}{\varepsilon \deg(u)}$$

Space Analysis

Random Permutation π :



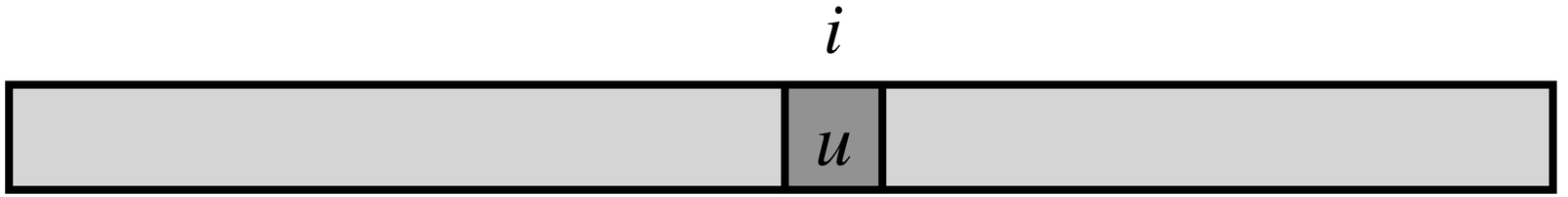
For u to be interesting: $i \leq \dfrac{n \log n}{\varepsilon \deg(u)}$

In other words: $\deg(u) \leq \dfrac{n \log n}{\varepsilon \cdot i}$

$$\tau_u \approx \frac{n \log n}{\varepsilon \deg(u)}$$

Space Analysis

Random Permutation π :



For u to be interesting: $i \leq \frac{n \log n}{\varepsilon \deg(u)}$

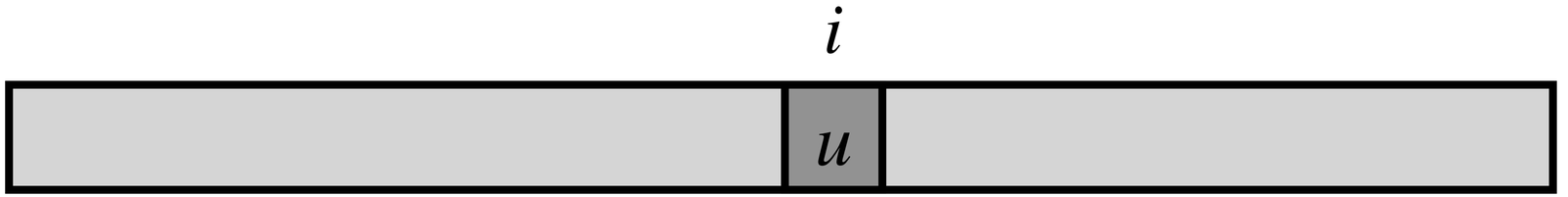
In other words: $\deg(u) \leq \frac{n \log n}{\varepsilon \cdot i}$

$$\tau_u \approx \frac{n \log n}{\varepsilon \deg(u)}$$

Total number of edges stored $= \frac{n \log n}{\varepsilon} \sum_{i=1}^{n} \frac{1}{i}$

Space Analysis

Random Permutation π :



For u to be interesting: $i \leq \frac{n \log n}{\varepsilon \deg(u)}$

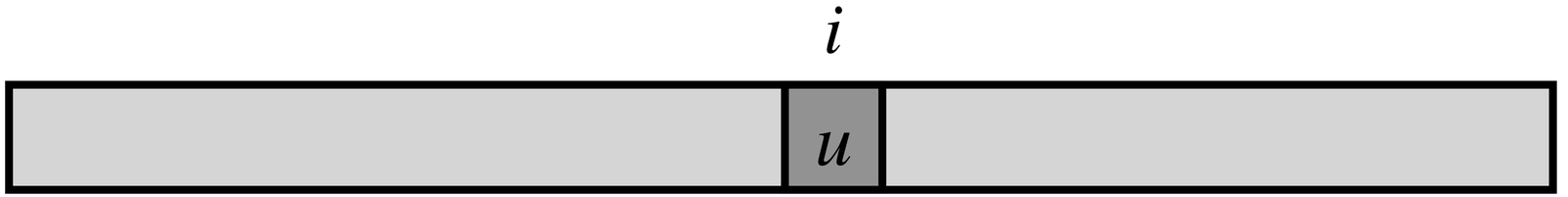
In other words: $\deg(u) \leq \frac{n \log n}{\varepsilon \cdot i}$

$$\tau_u \approx \frac{n \log n}{\varepsilon \deg(u)}$$

Total number of edges stored $= \frac{n \log n}{\varepsilon} \sum_{i=1}^{n} \frac{1}{i} \approx \frac{n \log^2 n}{\varepsilon}$

Space Analysis

Random Permutation π :



[CM23] improves this to $O(n/\varepsilon)$.

$$\tau_u \approx \frac{n \log n}{\varepsilon \deg(u)}$$

$$\text{Total number of edges stored} = \frac{n \log n}{\varepsilon} \sum_{i=1}^{n} \frac{1}{i} \approx \frac{n \log^2 n}{\varepsilon}$$

[CM23] Chakrabarty, Makarychev. NeurIPS 2023

Approximation Analysis

Approximation Analysis

Pivot Clusters

Approximation Analysis

Pivot Clusters

Singleton Clusters

Approximation Analysis

Pivot Clusters

Singleton Clusters

3-approximation

Approximation Analysis

Pivot Clusters

3-approximation

Singleton Clusters

“$+\varepsilon$”-approximation

Approximation Analysis

Pivot Clusters

3-approximation

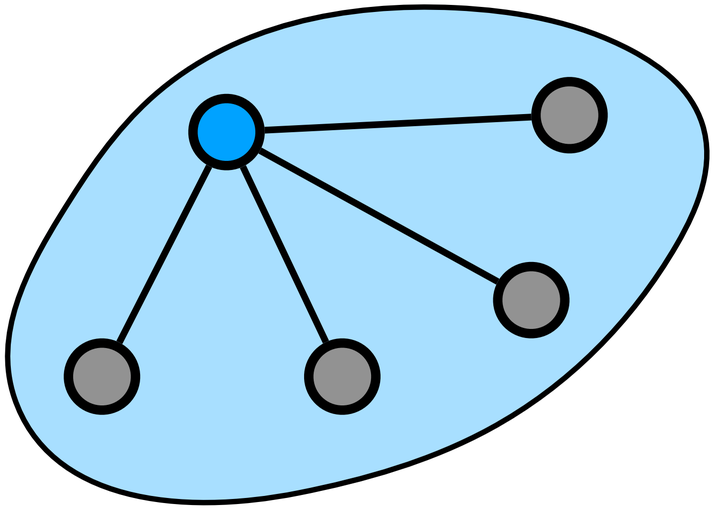
Singleton Clusters

“$+\varepsilon$”-approximation

Cost of Pivot Clusters

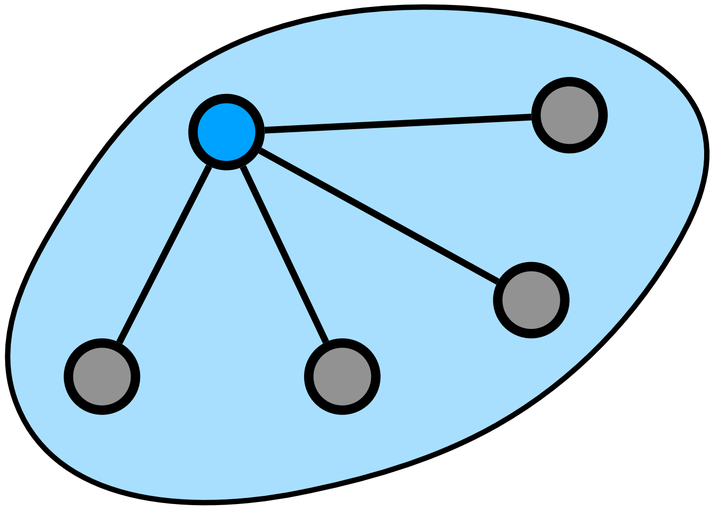
Cost of Pivot Clusters

- The Pivot Clusters created by Truncated-Pivot are different from the clusters created by Pivot.



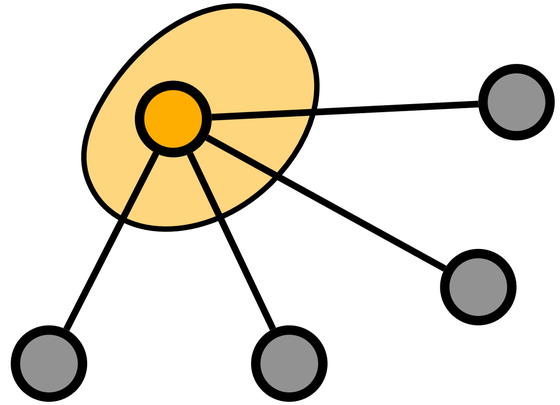
Cost of Pivot Clusters

- The Pivot Clusters created by Truncated-Pivot are different from the clusters created by Pivot.
    - We don’t allow uninteresting nodes to become pivots.



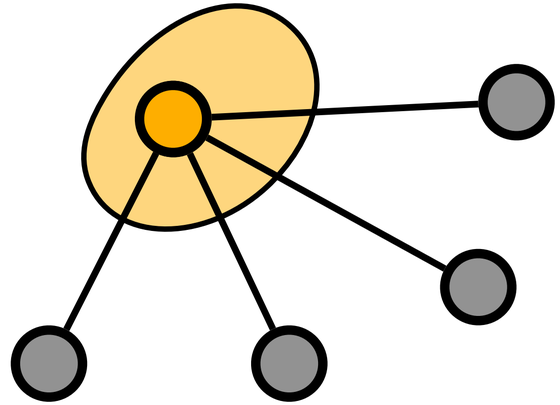
Cost of Pivot Clusters

- The Pivot Clusters created by Truncated-Pivot are different from the clusters created by Pivot.
    - We don’t allow uninteresting nodes to become pivots.



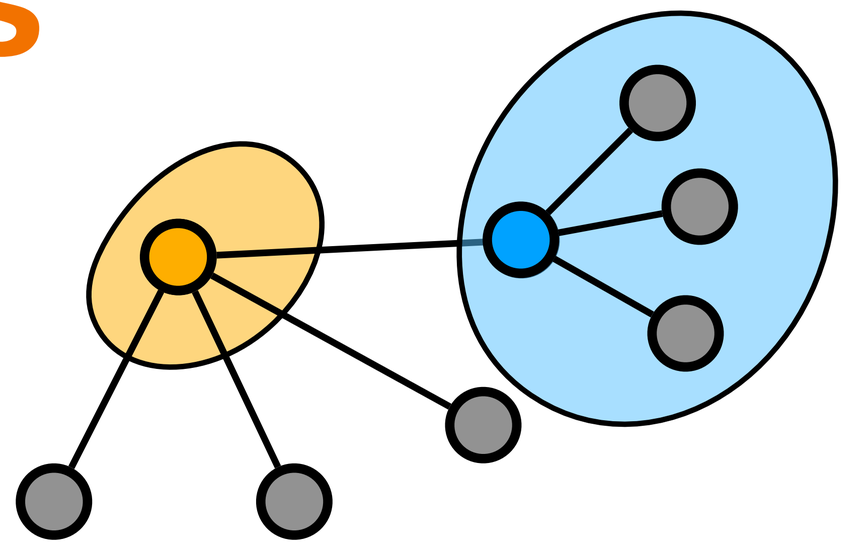
Cost of Pivot Clusters

- The Pivot Clusters created by Truncated-Pivot are different from the clusters created by Pivot.
    - We don’t allow uninteresting nodes to become pivots.
    - Their neighbors may create new pivot clusters.

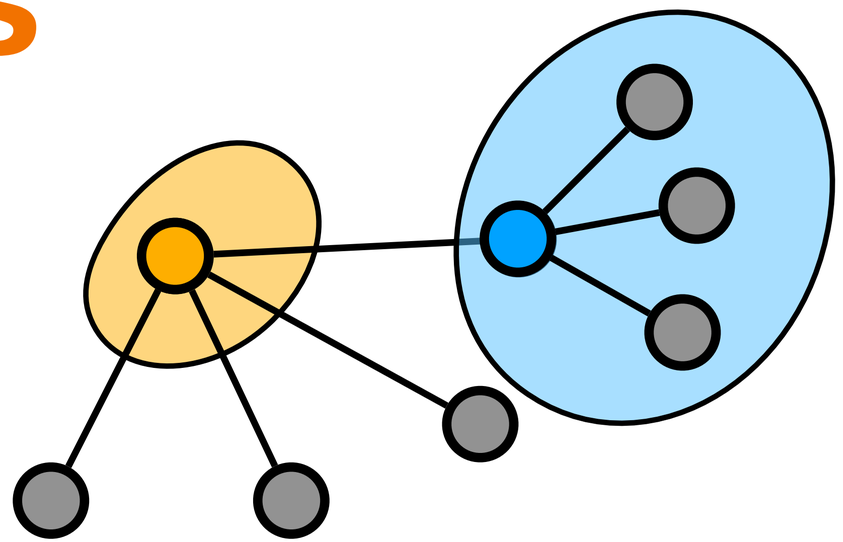


Cost of Pivot Clusters

- The Pivot Clusters created by Truncated-Pivot are different from the clusters created by Pivot.
    - We don’t allow uninteresting nodes to become pivots.
    - Their neighbors may create new pivot clusters.



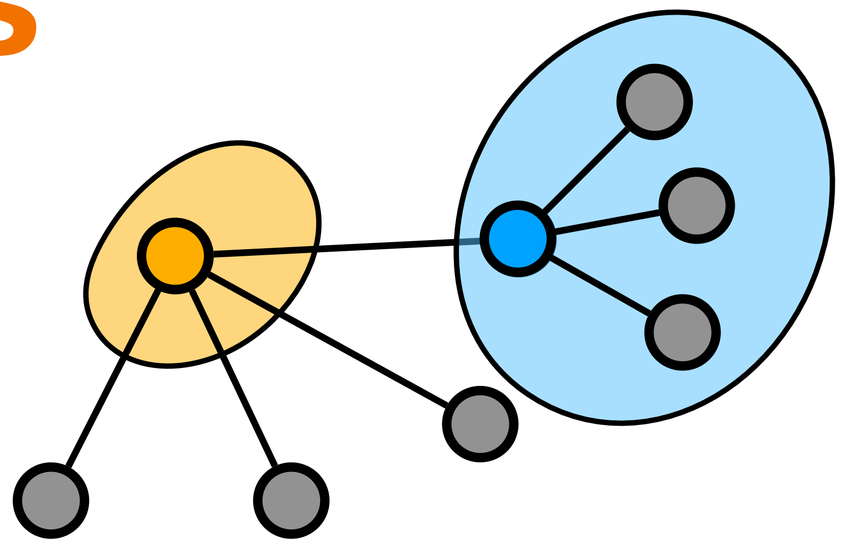
Cost of Pivot Clusters



- The Pivot Clusters created by Truncated-Pivot are different from the clusters created by Pivot.
    - We don’t allow uninteresting nodes to become pivots.
    - Their neighbors may create new pivot clusters.
- The analysis of [ACN08] shows that expected cost of the pivot clusters is a 3-approximation.

[ACN08] Ailon, Charikar, Newman, JACM 2008

Cost of Pivot Clusters

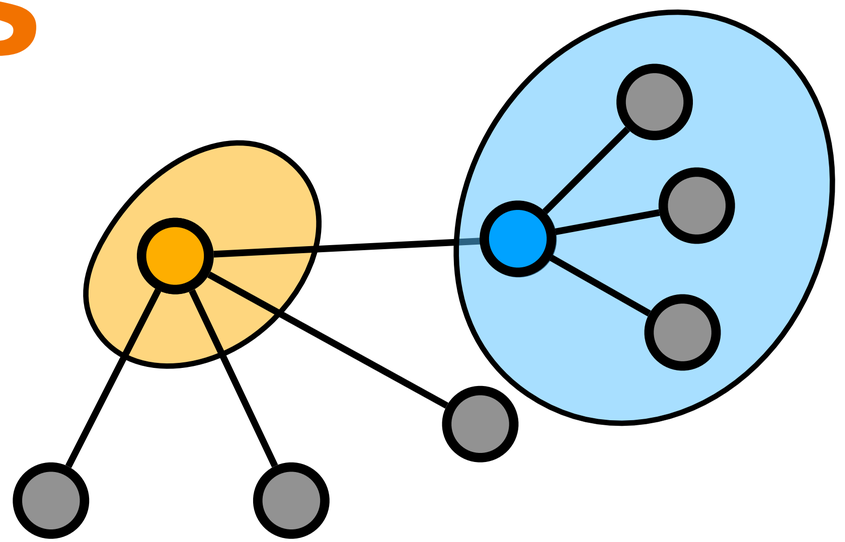


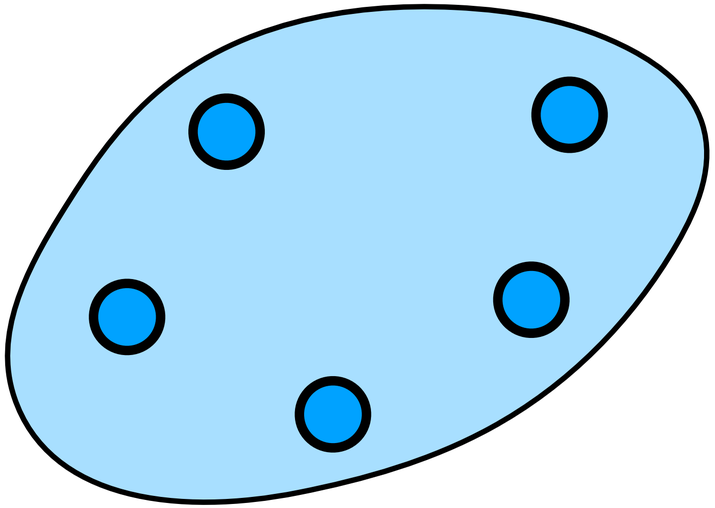
- The Pivot Clusters created by Truncated-Pivot are different from the clusters created by Pivot.
    - We don’t allow uninteresting nodes to become pivots.
    - Their neighbors may create new pivot clusters.
- The analysis of [ACN08] shows that expected cost of the pivot clusters is a 3-approximation.
    - Charging mistakes to bad triangles.

[ACN08] Ailon, Charikar, Newman, JACM 2008

Cost of Pivot Clusters

- The Pivot Clusters created by Truncated-Pivot are different from the clusters created by Pivot.
    - We don't allow uninteresting nodes to become pivots.
    - Their neighbors may create new pivot clusters.
- The analysis of [ACN08] shows that expected cost of the pivot clusters is a 3-approximation.
    - Charging mistakes to bad triangles.

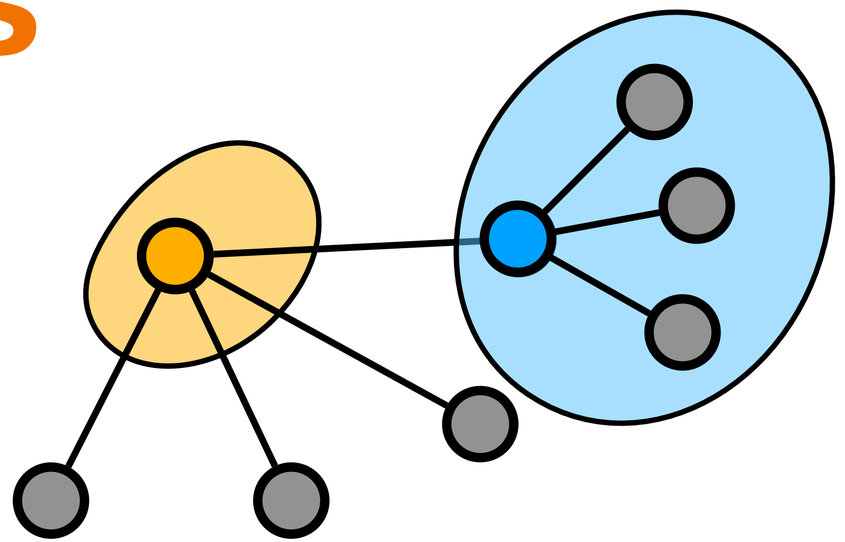


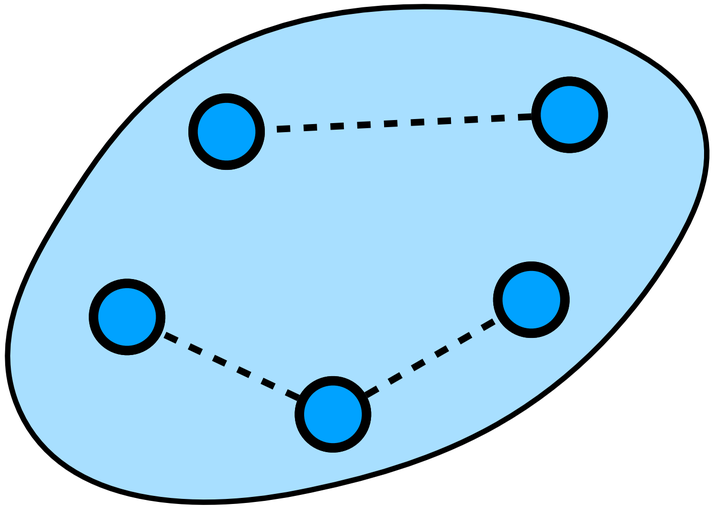


[ACN08] Ailon, Charikar, Newman, JACM 2008

Cost of Pivot Clusters

- The Pivot Clusters created by Truncated-Pivot are different from the clusters created by Pivot.
    - We don't allow uninteresting nodes to become pivots.
    - Their neighbors may create new pivot clusters.
- The analysis of [ACN08] shows that expected cost of the pivot clusters is a 3-approximation.
    - Charging mistakes to bad triangles.

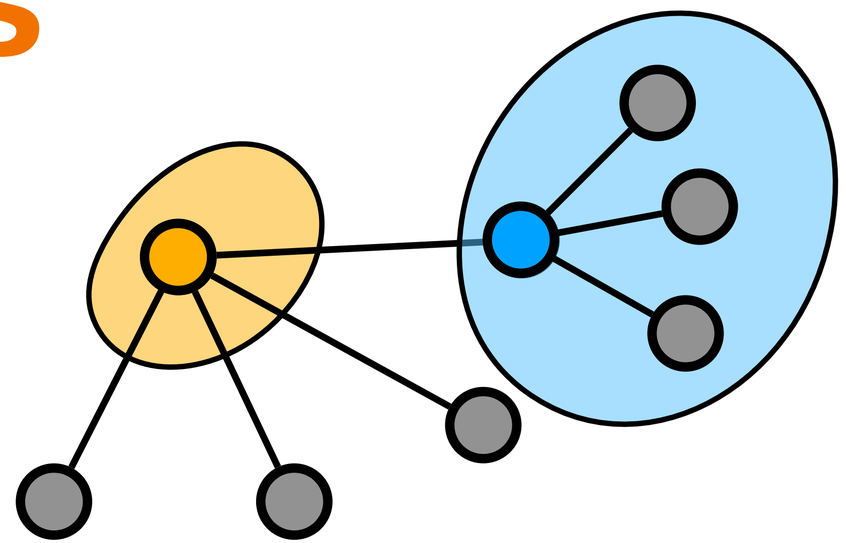


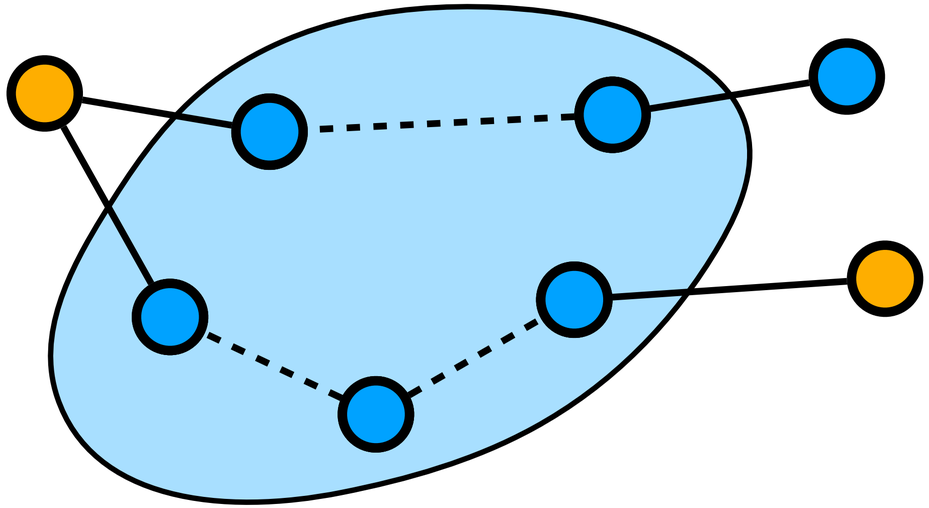


[ACN08] Ailon, Charikar, Newman, JACM 2008

Cost of Pivot Clusters

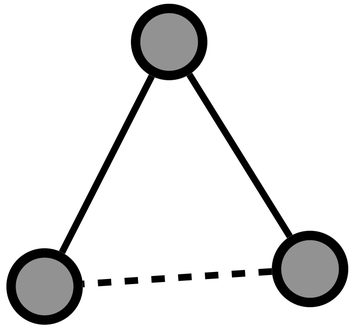
- The Pivot Clusters created by Truncated-Pivot are different from the clusters created by Pivot.
    - We don't allow uninteresting nodes to become pivots.
    - Their neighbors may create new pivot clusters.
- The analysis of [ACN08] shows that expected cost of the pivot clusters is a 3-approximation.
    - Charging mistakes to bad triangles.



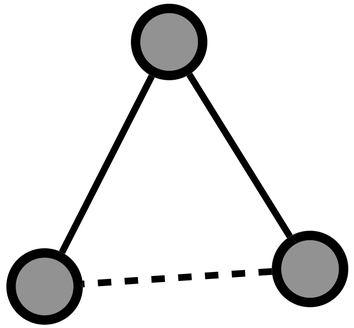


[ACN08] Ailon, Charikar, Newman, JACM 2008

Bad Triangles



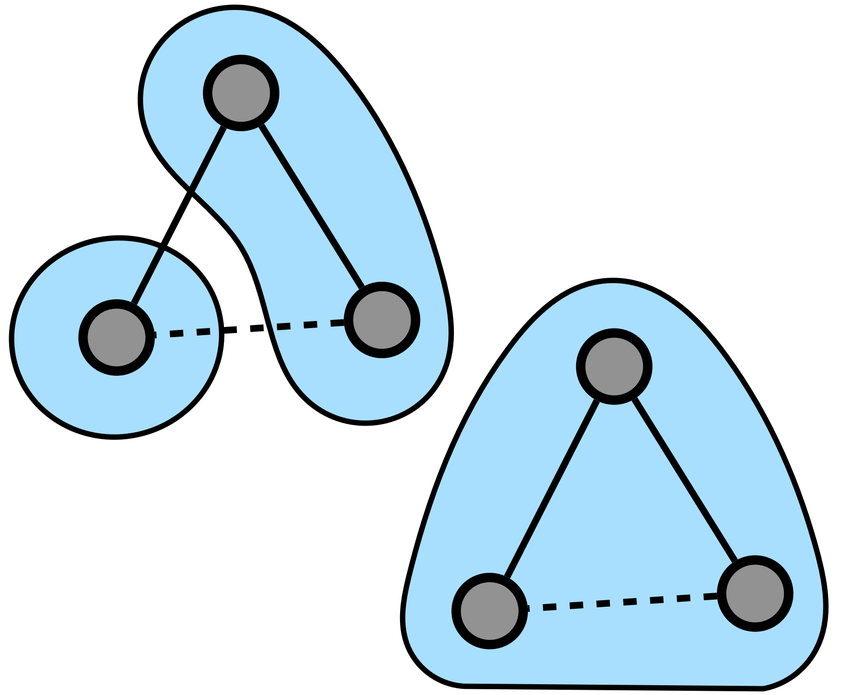
Bad Triangles



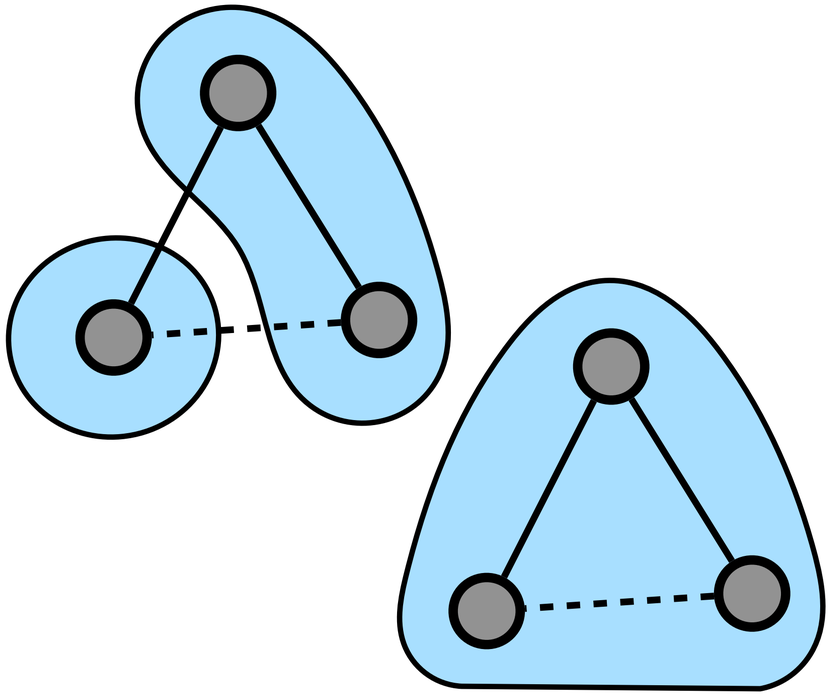
- A bad triangle is a triple containing two edges and one non-edge.

Bad Triangles

- A bad triangle is a triple containing two edges and one non-edge.
    - No matter how you cluster a bad triangle, you will make **at least one mistake**.

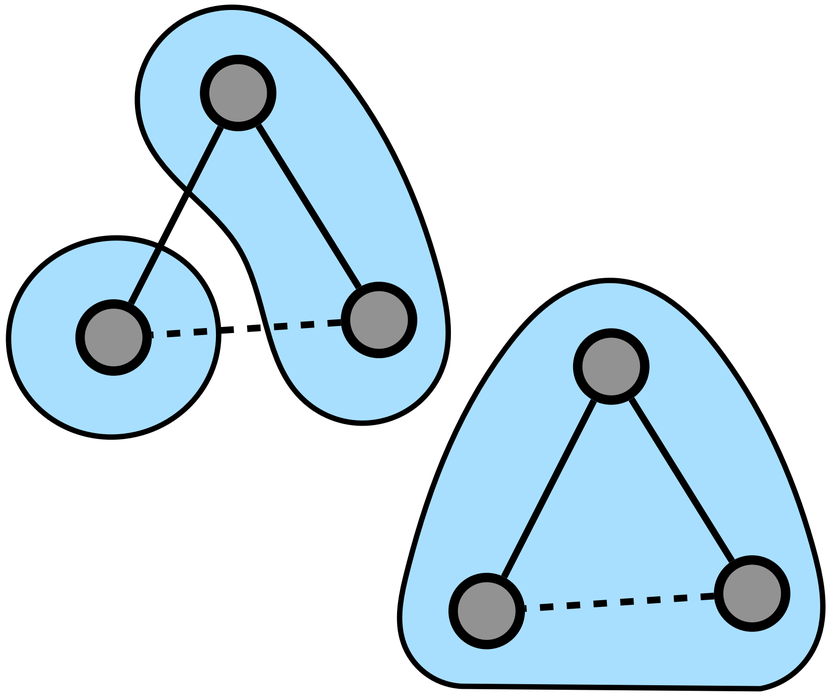


Bad Triangles



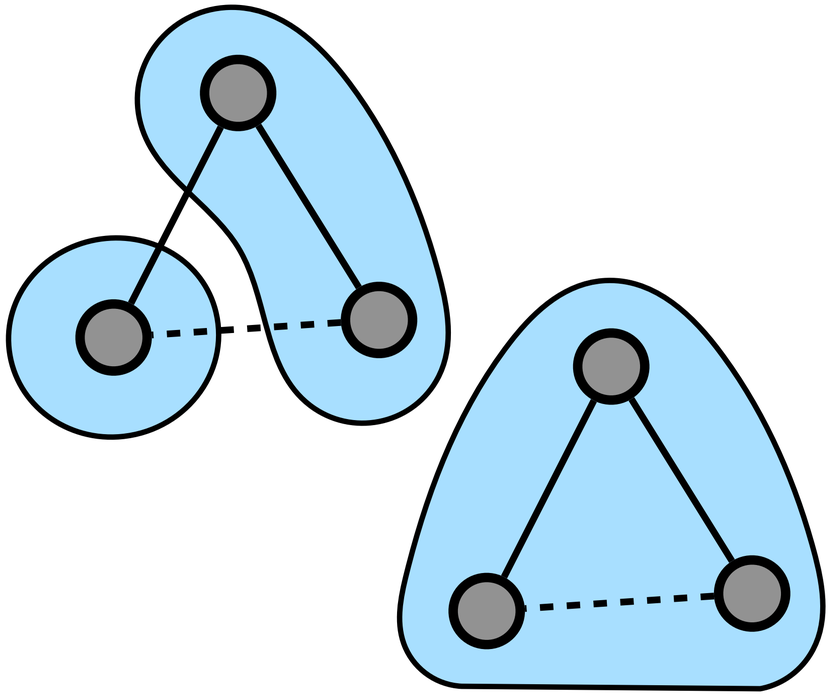
- A bad triangle is a triple containing two edges and one non-edge.
    - No matter how you cluster a bad triangle, you will make **at least one mistake**.
    - If the graph has t edge-disjoint bad triangles, we will make at least t mistakes.

Bad Triangles

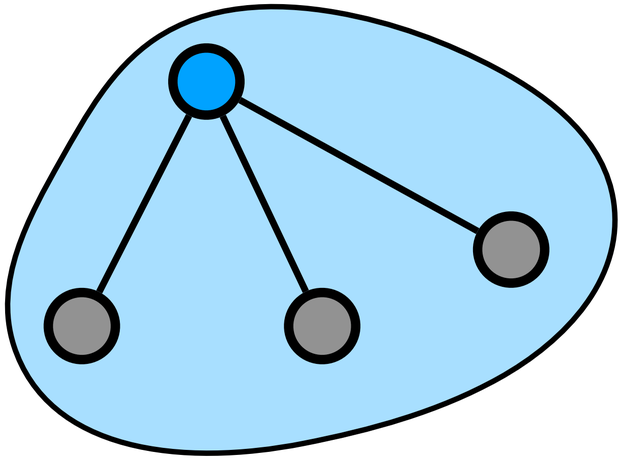


- A bad triangle is a triple containing two edges and one non-edge.
    - No matter how you cluster a bad triangle, you will make **at least one mistake**.
    - If the graph has t edge-disjoint bad triangles, we will make at least t mistakes.
- Each mistake made by creating a pivot cluster can be charged to a unique bad triangle.

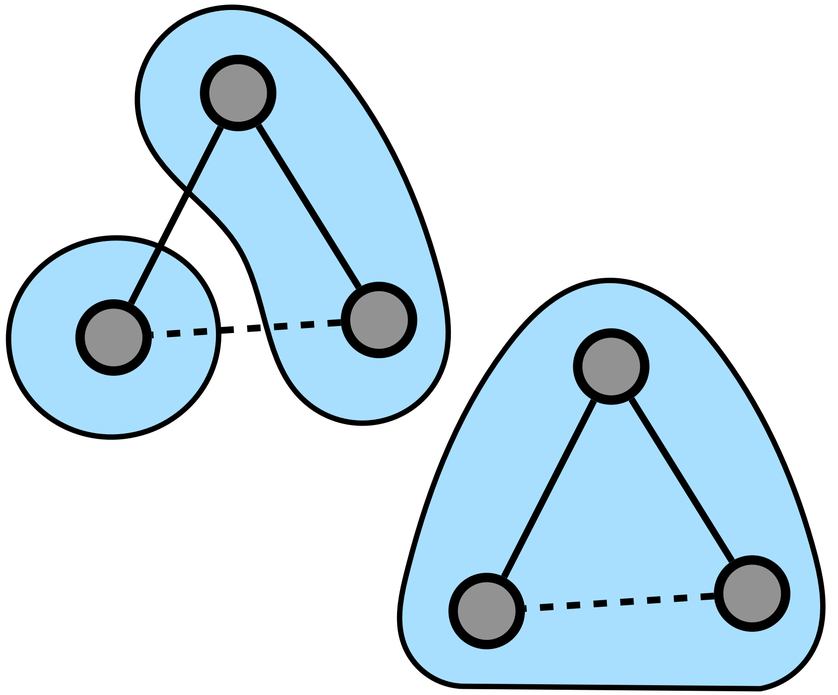
Bad Triangles



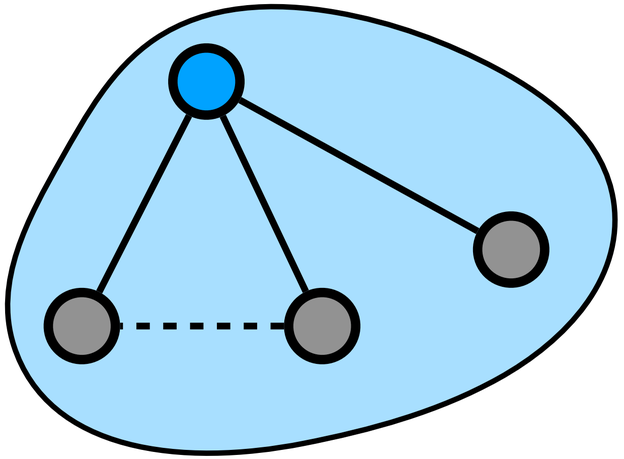
- A bad triangle is a triple containing two edges and one non-edge.
    - No matter how you cluster a bad triangle, you will make **at least one mistake**.
    - If the graph has t edge-disjoint bad triangles, we will make at least t mistakes.
- Each mistake made by creating a pivot cluster can be charged to a unique bad triangle.



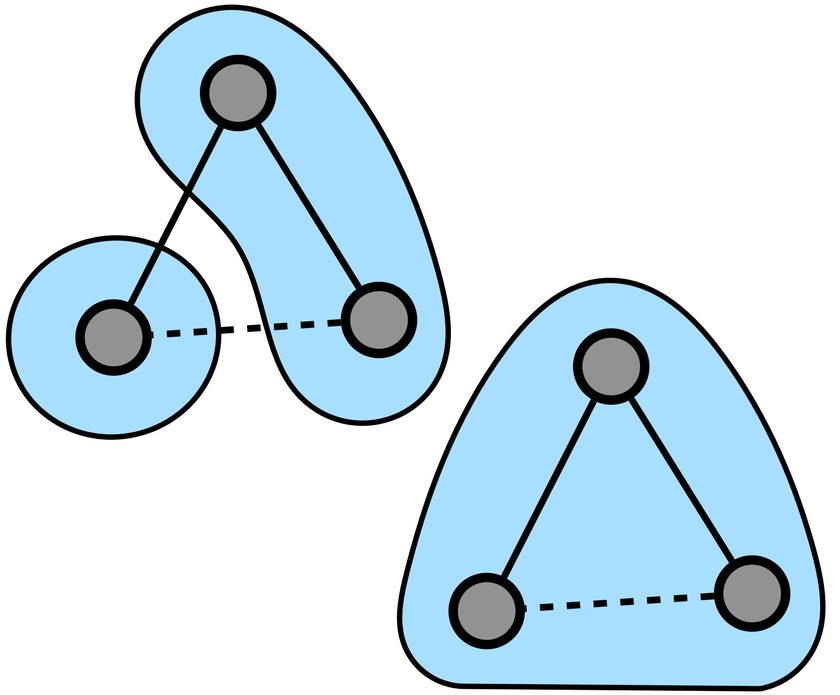
Bad Triangles



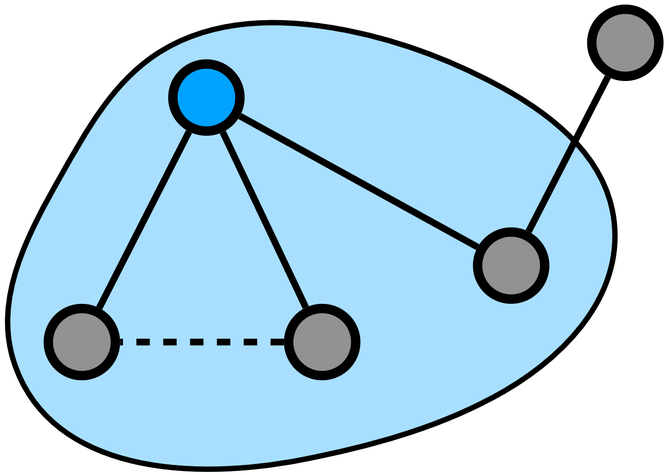
- A bad triangle is a triple containing two edges and one non-edge.
    - No matter how you cluster a bad triangle, you will make **at least one mistake**.
    - If the graph has t edge-disjoint bad triangles, we will make at least t mistakes.
- Each mistake made by creating a pivot cluster can be charged to a unique bad triangle.



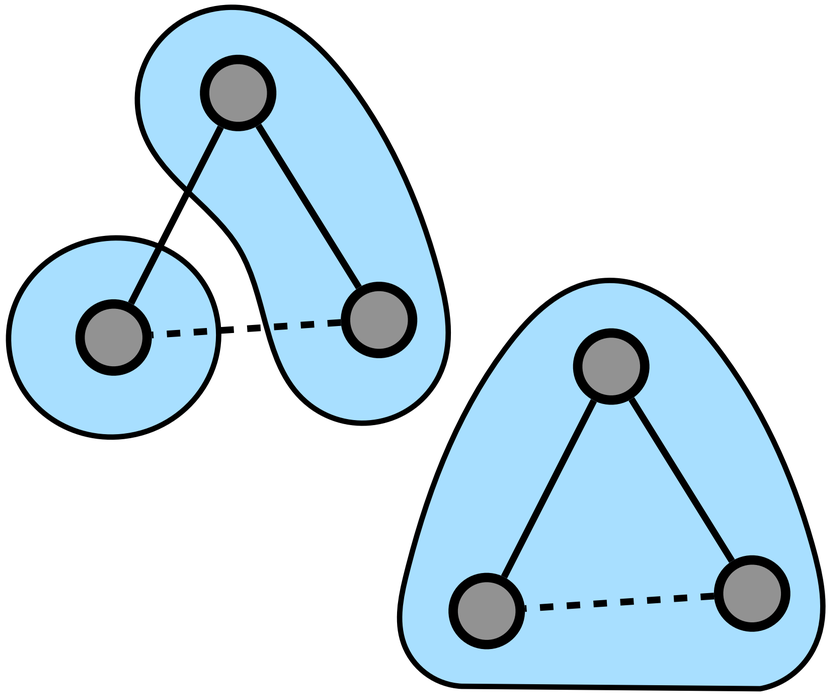
Bad Triangles



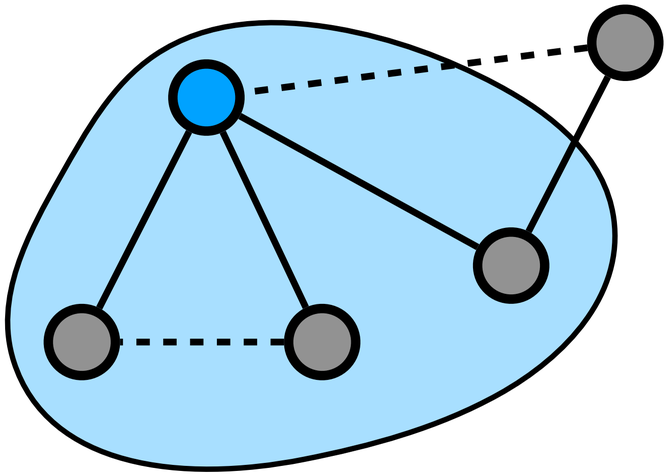
- A bad triangle is a triple containing two edges and one non-edge.
    - No matter how you cluster a bad triangle, you will make **at least one mistake**.
    - If the graph has t edge-disjoint bad triangles, we will make at least t mistakes.
- Each mistake made by creating a pivot cluster can be charged to a unique bad triangle.



Bad Triangles

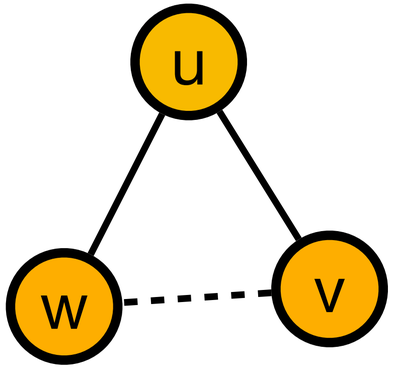


- A bad triangle is a triple containing two edges and one non-edge.
    - No matter how you cluster a bad triangle, you will make **at least one mistake**.
    - If the graph has t edge-disjoint bad triangles, we will make at least t mistakes.
- Each mistake made by creating a pivot cluster can be charged to a unique bad triangle.

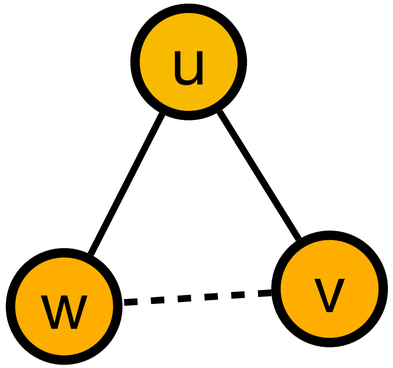


Charging Bad Triangles

Charging Bad Triangles

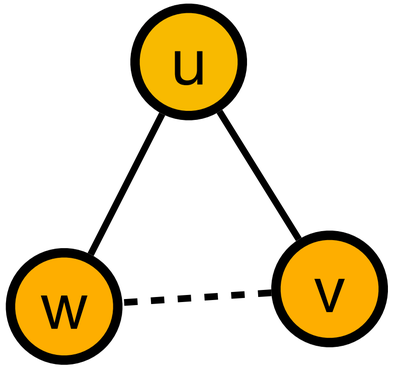


Charging Bad Triangles



A_t : **all** of u, v, w are **active** and
one is chosen as **pivot**.

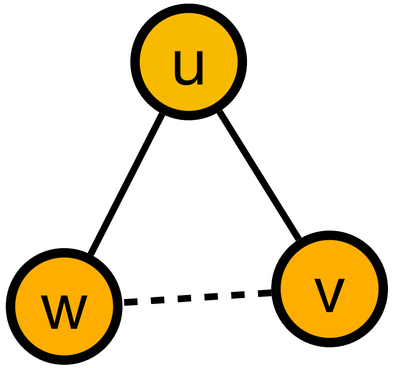
Charging Bad Triangles



A_t : **all** of u, v, w are **active** and **one** is chosen as **pivot**.

$$p_t = \Pr[A_t]$$

Charging Bad Triangles

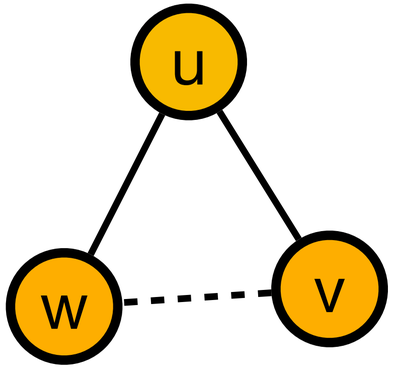


$$\mathbb{E}[C^{\text{pivot}}] \leq \sum_{t \in BT} p_t$$

A_t : **all** of u, v, w are **active** and **one** is chosen as **pivot**.

$$p_t = \Pr[A_t]$$

Charging Bad Triangles



$$\mathbb{E}[C^{\text{pivot}}] \leq \sum_{t \in BT} p_t$$

A_t : **all** of u, v, w are **active** and **one** is chosen as **pivot**.

$$p_t = \Pr[A_t]$$

How do we relate the p_t values to OPT?

Linear Program Relaxation

Linear Program Relaxation

Primal LP:

Linear Program Relaxation

Primal LP:

$$\min \sum_{e \in E^- \cup E^+} x_e$$

Linear Program Relaxation

Primal LP:

$$\min \sum_{e \in E^- \cup E^+} x_e$$

$$\text{s.t.} \sum_{e \in t} x_e \geq 1, \quad \forall t \in BT$$

Linear Program Relaxation

Primal LP:

$$\min \sum_{e \in E^- \cup E^+} x_e$$

$$\text{s.t.} \sum_{e \in t} x_e \geq 1, \quad \forall t \in BT$$

Dual LP:

$$\max \sum_{t \in BT} y_t$$

$$\text{s.t.} \sum_{t \ni e} y_t \leq 1, \quad \forall e \in E^- \cup E^+$$

Dual Feasible Solution

Dual LP:

$$\max \sum_{t \in BT} y_t$$

$$\text{s.t.} \sum_{t \ni e} y_t \leq 1, \quad \forall e \in E^- \cup E^+$$

Dual Feasible Solution

- By weak-duality:

Dual LP:

$$\max \sum_{t \in BT} y_t$$

$$\text{s.t.} \sum_{t \ni e} y_t \leq 1, \quad \forall e \in E^- \cup E^+$$

Dual Feasible Solution

- By weak-duality:
  - $\sum_{t \in BT} y_t \leq \mathrm{LP}^{\mathrm{OPT}} \leq \mathrm{OPT}.$

Dual LP:

$$\max \sum_{t \in BT} y_t$$

$$\text{s.t.} \sum_{t \ni e} y_t \leq 1, \quad \forall e \in E^- \cup E^+$$

Dual Feasible Solution

- By weak-duality:
  - $\sum_{t \in BT} y_t \leq \mathrm{LP}^{\mathrm{OPT}} \leq \mathrm{OPT}.$
- Candidate DFS: $y_t = p_t/3$.

Dual LP:

$$\max \sum_{t \in BT} y_t$$

$$\text{s.t.} \sum_{t \ni e} y_t \leq 1, \quad \forall e \in E^- \cup E^+$$

Dual Feasible Solution

- By weak-duality:
  - $\sum_{t \in BT} y_t \leq \mathrm{LP}^{\mathrm{OPT}} \leq \mathrm{OPT}.$
- Candidate DFS: $y_t = p_t/3$.
  - $\mathbb{E}[C^{\mathrm{pivot}}] \leq \sum_{t \in BT} p_t \leq 3\mathrm{OPT}.$

Dual LP:

$$\max \sum_{t \in BT} y_t$$

$$\text{s.t.} \sum_{t \ni e} y_t \leq 1, \quad \forall e \in E^- \cup E^+$$

Dual Feasible Solution

- By weak-duality:
  - $\sum_{t \in BT} y_t \leq \mathrm{LP}^{\mathrm{OPT}} \leq \mathrm{OPT}.$
- Candidate DFS: $y_t = p_t/3$.
  - $\mathbb{E}[C^{\mathrm{pivot}}] \leq \sum_{t \in BT} p_t \leq 3\mathrm{OPT}.$
- Why does $y_t = p_t/3$ satisfy all the dual constraints?

Dual LP:

$$\max \sum_{t \in BT} y_t$$

$$\text{s.t.} \sum_{t \ni e} y_t \leq 1, \quad \forall e \in E^- \cup E^+$$

Dual Feasible Solution

Dual Feasible Solution

Want: $\sum_{t \ni e} \frac{p_t}{3} \leq 1$

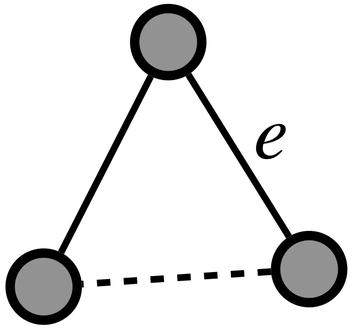
Dual Feasible Solution

Want: $\sum_{t \ni e} \frac{p_t}{3} \leq 1$

D_e: we make a mistake on e.

Dual Feasible Solution

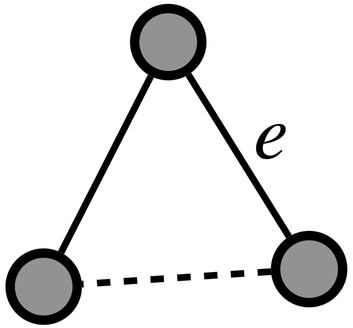
Want: $\sum_{t \ni e} \frac{p_t}{3} \leq 1$



D_e: we make a mistake on e.

Dual Feasible Solution

Want: $\sum_{t \ni e} \frac{p_t}{3} \leq 1$

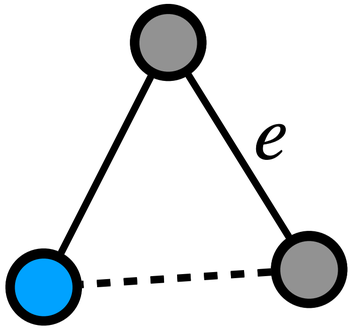


D_e: we make a mistake on e.

$$\Pr[D_e \mid A_t] = 1/3.$$

Dual Feasible Solution

Want: $\sum_{t \ni e} \frac{p_t}{3} \leq 1$

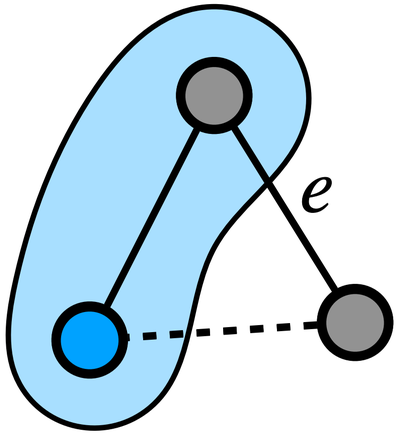


D_e: we make a mistake on e.

$$\Pr[D_e \mid A_t] = 1/3.$$

Dual Feasible Solution

Want: $\sum_{t \ni e} \frac{p_t}{3} \leq 1$

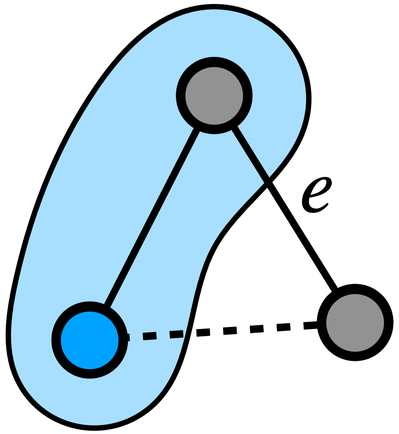


D_e: we make a mistake on e.

$$\Pr[D_e \mid A_t] = 1/3.$$

Dual Feasible Solution

Want: $\sum_{t \ni e} \frac{p_t}{3} \leq 1$



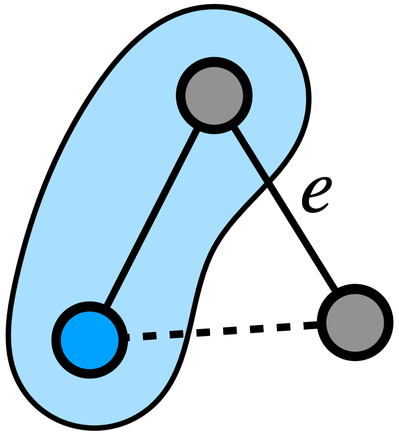
D_e: we make a mistake on e.

$$\Pr[D_e \mid A_t] = 1/3.$$

$$\Pr[D_e \cap A_t] = \Pr[D_e \mid A_t] \cdot \Pr[A_t]$$

Dual Feasible Solution

Want: $\sum_{t \ni e} \frac{p_t}{3} \leq 1$



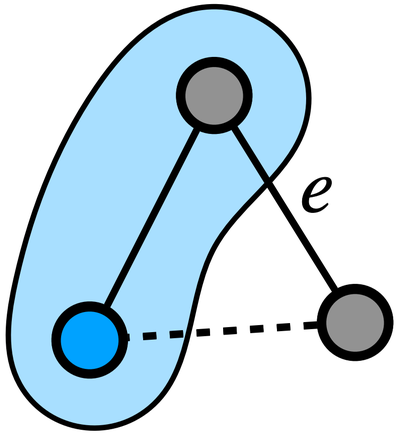
D_e: we make a mistake on e.

$$\Pr[D_e \mid A_t] = 1/3.$$

$$\Pr[D_e \cap A_t] = \Pr[D_e \mid A_t] \cdot \Pr[A_t] = \frac{p_t}{3}$$

Dual Feasible Solution

$$\sum_{t \ni e} \frac{p_t}{3} = \sum_{t \ni e} \Pr[D_e \cap A_t]$$



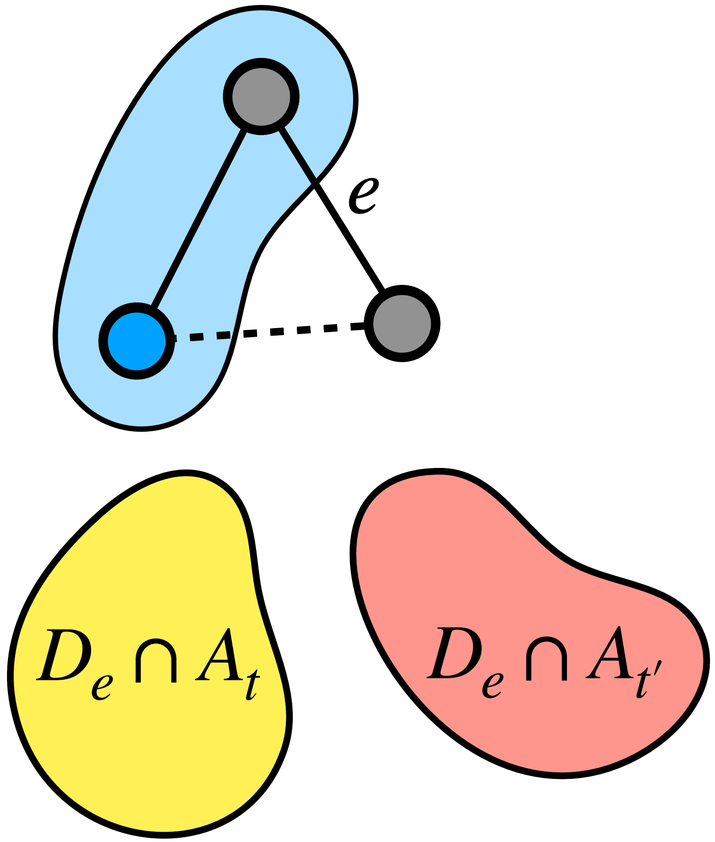
D_e: we make a mistake on e.

$$\Pr[D_e \mid A_t] = 1/3.$$

$$\Pr[D_e \cap A_t] = \Pr[D_e \mid A_t] \cdot \Pr[A_t] = \frac{p_t}{3}$$

Dual Feasible Solution

$$\sum_{t \ni e} \frac{p_t}{3} = \sum_{t \ni e} \Pr[D_e \cap A_t]$$



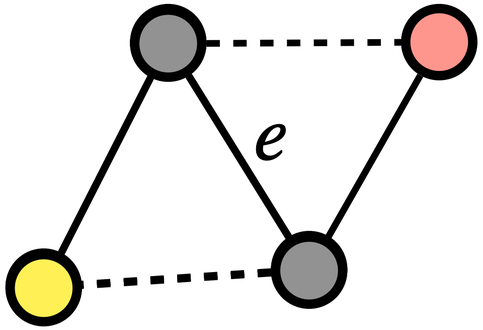
D_e: we make a mistake on e.

$$\Pr[D_e \mid A_t] = 1/3.$$

$$\Pr[D_e \cap A_t] = \Pr[D_e \mid A_t] \cdot \Pr[A_t] = \frac{p_t}{3}$$

Dual Feasible Solution

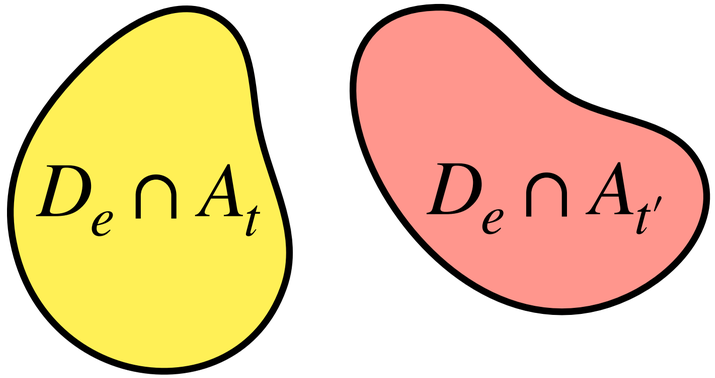
$$\sum_{t \ni e} \frac{p_t}{3} = \sum_{t \ni e} \Pr[D_e \cap A_t]$$



D_e: we make a mistake on e.

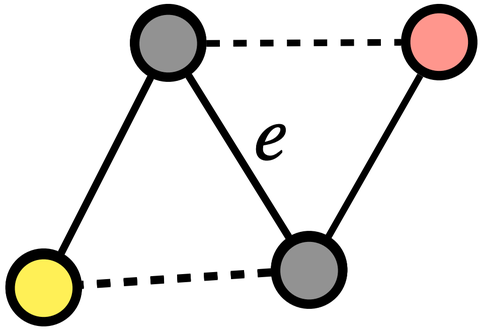
$$\Pr[D_e \mid A_t] = 1/3.$$

$$\Pr[D_e \cap A_t] = \Pr[D_e \mid A_t] \cdot \Pr[A_t] = \frac{p_t}{3}$$



Dual Feasible Solution

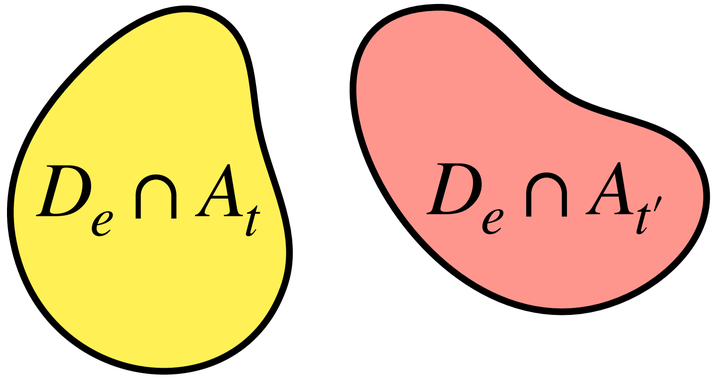
$$\sum_{t \ni e} \frac{p_t}{3} = \Pr\left[\cup_{t \ni e} (D_e \cap A_t)\right]$$



D_e: we make a mistake on e.

$$\Pr[D_e \mid A_t] = 1/3.$$

$$\Pr[D_e \cap A_t] = \Pr[D_e \mid A_t] \cdot \Pr[A_t] = \frac{p_t}{3}$$



Approximation Analysis

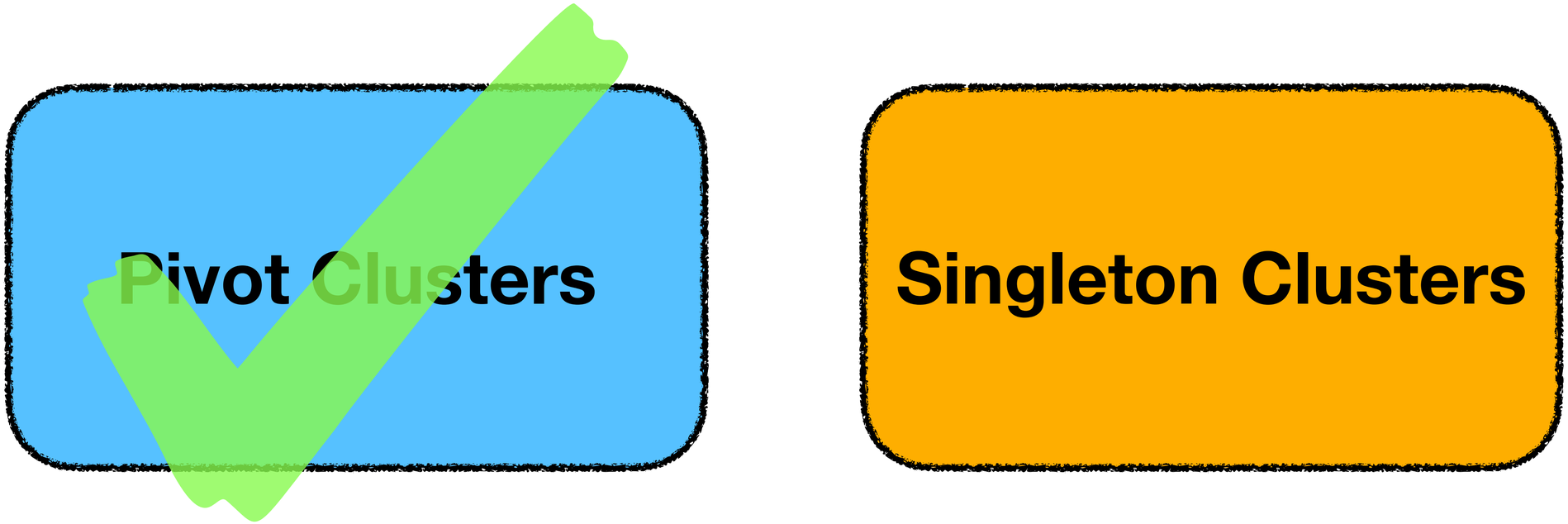
Pivot Clusters

3-approximation

Singleton Clusters

“$+\varepsilon$”-approximation

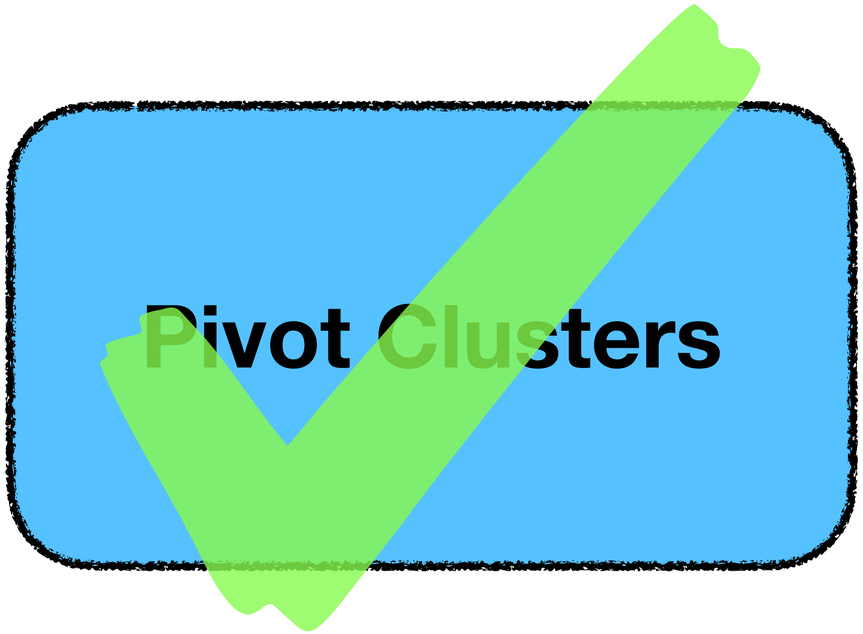
Approximation Analysis



3-approximation

“$+\varepsilon$”-approximation

Approximation Analysis

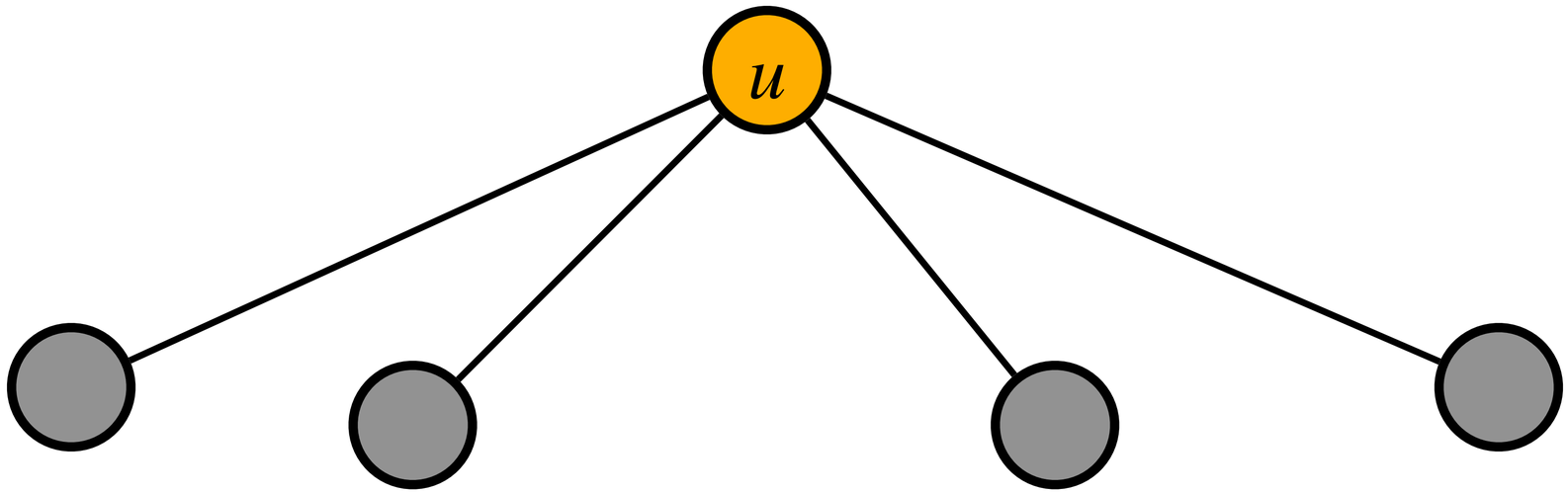


3-approximation

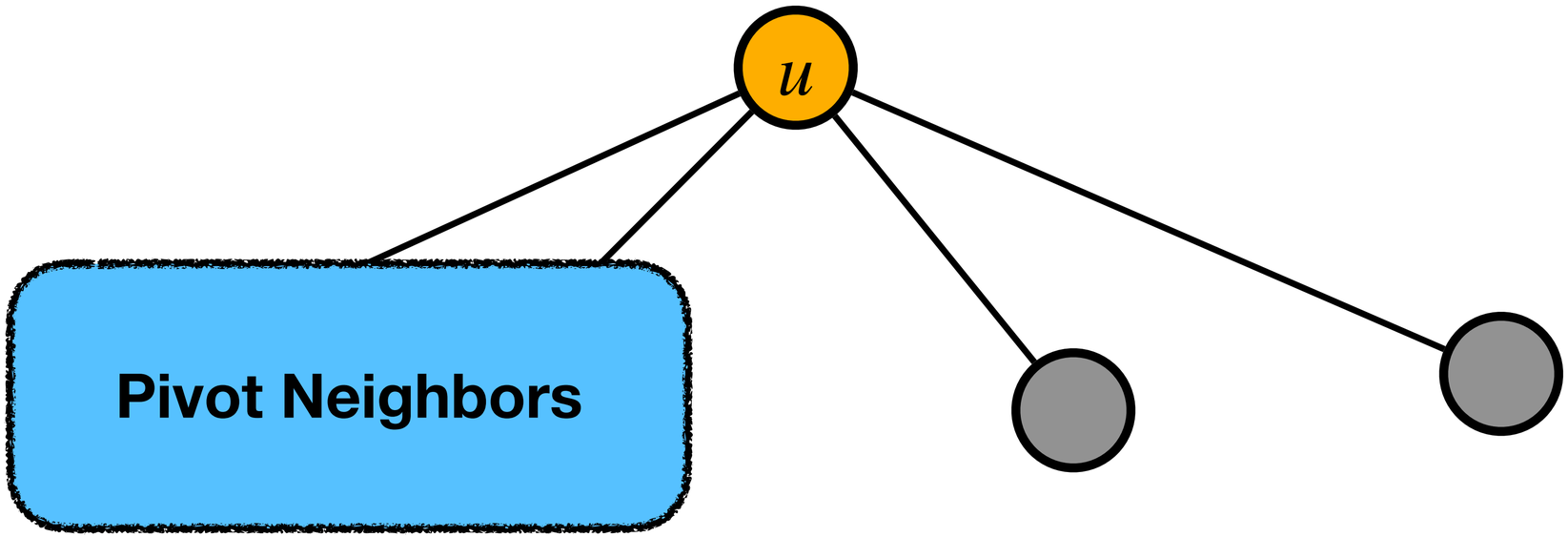
Singleton Clusters

“$+\varepsilon$”-approximation

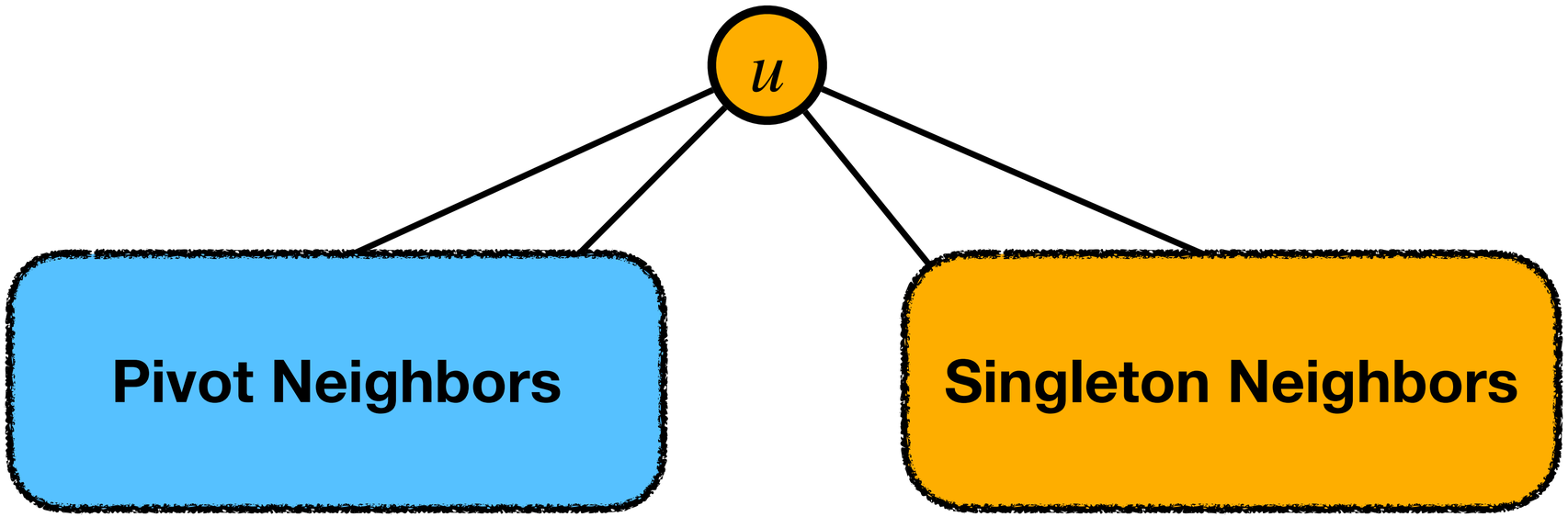
Cost of Singleton Clusters



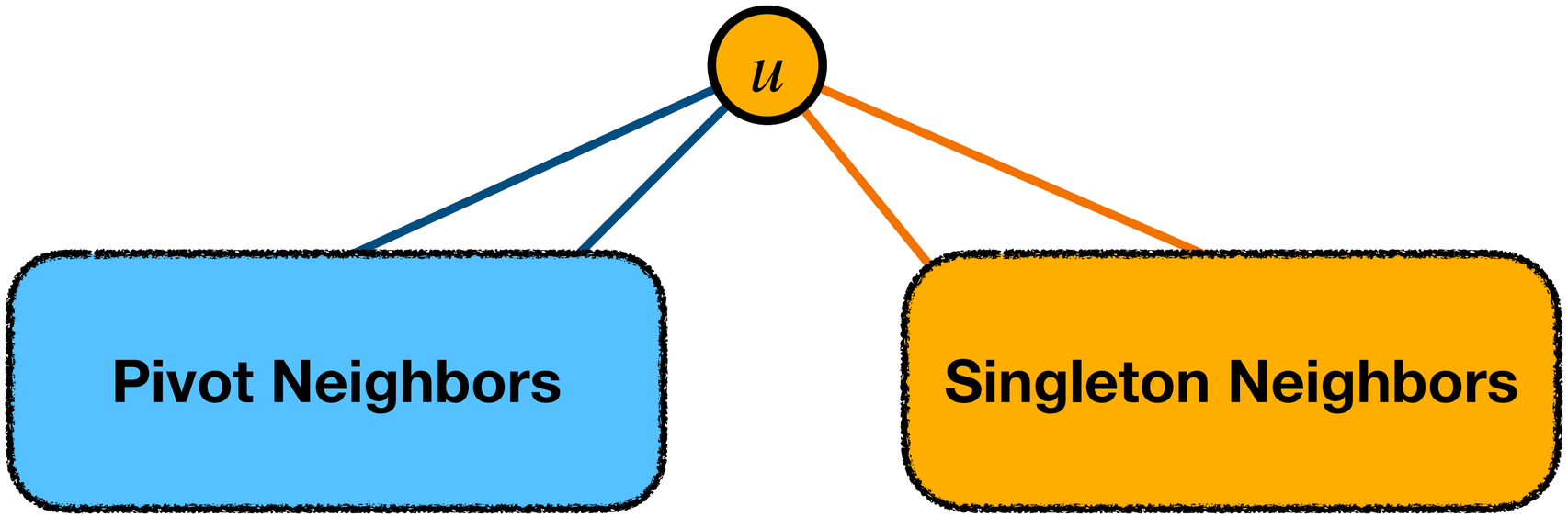
Cost of Singleton Clusters



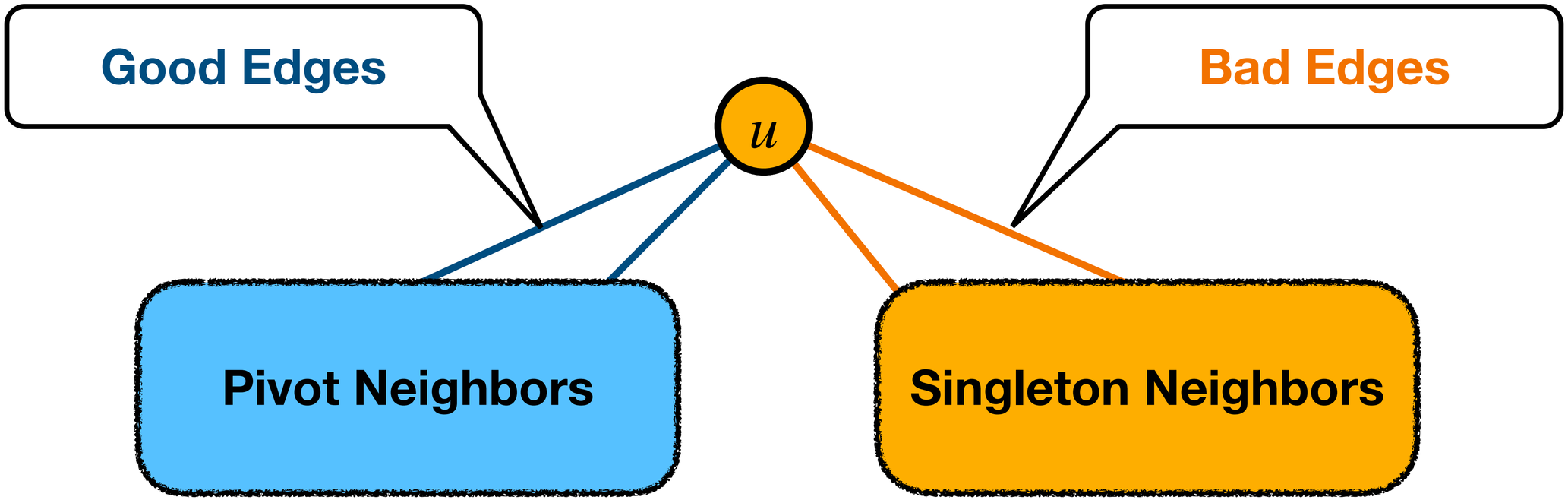
Cost of Singleton Clusters



Cost of Singleton Clusters



Cost of Singleton Clusters



Cost of Singleton Clusters

Singleton Edges = Good Edges + Bad Edges

Cost of Singleton Clusters

Singleton Edges = Good Edges + Bad Edges

- We make a mistake for each singleton edge.

Cost of Singleton Clusters

Singleton Edges = Good Edges + Bad Edges

- We make a mistake for each singleton edge.
- Good edges accounted in the analysis of Pivot clusters.

Cost of Singleton Clusters

Singleton Edges = Good Edges + Bad Edges

- We make a mistake for each singleton edge.
- Good edges accounted in the analysis of Pivot clusters.
- We charge the bad edges to the good edges by showing:

Cost of Singleton Clusters

Singleton Edges = Good Edges + Bad Edges

- We make a mistake for each singleton edge.
- Good edges accounted in the analysis of Pivot clusters.
- We charge the bad edges to the good edges by showing:

$$|\text{Bad Edges}| \leq O(\varepsilon) \cdot |\text{Good Edges}|$$

Cost of Singleton Clusters

Singleton Edges = Good Edges + Bad Edges

- We make a mistake for each singleton edge.
- Good edges accounted in the analysis of Pivot clusters.
- We charge the bad edges to the good edges by showing:

$$|\text{Bad Edges}| \leq O(\varepsilon) \cdot |\text{Good Edges}|$$

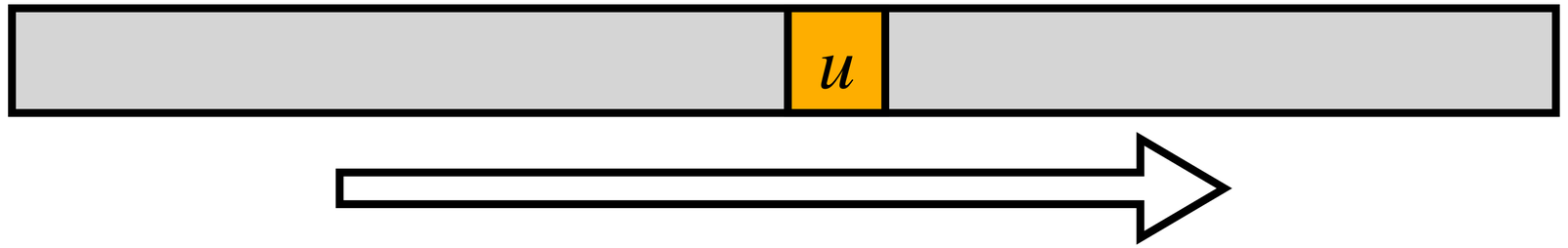
- This is not true for each node as all edges can be bad.

Counting Bad Edges

Random Permutation π :

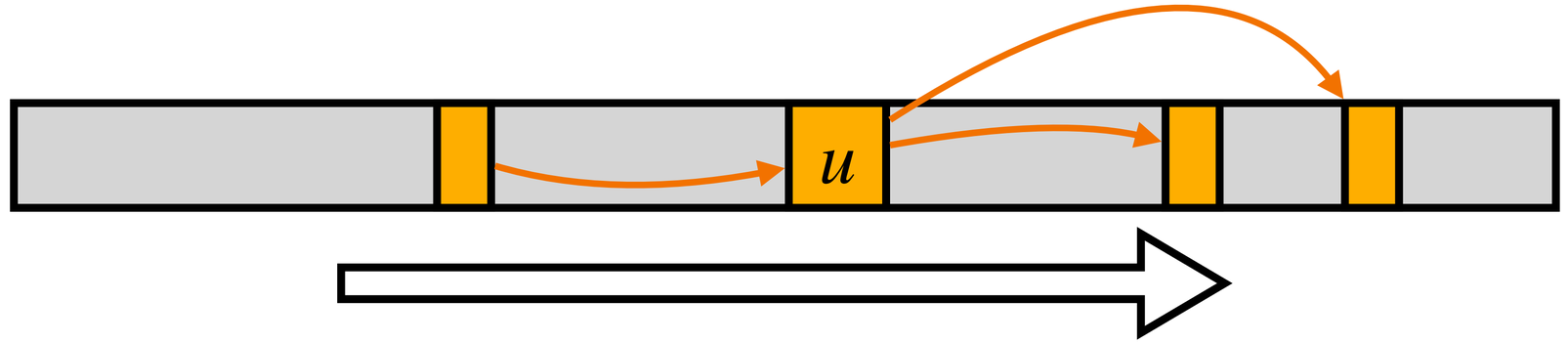
Counting Bad Edges

Random Permutation π :



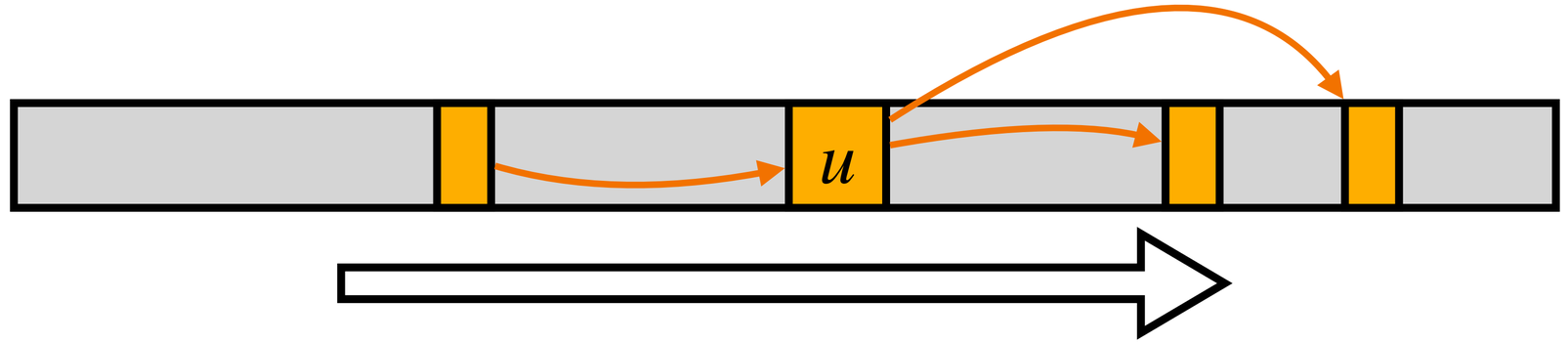
Counting Bad Edges

Random Permutation π :



Counting Bad Edges

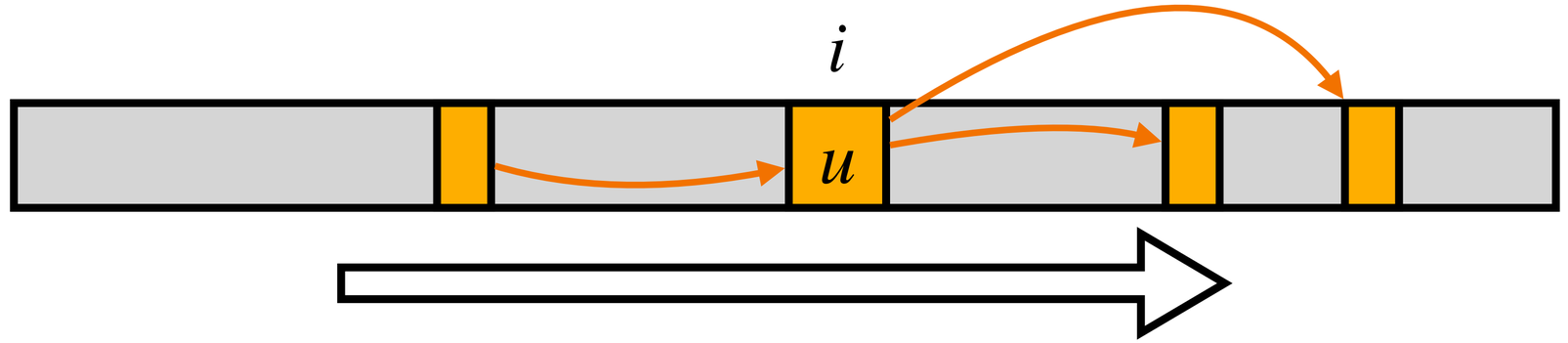
Random Permutation π :



With high probability, $\deg_{\text{out}}(u) \leq \varepsilon \deg(u)$

Counting Bad Edges

Random Permutation π :

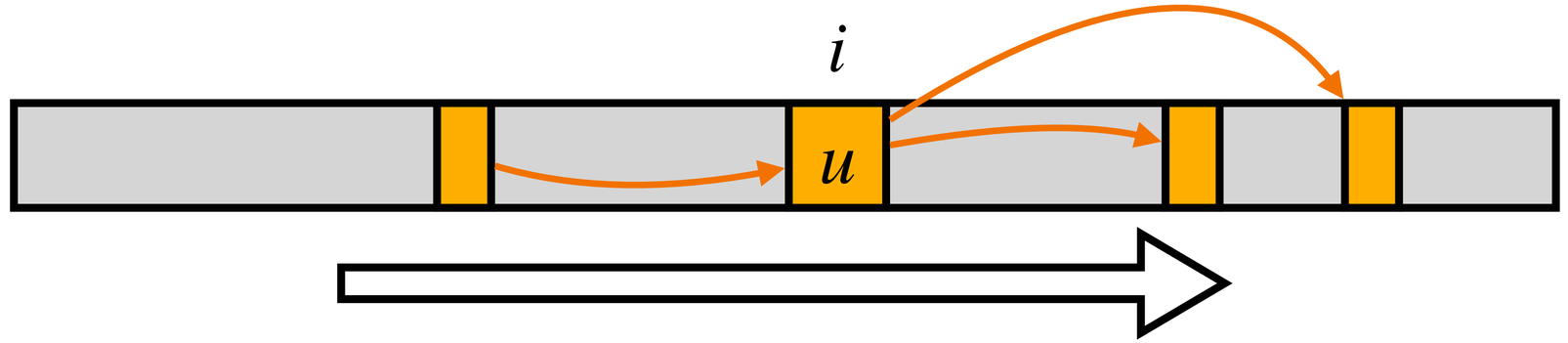


With high probability, $\deg_{\text{out}}(u) \leq \varepsilon \deg(u)$

$$i > \frac{n \log n}{\varepsilon \deg(u)}$$

Counting Bad Edges

Random Permutation π :



With high probability, $\deg_{\text{out}}(u) \leq \varepsilon \deg(u)$

$$i > \frac{n \log n}{\varepsilon \deg(u)}$$

It's unlikely that u is so far in the permutation and did not join a pivot cluster.

Counting Bad Edges

Counting Bad Edges

- We sum over all singleton nodes, and use degree-sum lemma.

Counting Bad Edges

- We sum over all singleton nodes, and use degree-sum lemma.

$$\sum_{\text{singleton } u} \deg_{\text{out}}(u) = \quad |\text{Bad Edges}|$$

Counting Bad Edges

- We sum over all singleton nodes, and use degree-sum lemma.

$$\sum_{\text{singleton } u} \deg_{\text{out}}(u) = \quad |\text{Bad Edges}|$$

$$\sum_{\text{singleton } u} \deg(u) \quad = \quad 2 \cdot |\text{Singleton Edges}|$$

Counting Bad Edges

- We sum over all singleton nodes, and use degree-sum lemma.

$$\sum_{\text{singleton } u} \deg_{\text{out}}(u) = \quad |\text{Bad Edges}|$$

$$\sum_{\text{singleton } u} \deg(u) \quad = \quad 2 \cdot |\text{Singleton Edges}|$$

$$\deg_{\text{out}}(u) \leq \varepsilon \deg(u)$$

Counting Bad Edges

- We sum over all singleton nodes, and use degree-sum lemma.

$$\sum_{\text{singleton } u} \deg_{\text{out}}(u) = \quad |\text{Bad Edges}|$$

$$\sum_{\text{singleton } u} \deg(u) \quad = \quad 2 \cdot |\text{Singleton Edges}|$$

$$\deg_{\text{out}}(u) \leq \varepsilon \deg(u)$$

- Therefore, $|\text{Bad Edges}| \leq 2\varepsilon \cdot |\text{Singleton Edges}|$

Counting Bad Edges

Counting Bad Edges

- The rest of the edges must be good…

Counting Bad Edges

- The rest of the edges must be good…

$$|\text{Good Edges}| \geq (1 - 2\varepsilon) \cdot |\text{Singleton Edges}|$$

Counting Bad Edges

- The rest of the edges must be good…

$$|\text{Good Edges}| \geq (1 - 2\varepsilon) \cdot |\text{Singleton Edges}|$$

- A better way to write it,

Counting Bad Edges

- The rest of the edges must be good…

$$|\text{Good Edges}| \geq (1 - 2\varepsilon) \cdot |\text{Singleton Edges}|$$

- A better way to write it,

$$|\text{Singleton Edges}| \leq \frac{1}{1 - 2\varepsilon} \cdot |\text{Good Edges}|$$

Counting Bad Edges

- The rest of the edges must be good…

$$|\text{Good Edges}| \geq (1 - 2\varepsilon) \cdot |\text{Singleton Edges}|$$

- A better way to write it,

$$|\text{Singleton Edges}| \leq \frac{1}{1 - 2\varepsilon} \cdot |\text{Good Edges}|$$

- Finally we get,

Counting Bad Edges

- The rest of the edges must be good…

$$|\text{Good Edges}| \geq (1 - 2\varepsilon) \cdot |\text{Singleton Edges}|$$

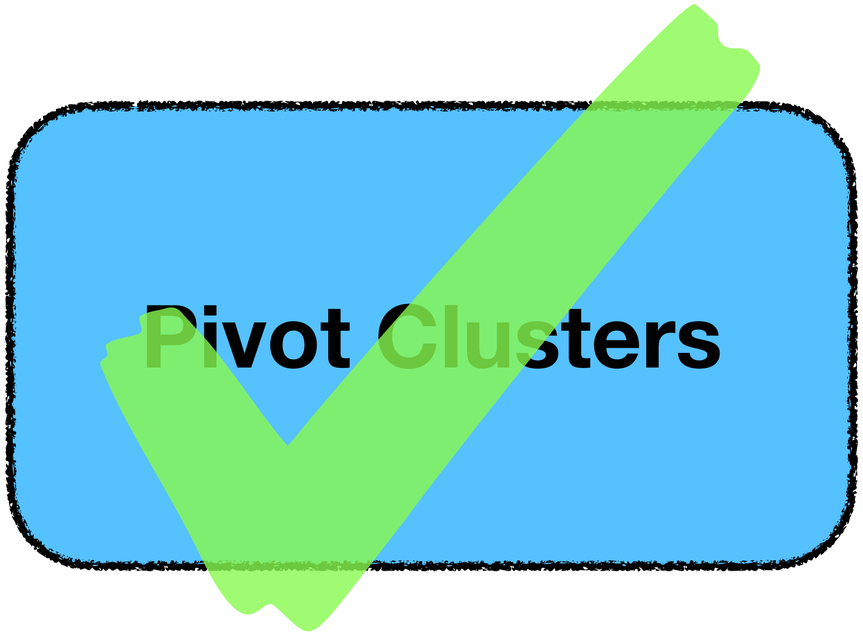
- A better way to write it,

$$|\text{Singleton Edges}| \leq \frac{1}{1 - 2\varepsilon} \cdot |\text{Good Edges}|$$

- Finally we get,

$$|\text{Bad Edges}| \leq 2\varepsilon \cdot |\text{Singleton Edges}| \leq \frac{2\varepsilon}{1 - 2\varepsilon} \cdot |\text{Good Edges}|$$

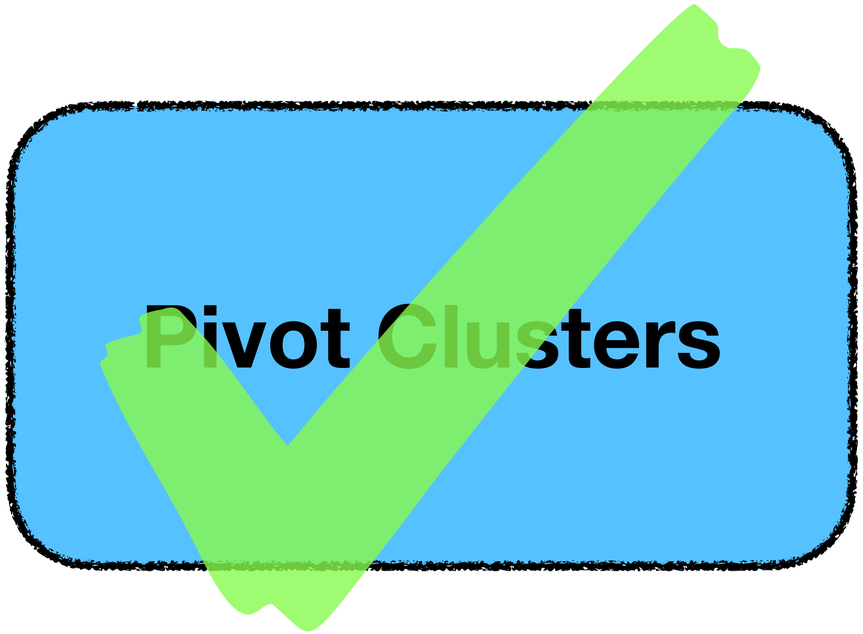
Approximation Analysis



3-approximation

“$+\varepsilon$”-approximation

Approximation Analysis



3-approximation

“$+\varepsilon$”-approximation

Correlation Clustering Summary

A $(3+\varepsilon)$-approximation algorithm using near-linear time and space.

Can be implemented in:

- Massively Parallel Computing: $O(1)$ rounds with $\tilde{O}(n/\varepsilon)$ local space.
- Dynamic Streaming: single pass with $\tilde{O}(n/\varepsilon)$ space.

Open Problems

Open Problems

- MIS: can we go below $\log \log \Delta$ rounds in MPC?

Open Problems

- MIS: can we go below $\log \log \Delta$ rounds in MPC?
  - Sub-linear MPC? MIS: $\sqrt{\log \Delta}$ [GU19] and 2-RS: $\log^{1/6} \Delta$ [KPP20].

[GU19] Ghaffari, and Uitto. SODA 2019

[KPP20] Kothapalli, Pai, and Pemmaraju. FSTTCS 2020

Open Problems

- MIS: can we go below $\log \log \Delta$ rounds in MPC?
  - Sub-linear MPC? MIS: $\sqrt{\log \Delta}$ [GU19] and 2-RS: $\log^{1/6} \Delta$ [KPP20].
- Can we solve 2-RS in a single pass of streaming with $O(n)$ space?

[GU19] Ghaffari, and Uitto. SODA 2019

[KPP20] Kothapalli, Pai, and Pemmaraju. FSTTCS 2020

Open Problems

- MIS: can we go below $\log \log \Delta$ rounds in MPC?
  - Sub-linear MPC? MIS: $\sqrt{\log \Delta}$ [GU19] and 2-RS: $\log^{1/6} \Delta$ [KPP20].
- Can we solve 2-RS in a single pass of streaming with $O(n)$ space?
  - Possible for random-order [AA21]. How about adversarial streams?

[GU19] Ghaffari, and Uitto. SODA 2019

[KPP20] Kothapalli, Pai, and Pemmaraju. FSTTCS 2020

[AA21] Assadi and Dudeja. DISC 2021

Open Problems

- MIS: can we go below $\log \log \Delta$ rounds in MPC?
  - Sub-linear MPC? MIS: $\sqrt{\log \Delta}$ [GU19] and 2-RS: $\log^{1/6} \Delta$ [KPP20].
- Can we solve 2-RS in a single pass of streaming with $O(n)$ space?
  - Possible for random-order [AA21]. How about adversarial streams?
- What is the best approximation we can get for correlation clustering in $O(1)$ MPC rounds?

[GU19] Ghaffari, and Uitto. SODA 2019
[KPP20] Kothapalli, Pai, and Pemmaraju. FSTTCS 2020
[AA21] Assadi and Dudeja. DISC 2021

Open Problems

Thank you!

- MIS: can we go below $\log \log \Delta$ rounds in MPC?
  - Sub-linear MPC? MIS: $\sqrt{\log \Delta}$ [GU19] and 2-RS: $\log^{1/6} \Delta$ [KPP20].
- Can we solve 2-RS in a single pass of streaming with $O(n)$ space?
  - Possible for random-order [AA21]. How about adversarial streams?
- What is the best approximation we can get for correlation clustering in $O(1)$ MPC rounds?

[GU19] Ghaffari, and Uitto. SODA 2019

[KPP20] Kothapalli, Pai, and Pemmaraju. FSTTCS 2020

[AA21] Assadi and Dudeja. DISC 2021