Recent Advances in Deterministic Distributed Algorithms

Christoph Grunau (ETH Zurich)

Overview

Goal: Efficient **deterministic** distributed algorithms for **LOCAL** problems

- **Examples:** Maximal Independent Set (MIS), Coloring, Network Decomposition,…

Overview

Goal: Efficient **deterministic** distributed algorithms for **LOCAL** problems

- **Examples:** Maximal Independent Set (MIS), Coloring, Network Decomposition,...

For ≈30 years: [AGLP FOCS'89], [Panconesi, Srinivasan STOC'93]

- $2^{O(\sqrt{\log n})}$-round deterministic Network Decomposition

Overview

Goal: Efficient **deterministic** distributed algorithms for **LOCAL** problems

- **Examples:** Maximal Independent Set (MIS), Coloring, Network Decomposition,...

For ≈30 years: [AGLP FOCS’89], [Panconesi, Srinivasan STOC’93]

- $2^{O(\sqrt{\log n})}$-round deterministic Network Decomposition

Breakthrough: [Rozhoň, Ghaffari STOC’20]

- (polylog n)-round deterministic Network Decomposition

Overview

Goal: Efficient **deterministic** distributed algorithms for **LOCAL** problems

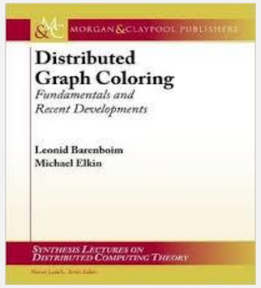
- **Examples:** Maximal Independent Set (MIS), Coloring, Network Decomposition,…

For ≈30 years: [AGLP FOCS'89], [Panconesi, Srinivasan STOC'93]

- $2^{O(\sqrt{\log n})}$-round deterministic Network Decomposition

Breakthrough: [Rozhoň, Ghaffari STOC'20]

- (polylog n)-round deterministic Network Decomposition



Answered
8 out of the 17
open problems

Barenboim & Elkin (2013)

Overview

Goal: Efficient **deterministic** distributed algorithms for **LOCAL** problems

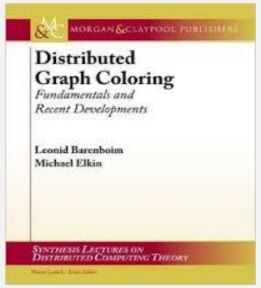
- **Examples:** Maximal Independent Set (MIS), Coloring, Network Decomposition,...

For ≈30 years: [AGLP FOCS'89], [Panconesi, Srinivasan STOC'93]

- $2^{O(\sqrt{\log n})}$-round deterministic Network Decomposition

Breakthrough: [Rozhoň, Ghaffari STOC'20]

- (polylog n)-round deterministic Network Decomposition



Barenboim & Elkin (2013)

Answered
8 out of the 17
open problems

Current State of the Art: [Ghaffari, **G** FOCS '24]

- $\tilde{O}(\log^2 n)$ rounds for Network Decomposition
- $\tilde{O}(\log^{5/3} n)$ rounds for MIS

Overview

Goal: Efficient **deterministic** distributed algorithms for **LOCAL** problems

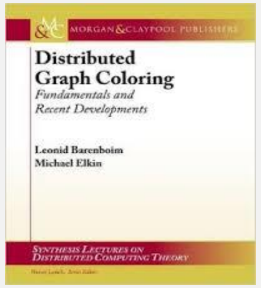
- **Examples:** Maximal Independent Set (MIS), Coloring, Network Decomposition,…

For ≈30 years: [AGLP FOCS'89], [Panconesi, Srinivasan STOC'93]

- $2^{O(\sqrt{\log n})}$-round deterministic Network Decomposition

Breakthrough: [Rozhoň, Ghaffari STOC'20]

- (polylog n)-round deterministic Network Decomposition



Barenboim & Elkin (2013)

Answered
8 out of the 17
open problems

Current State of the Art: [Ghaffari, **G** FOCS '24]

- $\widetilde{O}(\log^2 n)$ rounds for Network Decomposition
- $\tilde{O}(\log^{5/3} n)$ rounds for MIS

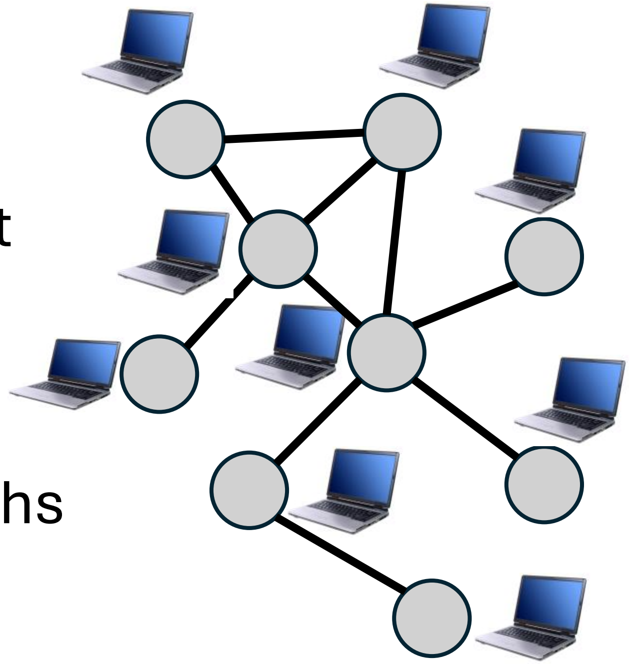
This talk: Present some of the key techniques used in the SOTA algorithms

The LOCAL Model of Distributed Computing *[Linial'87]*

- Undirected graph G = (V,E)
  - V = Processors with unique IDs from $\{1, \ldots, poly(n)\}$
  - E = Communication link

- Synchronous rounds
  - **Unbounded** local computation
  - **Unbounded** message sizes

- After **r** founds: each node needs to know its part of the output
  r = O(diameter) is always possible



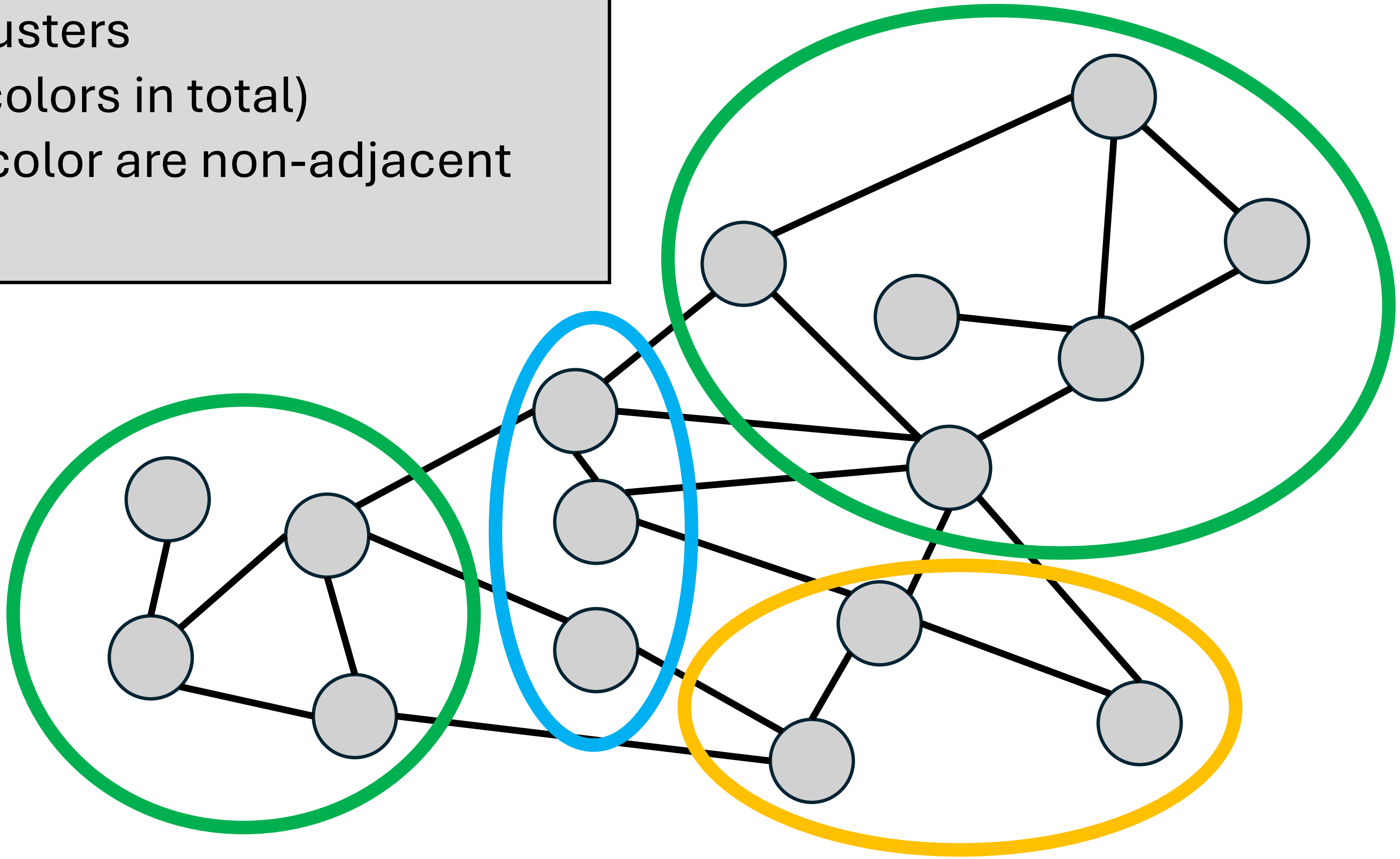
Generic approach: Break the graph into small-diameter subgraphs

Reminder: Network Decomposition & MIS

(C,D) Network Decomposition

- Graph is partitioned into clusters
- Each cluster is colored (**C** colors in total)
  - Clusters with identical color are non-adjacent
- Cluster diameter at most **D**

(3,3) Network Decomposition



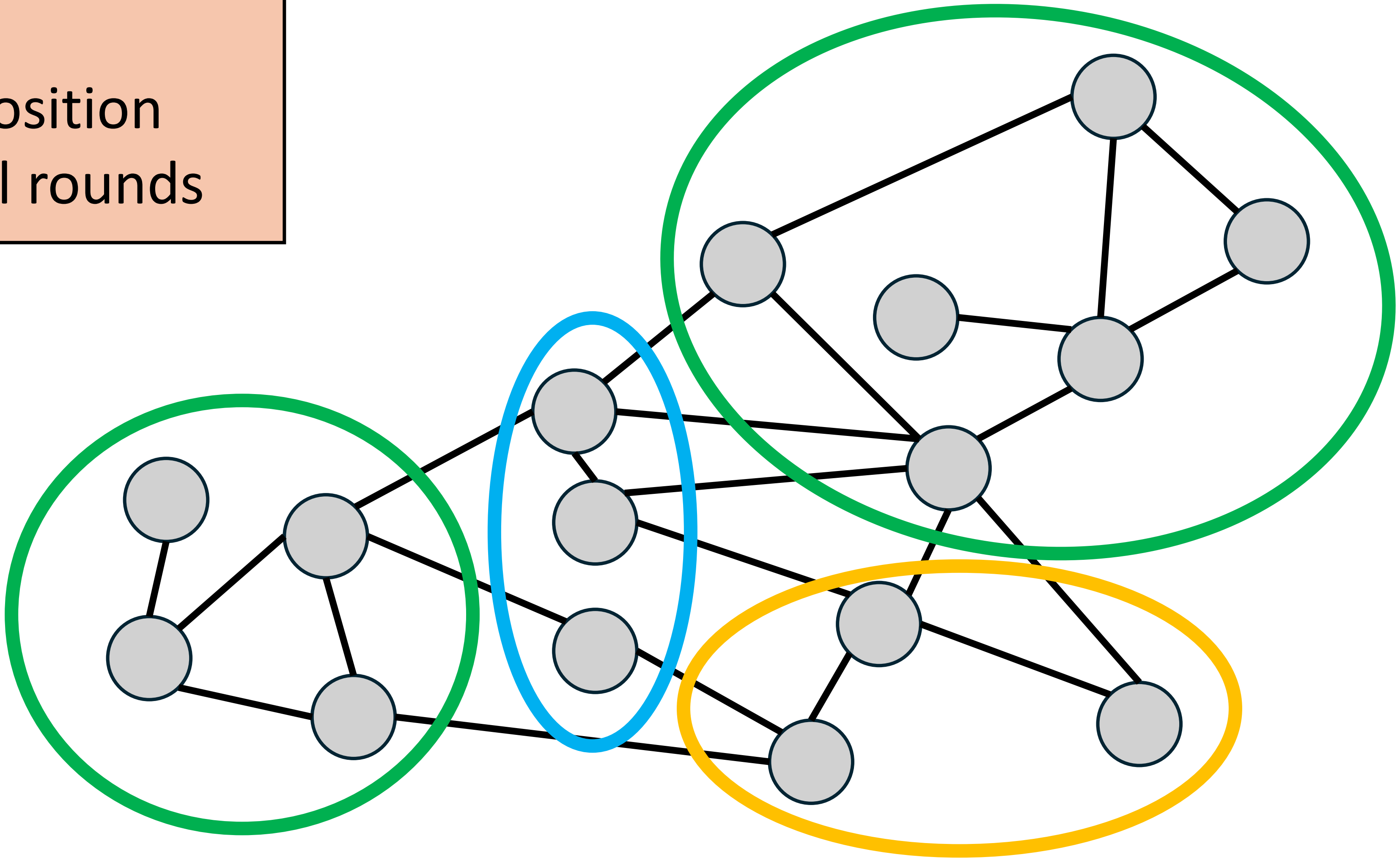
(C,D) Network Decomposition

Claim:

(C,D) Network Decomposition

⇨ MIS in O(CD) additional rounds

(3,3) Network Decomposition



Network Decomposition Results

Network Decomposition Results

Existential: can get C & D = O(log n). Implicit in [Awerbuch, Peleg '89]

Simple sequential ball growing to generate each color, which includes ½ of remaining nodes.

Network Decomposition Results

Existential: can get C & D = O(log n). Implicit in [Awerbuch, Peleg '89]

Simple sequential ball growing to generate each color, which includes ½ of remaining nodes.

Lower bound: For any (C, D) net decomposition, we need CD=$\Omega(\frac{\log^2 n}{\log^2 \log n})$. [Linial, Saks'93]

Network Decomposition Results

Existential: can get C & D = O(log n). Implicit in [Awerbuch, Peleg '89]

Simple sequential ball growing to generate each color, which includes ½ of remaining nodes.

Lower bound: For any (C, D) net decomposition, we need CD=$\Omega(\frac{\log^2 n}{\log^2 \log n})$. [Linial, Saks'93]

Randomized distributed algorithm: T=O($\log^2 n$) rounds. [Linial, Saks'93]

*originally weak-diameter, in 2016 made strong diameter by Elkin and Neiman using an algo of Miller, Peng, Xu

Network Decomposition Results

Existential: can get C & D = O(log n). Implicit in [Awerbuch, Peleg '89]

Simple sequential ball growing to generate each color, which includes ½ of remaining nodes.

Lower bound: For any (C, D) net decomposition, we need CD=$\Omega(\frac{\log^2 n}{\log^2 \log n})$. [Linial, Saks'93]

Randomized distributed algorithm: T=O($\log^2 n$) rounds. [Linial, Saks'93]

*originally weak-diameter, in 2016 made strong diameter by Elkin and Neiman using an algo of Miller, Peng, Xu

[Ghaffari, **G**'24] : Deterministic (O(log n), O(log n))-ND in $\tilde{O}(\log^2 n)$ rounds.

MIS Results

Randomized state of the art: O(log n) [Luby ’86] and [Alon, Babai, and Itai ’86]

MIS Results

Randomized state of the art: O(log n) [Luby ’86] and [Alon, Babai, and Itai ’86]

[Ghaffari, **G**‘24] : Deterministic MIS in $\tilde{O}(\log^{5/3} n)$ rounds.

MIS Results

Randomized state of the art: O(log n) [Luby '86] and [Alon, Babai, and Itai '86]

[Ghaffari, **G**'24] : Deterministic MIS in $\tilde{O}(\log^{5/3} n)$ rounds.

Remarks:

1.) Breaks a natural barrier, going below network decomposition-based methods.

2.) Comes close to $\tilde{\Omega}(log\ n)$ lower bound [Balliu, Brandt, Hirvonen, Olivetti, Rabie, Suomela'19]

MIS Results

Randomized state of the art: O(log n) [Luby '86] and [Alon, Babai, and Itai '86]

[Ghaffari, **G**'24] **:** Deterministic MIS in $\tilde{O}(\log^{5/3} n)$ rounds.

Remarks:

1.) Breaks a natural barrier, going below network decomposition-based methods.

2.) Comes close to $\tilde{\Omega}(log\ n)$ lower bound [Balliu, Brandt, Hirvonen, Olivetti, Rabie, Suomela'19]

Corollary 1: Deterministic Maximal Matching and $(\Delta + 1)$-coloring in $\tilde{O}(\log^{5/3} n)$ rounds.

Corollary 2: Randomized $(\Delta + 1)$-coloring in $\tilde{O}(\log^{\frac{5}{3}} log\ n)$ rounds.

Technique #1: Local Rounding of Pairwise-Independent Sampling

Technique #1: Local Rounding of Pairwise-Independent Sampling

Theorem (very informal) [Kuhn, Ghaffari ‘21, FG**G**KR ‘23]:
Any randomized algorithm using only **pairwise independence** can be turned into an almost as efficient deterministic algorithm.

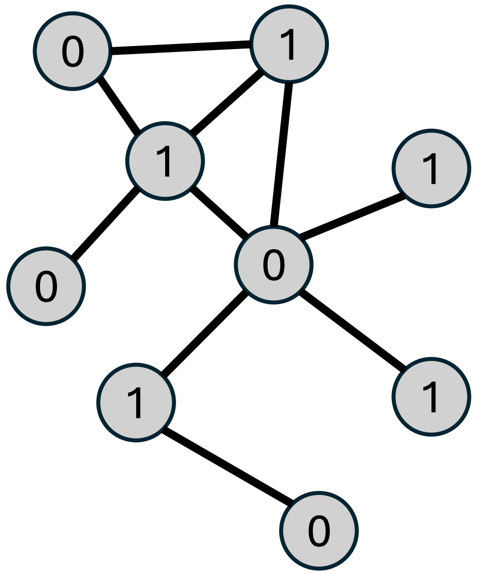
Technique #1: Local Rounding of Pairwise-Independent Sampling

Theorem (very informal) [Kuhn, Ghaffari '21, FG**G**KR '23]:
Any randomized algorithm using only **pairwise independence** can be turned into an almost as efficient deterministic algorithm.

Let's start with a toy problem: Max Cut

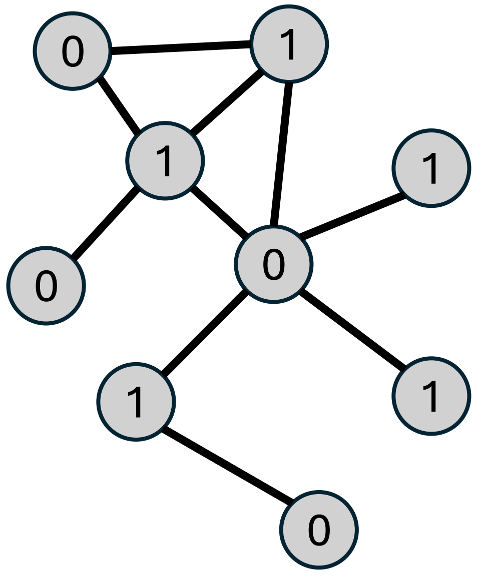
Toy Problem: Max-Cut

- Let $X_v \in \{0, 1\}$ indicate the side of the cut for node v.



Toy Problem: Max-Cut

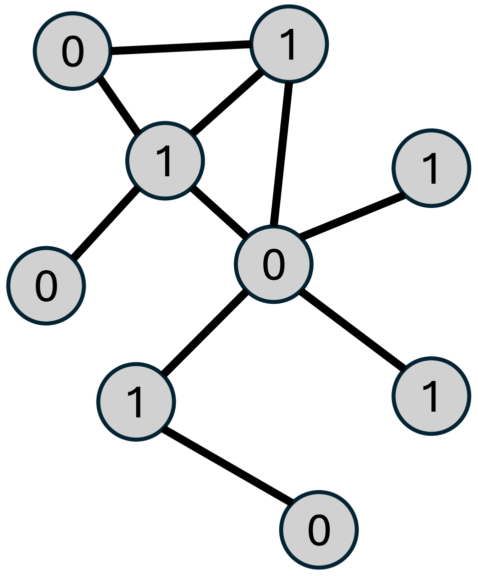
- Let $X_v \in \{0, 1\}$ indicate the side of the cut for node v.
- Cut size = $\sum_{(u,v)\in E} X_u + X_v - 2X_uX_v$



Toy Problem: Max-Cut

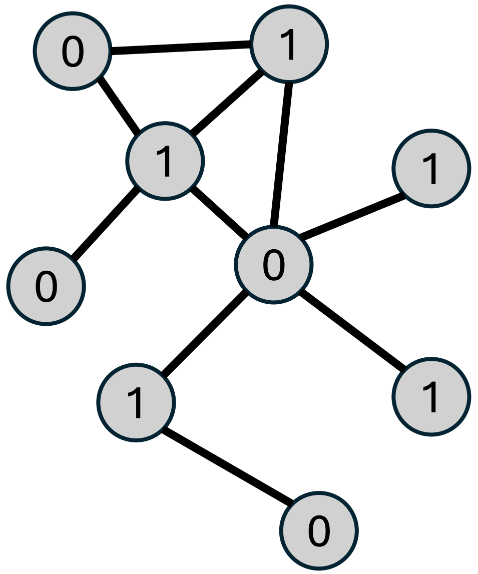
- Let $\mathrm{X_v} \in \{0, 1\}$ indicate the side of the cut for node v.
- Cut size = $\sum_{(u,v)\in E} X_u + X_v - 2X_uX_v$

<u>Simple (½)-approx. randomized Algorithm</u>:
Set each $X_v \in \{0,1\}$ randomly.



Toy Problem: Max-Cut

- Let $X_v \in \{0, 1\}$ indicate the side of the cut for node v.
- Cut size = $\sum_{(u,v)\in E} X_u + X_v - 2X_uX_v$

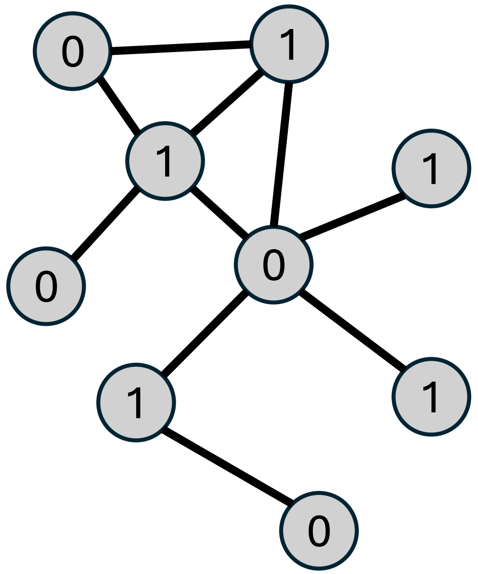


<u>Simple (½)-approx. randomized Algorithm</u>:
Set each $X_v \in \{0,1\}$ randomly.

<u>Deterministic Algorithm via Conditional Expectations:</u>
Fix variables **one by one**, such that the conditional expected cut value never decreases.

Toy Problem: Max-Cut

- Let $X_v \in \{0,1\}$ indicate the side of the cut for node v.
- Cut size = $\sum_{(u,v)\in E} X_u + X_v - 2X_uX_v$



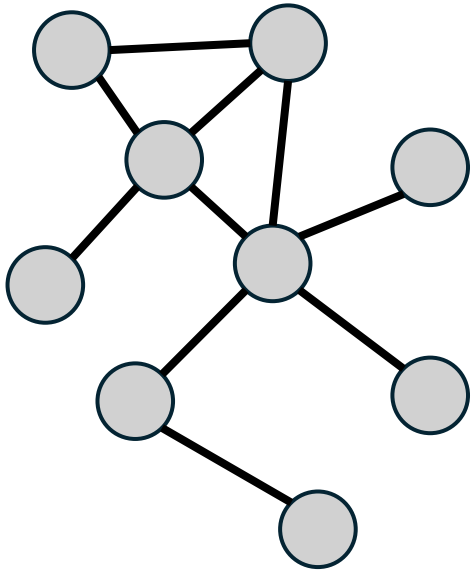
<u>Simple (½)-approx. randomized Algorithm</u>:
Set each $X_v \in \{0,1\}$ randomly.

<u>Deterministic Algorithm via Conditional Expectations:</u>
Fix variables **one by one**, such that the conditional expected cut value never decreases.

Issue: Highly Sequential

Determinstic Distributed $(\frac{1}{2} - \varepsilon)$-Approx.

Algorithm:
1.) Compute a coloring such that at most an ε-fraction of edges are monochromatic
2.) Remove all monochromatic edges.
3.) Iterate through the colors, fixing all nodes of the same color at once



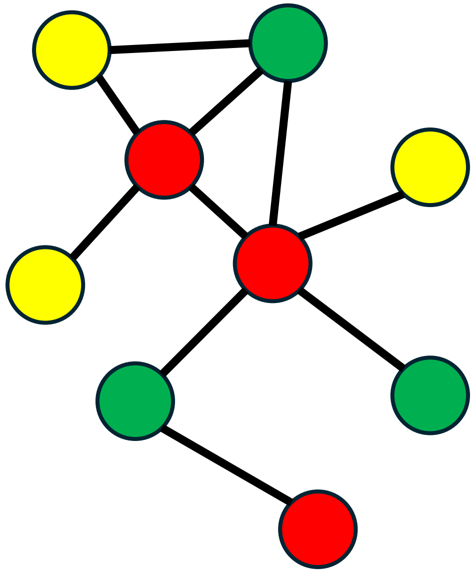
Determinstic Distributed $(\frac{1}{2} - \varepsilon)$-Approx.

Algorithm:

1.) Compute a coloring such that at most an ε-fraction of edges are monochromatic

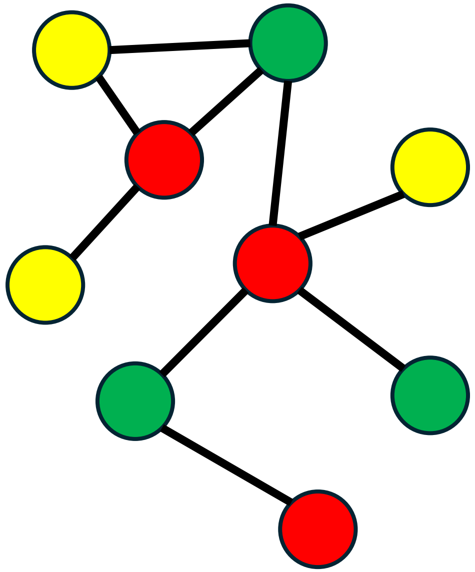
2.) Remove all monochromatic edges.

3.) Iterate through the colors, fixing all nodes of the same color at once



Determinstic Distributed $(\frac{1}{2} - \varepsilon)$-Approx.

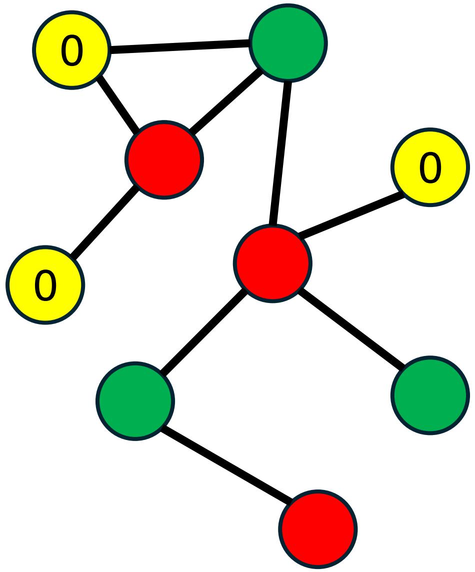
Algorithm:
1.) Compute a coloring such that at most an ε-fraction of edges are monochromatic
2.) Remove all monochromatic edges.
3.) Iterate through the colors, fixing all nodes of the same color at once



Determinstic Distributed $(\frac{1}{2} - \varepsilon)$-Approx.

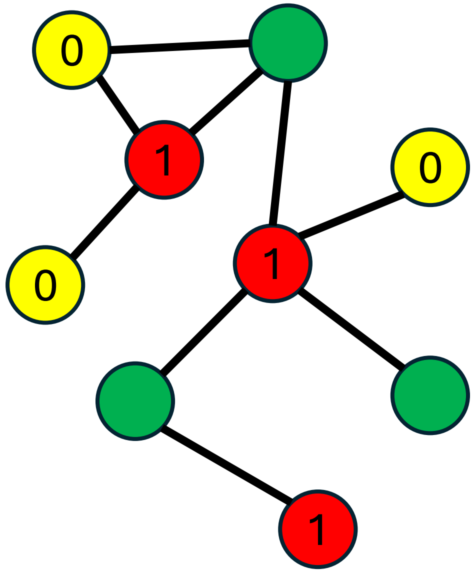
Algorithm:

1.) Compute a coloring such that at most an ε-fraction of edges are monochromatic
2.) Remove all monochromatic edges.
3.) Iterate through the colors, fixing all nodes of the same color at once



Determinstic Distributed $(\frac{1}{2} - \varepsilon)$-Approx.

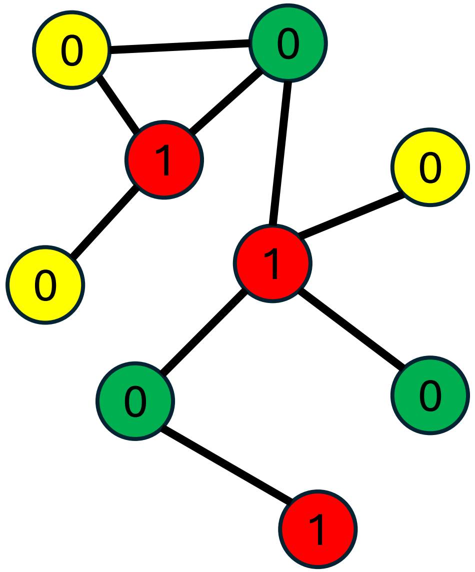
Algorithm:
1.) Compute a coloring such that at most an ε-fraction of edges are monochromatic
2.) Remove all monochromatic edges.
3.) Iterate through the colors, fixing all nodes of the same color at once



Determinstic Distributed $(\frac{1}{2} - \varepsilon)$-Approx.

Algorithm:

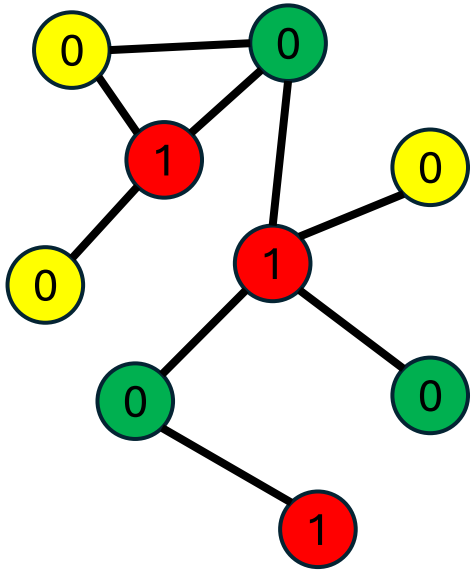
1.) Compute a coloring such that at most an ε-fraction of edges are monochromatic
2.) Remove all monochromatic edges.
3.) Iterate through the colors, fixing all nodes of the same color at once



Determinstic Distributed $(\frac{1}{2} - \varepsilon)$-Approx.

Algorithm:
1.) Compute a coloring such that at most an ε-fraction of edges are monochromatic
2.) Remove all monochromatic edges.
3.) Iterate through the colors, fixing all nodes of the same color at once

[Ghaffari, Kuhn' 21]:
Such a coloring with O(1/ ε) colors can be computed in O(1/ ε+$\log^* n$) rounds



Local Rounding of Pairwise Independent Sampling

Goal: Turn a fractional solution $\vec{p}$ into an almost-as-good integral solution $\vec{y}$ *locally*.

Local Rounding of Pairwise Independent Sampling

Goal: Turn a fractional solution $\vec{p}$ into an almost-as-good integral solution $\vec{y}$ *locally*.

Input:

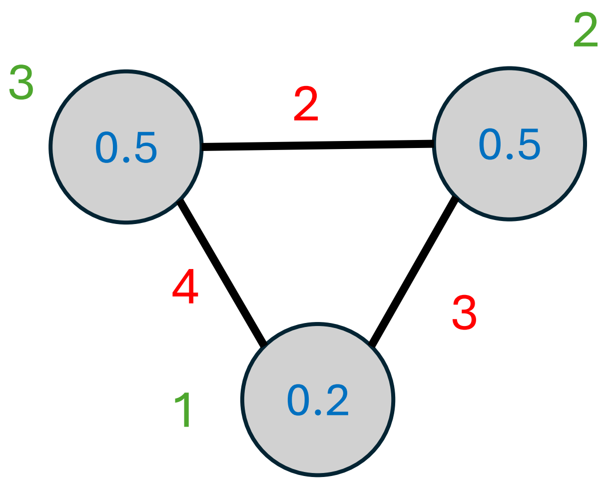
- $\vec{p} \in (0,1]^V$ ("sampling probabilities")
- $\vec{ut} \in \mathbb{R}^V_{\geq 0}$ ("utility of including a given node")
- $\vec{cost} \in \mathbb{R}^E_{\geq 0}$ ("cost of including both endpoints of a given edge")

Local Rounding of Pairwise Independent Sampling

Goal: Turn a fractional solution $\vec{p}$ into an almost-as-good integral solution $\vec{y}$ *locally*.

Input:

- $\vec{p} \in (0,1]^V$ ("sampling probabilities")
- $\vec{ut} \in \mathbb{R}_{\geq 0}^V$ ("utility of including a given node")
- $\vec{cost} \in \mathbb{R}_{\geq 0}^E$ ("cost of including both endpoints of a given edge")



Local Rounding of Pairwise Independent Sampling

Goal: Turn a fractional solution $\vec{p}$ into an almost-as-good integral solution $\vec{y}$ *locally*.

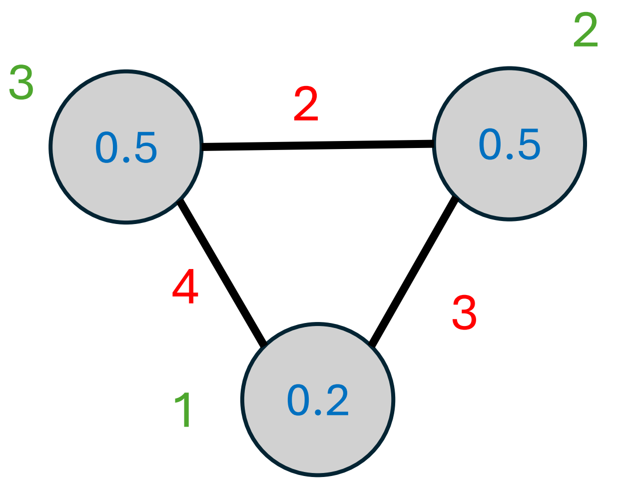
Input:

- $\vec{p} \in (0,1]^V$ ("sampling probabilities")
- $\vec{ut} \in \mathbb{R}^V_{\geq 0}$ ("utility of including a given node")
- $\vec{cost} \in \mathbb{R}^E_{\geq 0}$ ("cost of including both endpoints of a given edge")

Notation:

$$\Phi(\vec{x}) := \sum_{v\in V} ut(v)x(v) - \sum_{\{u,v\}\in E} cost(uv)x(u)x(v) \; (\vec{x} \in [0,1]^V)$$

"Expected utility" – "Expected cost"

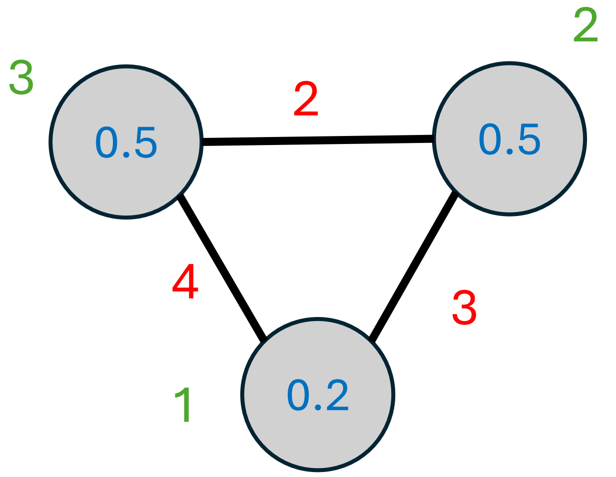


Local Rounding of Pairwise Independent Sampling

Goal: Turn a fractional solution $\vec{p}$ into an almost-as-good integral solution $\vec{y}$ *locally*.

Input:

- $\vec{p} \in (0,1]^V$ ("sampling probabilities")
- $\vec{ut} \in \mathbb{R}^V_{\geq 0}$ ("utility of including a given node")
- $\vec{cost} \in \mathbb{R}^E_{\geq 0}$ ("cost of including both endpoints of a given edge")



Notation:

$$\Phi(\vec{x}) \coloneqq \sum_{v \in V} ut(v)x(v) - \sum_{\{u,v\} \in E} cost(uv)x(u)x(v) \quad (\vec{x} \in [0,1]^V)$$

"Expected utility" – "Expected cost"

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň'23]

In $O\left(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n)\right)$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that

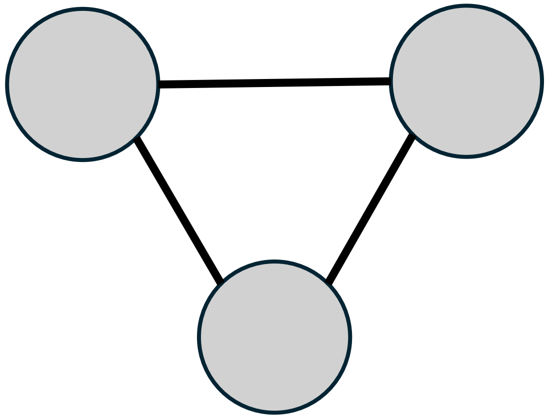
$$\Phi(\vec{y}) \geq 0.9\Phi(\vec{p}).$$

Local Rounding of Pairwise Independent Sampling

Goal: Turn a fractional solution $\vec{p}$ into an almost-as-good integral solution $\vec{y}$ *locally*.

Input:

- $\vec{p} \in (0,1]^V$ ("sampling probabilities")
- $\vec{ut} \in \mathbb{R}^V_{\geq 0}$ ("utility of including a given node")
- $\vec{cost} \in \mathbb{R}^E_{\geq 0}$ ("cost of including both endpoints of a given edge")



Notation:

$$\Phi(\vec{x}) := \sum_{v \in V} ut(v)x(v) - \sum_{\{u,v\} \in E} cost(uv)x(u)x(v) \; (\vec{x} \in [0,1]^V)$$

"Expected utility " – "Expected cost"

Max Cut:

- $p(v)$=
- $ut(v)$ =
- cost(u,v) =

Local Rounding of Pairwise Independent Sampling

Goal: Turn a fractional solution $\vec{p}$ into an almost-as-good integral solution $\vec{y}$ *locally*.

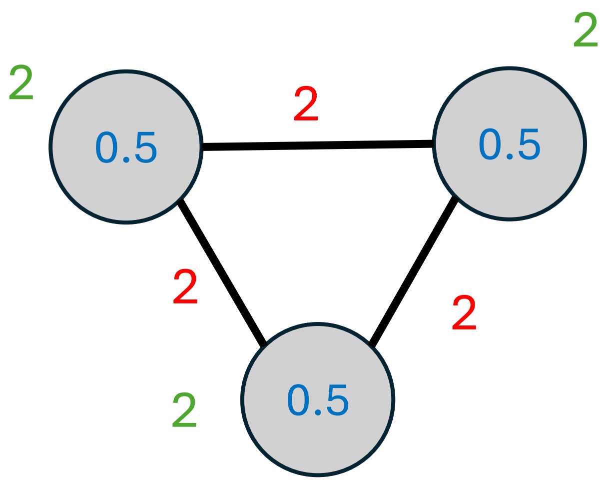
Input:

- $\vec{p} \in (0,1]^V$ ("sampling probabilities")
- $\vec{ut} \in \mathbb{R}^V_{\geq 0}$ ("utility of including a given node")
- $\vec{cost} \in \mathbb{R}^E_{\geq 0}$ ("cost of including both endpoints of a given edge")

Notation:

$$\Phi(\vec{x}) := \sum_{v \in V} ut(v)x(v) - \sum_{\{u,v\} \in E} cost(uv)x(u)x(v) \ (\vec{x} \in [0,1]^V)$$

"Expected utility " – "Expected cost"



Max Cut:

- $p(v)$= 0.5
- ut(v) = deg(v)
- cost(u,v) = 2

Local Rounding of Pairwise Independent Sampling

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň’23]

In $O\left(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n)\right)$ rounds, one can compute an integral $\mathrm{y} \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

Local Rounding of Pairwise Independent Sampling

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň'23]
In $O(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n))$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

Sequential algorithm:
Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\vec{p})$.

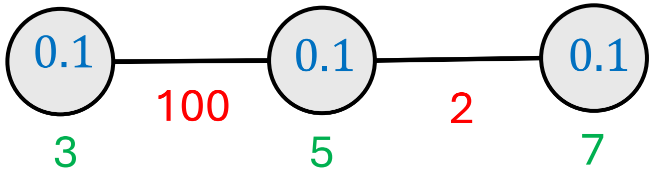
Local Rounding of Pairwise Independent Sampling

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň'23]

In $O\left(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n)\right)$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

Sequential algorithm:

Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\vec{p})$.



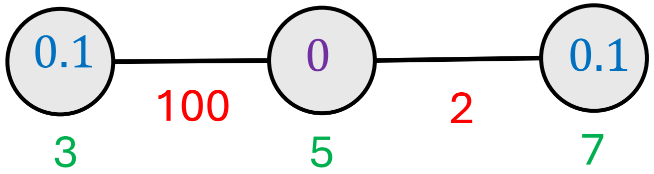
Local Rounding of Pairwise Independent Sampling

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň'23]

In $O(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n))$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

Sequential algorithm:

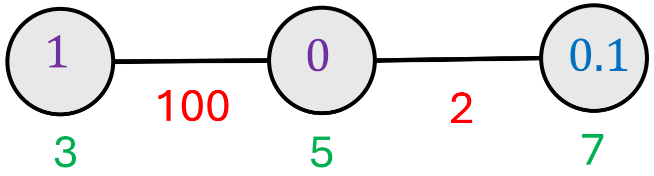
Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\vec{p})$.



Local Rounding of Pairwise Independent Sampling

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň'23]
In $O\left(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n)\right)$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

Sequential algorithm:
Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\vec{p})$.



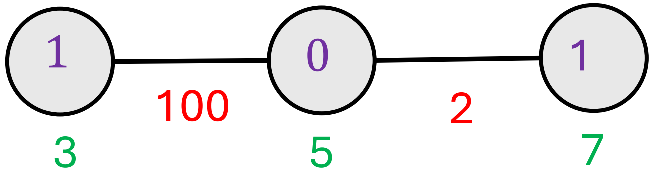
Local Rounding of Pairwise Independent Sampling

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň'23]

In $O\left(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n)\right)$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

Sequential algorithm:

Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\vec{p})$.



Local Rounding of Pairwise Independent Sampling

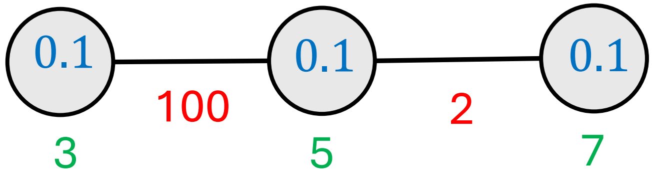
Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň'23]
In $O(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n))$ rounds, one can compute an integral $\mathrm{y} \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

Sequential algorithm:
Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\vec{p})$.
Can be parallelized, given a coloring with few colors.

Local Rounding of Pairwise Independent Sampling

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň'23]
In $O\left(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n)\right)$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

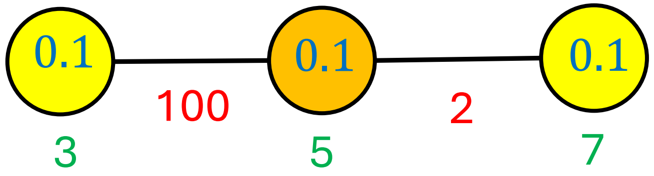
Sequential algorithm:
Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\vec{p})$.
Can be parallelized, given a coloring with few colors.



Local Rounding of Pairwise Independent Sampling

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň'23]
In $O(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n))$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

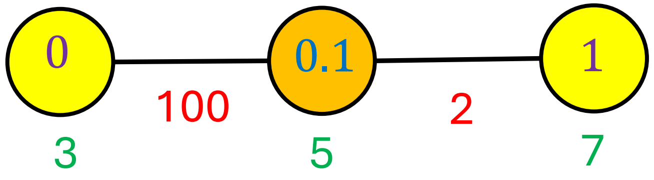
Sequential algorithm:
Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\vec{p})$.
Can be parallelized, given a coloring with few colors.



Local Rounding of Pairwise Independent Sampling

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň'23]
In $O\left(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n)\right)$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

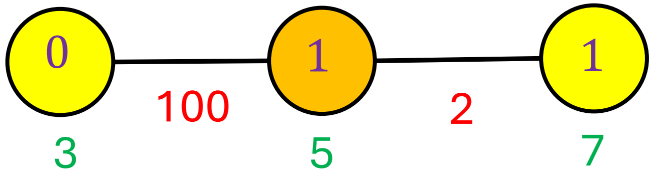
Sequential algorithm:
Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\vec{p})$.
Can be parallelized, given a coloring with few colors.



Local Rounding of Pairwise Independent Sampling

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň'23]
In $O\left(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n)\right)$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

Sequential algorithm:
Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\vec{p})$.
Can be parallelized, given a coloring with few colors.



Local Rounding of Pairwise Independent Sampling

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň'23]
In $O(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n))$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

Sequential algorithm:
Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\vec{p})$.
Can be parallelized, given a coloring with few colors.

Problem:
Valid coloring might need too many colors.

Local Rounding of Pairwise Independent Sampling

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň'23]

In $O(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n))$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that $\Phi(\overrightarrow{y}) \geq 0.9\Phi(\overrightarrow{p})$.

Sequential algorithm:

Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\overrightarrow{p})$.

Can be parallelized, given a coloring with few colors.

Problem:

Valid coloring might need too many colors.

Approach:

1. Use coloring with few monochromatic edges
2. Ignore those edges and round the rest.

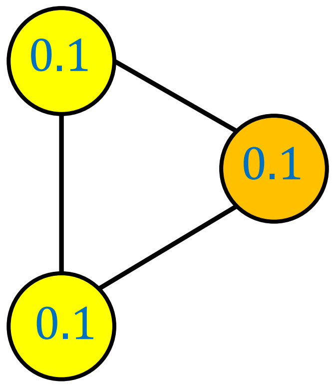
Local Rounding of Pairwise Independent Sampling

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň'23]
In $O\left(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n)\right)$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

Sequential algorithm:
Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\vec{p})$.
Can be parallelized, given a coloring with few colors.

Problem:
Valid coloring might need too many colors.

Approach:
1. Use coloring with few monochromatic edges
2. Ignore those edges and round the rest.

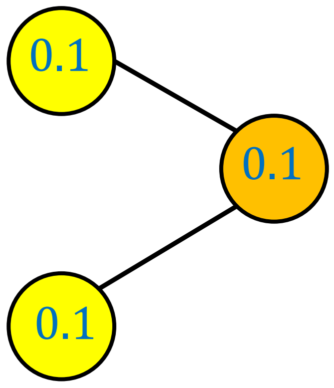


Local Rounding of Pairwise Independent Sampling

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň'23]

In $O\left(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n)\right)$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

Sequential algorithm:
Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\vec{p})$.
Can be parallelized, given a coloring with few colors.



Problem:
Valid coloring might need too many colors.

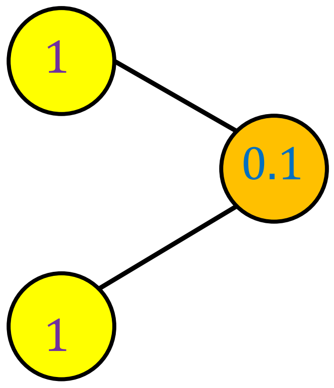
Approach:
1. Use coloring with few monochromatic edges
2. Ignore those edges and round the rest.

Local Rounding of Pairwise Independent Sampling

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň'23]
In $O\left(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n)\right)$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

Sequential algorithm:
Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\vec{p})$.
Can be parallelized, given a coloring with few colors.

Problem:
Valid coloring might need too many colors.



Approach:
1. Use coloring with few monochromatic edges
2. Ignore those edges and round the rest.

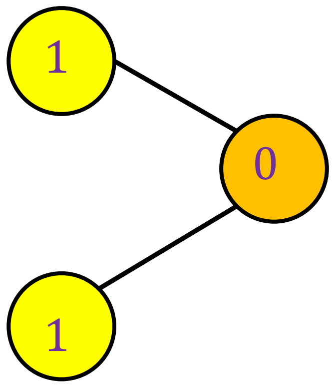
Local Rounding of Pairwise Independent Sampling

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň’23]
In $O\left(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n)\right)$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

Sequential algorithm:
Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\vec{p})$.
Can be parallelized, given a coloring with few colors.

Problem:
Valid coloring might need too many colors.

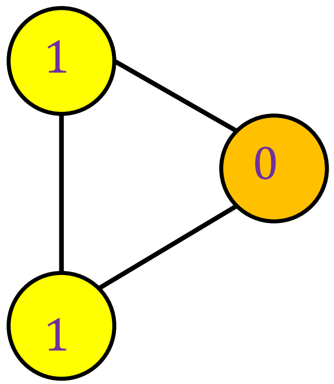
Approach:
1. Use coloring with few monochromatic edges
2. Ignore those edges and round the rest.



Local Rounding of Pairwise Independent Sampling

Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň'23]
In $O\left(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n)\right)$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

Sequential algorithm:
Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\vec{p})$.
Can be parallelized, given a coloring with few colors.



Problem:
Valid coloring might need too many colors.

Approach:
1. Use coloring with few monochromatic edges
2. Ignore those edges and round the rest.

Issue: "Contribution" of an edge can increase a lot

Local Rounding of Pairwise Independent Sampling

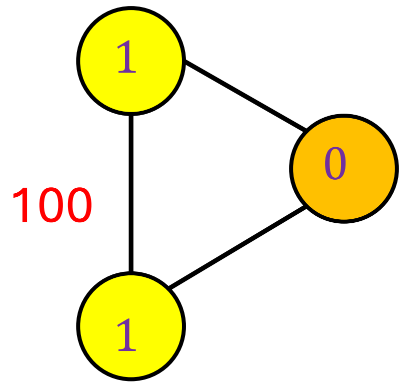
Theorem [Faour, Ghaffari, G, Kuhn, Rozhoň’23]
In $O(\log^2\left(\frac{1}{p_{min}}\right) + \log^*(n))$ rounds, one can compute an integral $y \in \{0,1\}^V$ such that $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$.

Sequential algorithm:
Sequentially computes $\vec{y} \in \{0,1\}^V$ with $\Phi(\vec{y}) \geq \Phi(\vec{p})$.
Can be parallelized, given a coloring with few colors.

Problem:
Valid coloring might need too many colors.

Approach:
1. Use coloring with few monochromatic edges
2. Ignore those edges and round the rest.

Issue: “Contribution” of an edge can increase a lot



Before: $100 \cdot 0.1 \cdot 0.1$
After: $100 \cdot 1 \cdot 1$

Algorithm: High-Level Outline

Algorithm: High-Level Outline

- Assume the input vector $\vec{p}$ is $\frac{1}{2^I}$-integral

q-integral: each entry is either 0 or at least q

Algorithm: High-Level Outline

q-integral: each entry is either 0 or at least q

- Assume the input vector $\vec{p}$ is $\frac{1}{2^I}$-integral
- The algorithm has *I* iterations, in each iteration we increase the integrality by a 2-factor

Algorithm: High-Level Outline

q-integral: each entry is either 0 or at least q

- Assume the input vector $\vec{p}$ is $\frac{1}{2^I}$-integral
- The algorithm has *I* iterations, in each iteration we increase the integrality by a 2-factor
- In each iteration, the potential decreases by at most a $\left(1-\frac{1}{10I}\right)$-factor

Algorithm: High-Level Outline

q-integral: each entry is either 0 or at least q

- Assume the input vector $\vec{p}$ is $\frac{1}{2^I}$-integral
- The algorithm has *I* iterations, in each iteration we increase the integrality by a 2-factor
- In each iteration, the potential decreases by at most a $\left(1-\frac{1}{10I}\right)$-factor
- As $\left(1-\frac{1}{10I}\right)^I \geq 0.9$, we therefore get $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$, as needed

Algorithm: High-Level Outline

q-integral: each entry is either 0 or at least q

- Assume the input vector $\vec{p}$ is $\frac{1}{2^I}$-integral
- The algorithm has *I* iterations, in each iteration we increase the integrality by a 2-factor
- In each iteration, the potential decreases by at most a $\left(1-\frac{1}{10I}\right)$-factor
- As $\left(1-\frac{1}{10I}\right)^I \geq 0.9$, we therefore get $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$, as needed

Main Technical Lemma:
Given a $\frac{1}{2^i}$-integral $\overrightarrow{p_i}$, one can compute a $\frac{1}{2^{i-1}}$ -integral $\overrightarrow{p_{i-1}}$ with $\Phi(\overrightarrow{p_{i-1}}) \geq \left(1-\frac{1}{10I}\right)\Phi(\overrightarrow{p_i})$ in $O(I)$ rounds

Algorithm: High-Level Outline

q-integral: each entry is either 0 or at least q

- Assume the input vector $\vec{p}$ is $\frac{1}{2^I}$-integral
- The algorithm has *I* iterations, in each iteration we increase the integrality by a 2-factor
- In each iteration, the potential decreases by at most a $\left(1-\frac{1}{10I}\right)$-factor
- As $\left(1-\frac{1}{10I}\right)^I \geq 0.9$, we therefore get $\Phi(\vec{y}) \geq 0.9\Phi(\vec{p})$, as needed

Main Technical Lemma:
Given a $\frac{1}{2^i}$-integral $\overrightarrow{p_i}$, one can compute a $\frac{1}{2^{i-1}}$ -integral $\overrightarrow{p_{i-1}}$ with $\Phi(\overrightarrow{p_{i-1}}) \geq \left(1-\frac{1}{10I}\right)\Phi(\overrightarrow{p_i})$ in $O(I)$ rounds

- Overall round complexity: $O(I^2)$

Algorithm: One Rounding Step

Main Technical Lemma:
Given a $\frac{1}{2^i}$-integral $\overrightarrow{p_i}$, one can compute a $\frac{1}{2^{i-1}}$-integral $\overrightarrow{p_{i-1}}$ with $\Phi(\overrightarrow{p_{i-1}}) \geq \left(1 - \frac{1}{10I}\right)\Phi(\overrightarrow{p_i})$ in $O(I)$ rounds

Algorithm: One Rounding Step

Main Technical Lemma:
Given a $\frac{1}{2^i}$-integral $\overrightarrow{p_i}$, one can compute a $\frac{1}{2^{i-1}}$-integral $\overrightarrow{p_{i-1}}$ with $\Phi(\overrightarrow{p_{i-1}}) \geq \left(1 - \frac{1}{10I}\right)\Phi(\overrightarrow{p_i})$ in $O(I)$ rounds

Algorithm:

1.) Compute a coloring with $O(I)$ colors so that monochromatic edges contribute $\leq \frac{1}{40I}\Phi(\overrightarrow{p_i})$.

Formally: $\sum_{\{u,v\}\in E^{mono}} cost(uv)(\overrightarrow{p_i})_u(\overrightarrow{p_i})_v \leq \frac{1}{40I}\Phi(\overrightarrow{p_i})$

2.) Drop all edges in E^{mono}

3.) double integrality: $\frac{1}{2^i} \to \frac{1}{2^{i-1}}$, increasing each value by at most a 2-factor.

Algorithm: One Rounding Step

Main Technical Lemma:
Given a $\frac{1}{2^i}$-integral $\overrightarrow{p_i}$, one can compute a $\frac{1}{2^{i-1}}$ -integral $\overrightarrow{p_{i-1}}$ with $\Phi(\overrightarrow{p_{i-1}}) \geq \left(1 - \frac{1}{10I}\right)\Phi(\overrightarrow{p_i})$ in $O(I)$ rounds

Algorithm:

1.) Compute a coloring with $O(I)$ colors so that monochromatic edges contribute $\leq \frac{1}{40I}\Phi(\overrightarrow{p_i})$.

Formally: $\sum_{\{u,v\}\in E^{mono}} cost(uv)(\overrightarrow{p_i})_u(\overrightarrow{p_i})_v \leq \frac{1}{40I}\Phi(\overrightarrow{p_i})$

2.) Drop all edges in E^{mono}

3.) double integrality: $\frac{1}{2^i} \to \frac{1}{2^{i-1}}$, increasing each value by at most a 2-factor.

Potential Analysis:

$$\begin{aligned}
\Phi(\overrightarrow{p_{i-1}}) &= \Phi_{E\setminus E^{mono}}(\overrightarrow{p_{i-1}}) - \sum_{\{u,v\}\in E^{mono}} cost(uv)(\overrightarrow{p_{i-1}})_u(\overrightarrow{p_{i-1}})_v \\
&\geq \Phi_{E\setminus E^{mono}}(\overrightarrow{p_i}) - 4\sum_{\{u,v\}\in E^{mono}} cost(uv)(\overrightarrow{p_i})_u(\overrightarrow{p_i})_v \\
&\geq \Phi(\overrightarrow{p_i}) - 4\frac{1}{40I}\Phi(\overrightarrow{p_i}) = \left(1 - \frac{1}{10I}\right)\Phi(\overrightarrow{p_i})
\end{aligned}$$

Pairwise Analysis #2:
Derandomizing One Round of Luby’s MIS algorithm on a regular graph

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

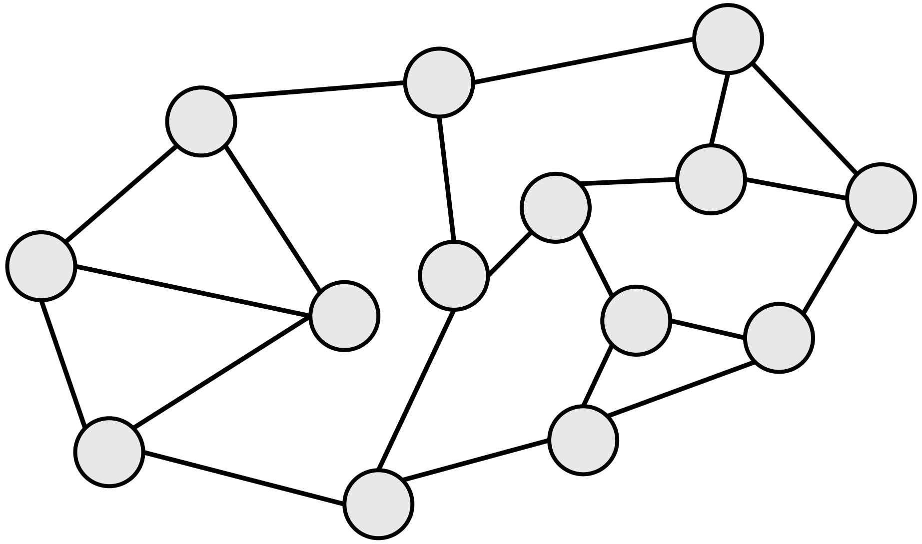
Input: Δ-regular graph

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set I

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

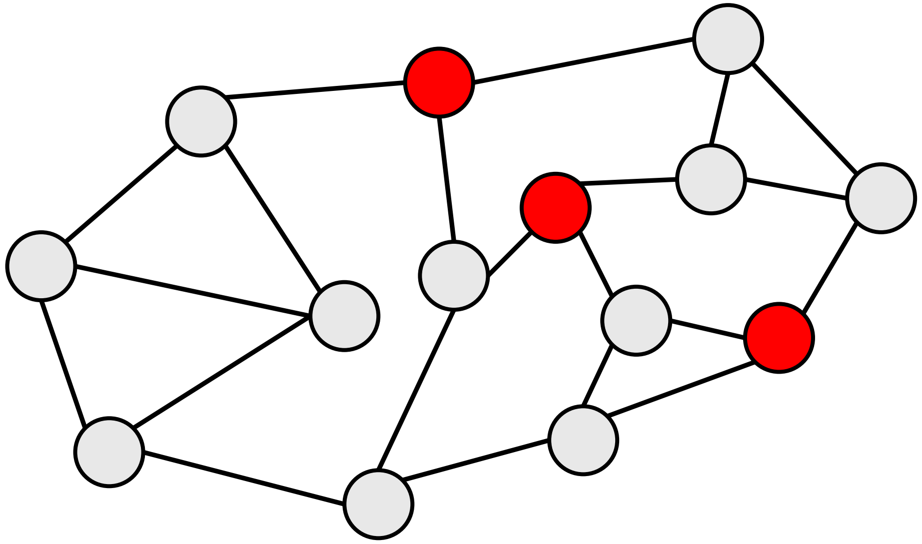
Input: Δ-regular graph
Output: Independent Set *I*



$\Delta = 3$

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

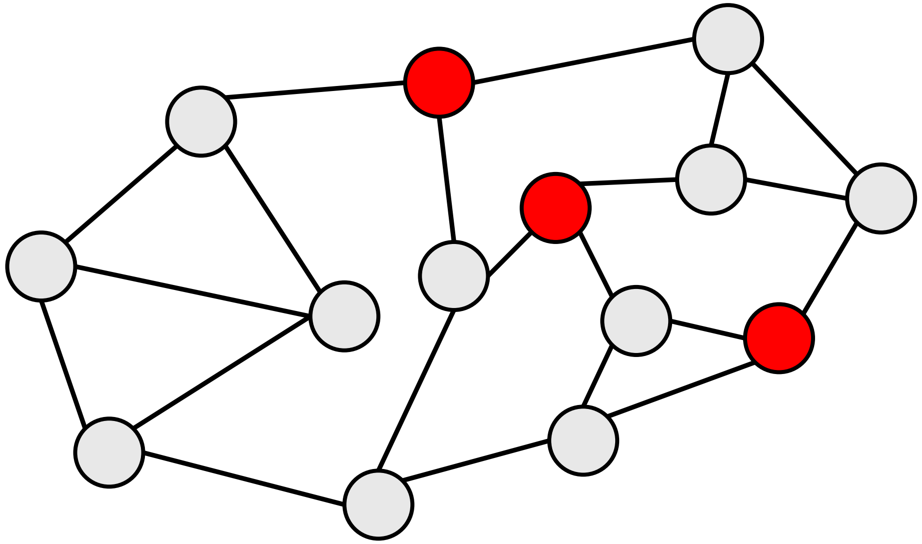
Input: Δ-regular graph
Output: Independent Set /



$\Delta = 3$

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

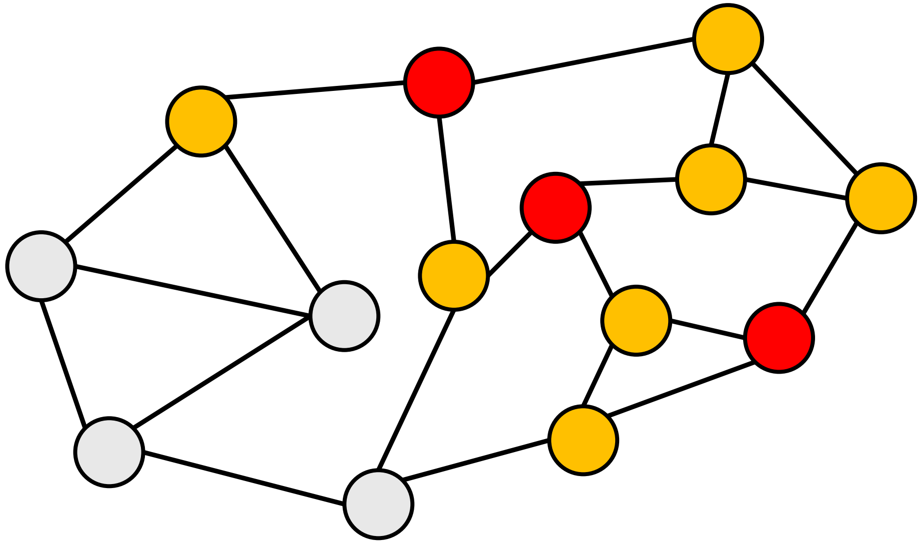
Input: Δ-regular graph
Output: Independent Set I
Goal: $\Omega(n)$ nodes neighbor I (in expectation)



$\Delta = 3$

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set I
Goal: $\Omega(n)$ nodes neighbor I (in expectation)

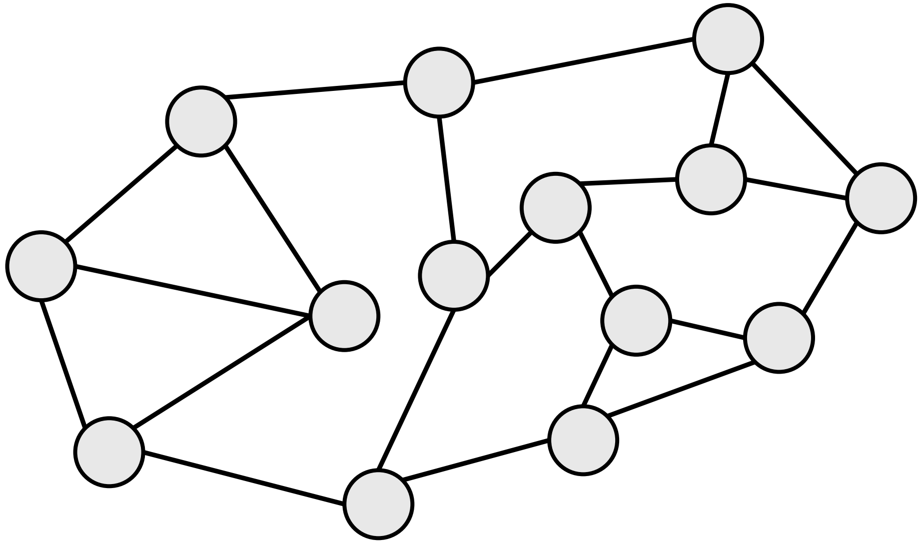


$\Delta = 3$

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set I
Goal: $\Omega(n)$ nodes neighbor I (in expectation)

Randomized Algorithm:

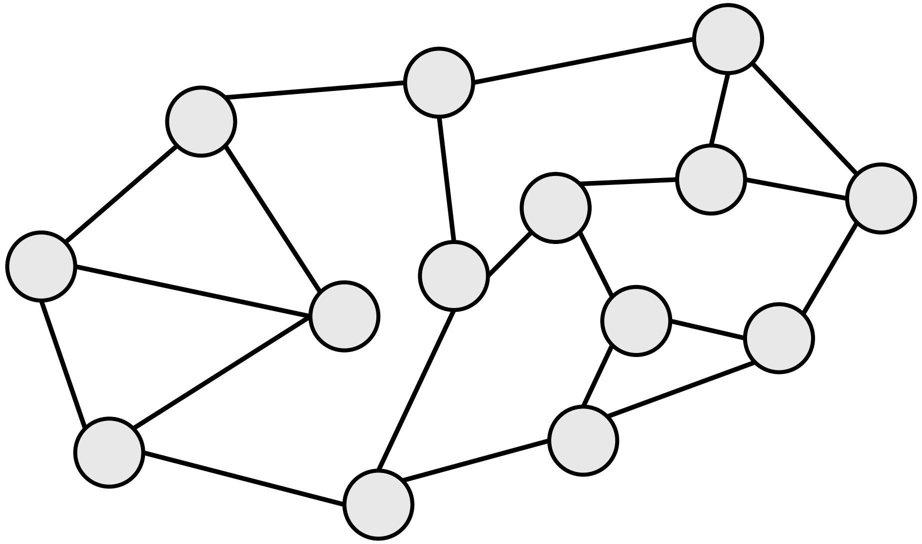


Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set I
Goal: $\Omega(n)$ nodes neighbor I (in expectation)

Randomized Algorithm:

1.) Sample each node u with prob. $\mathrm{p} = \frac{1}{10\Delta}$ (pw. ind.)

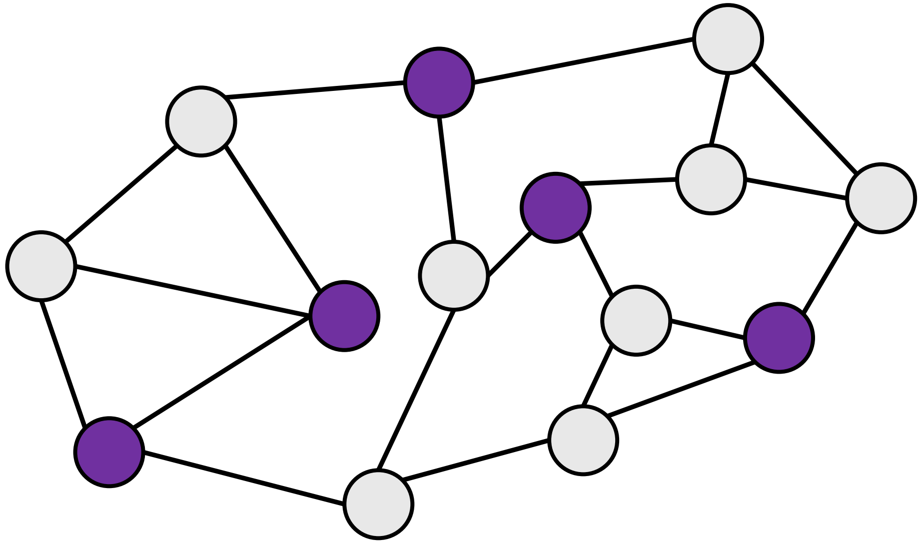


Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set I
Goal: $\Omega(n)$ nodes neighbor I (in expectation)

Randomized Algorithm:

1.) Sample each node *u* with prob. $\mathrm{p} = \frac{1}{10\Delta}$ (pw. ind.)

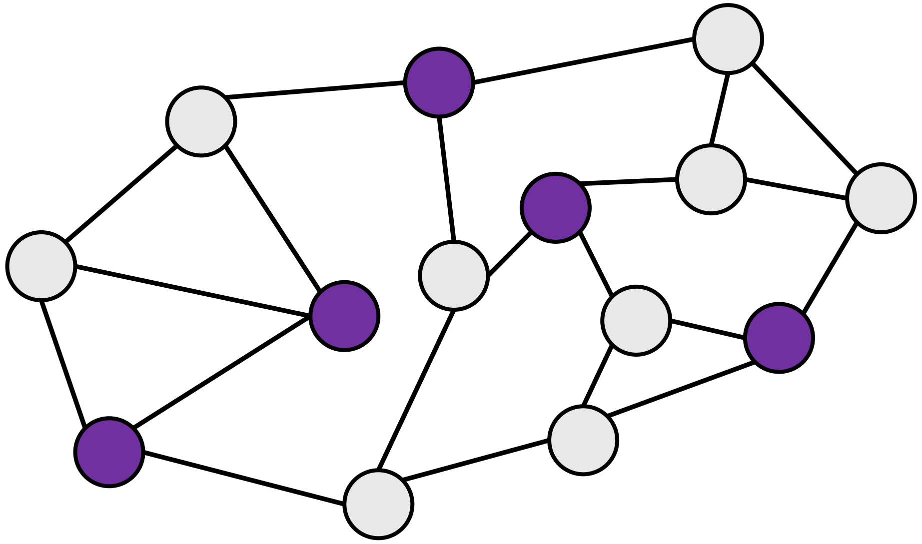


Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set *I*
Goal: $\Omega(n)$ nodes neighbor *I* (in expectation)

Randomized Algorithm:

1.) Sample each node *u* with prob. $\mathrm{p} = \frac{1}{10\Delta}$ (pw. ind.)
2.) Include *u* in *I* if a) *u* is sampled
b) no neighbor of *u* is sampled



Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

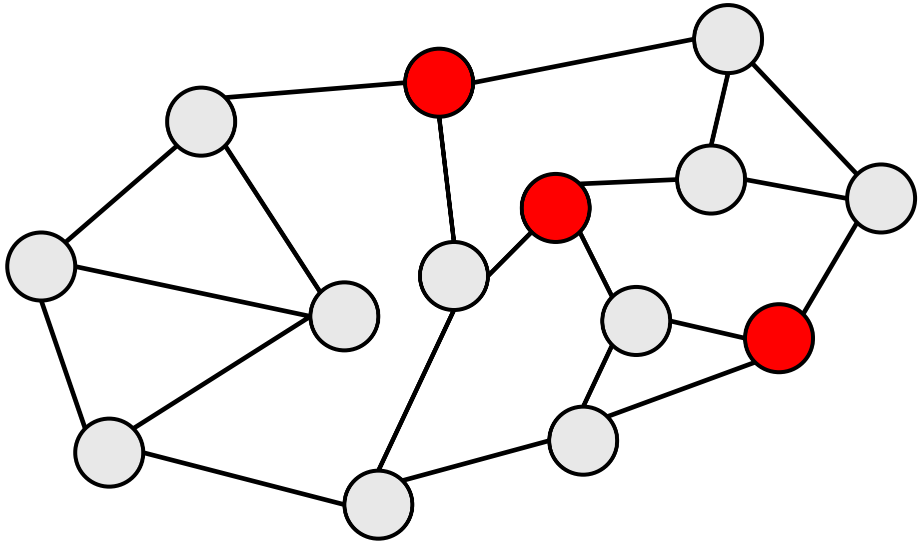
Input: Δ-regular graph
Output: Independent Set I
Goal: $\Omega(n)$ nodes neighbor I (in expectation)

Randomized Algorithm:

1.) Sample each node u with prob. $\mathrm{p} = \frac{1}{10\Delta}$ (pw. ind.)

2.) Include u in I if a) u is sampled

b) no neighbor of u is sampled



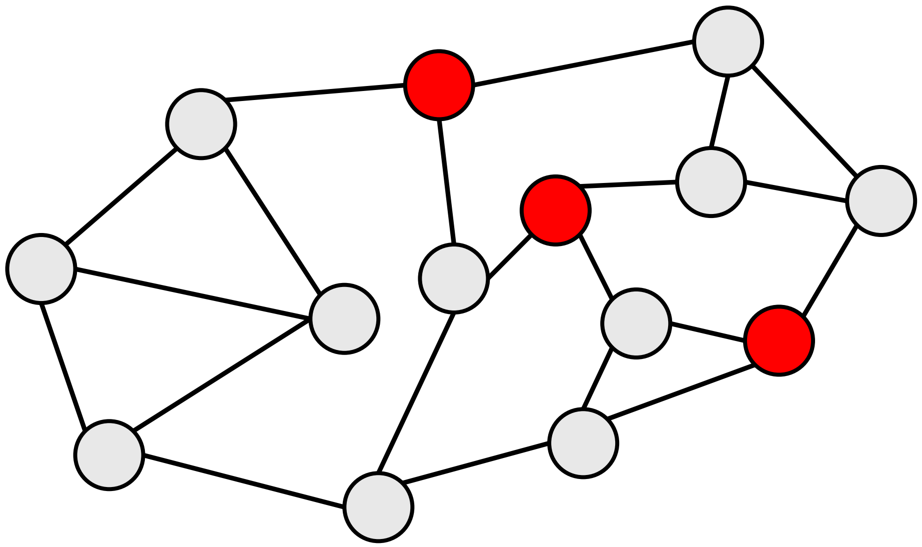
Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set I
Goal: $\Omega(n)$ nodes neighbor I (in expectation)

Randomized Algorithm:

1.) Sample each node u with prob. $\mathrm{p} = \frac{1}{10\Delta}$ (pw. ind.)

2.) Include u in I if a) u is sampled
b) no neighbor of u is sampled



Notation:

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set *I*
Goal: $\Omega(n)$ nodes neighbor *I* (in expectation)

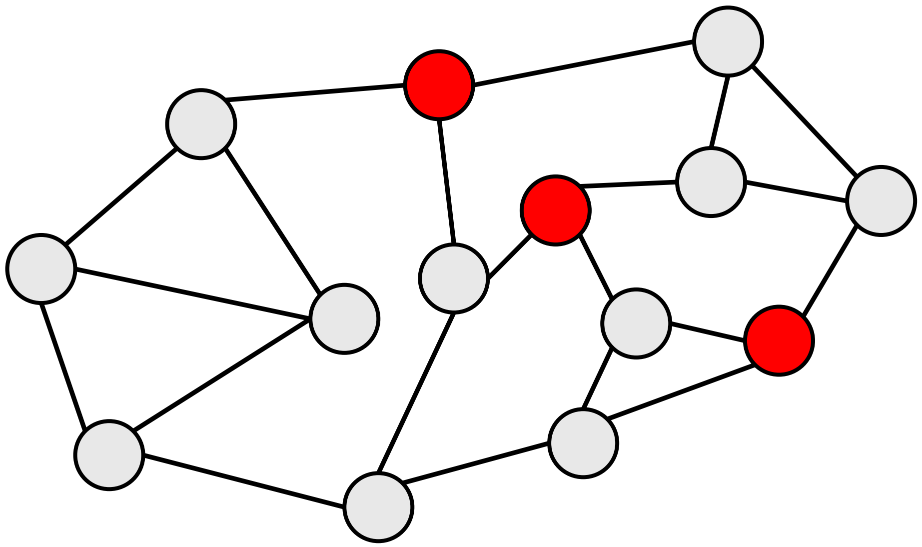
Randomized Algorithm:

1.) Sample each node *u* with prob. $\mathrm{p} = \frac{1}{10\Delta}$ (pw. ind.)

2.) Include *u* in *I* if a) *u* is sampled
b) no neighbor of *u* is sampled

Notation:

Y_v := Indicator that *v* is neighboring a node in *I*



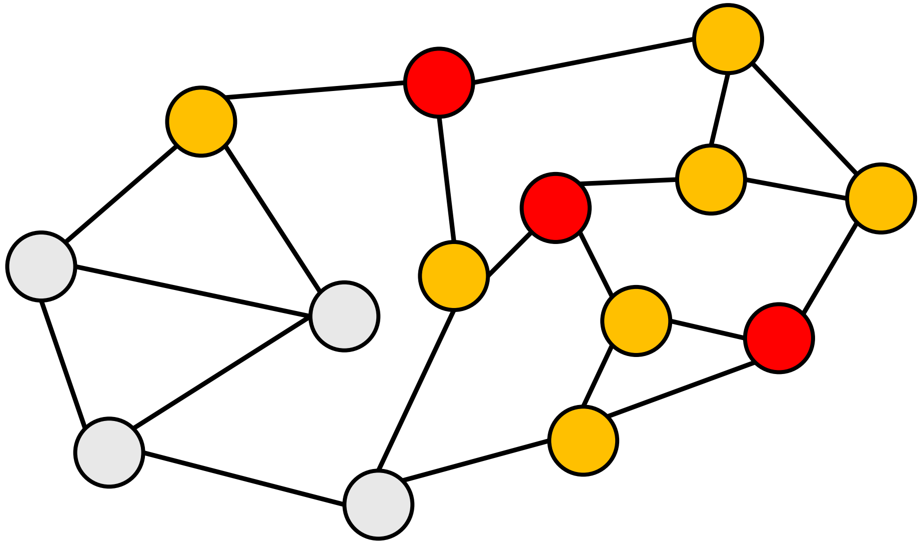
Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set I
Goal: $\Omega(n)$ nodes neighbor I (in expectation)

Randomized Algorithm:

1.) Sample each node u with prob. $\mathrm{p} = \frac{1}{10\Delta}$ (pw. ind.)

2.) Include u in I if a) u is sampled
b) no neighbor of u is sampled



Notation:

Y_v := Indicator that v is neighboring a node in I

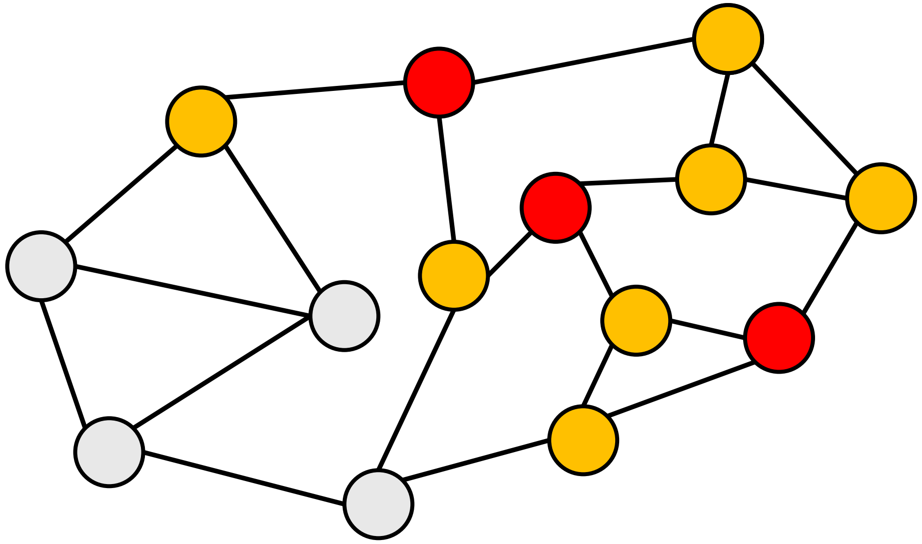
Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set *I*
Goal: $\Omega(n)$ nodes neighbor *I* (in expectation)

Randomized Algorithm:

1.) Sample each node *u* with prob. $\mathrm{p} = \frac{1}{10\Delta}$ (pw. ind.)

2.) Include *u* in *I* if a) *u* is sampled
b) no neighbor of *u* is sampled



Notation:

Y_v := Indicator that *v* is neighboring a node in *I*

X_u:= Indicator that *u* is sampled

Pairwise Analysis #2: Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set *I*
Goal: $\Omega(n)$ nodes neighbor *I* (in expectation)

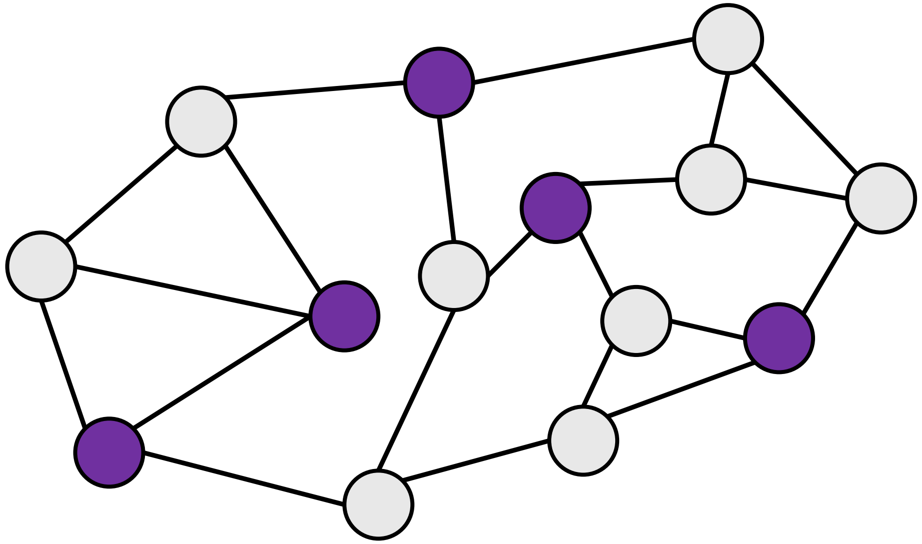
Randomized Algorithm:

1.) Sample each node *u* with prob. $\mathrm{p} = \frac{1}{10\Delta}$ (pw. ind.)

2.) Include *u* in *I* if a) *u* is sampled
b) no neighbor of *u* is sampled

Notation:

Y_v := Indicator that *v* is neighboring a node in *I*
X_u:= Indicator that *u* is sampled



Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set I
Goal: $\Omega(n)$ nodes neighbor I (in expectation)

Randomized Algorithm:

1.) Sample each node u with prob. $\mathrm{p} = \frac{1}{10\Delta}$ (pw. ind.)

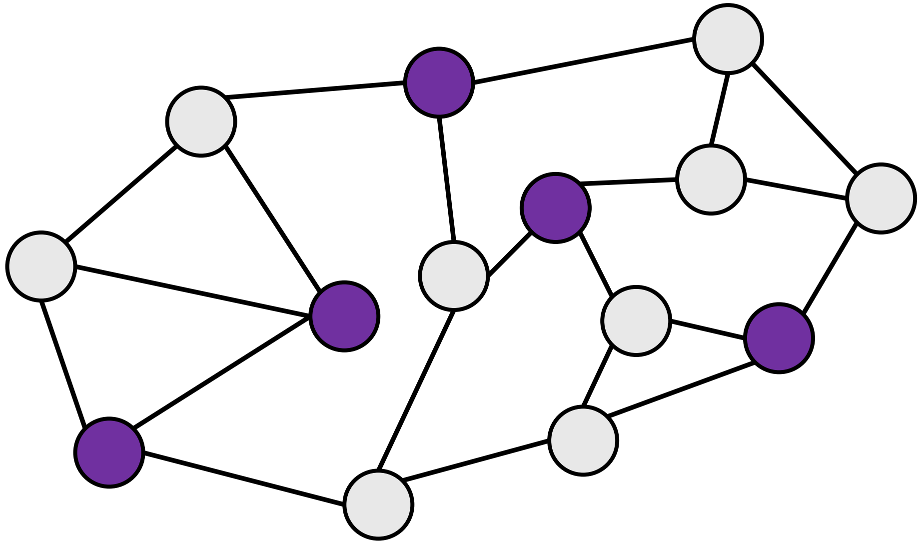
2.) Include u in I if a) u is sampled
b) no neighbor of u is sampled

Notation:

Y_v := Indicator that v is neighboring a node in I

X_u := Indicator that u is sampled

$$Z_v := \sum_{u \in N(v)} X_u - \sum_{u \neq u' \in N(v)} X_u X_{u'} - \sum_{u \in N(v)} \sum_{w \in N(u)} X_u X_w$$



Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set *I*
Goal: $\Omega(n)$ nodes neighbor *I* (in expectation)

Randomized Algorithm:

1.) Sample each node *u* with prob. $\mathrm{p} = \frac{1}{10\Delta}$ (pw. ind.)

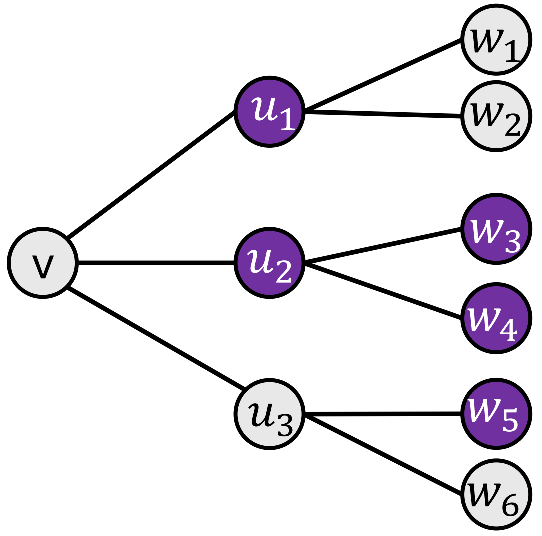
2.) Include *u* in *I* if a) *u* is sampled
b) no neighbor of *u* is sampled

Notation:

Y_v := Indicator that *v* is neighboring a node in *I*

X_u := Indicator that *u* is sampled

$$Z_v := \sum_{u \in N(v)} X_u - \sum_{u \neq u' \in N(v)} X_u X_{u'} - \sum_{u \in N(v)} \sum_{w \in N(u)} X_u X_w$$



Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set *I*
Goal: $\Omega(n)$ nodes neighbor *I* (in expectation)

Randomized Algorithm:

1.) Sample each node *u* with prob. $\mathrm{p} = \frac{1}{10\Delta}$ (pw. ind.)

2.) Include *u* in *I* if a) *u* is sampled

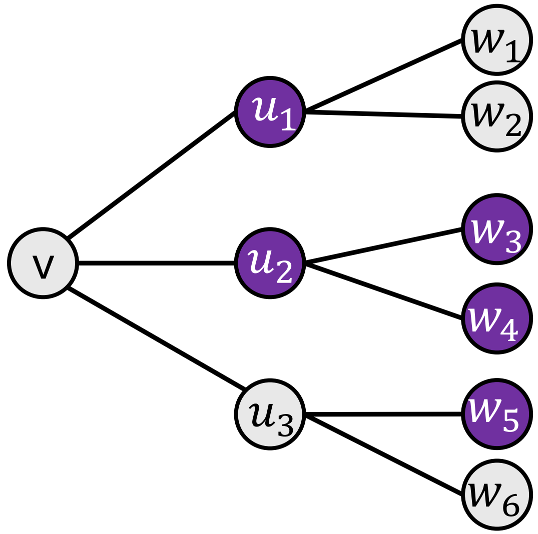
b) no neighbor of *u* is sampled

Notation:

Y_v := Indicator that *v* is neighboring a node in *I*

X_u:= Indicator that *u* is sampled

$$Z_v := \sum_{u \in N(v)} X_u - \sum_{u \neq u' \in N(v)} X_u X_{u'} - \sum_{u \in N(v)} \sum_{w \in N(u)} X_u X_w$$



What is Y_v?

What is Z_v?

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set *I*
Goal: $\Omega(n)$ nodes neighbor *I* (in expectation)

Randomized Algorithm:

1.) Sample each node *u* with prob. $\mathrm{p} = \frac{1}{10\Delta}$ (pw. ind.)

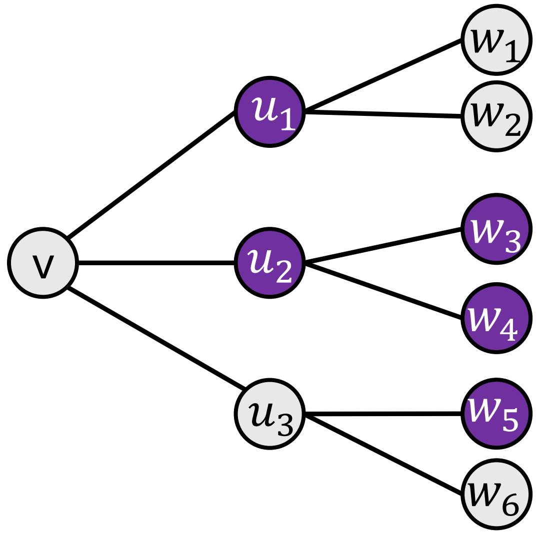
2.) Include *u* in *I* if a) *u* is sampled
b) no neighbor of *u* is sampled

Notation:

Y_v := Indicator that *v* is neighboring a node in *I*

X_u := Indicator that *u* is sampled

$$Z_v := \sum_{u \in N(v)} X_u - \sum_{u \neq u' \in N(v)} X_u X_{u'} - \sum_{u \in N(v)} \sum_{w \in N(u)} X_u X_w$$



What is Y_v? *1*

What is Z_v?

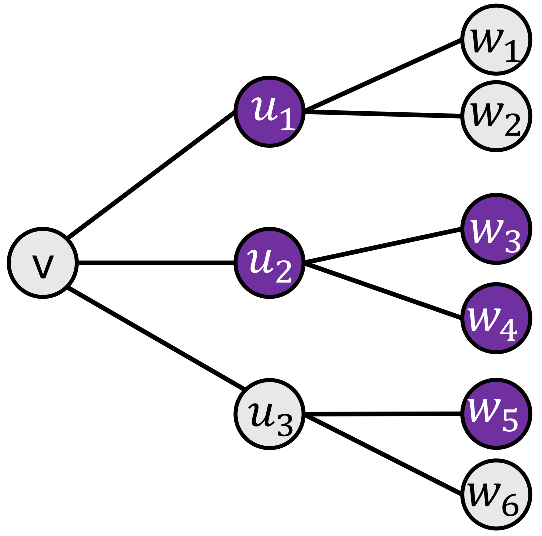
Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set *I*
Goal: $\Omega(n)$ nodes neighbor *I* (in expectation)

Randomized Algorithm:
1.) Sample each node *u* with prob. $\mathrm{p} = \frac{1}{10\Delta}$ (pw. ind.)
2.) Include *u* in *I* if a) *u* is sampled
b) no neighbor of *u* is sampled

Notation:
Y_v := Indicator that *v* is neighboring a node in *I*
X_u := Indicator that *u* is sampled

$$Z_v := \sum_{u \in N(v)} X_u - \sum_{u \neq u' \in N(v)} X_u X_{u'} - \sum_{u \in N(v)} \sum_{w \in N(u)} X_u X_w$$



What is Y_v? *1*

What is Z_v? *-1*

Pairwise Analysis #2: Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set I
Goal: $\Omega(n)$ nodes neighbor I (in expectation)

Randomized Algorithm:

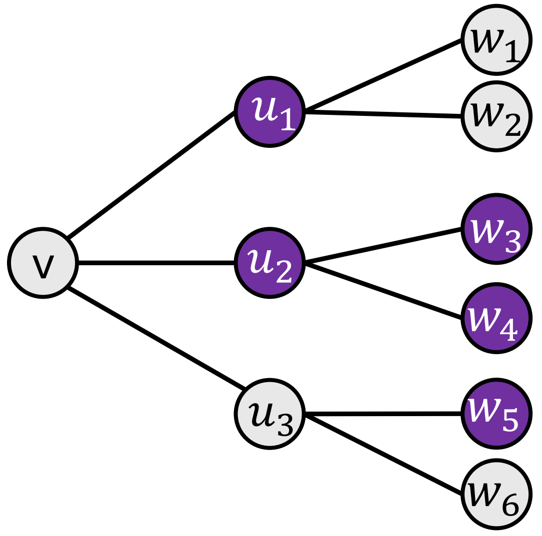
1.) Sample each node *u* with prob. $\mathrm{p} = \frac{1}{10\Delta}$ (pw. ind.)

2.) Include *u* in I if a) *u* is sampled
b) no neighbor of *u* is sampled

Notation:

Y_v := Indicator that *v* is neighboring a node in I
X_u:= Indicator that *u* is sampled

$$Z_v := \sum_{u \in N(v)} X_u - \sum_{u \neq u' \in N(v)} X_u X_{u'} - \sum_{u \in N(v)} \sum_{w \in N(u)} X_u X_w$$



What is Y_v? *1*

What is Z_v? *-1*

$\sum_{u \in N(v)} X_u = 2$

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set *I*
Goal: $\Omega(n)$ nodes neighbor *I* (in expectation)

Randomized Algorithm:

1.) Sample each node *u* with prob. $\mathrm{p} = \frac{1}{10\Delta}$ (pw. ind.)

2.) Include *u* in *I* if a) *u* is sampled

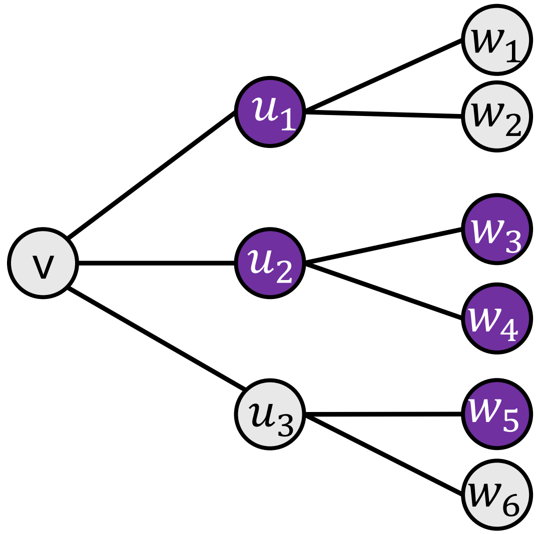
b) no neighbor of *u* is sampled

Notation:

Y_v := Indicator that *v* is neighboring a node in *I*

X_u:= Indicator that *u* is sampled

$$Z_v := \sum_{u \in N(v)} X_u - \sum_{u \neq u' \in N(v)} X_u X_{u'} - \sum_{u \in N(v)} \sum_{w \in N(u)} X_u X_w$$



What is Y_v? *1*

What is Z_v? *-1*

$\sum_{u \in N(v)} X_u = 2$

$\sum_{u \neq u' \in N(v)} X_u X_{u'} = 1$

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set I
Goal: $\Omega(n)$ nodes neighbor I (in expectation)

Randomized Algorithm:

1.) Sample each node u with prob. $p = \frac{1}{10\Delta}$ (pw. ind.)

2.) Include u in I if a) u is sampled

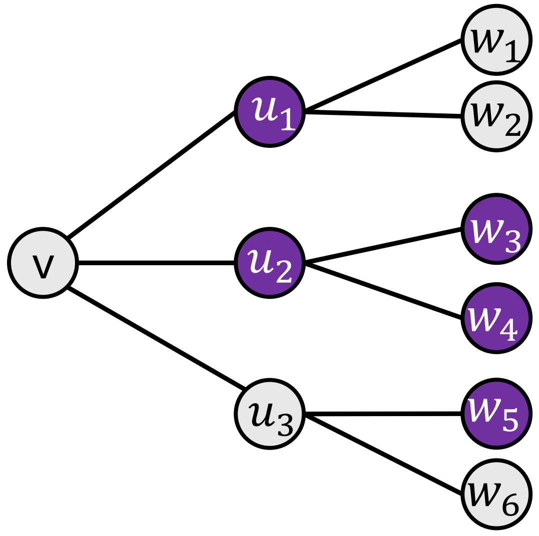
b) no neighbor of u is sampled

Notation:

Y_v := Indicator that v is neighboring a node in I

X_u := Indicator that u is sampled

$$Z_v := \sum_{u \in N(v)} X_u - \sum_{u \neq u' \in N(v)} X_u X_{u'} - \sum_{u \in N(v)} \sum_{w \in N(u)} X_u X_w$$



What is Y_v? *1*

What is Z_v? *-1*

$\sum_{u \in N(v)} X_u = 2$

$\sum_{u \neq u' \in N(v)} X_u X_{u'} = 1$

$\sum_{u \in N(v)} \sum_{w \in N(u)} X_u X_w = 2$

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Input: Δ-regular graph
Output: Independent Set *I*
Goal: $\Omega(n)$ nodes neighbor *I* (in expectation)

Randomized Algorithm:

1.) Sample each node *u* with prob. $p = \frac{1}{10\Delta}$ (pw. ind.)

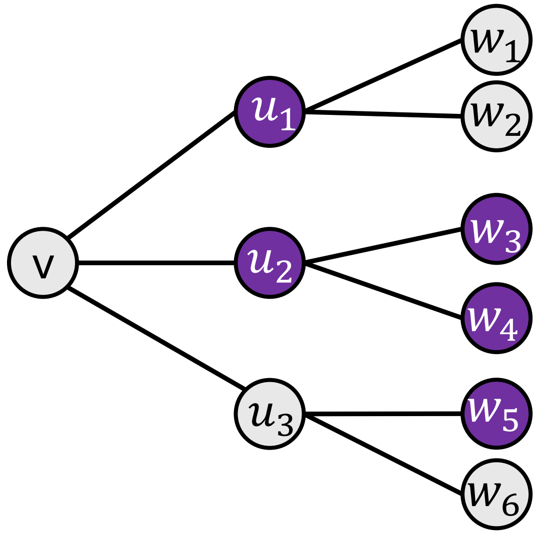
2.) Include *u* in *I* if a) *u* is sampled
b) no neighbor of *u* is sampled

Notation:

Y_v := Indicator that *v* is neighboring a node in *I*

X_u := Indicator that *u* is sampled

$$Z_v := \sum_{u \in N(v)} X_u - \sum_{u \neq u' \in N(v)} X_u X_{u'} - \sum_{u \in N(v)} \sum_{w \in N(u)} X_u X_w$$



Claim 1:
Z_v is a pessimistic estimator of Y_v ($Z_v \leq Y_v$)

Claim follows from:

1.) $Z_v \leq 1$

2.) $Y_v = 0$ implies $Z_v \leq 0$

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Claim 2: $E[Z_v] \geq \frac{1}{20}$

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Claim 2: $E[Z_v] \geq \frac{1}{20}$

Note that $E[X_u] = \frac{1}{10\Delta}, E\left[X_{u_1} X_{u_2}\right] = \frac{1}{100\Delta^2} \quad (u_1 \neq u_2)$

Pairwise Analysis #2: Derandomizing One Round of Luby's MIS algorithm on a regular graph

Claim 2: $E[Z_v] \geq \frac{1}{20}$

Note that $E[X_u] = \frac{1}{10\Delta}, E\left[X_{u_1}X_{u_2}\right] = \frac{1}{100\Delta^2} \quad (u_1 \neq u_2)$

Therefore,

$$E[Z_v] = \sum_{u \in N(v)} E[X_u] - \sum_{u \neq u' \in N(v)} E[X_u X_{u'}] - \sum_{u \in N(v)} \sum_{w \in N(u)} E[X_u X_w]$$
$$= \Delta \cdot \frac{1}{10\Delta} - \binom{\Delta}{2} \cdot \frac{1}{100\Delta^2} - \Delta^2 \cdot \frac{1}{100\Delta^2}$$
$$\geq \frac{1}{10} - \frac{1}{100} - \frac{1}{100}$$

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Goal: $\Omega(n)$ nodes neighbor ***I*** (in expectation)

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Goal: $\Omega(n)$ nodes neighbor ***I*** (in expectation)

By definition, $\sum_v Y_v$ nodes are neighboring ***I***
Claim 1 implies: $\sum_v Y_v \geq \sum_v Z_v$
Claim 2 implies: E[$\sum_v Z_v$] ≥ n/20
Therefore, the expected number of nodes neighboring **I** is at least n/20 ☺

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Goal: $\Omega(n)$ nodes neighbor *I* (in expectation)

By definition, $\sum_v Y_v$ nodes are neighboring *I*
Claim 1 implies: $\sum_v Y_v \geq \sum_v Z_v$
Claim 2 implies: E$[\sum_v Z_v] \geq$ n/20
Therefore, the expected number of nodes neighboring I is at least n/20 ☺

By the Local Rounding Theorem, this gives:

Corollary:
In $O(\log^2(\Delta) + \log^*(n))$ rounds, one can deterministically compute an independent set neighboring $\Omega(n)$ nodes.

Pairwise Analysis #2:
Derandomizing One Round of Luby's MIS algorithm on a regular graph

Goal: $\Omega(n)$ nodes neighbor *I* (in expectation)

By definition, $\sum_v Y_v$ nodes are neighboring *I*
Claim 1 implies: $\sum_v Y_v \geq \sum_v Z_v$
Claim 2 implies: $E[\sum_v Z_v] \geq n/20$
Therefore, the expected number of nodes neighboring I is at least n/20 ☺

By the Local Rounding Theorem, this gives:

Corollary:
In $O(\log^2(\Delta) + \log^*(n))$ rounds, one can deterministically compute an independent set neighboring $\Omega(n)$ nodes.

In fact, with slightly more work, one can show that each round of Luby's MIS algorithm can be derandomized in $O(\log^2(\Delta))$ rounds

MIS via Local Rounding

Theorem [FG**G**KR ’23] **:**
An MIS can be deterministically computed in $O(log\, n\, log^2\, \Delta)$ rounds.

MIS via Local Rounding

Theorem [FG**G**KR '23] **:**
An MIS can be deterministically computed in $O(log\, n\, log^2\, \Delta)$ rounds.

If $\Delta \leq\ 2^{log^{\frac{1}{3}}(n)}$, we get a round complexity of $\mathrm{O}(\log^{5/3} n)$!

MIS via Local Rounding

Theorem [FGGKR '23] **:**
An MIS can be deterministically computed in $O(log\, n\, log^2\, \Delta)$ rounds.

If $\Delta \leq\ 2^{log^{\frac{1}{3}}(n)}$, we get a round complexity of $\mathrm{O}(\log^{5/3} n)$!

What if Δ is larger?

Technique #2: Intra-Cluster Rounding

Key Definition: Cluster Degree

Let C be a partition of the nodes into clusters.

The **cluster degree** of a node v is defined as

$$\deg_C(v) = \text{number of clusters containing a neighbor of } v.$$

The **maximum cluster degree** is

$$\Delta_C \coloneqq \max_{v \in V} \deg_C(v).$$

Key Definition: Cluster Degree

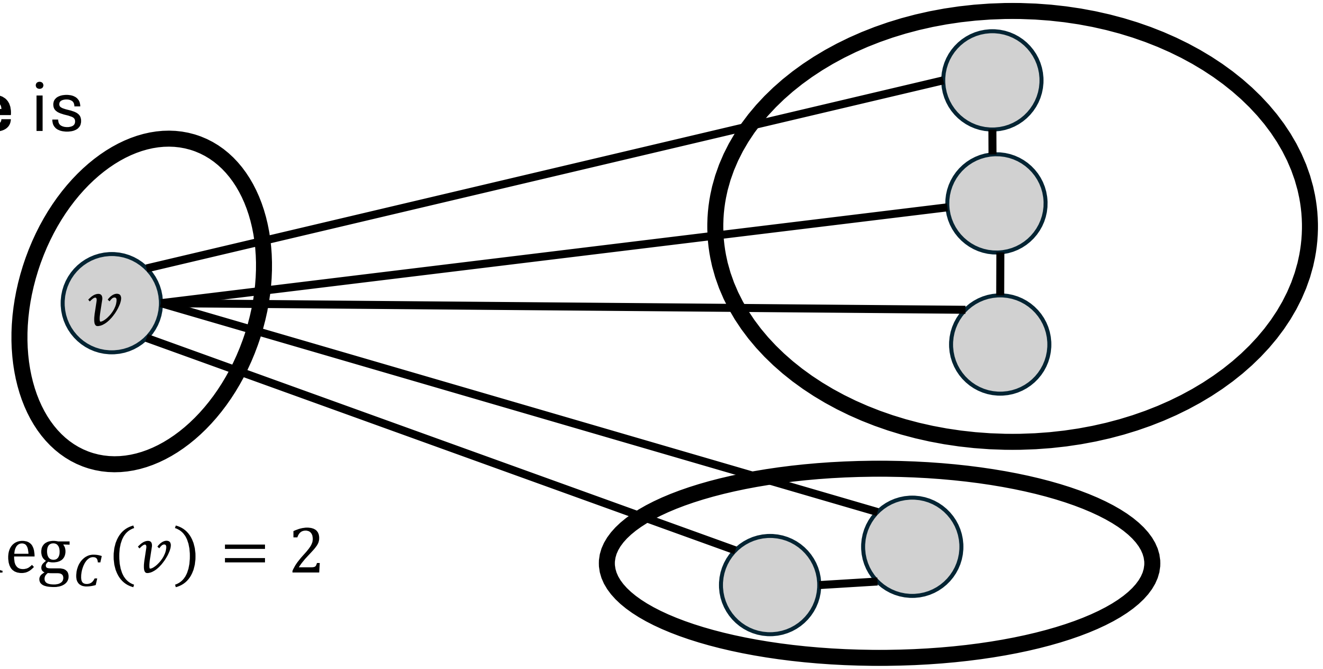
Let C be a partition of the nodes into clusters.

The **cluster degree** of a node v is defined as

$$\deg_C(v) = \text{number of clusters containing a neighbor of } v.$$

The **maximum cluster degree** is

$$\Delta_C \coloneqq \max_{v \in V} \deg_C(v).$$



$\deg_C(v) = 2$

Intra-Cluster Rounding

High-Level Idea: Round within clusters to increase integrality from $\approx \frac{1}{\Delta}$ to $\approx \frac{1}{\Delta_C}$

Intra-Cluster Rounding

High-Level Idea: Round within clusters to increase integrality from $\approx \frac{1}{\Delta}$ to $\approx \frac{1}{\Delta_C}$

Partial Intra-Cluster Rounding Theorem [Ghaffari, G '23] **:**
Let C be a partition of diameter D. In O(D) rounds, we can compute a $\frac{1}{10^9 \Delta_C \log(n)}$- integral vector such that if we treat the entries as sampling probabilities, we remove a constant fraction of the edges, in expectation.

Intra-Cluster Rounding

High-Level Idea: Round within clusters to increase integrality from $\approx \frac{1}{\Delta}$ to $\approx \frac{1}{\Delta_C}$

Partial Intra-Cluster Rounding Theorem [Ghaffari, G '23] **:**
Let C be a partition of diameter D. In O(D) rounds, we can compute a $\frac{1}{10^9 \Delta_C \log(n)}$- integral vector such that if we treat the entries as sampling probabilities, we remove a constant fraction of the edges, in expectation.

- Each cluster rounds independently – hence O(D) rounds.
- A "good" local solution is guaranteed by a probabilistic method argument.

Intra-Cluster + Local Rounding

Intra-Cluster + Local Rounding Theorem:
Given a partition C into clusters of diameter D, one can compute an MIS in $\widetilde{\boldsymbol{O}}(\log(n)\,(D + \log^2(\Delta_C)))$ rounds.

Intra-Cluster + Local Rounding

Intra-Cluster + Local Rounding Theorem:
Given a partition C into clusters of diameter D, one can compute an MIS in $\tilde{\boldsymbol{O}}(\log(n)\,(D + \log^2(\Delta_C)))$ rounds.

Proof Sketch: Derandomize each round of Luby's algorithm by combining partial intra-cluster rounding with local rounding.

Intra-Cluster + Local Rounding

Intra-Cluster + Local Rounding Theorem:
Given a partition C into clusters of diameter D, one can compute an MIS in $\tilde{O}(\log(n)\,(D + \log^2(\Delta_C)))$ rounds.

Proof Sketch: Derandomize each round of Luby's algorithm by combining partial intra-cluster rounding with local rounding.

Key Question:

What's the right trade-off between the diameter D and the max cluster degree Δ_C?

Clustering with Small Cluster Degree

Randomized Low-Diameter Decomposition Miller, Peng, Xu [SPAA'13] **:**
In $O(D)$ rounds, the MPX algorithm computes a partition with diameter D and cluster degree $n^{O(1/D)}$ (for $D \leq \log^{0.99}(n)$).

Clustering with Small Cluster Degree

Randomized Low-Diameter Decomposition Miller, Peng, Xu [SPAA'13] **:**
In $O(D)$ rounds, the MPX algorithm computes a partition with diameter D and cluster degree $n^{O(1/D)}$ (for $D \leq \log^{0.99}(n)$).

Given such a partition, we can compute an MIS in $\tilde{O}(\log(n)\,(D + \frac{\log^2(n))}{D^2}))$.

Clustering with Small Cluster Degree

Randomized Low-Diameter Decomposition Miller, Peng, Xu [SPAA'13] **:**
In $O(D)$ rounds, the MPX algorithm computes a partition with diameter D and cluster degree $n^{O(1/D)}$ (for $D \leq \log^{0.99}(n)$)).

Given such a partition, we can compute an MIS in $\tilde{O}(\log(n)\,(D + \frac{\log^2(n))}{D^2}))$.

Setting $D = \log^{2/3}(n)))$, we get the desired round complexity of $\tilde{O}(\log^{5/3}(n))$.

Clustering with Small Cluster Degree

Randomized Low-Diameter Decomposition Miller, Peng, Xu [SPAA'13] **:**
In $O(D)$ rounds, the MPX algorithm computes a partition with diameter D and cluster degree $n^{O(1/D)}$ (for $D \leq \log^{0.99}(n)$).

MPX Algorithm:

Clustering with Small Cluster Degree

Randomized Low-Diameter Decomposition Miller, Peng, Xu [SPAA'13] **:**
In $O(D)$ rounds, the MPX algorithm computes a partition with diameter D and cluster degree $n^{O(1/D)}$ (for $D \leq \log^{0.99}(n)$).

MPX Algorithm:

1.) Each node v gets a random head start $h_v \sim Geo\left(1 - \frac{1}{n^{O\left(\frac{1}{D}\right)}}\right)$

$$\Rightarrow \Pr[h_v \geq i] = \frac{1}{n^{O((i-1)/D)}}$$

Clustering with Small Cluster Degree

Randomized Low-Diameter Decomposition Miller, Peng, Xu [SPAA'13] **:**
In $O(D)$ rounds, the MPX algorithm computes a partition with diameter D and cluster degree $n^{O(1/D)}$ (for $D \leq \log^{0.99}(n)$).

MPX Algorithm:

1.) Each node *v* gets a random head start $h_v \sim Geo\left(1 - \frac{1}{n^{O\left(\frac{1}{D}\right)}}\right)$

$$\Rightarrow \Pr[h_v \geq i] = \frac{1}{n^{O((i-1)/D)}}$$

2.) Cluster each node *u* into the cluster of the node v minimizing dist(v,u)-h_v (ties to higher ID).

Clustering with Small Cluster Degree

Randomized Low-Diameter Decomposition Miller, Peng, Xu [SPAA'13] **:**
In $O(D)$ rounds, the MPX algorithm computes a partition with diameter D and cluster degree $n^{O(1/D)}$ (for $D \leq \log^{0.99}(n)$).

MPX Algorithm:

1.) Each node *v* gets a random head start $h_v \sim Geo\left(1 - \frac{1}{n^{O\left(\frac{1}{D}\right)}}\right)$

$$\Rightarrow \Pr[h_v \geq i] = \frac{1}{n^{O((i-1)/D)}}$$

2.) Cluster each node *u* into the cluster of the node v minimizing dist(v,u)-h_v (ties to higher ID).

<u>Observation</u>: Each cluster has diameter $O(D)$, w.h.p.

Technique #3: Derandomizing the randomized MPX algorithm

LDD with Small Cluster Degree

Derandomized MPX Decomposition Ghaffari, G [STOC'23] **:**
In $\widetilde{\boldsymbol{O}}(\log(n)\, D^2)$ rounds, one can compute a partition with diameter D and maximum cluster degree $n^{O(1/D)}$ (assuming $D \leq \log^{0.99}(n)$).

LDD with Small Cluster Degree

Derandomized MPX Decomposition Ghaffari, G [STOC'23] **:**
In $\widetilde{O}(\log(n)\, D^2)$ rounds, one can compute a partition with diameter D and maximum cluster degree $n^{O(1/D)}$ (assuming $D \leq \log^{0.99}(n)$).

High-Level Idea:

- Each node has D random coin tosses that define its head start.
- Derandomize coin #1 of all nodes, then coin #2, and so on -- for D rounds.
- Each derandomization step reduces to computing a **hitting set**.
- Each round takes $\widetilde{O}(\log(n) D)$ rounds.

LDD with Small Cluster Degree

Derandomized MPX Decomposition Ghaffari, G [STOC'23] **:**
In $\widetilde{\boldsymbol{O}}(\log(n)\, D^2)$ rounds, one can compute a partition with diameter D and maximum cluster degree $n^{O(1/D)}$ (assuming $D \leq \log^{0.99}(n)$).

High-Level Idea:

- Each node has D random coin tosses that define its head start.
- Derandomize coin #1 of all nodes, then coin #2, and so on -- for D rounds.
- Each derandomization step reduces to computing a **hitting set**.
- Each round takes $\widetilde{\boldsymbol{O}}(\log(n) D)$ rounds.

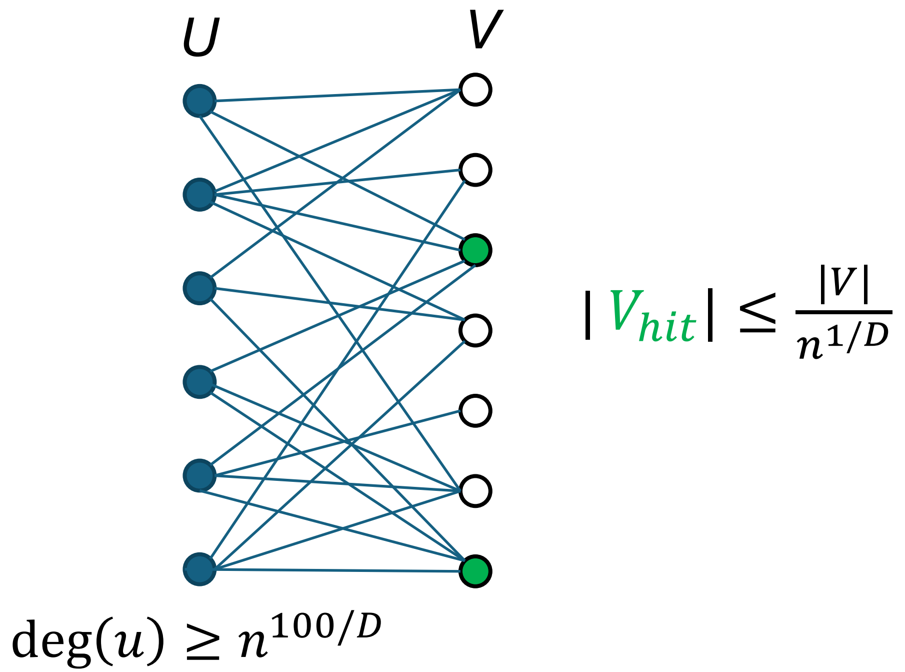
Corollary Ghaffari, G [STOC'23] **:**
An MIS can be computed in $\tilde{O}(\log^2 n)$ rounds.

Technique #4: Deterministic Hitting Set

Deterministic Hitting Set

Deterministic Hitting Set Ghaffari, G [STOC'23] **:**
Consider a bipartite graph with bipartition $U \sqcup V$, where each $u \in U$ has degree $\deg(u) \geq n^{100/D}$.
In $\widetilde{\boldsymbol{O}}(\log(n))$ rounds, one can compute a subset $V_{hit} \subseteq V$ such that:

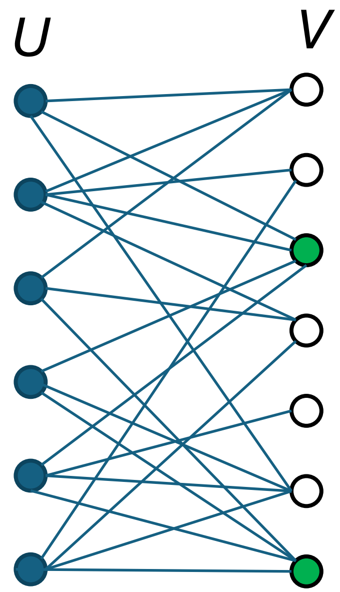
1. $|V_{hit}| \leq \frac{|V|}{n^{1/D}}$
2. $N(u) \cap V_{hit} \neq \emptyset$ for every $u \in U$.



Deterministic Hitting Set

Deterministic Hitting Set Ghaffari, G [STOC'23] **:**
Consider a bipartite graph with bipartition $U \sqcup V$, where each $u \in U$ has degree $\deg(u) \geq n^{100/D}$.
In $\widetilde{\boldsymbol{O}}(\log(n))$ rounds, one can compute a subset $V_{hit} \subseteq V$ such that:

1. $|V_{hit}| \leq \frac{|V|}{n^{1/D}}$
2. $N(u) \cap V_{hit} \neq \emptyset$ for every $u \in U$.



$\deg(u) \geq n^{100/D}$

$|V_{hit}| \leq \frac{|V|}{n^{1/D}}$

Simple 0-round randomized algorithm!

Open Problems

Thanks!

- Deterministic MIS (or Maximal Matching, $(\Delta + 1)$-coloring) in o($\log^{5/3} n$) or o($log\, n\, log^2\, \Delta$) rounds

- Derandomization Gap in Distributed Graph Algorithms (for locally checkable problems)
  - Det/Rand =$\tilde{O}(\log^2 n)$ G., Grunau FOCS'24
  - Det/Rand = $\tilde{\Omega}(\log n)$ Brandt et al. STOC'16; Chang, Pettie, Kopelowitz FOCS'17; G., Su SODA'17
  - ? Close the gap?

- Use Local Rounding in other models
  - Parallel derandomization, towards work-efficiency G., Grunau '25
  - ? Dynamic, ...?