Distributed Algorithms for Local Potential Problems

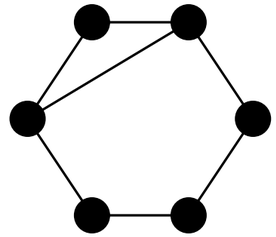
Francesco d'Amore

Joint work with A. Balliu, T. Boudier, F. Kuhn, D. Olivetti, G. Schmid, and J. Suomela
To appear at PODC 2026

Workshop on Foundations of Distributed and Parallel Graph Algorithms (Venice 2026)
18 May 2026

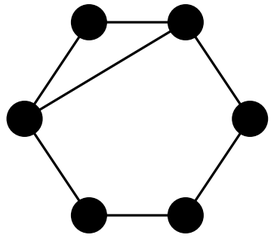
Locally Optimal Cut (LOC)

Input: - graph $G = (V, E)$
- two colors red and green



Locally Optimal Cut (LOC)

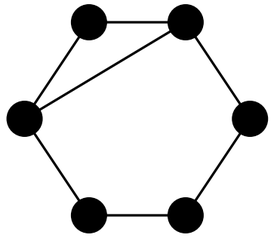
Input: - graph $G = (V, E)$
- two colors red and green



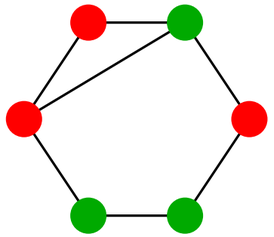
Output: - a (not necessarily proper) 2-coloring $c: V \to \{\text{red}, \text{green}\}$ of G
- for each $v \in V$, at least $\geq \deg(v)/2$ *neighbors of different color* (w.r.t. v) ⟶ *locally checkable*

Locally Optimal Cut (LOC)

Input: - graph $G = (V, E)$
- two colors red and green



Output: - a (not necessarily proper) 2-coloring $c: V \to \{\text{red}, \text{green}\}$ of G
- for each $v \in V$, at least $\geq \deg(v)/2$ *neighbors of different color* (w.r.t. v) ⟶ *locally checkable*

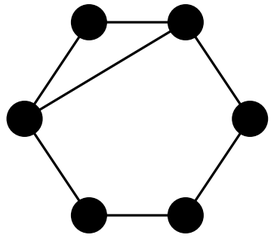


valid solution

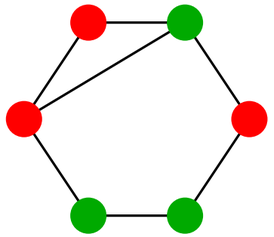
2-apx of MAX-CUT (locally optimal)

Locally Optimal Cut (LOC)

Input: - graph $G = (V, E)$
- two colors red and green

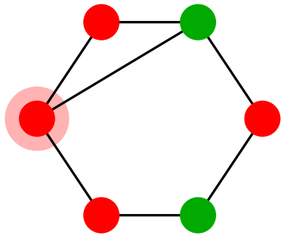


Output: - a (not necessarily proper) 2-coloring $c: V \to \{\text{red}, \text{green}\}$ of G
- for each $v \in V$, at least $\geq \deg(v)/2$ *neighbors of different color* (w.r.t. v) ⟶ *locally checkable*



valid solution

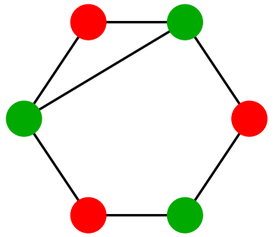
2-apx of MAX-CUT (locally optimal)



invalid solution

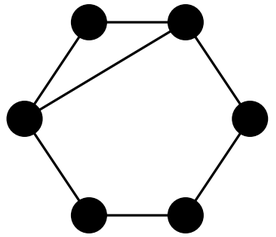
fixing procedure ⟶

flip color of invalid node

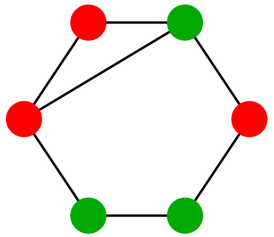


Locally Optimal Cut (LOC)

Input: - graph $G = (V, E)$
- two colors red and green

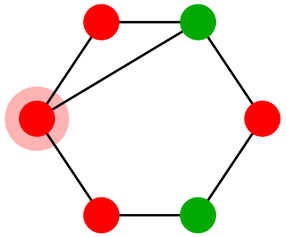


Output: - a (not necessarily proper) 2-coloring $c: V \to \{\text{red}, \text{green}\}$ of G
- for each $v \in V$, at least $\geq \deg(v)/2$ *neighbors of different color* (w.r.t. v) ⟶ *locally checkable*



valid solution

2-apx of MAX-CUT (locally optimal)

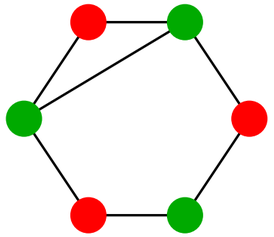


invalid solution

fixing procedure

⟶

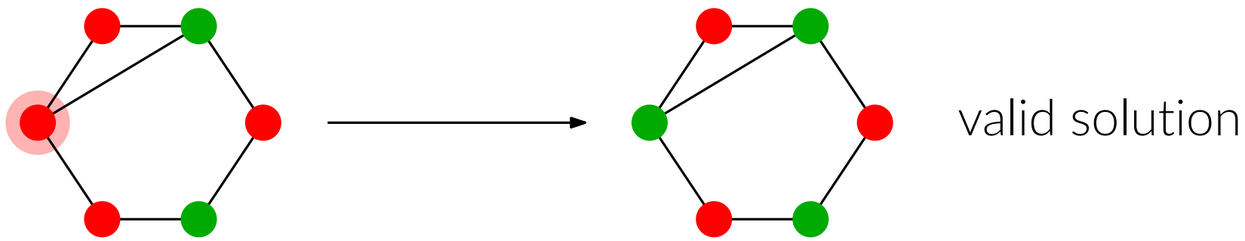
flip color of invalid node



Question: can we *always* solve the problem?

LOC: fixing procedure

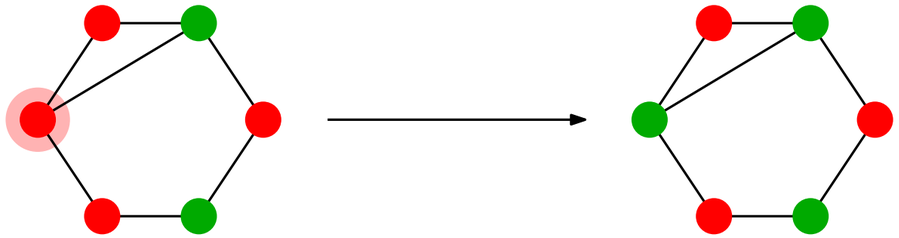
Fixing procedure: *pick an arbitrary invalid node and flip its color*



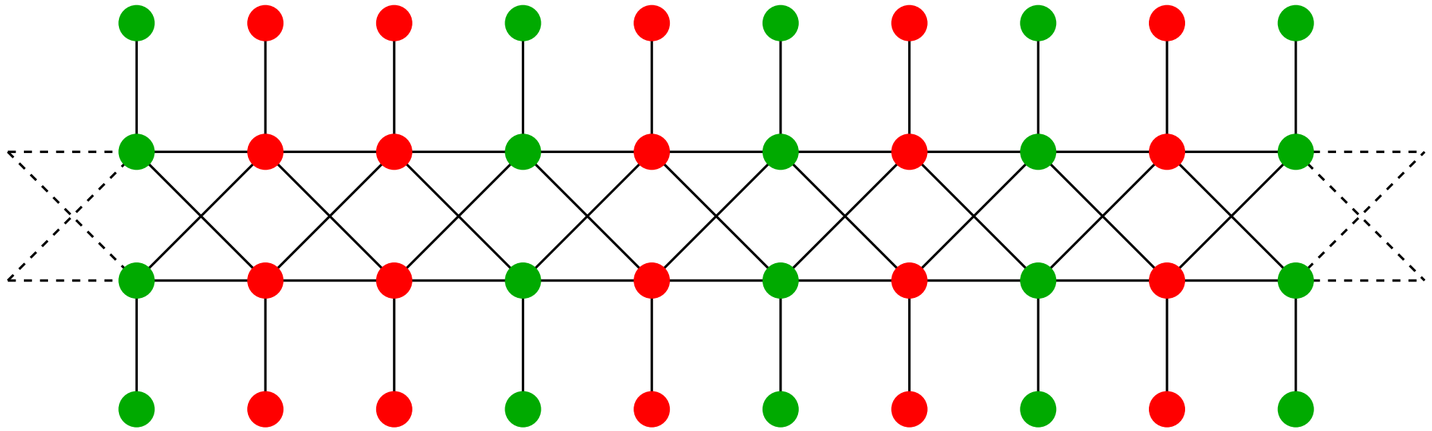
valid solution

LOC: fixing procedure

Fixing procedure: *pick an arbitrary invalid node and flip its color*

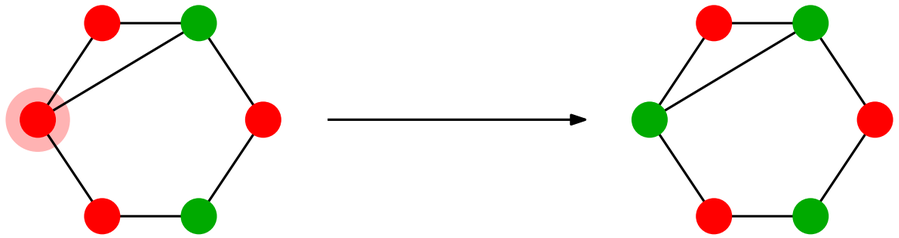


valid solution

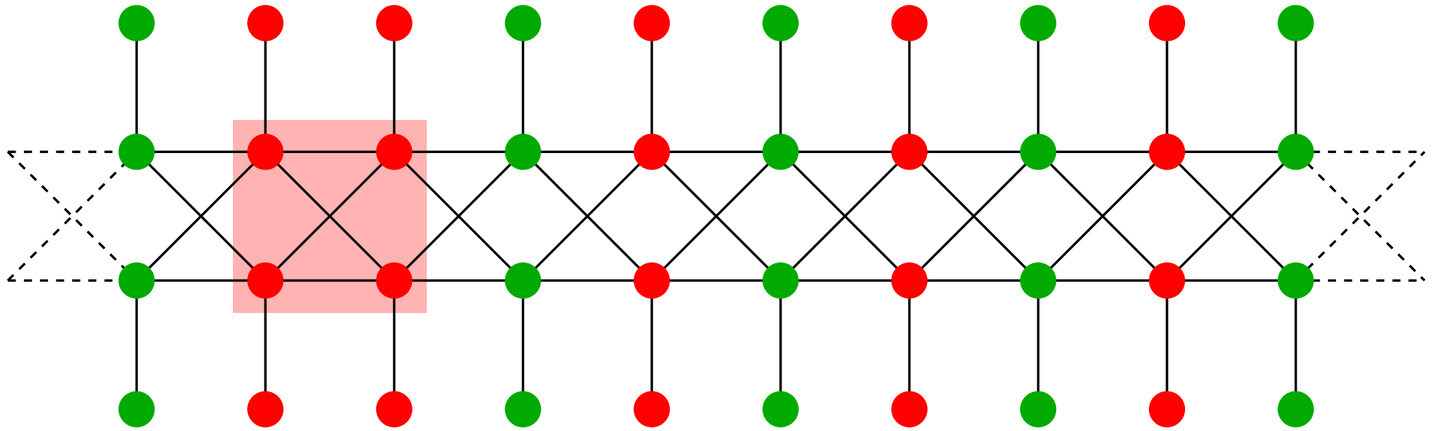


LOC: fixing procedure

Fixing procedure: *pick an arbitrary invalid node and flip its color*

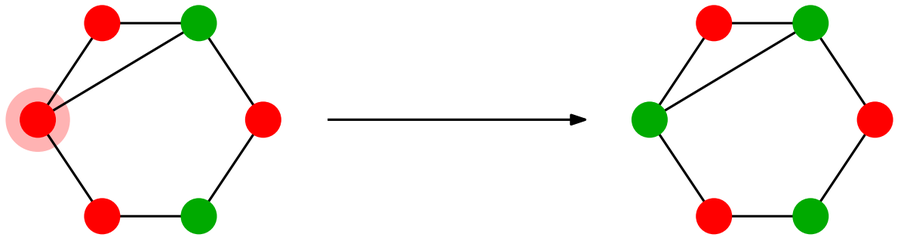


valid solution

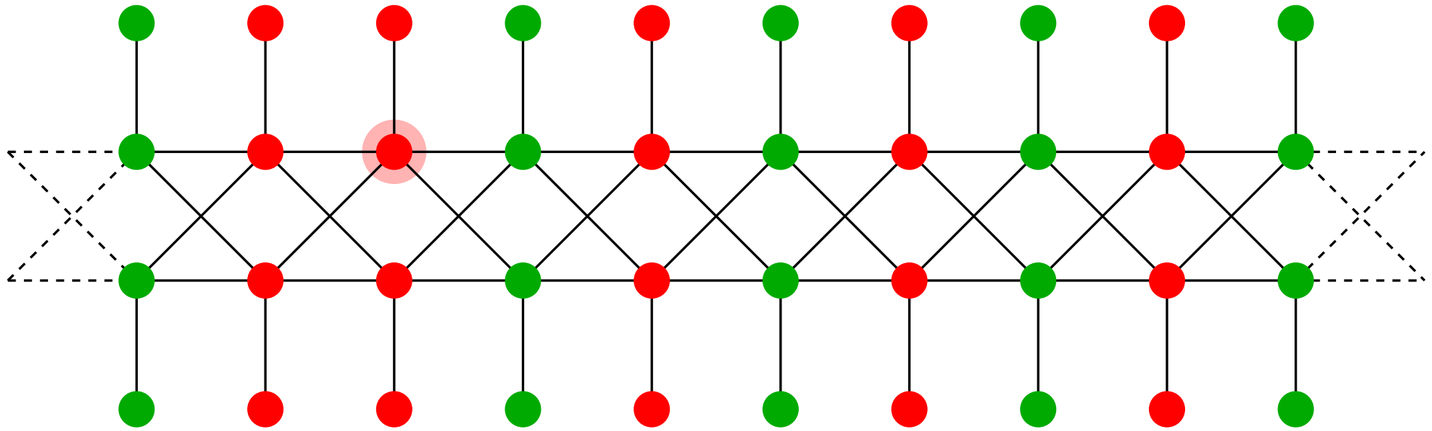


LOC: fixing procedure

Fixing procedure: *pick an arbitrary invalid node and flip its color*

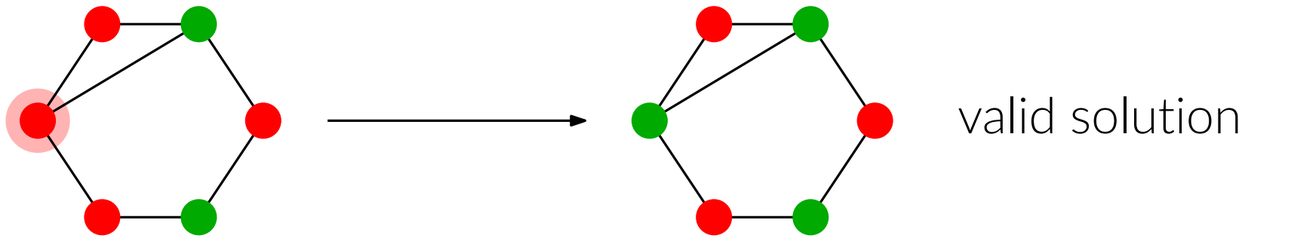


valid solution

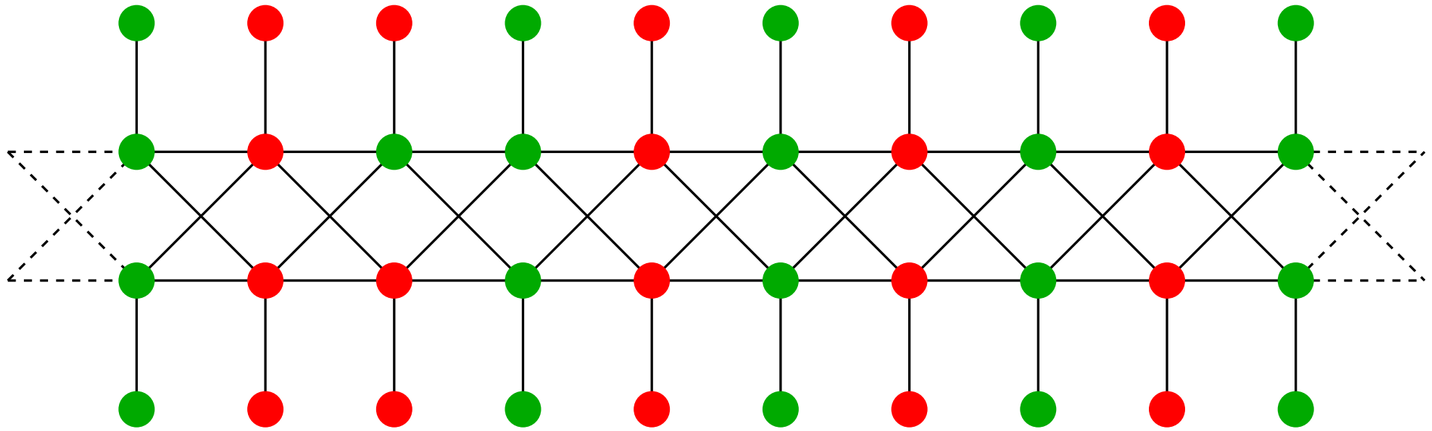


LOC: fixing procedure

Fixing procedure: *pick an arbitrary invalid node and flip its color*



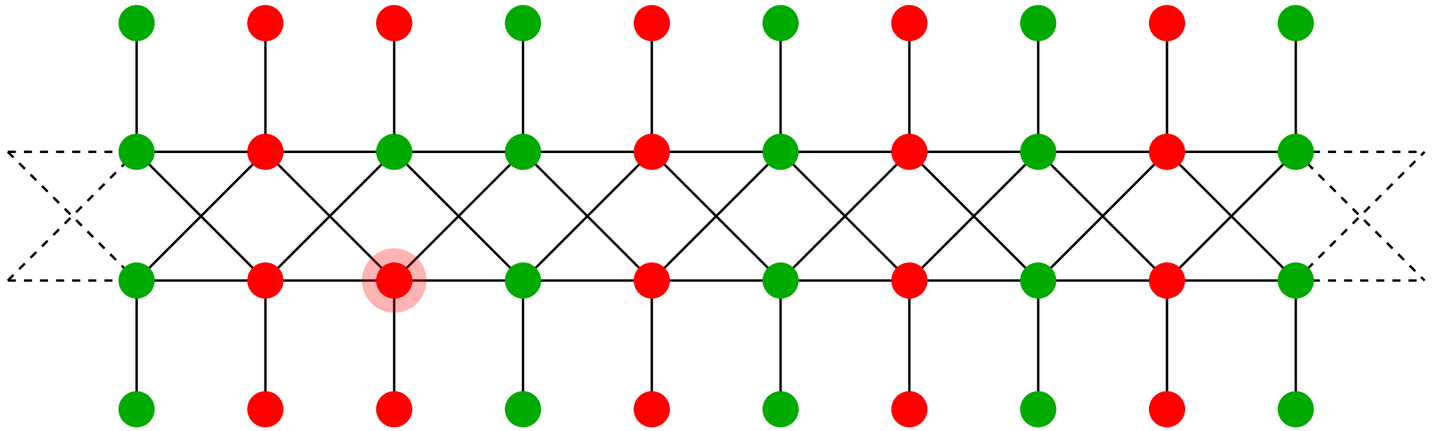
valid solution



LOC: fixing procedure

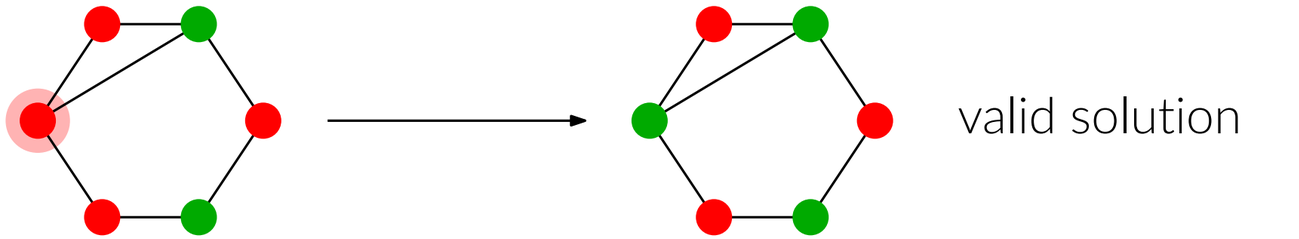
Fixing procedure: *pick an arbitrary invalid node and flip its color*

valid solution

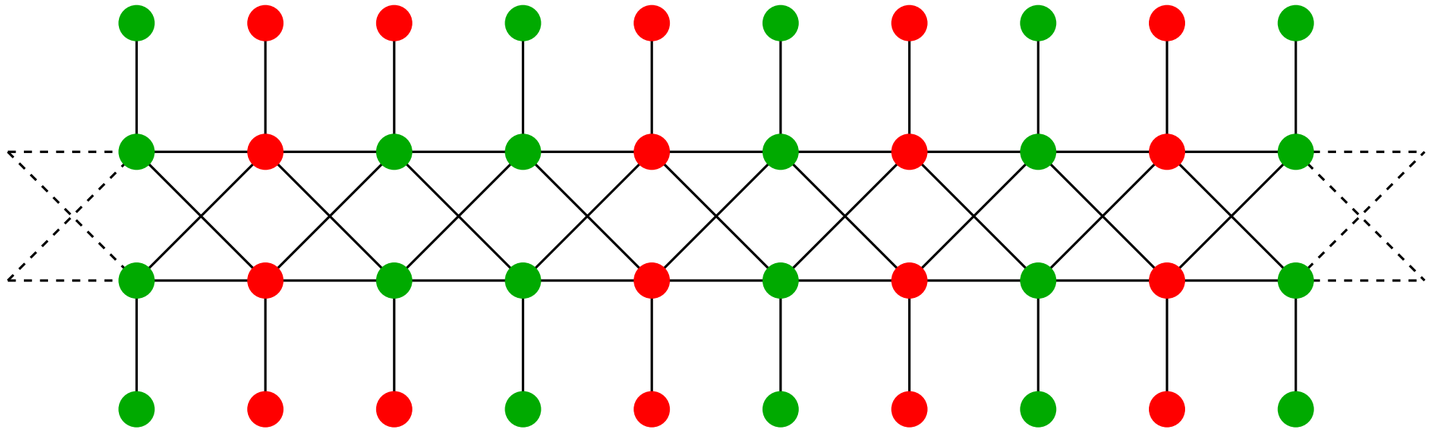


LOC: fixing procedure

Fixing procedure: *pick an arbitrary invalid node and flip its color*

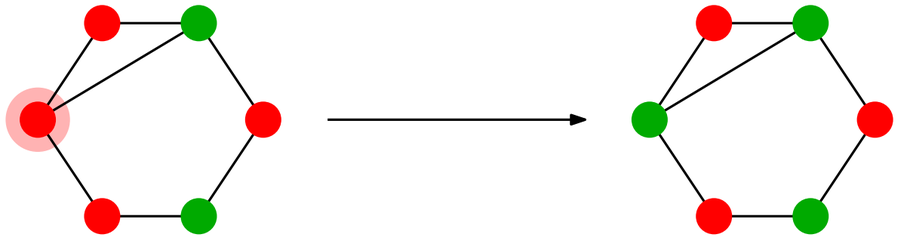


valid solution

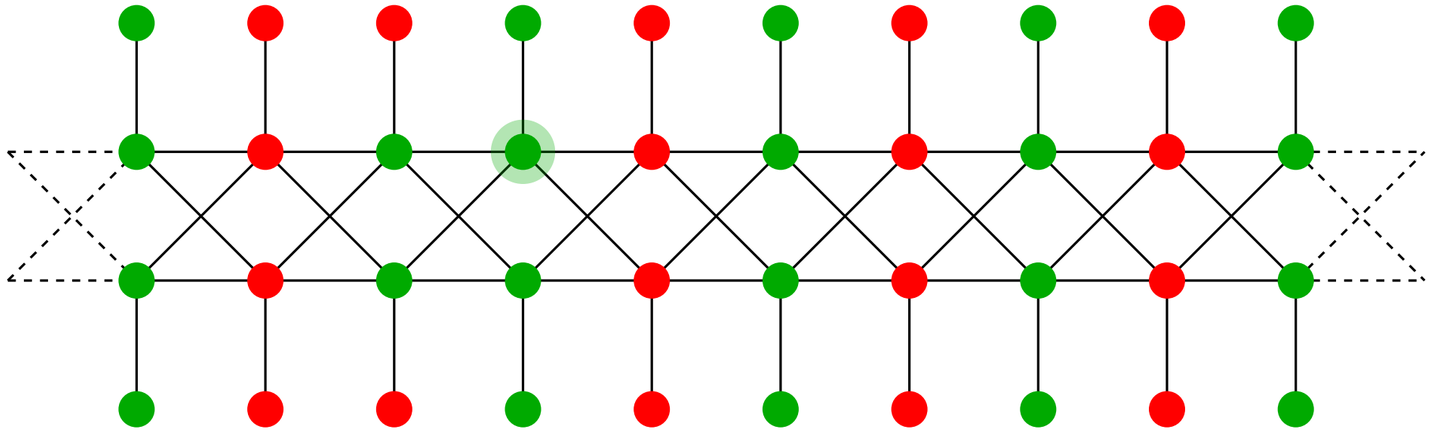


LOC: fixing procedure

Fixing procedure: *pick an arbitrary invalid node and flip its color*

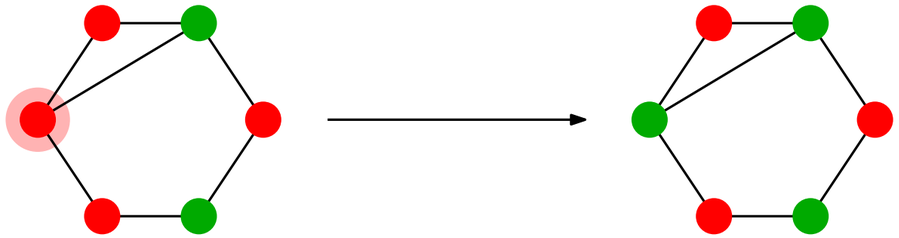


valid solution

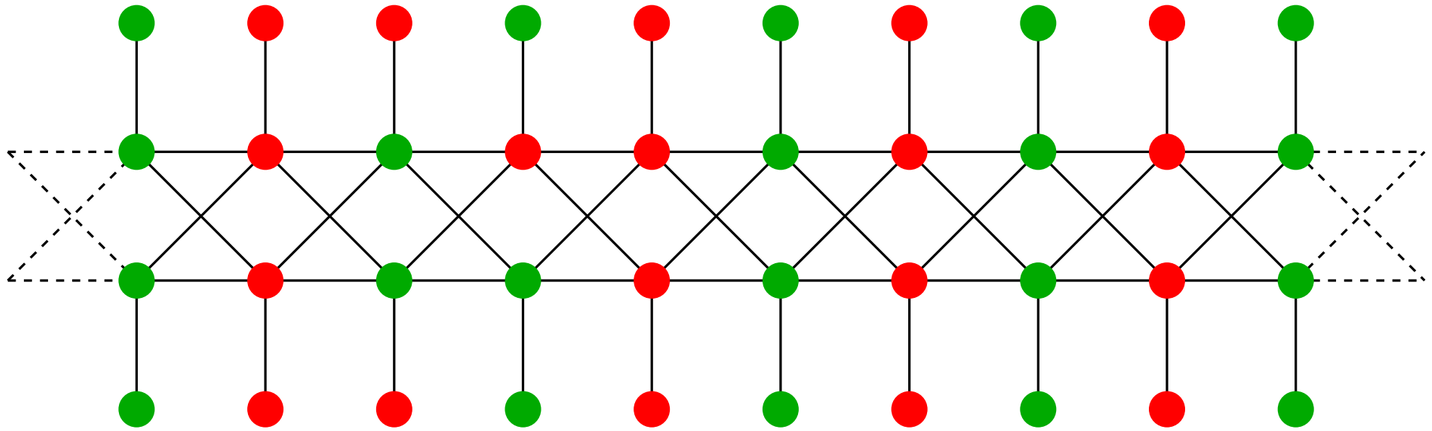


LOC: fixing procedure

Fixing procedure: *pick an arbitrary invalid node and flip its color*

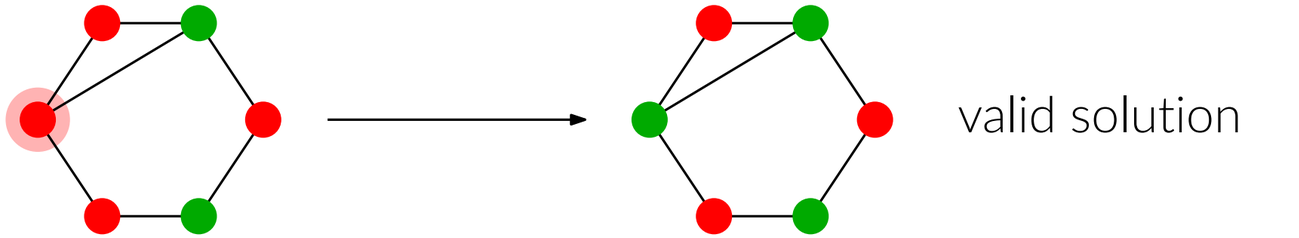


valid solution

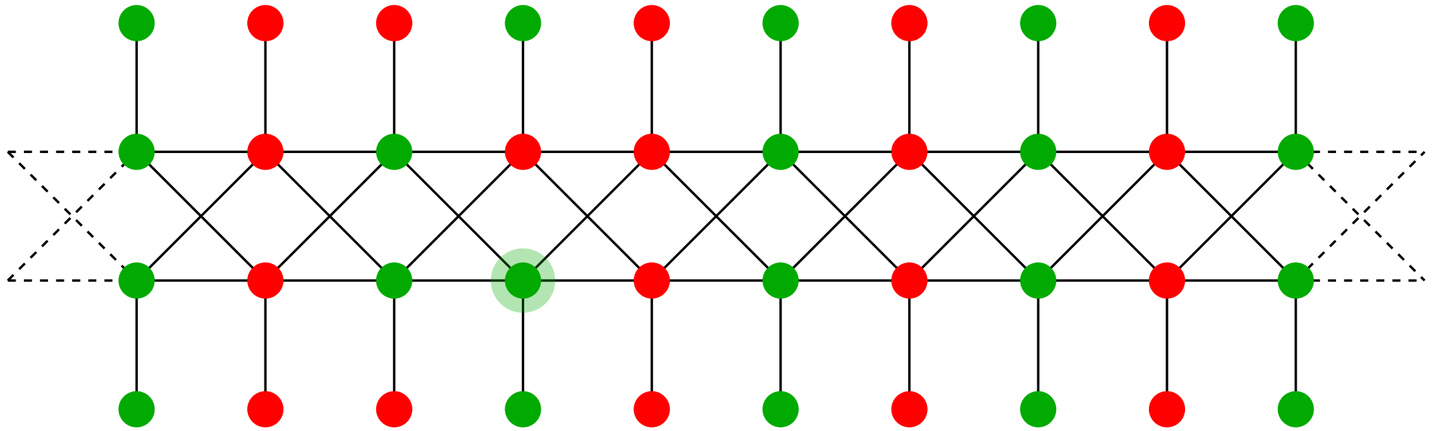


LOC: fixing procedure

Fixing procedure: *pick an arbitrary invalid node and flip its color*

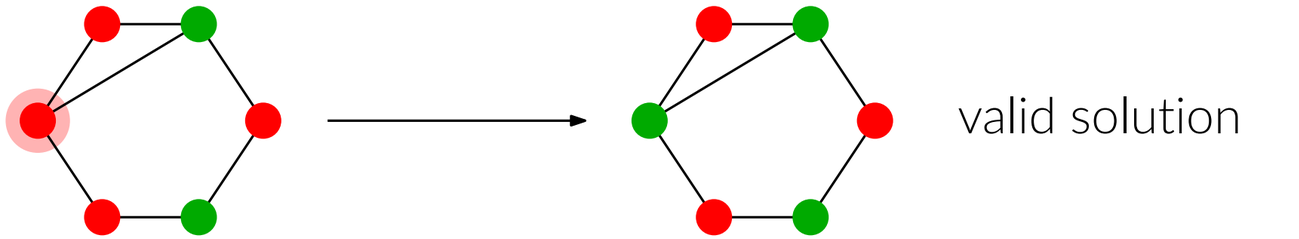


valid solution

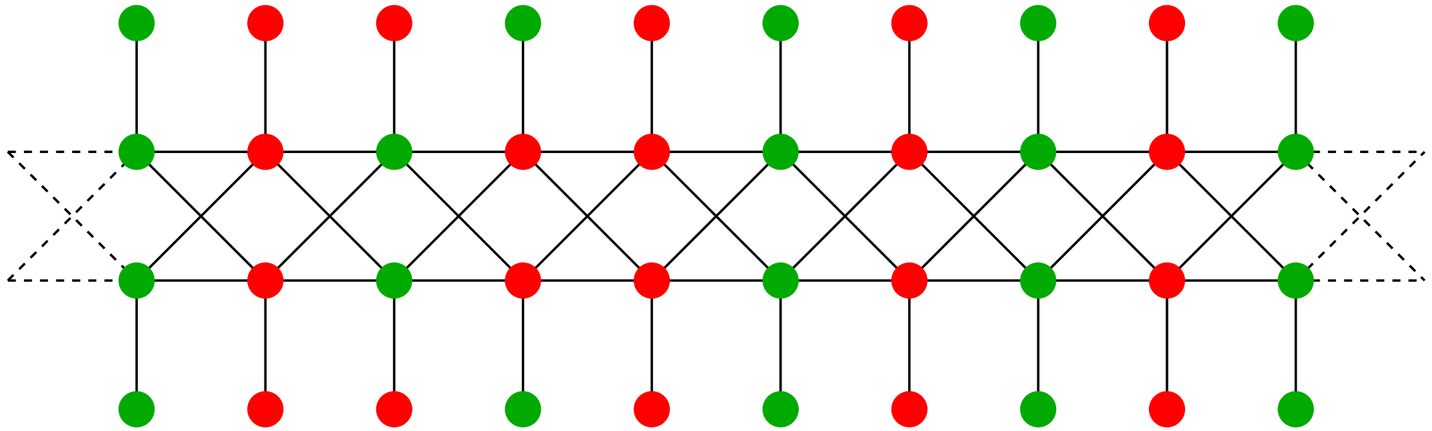


LOC: fixing procedure

Fixing procedure: *pick an arbitrary invalid node and flip its color*

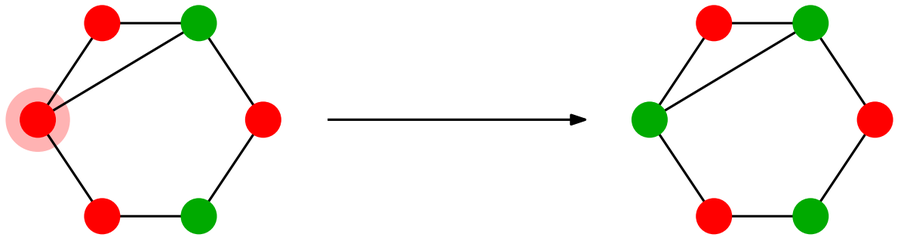


valid solution

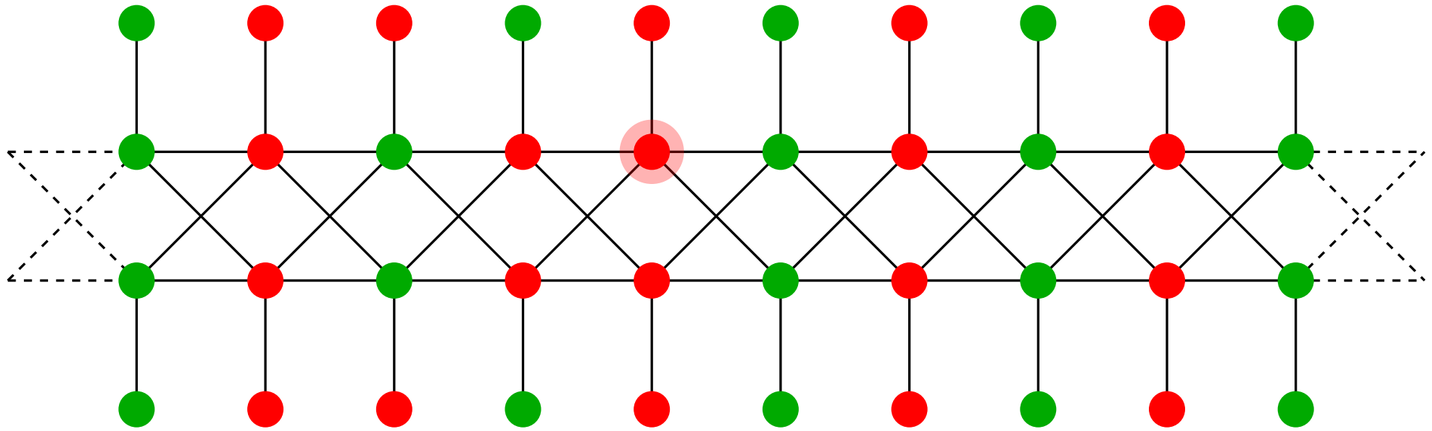


LOC: fixing procedure

Fixing procedure: *pick an arbitrary invalid node and flip its color*

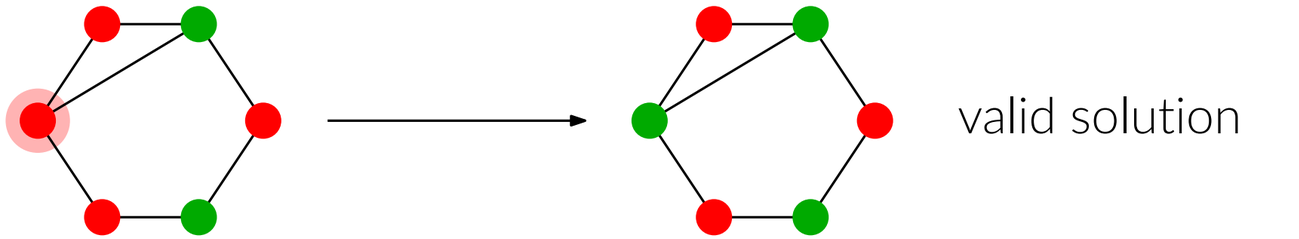


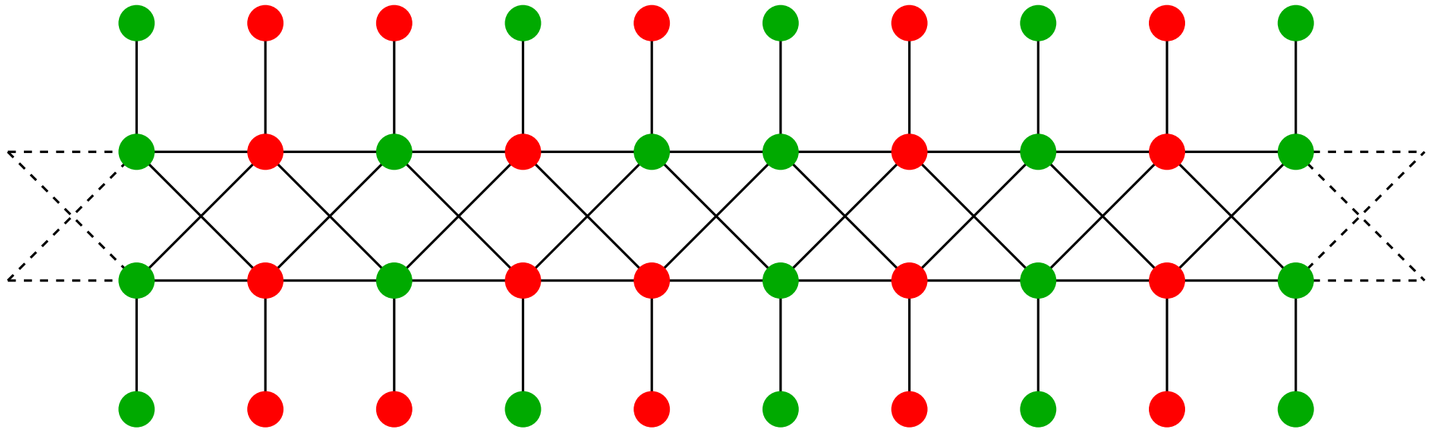
valid solution



LOC: fixing procedure

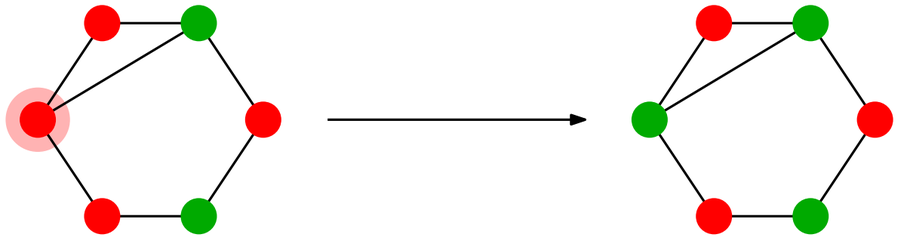
Fixing procedure: *pick an arbitrary invalid node and flip its color*



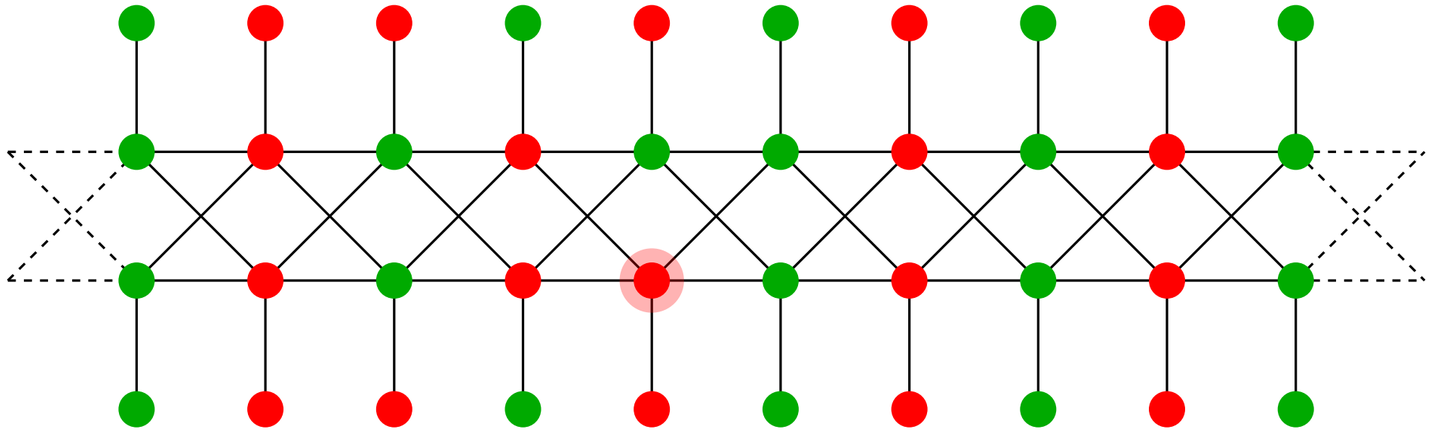


LOC: fixing procedure

Fixing procedure: *pick an arbitrary invalid node and flip its color*

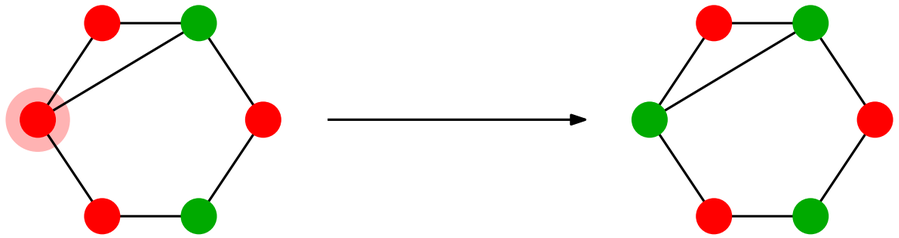


valid solution

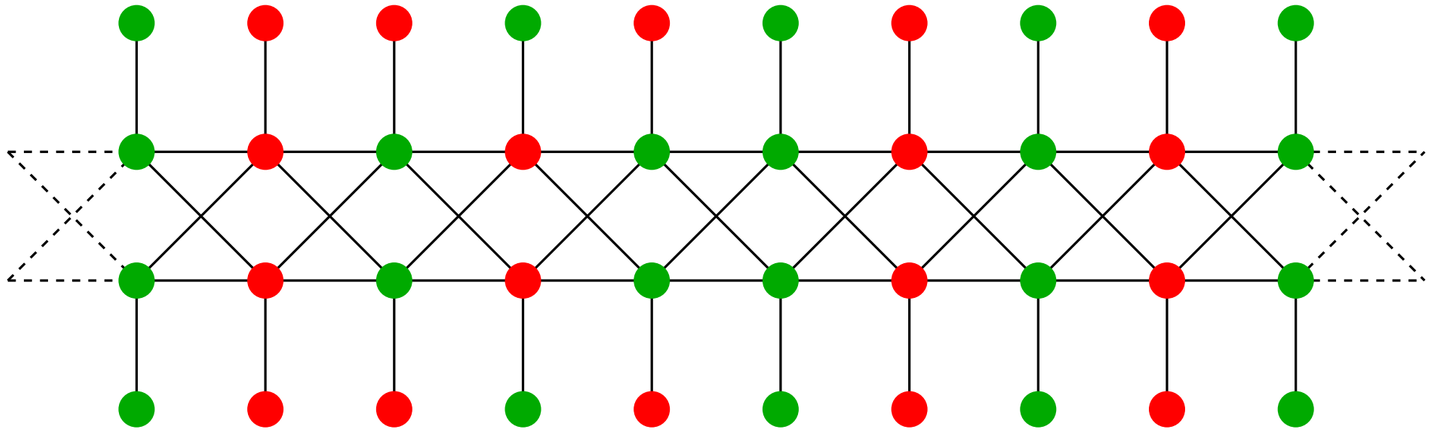


LOC: fixing procedure

Fixing procedure: *pick an arbitrary invalid node and flip its color*

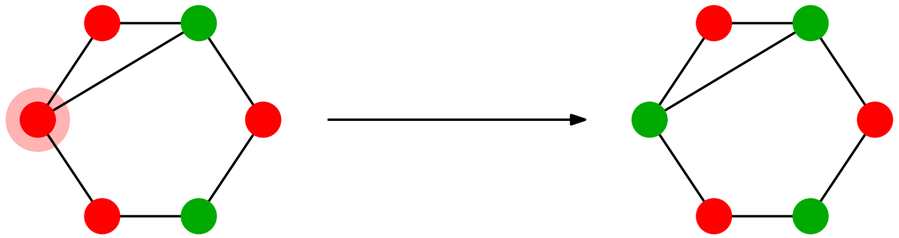


valid solution

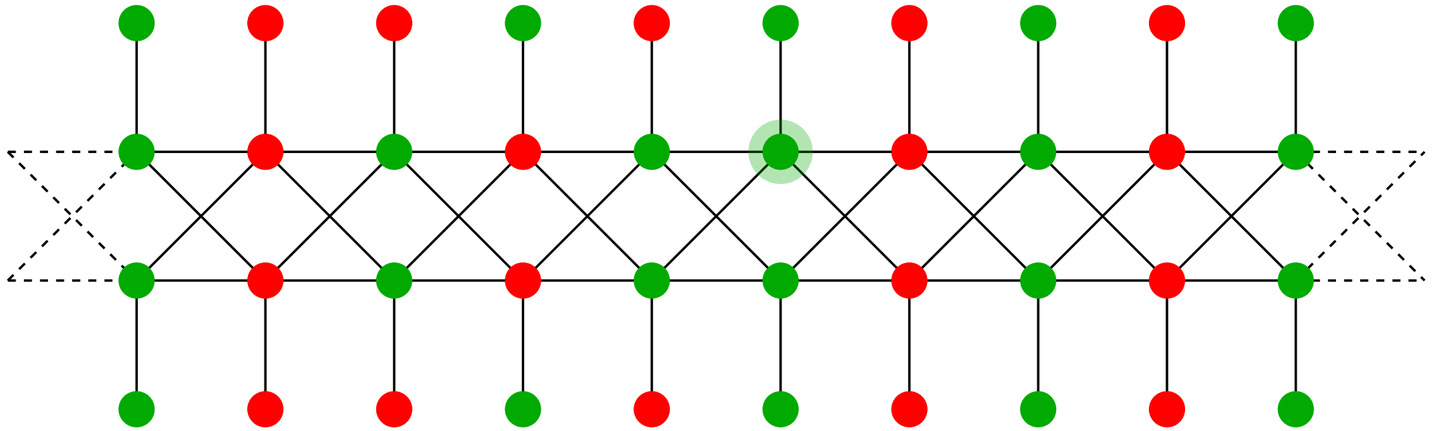


LOC: fixing procedure

Fixing procedure: *pick an arbitrary invalid node and flip its color*

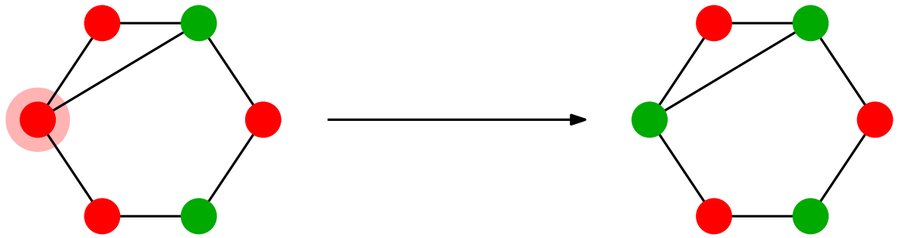


valid solution

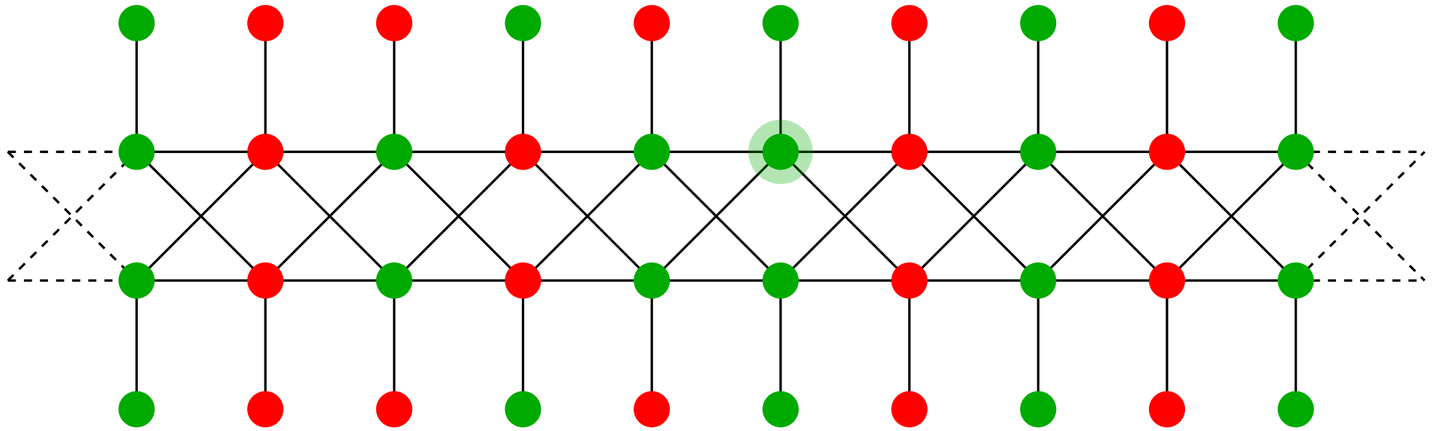


LOC: fixing procedure

Fixing procedure: *pick an arbitrary invalid node and flip its color*



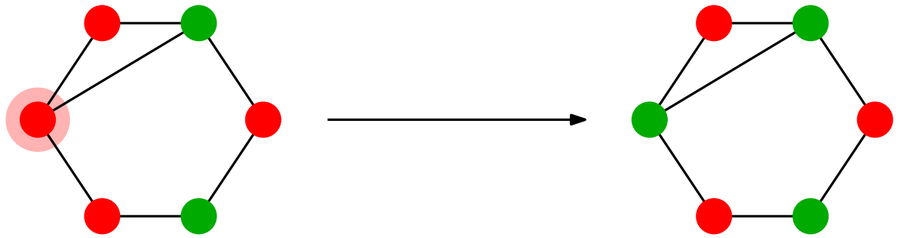
valid solution



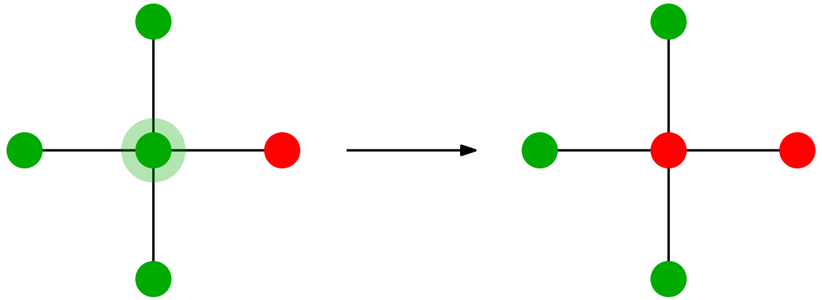
- **Convergence?**

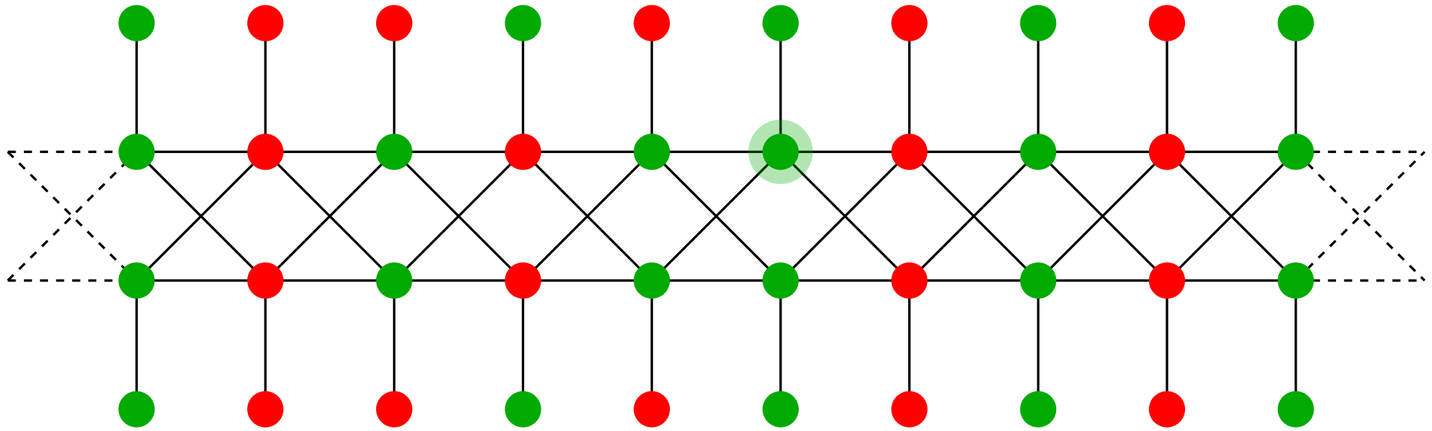
LOC: fixing procedure

Fixing procedure: *pick an arbitrary invalid node and flip its color*



valid solution



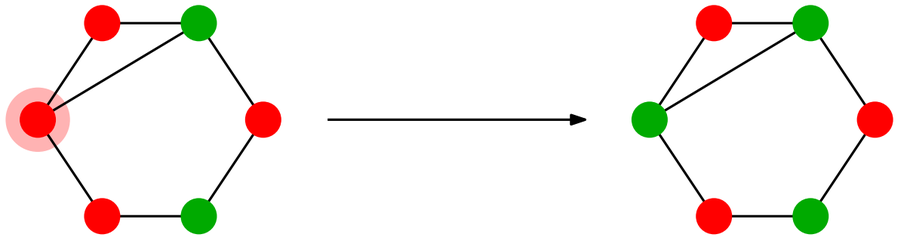


- **Convergence?** Potential function!

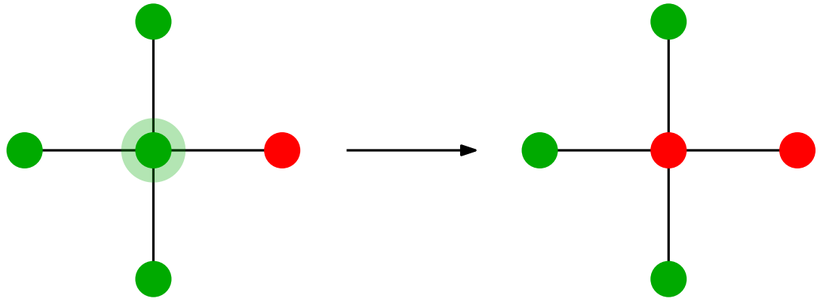
  - # of monochromatic edges decreases

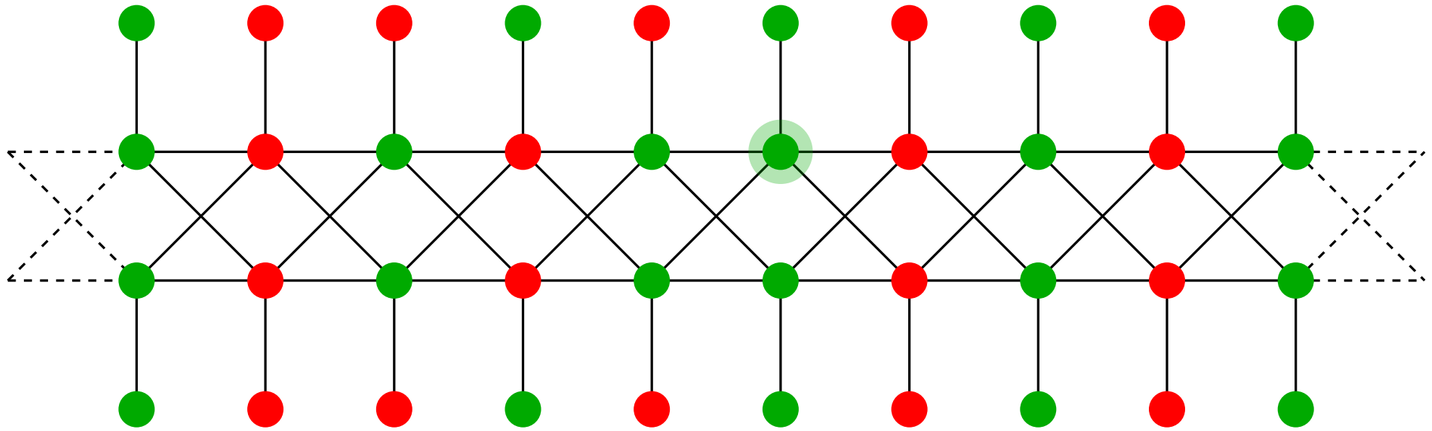
LOC: fixing procedure

Fixing procedure: *pick an arbitrary invalid node and flip its color*



valid solution

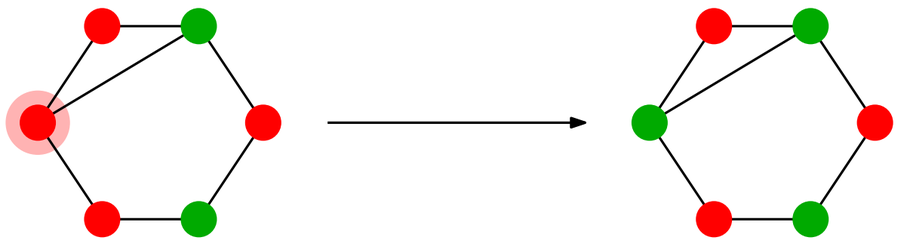




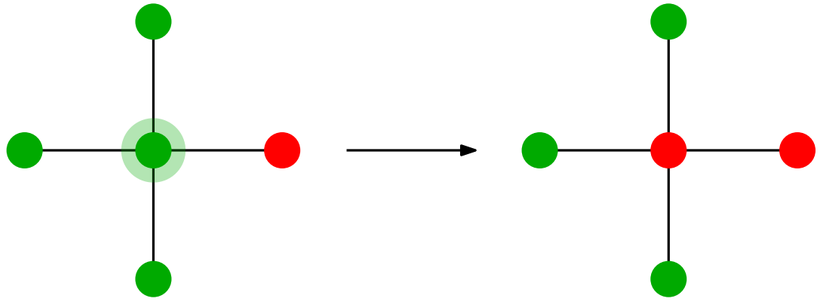
- **Convergence?** Potential function!
  - # of monochromatic edges decreases
- Possibly, very long chains…
  - at most $O(|E|)$ flips

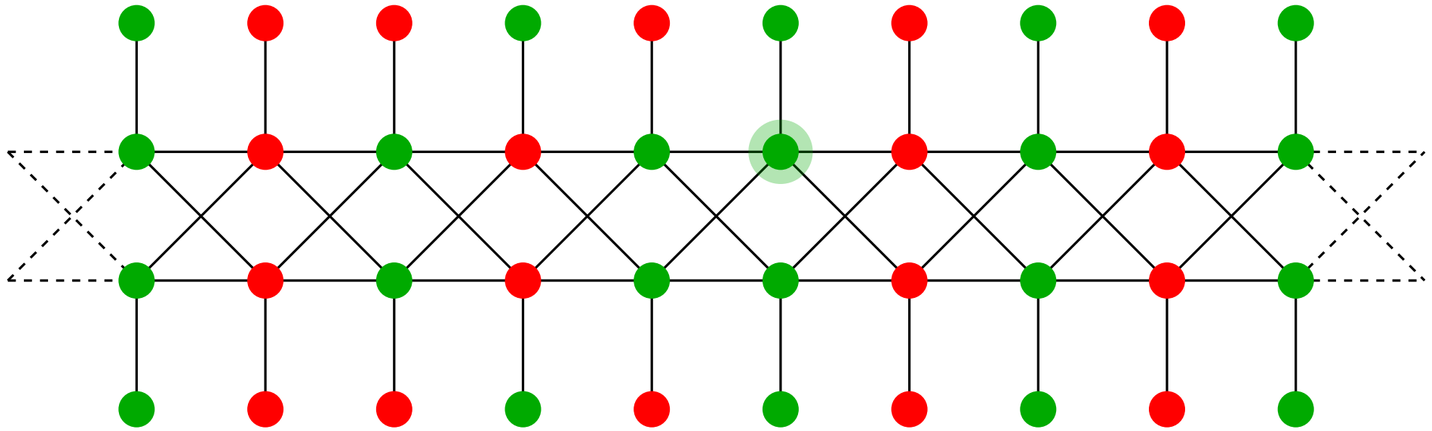
LOC: fixing procedure

Fixing procedure: *pick an arbitrary invalid node and flip its color*



valid solution



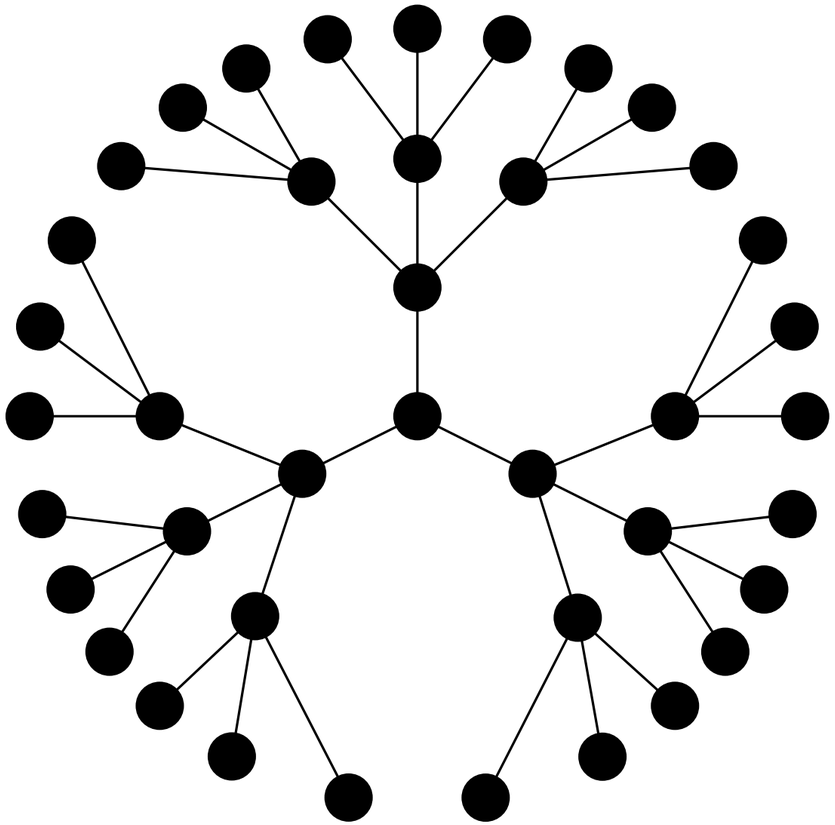


- **Convergence?** Potential function!
  - # of monochromatic edges decreases
- Possibly, very long chains…
  - at most $O(|E|)$ flips
- With distributed algorithms?

The LOCAL model

[Linial FOCS ’87 & SICOMP ’92]

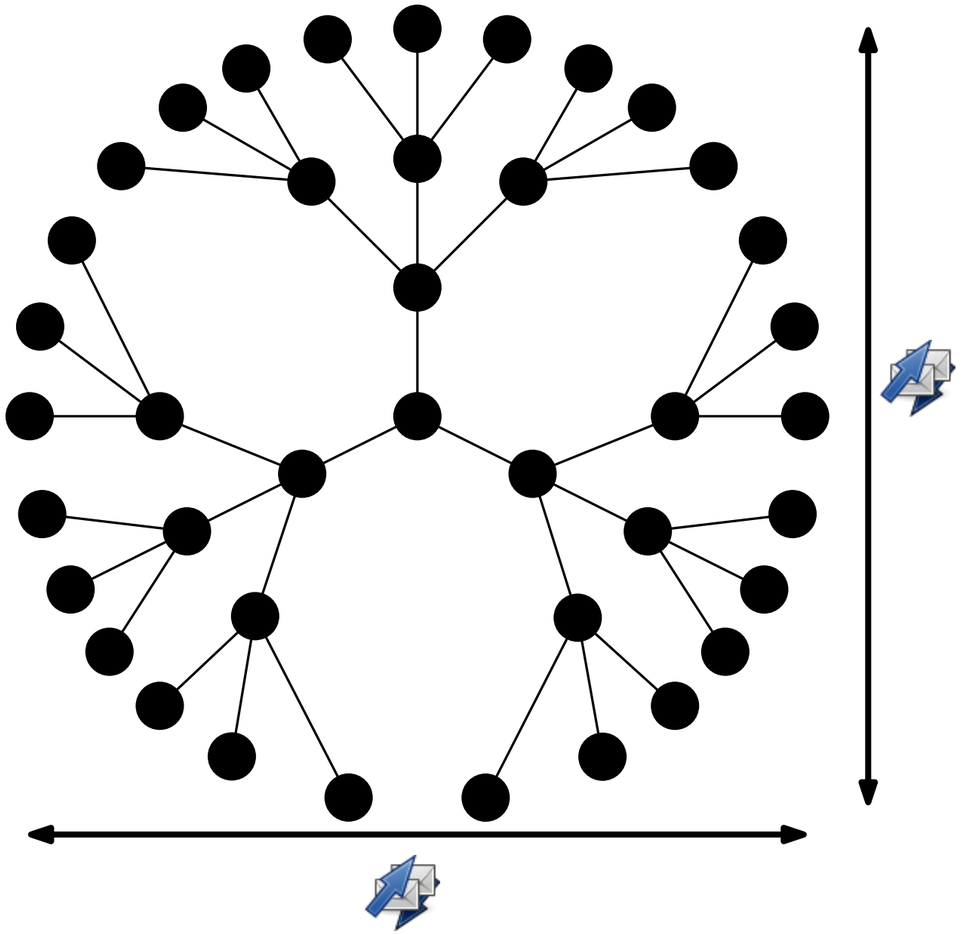
- **Distributed network** of n processors/nodes
  - graph $G = (V, E)$ with $|V| = n$
  - E: communication links
  - each node in V runs the same algorithm



The LOCAL model

[Linial FOCS '87 & SICOMP '92]

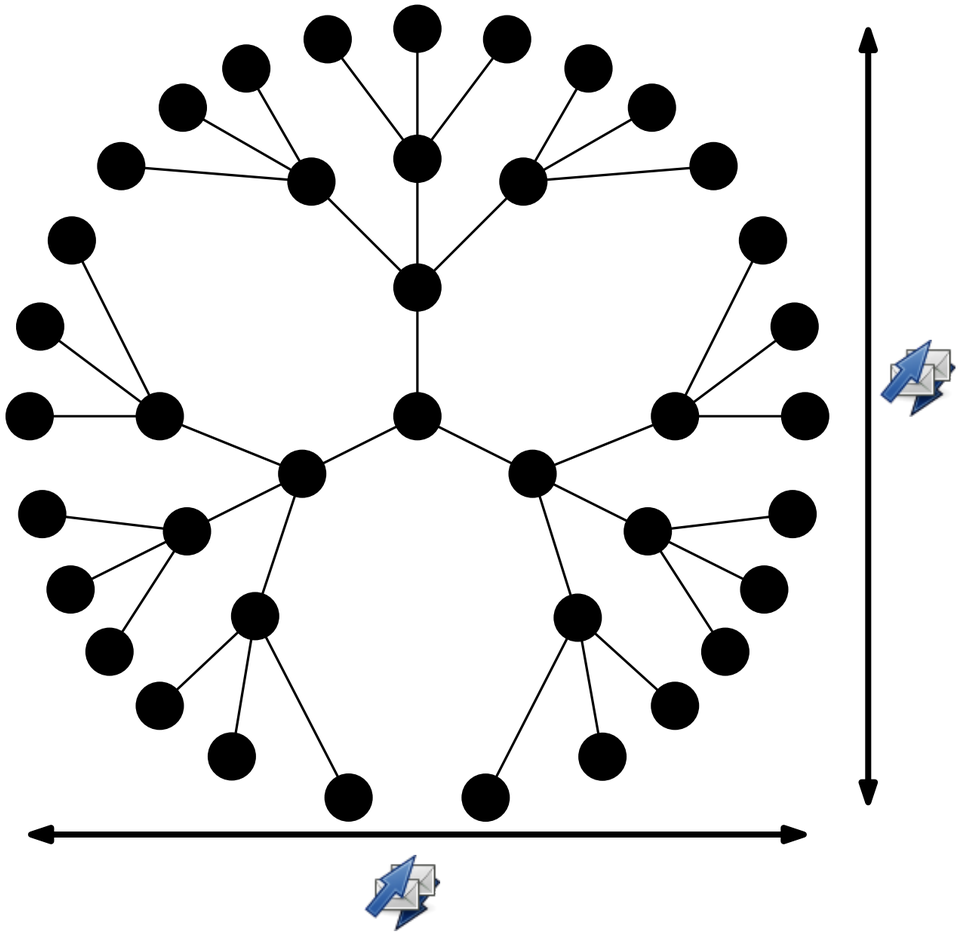
- **Distributed network** of n processors/nodes
  - graph $G = (V, E)$ with $|V| = n$
  - E: communication links
  - each node in V runs the same algorithm
- **Time is synchronous**: nodes alternate
  - arbitrary local computation & update of state variables
  - sending of messages to all neighbors
    * no bandwidth constraints



The LOCAL model

[Linial FOCS '87 & SICOMP '92]

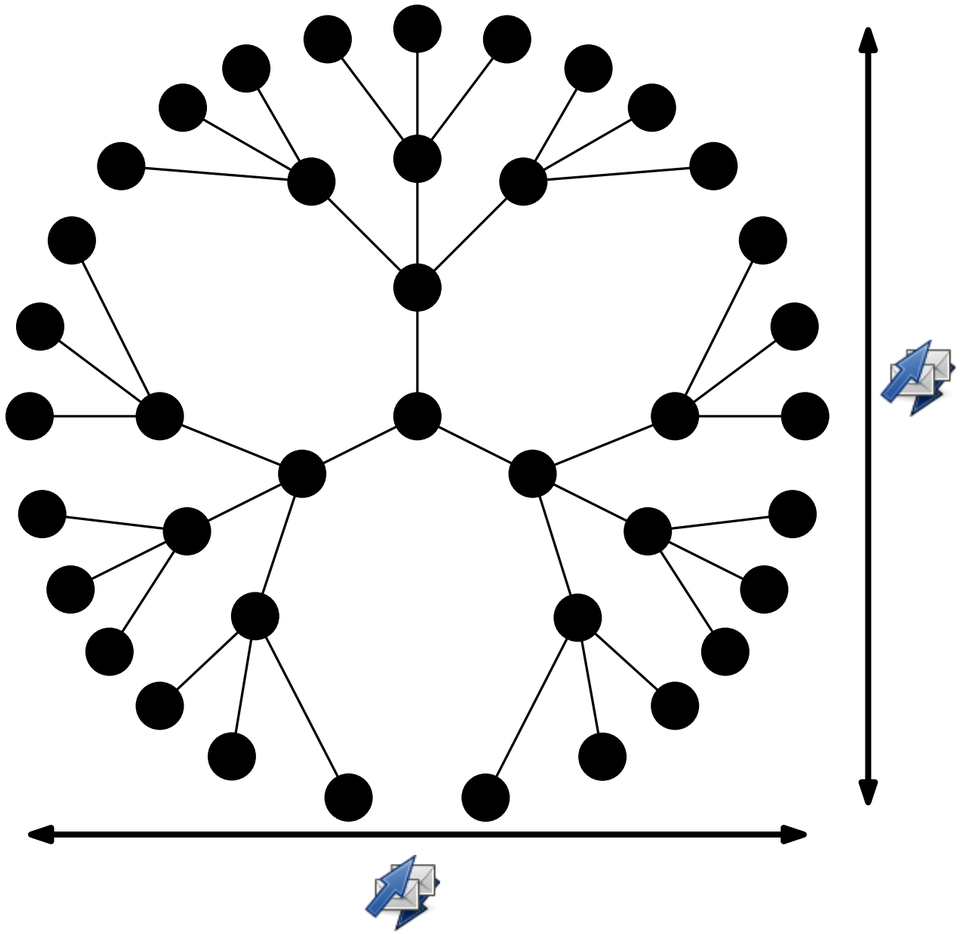
- **Distributed network** of n processors/nodes
  - graph $G = (V, E)$ with $|V| = n$
  - E: communication links
  - each node in V runs the same algorithm
- **Time is synchronous**: nodes alternate
  - arbitrary local computation & update of state variables
  - sending of messages to all neighbors
    * no bandwidth constraints
- **Unique identifiers** to nodes in the set $1, \ldots, \text{poly}(n)$
  * adversarially chosen * n is known to the nodes
  - needed to solve even basic problems (2-coloring a 2-path)
- **Possible randomness**: i.i.d. infinite random bit strings to nodes



The LOCAL model

[Linial FOCS '87 & SICOMP '92]

- **Distributed network** of n processors/nodes
  - graph $G = (V, E)$ with $|V| = n$
  - E: communication links
  - each node in V runs the same algorithm
- **Time is synchronous**: nodes alternate
  - arbitrary local computation & update of state variables
  - sending of messages to all neighbors
    * no bandwidth constraints
- **Unique identifiers** to nodes in the set $1, \ldots, \text{poly}(n)$
  * adversarially chosen * n is known to the nodes
  - needed to solve even basic problems (2-coloring a 2-path)
- **Possible randomness**: i.i.d. infinite random bit strings to nodes
- **Complexity measure**: number of communication rounds



Previous results about LOC

Lower bound: - $\Omega(\log n)$-rounds in deterministic LOCAL (in *bounded-degree* trees)

- $\Omega(\log\log n)$-rounds in randomized LOCAL (in *bounded-degree* trees)

- reduction from Sinkless Orientation [Balliu, Hirvonen, Lenzen, Olivetti, and Suomela, SIROCCO '19]
- fixed point in round elimination [Balliu, Brandt, Kuhn, Olivetti, and Saarhelo, DISC '25]

Previous results about LOC

Lower bound: - $\Omega(\log n)$-rounds in deterministic LOCAL (in *bounded-degree* trees)

- $\Omega(\log\log n)$-rounds in randomized LOCAL (in *bounded-degree* trees)

- reduction from Sinkless Orientation [Balliu, Hirvonen, Lenzen, Olivetti, and Suomela, SIROCCO '19]
- fixed point in round elimination [Balliu, Brandt, Kuhn, Olivetti, and Saarhelo, DISC '25]

Upper bound: - $O(n)$-rounds in both deterministic and randomized LOCAL (even in bounded-degree graphs)

Previous results about LOC

Lower bound: - $\Omega(\log n)$-rounds in deterministic LOCAL (in *bounded-degree* trees)

- $\Omega(\log\log n)$-rounds in randomized LOCAL (in *bounded-degree* trees)

- reduction from Sinkless Orientation [Balliu, Hirvonen, Lenzen, Olivetti, and Suomela, SIROCCO '19]
- fixed point in round elimination [Balliu, Brandt, Kuhn, Olivetti, and Saarhelo, DISC '25]

Upper bound: - $O(n)$-rounds in both deterministic and randomized LOCAL (even in bounded-degree graphs)

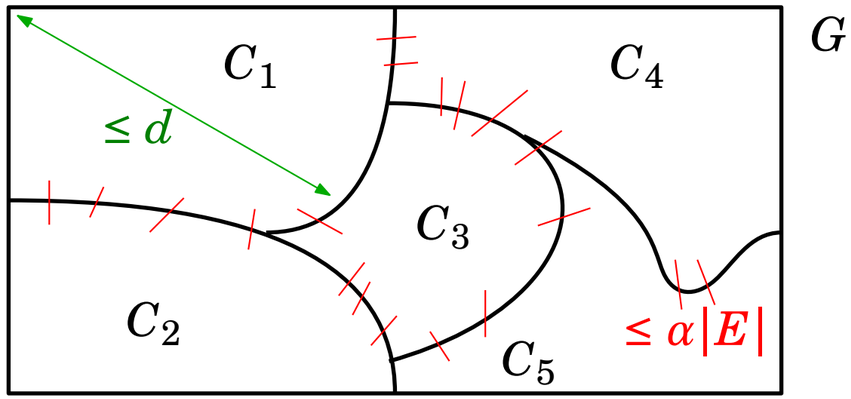
HUGE GAP!

Let's find a *better* distributed algorithm...

Network decomposition

(α, d)-**small boundary decomposition** of a graph $G = (V, E)$:

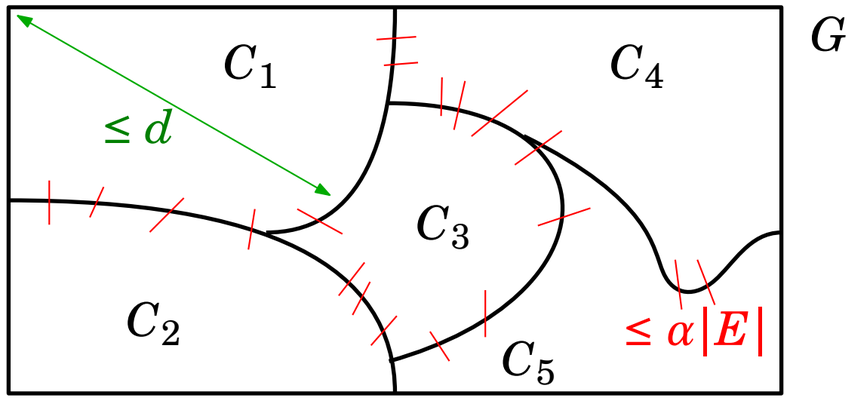
- partition of V into clusters (sets) $C_1, \ldots, C_k$
- $\text{diam}(C_i) \leq d$ for all i
- # inter-clusters edges $\leq \alpha |E|$



Network decomposition

(α, d)-**small boundary decomposition** of a graph $G = (V, E)$:

- partition of V into clusters (sets) $C_1, \ldots, C_k$
- $\text{diam}(C_i) \leq d$ for all i
- # inter-clusters edges $\leq \alpha |E|$

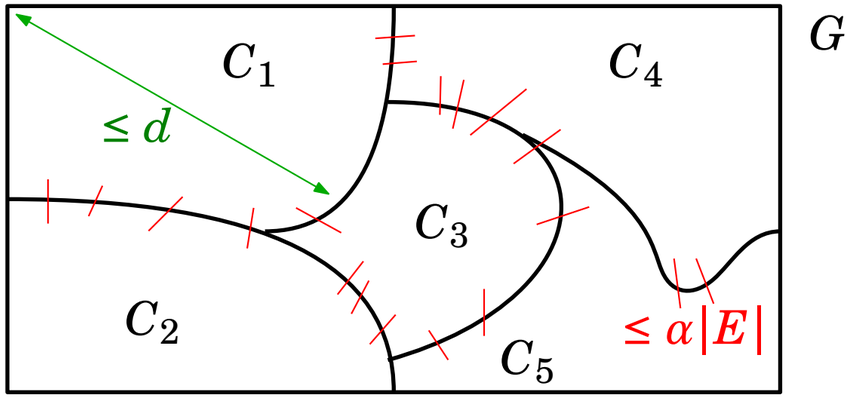


Theorem (adaptation of [Miller, Peng, and Xu, SPAA '13]):

There exists a randomized LOCAL algorithm $\mathcal{MPX}$ that computes an (α, d)-small boundary decomposition of a graph $G = (V, E)$ with the following properties:

- *Running time $O(\log n / \alpha)$.*
- *UB on the clusters' diameter is $d = O(\log n / \alpha)$.*

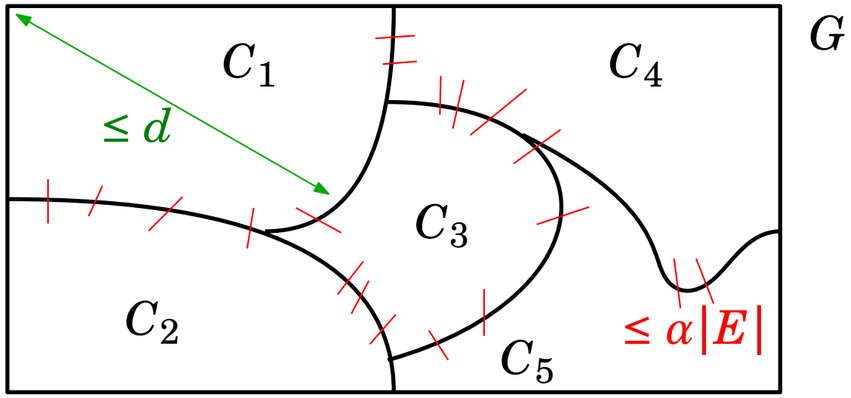
A “simple” algorithm



- *Brute-force solution with minimum potential in each cluster (time $O(d)$)*

  - distance from global minimum of the potential is $O(\alpha|E|)$

A “simple” algorithm

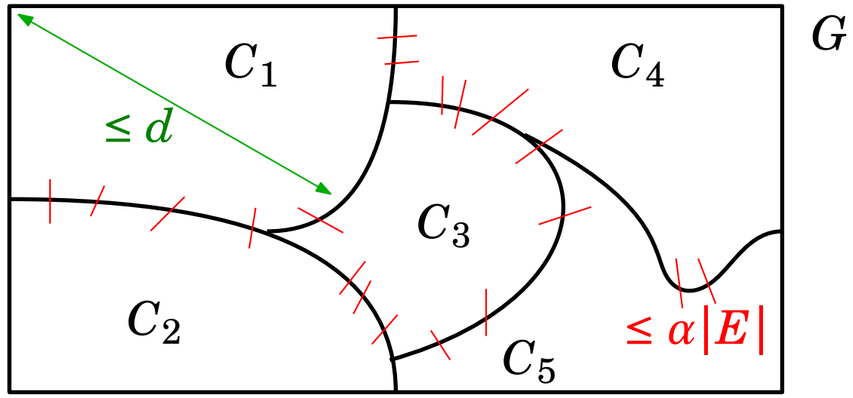


- *Brute-force solution with minimum potential in each cluster (time $O(d)$)*

  - distance from global minimum of the potential is $O(\alpha|E|)$

- *Simulate “fixing procedure” (at most $O(\alpha|E|)$ rounds)*

A “simple” algorithm



- *Brute-force solution with minimum potential in each cluster (time $O(d)$)*

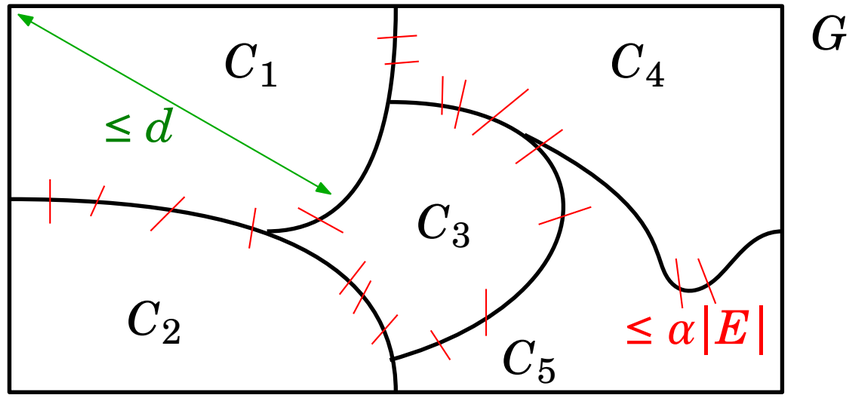
  - distance from global minimum of the potential is $O(\alpha|E|)$

- *Simulate “fixing procedure” (at most $O(\alpha|E|)$ rounds)*

Overall running time: cost($\mathcal{MPX}$) + $O(d) + O(\alpha|E|) = O(\log n/\alpha + \alpha|E|)$

- **max-degree** Δ: running time $O(\sqrt{\Delta n \log n})$ (minimized by $\alpha = \sqrt{\log n/(\Delta n)}$)

A “simple” algorithm



- *Brute-force solution with minimum potential in each cluster (time $O(d)$)*

  - distance from global minimum of the potential is $O(\alpha|E|)$

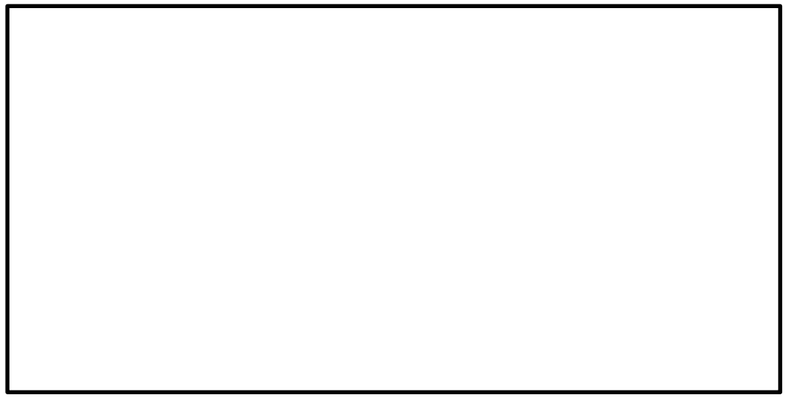
- *Simulate “fixing procedure” (at most $O(\alpha|E|)$ rounds)*

Overall running time: $\text{cost}(\mathcal{MPX}) + O(d) + O(\alpha|E|) = O(\log n/\alpha + \alpha|E|)$

- **max-degree** Δ: running time $O(\sqrt{\Delta n \log n})$ (minimized by $\alpha = \sqrt{\log n/(\Delta n)}$)

- still far from the lower bounds … How to do better?

Repeating “brute-force” does not work

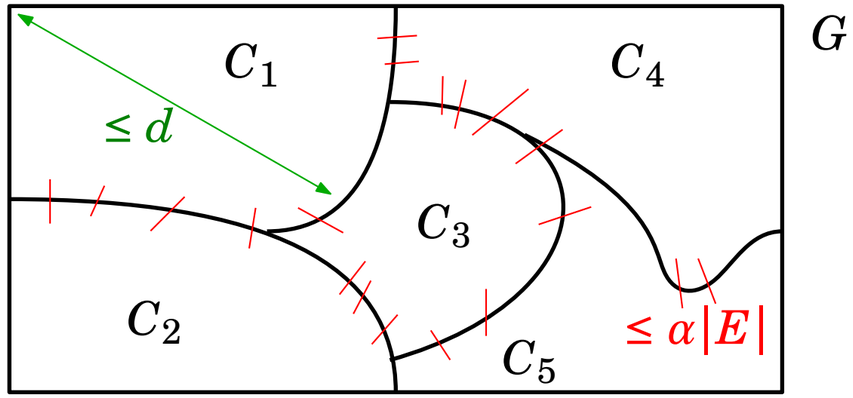


- *Repeat:*

  - *Run $\mathcal{MPX}$ to get (α, d)-network decomposition (time $O(d)$)*

  - *Brute-force solution with minimum potential in each cluster (time $O(d)$)*

Repeating “brute-force” does not work

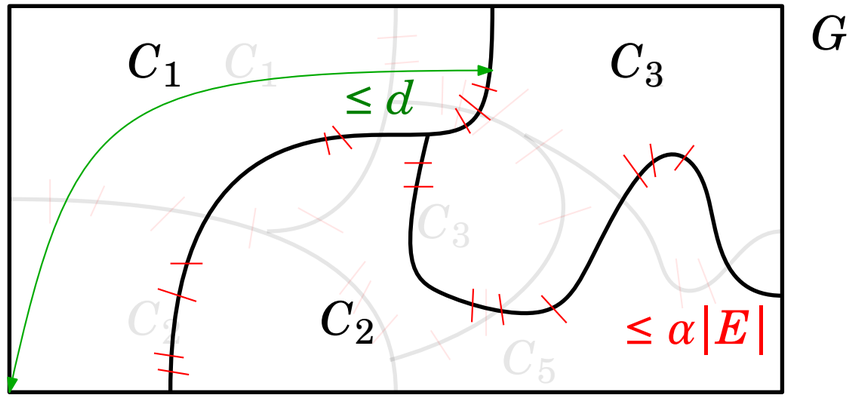


- *Repeat:*

  - *Run $\mathcal{MPX}$ to get (α, d)-network decomposition (time $O(d)$)*

  - *Brute-force solution with minimum potential in each cluster (time $O(d)$)*

Repeating “brute-force” does not work

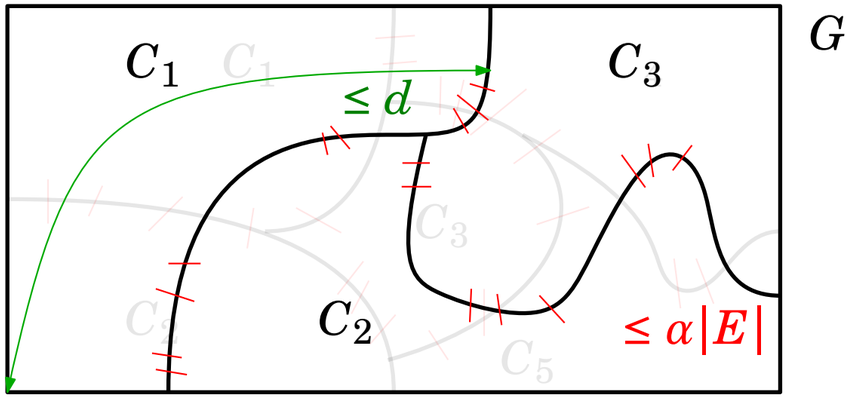


- *Repeat:*

  - *Run $\mathcal{MPX}$ to get (α, d)-network decomposition (time $O(d)$)*

  - *Brute-force solution with minimum potential in each cluster (time $O(d)$)*

Repeating “brute-force” does not work



- *Repeat:*

  - *Run $\mathcal{MPX}$ to get (α, d)-network decomposition (time $O(d)$)*

  - *Brute-force solution with minimum potential in each cluster (time $O(d)$)*

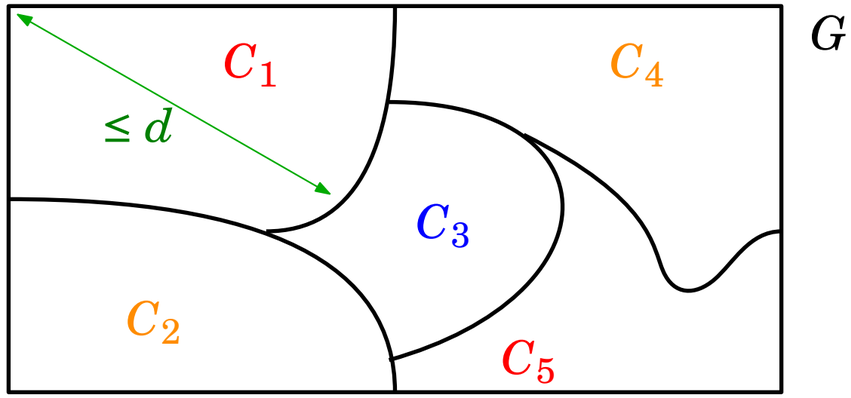
- Distance from global minimum of the potential keeps at $O(\alpha|E|)$

  - what to do?

Ingedient 1: colored network decomposition

(c,d)-**colored decomposition** of a graph $G = (V,E)$:

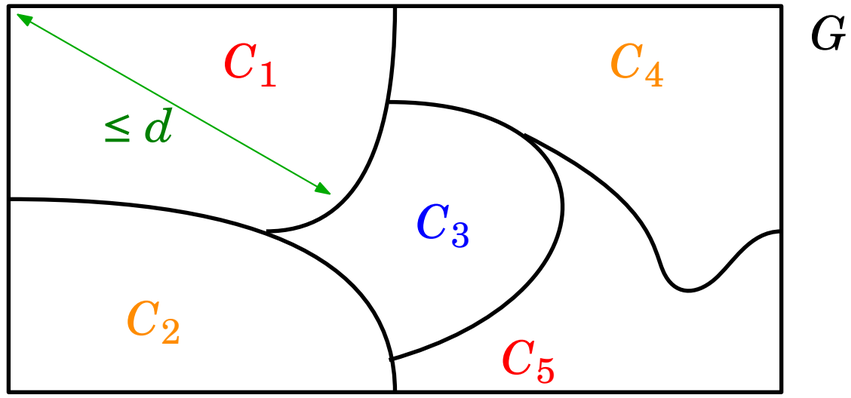
- partition of V into clusters (sets) $C_1,\dots,C_k$
- $\mathrm{diam}(C_i) \le d$ for all i
- $\varphi\colon \{C_1,\dots,C_k\} \to [c]$ is a coloring of the clusters
  - adjacent clusters have different colors



Ingedient 1: colored network decomposition

(c,d)-**colored decomposition** of a graph $G = (V,E)$:

- partition of V into clusters (sets) $C_1,\dots,C_k$
- $\text{diam}(C_i) \le d$ for all i
- $\varphi\colon \{C_1,\dots,C_k\} \to [c]$ is a coloring of the clusters
  - adjacent clusters have different colors



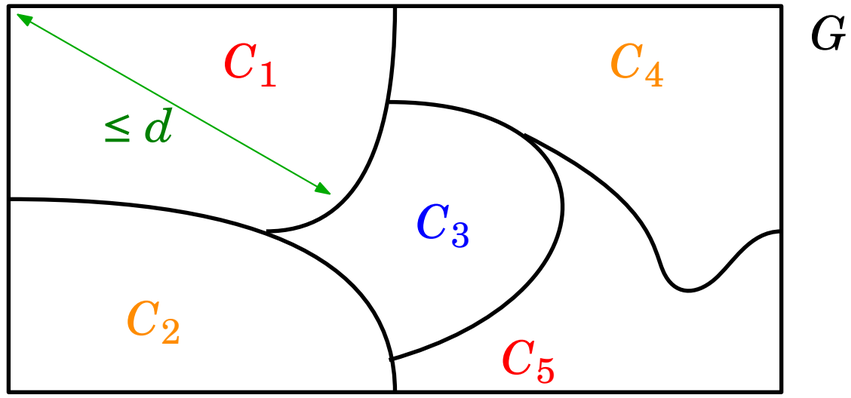
Theorem ([Ghaffari and Grunau, FOCS '24]):

There exists a deterministic LOCAL algorithm that computes an $(O(\log n), O(\log n))$*-colored decomposition of a graph* $G = (V,E)$ *in time* $\tilde{O}(\log^2 n)$.

Ingedient 1: colored network decomposition

(c,d)-**colored decomposition** of a graph $G = (V,E)$:

- partition of V into clusters (sets) $C_1,\dots,C_k$
- $\text{diam}(C_i) \le d$ for all i
- $\varphi: \{C_1,\dots,C_k\} \to [c]$ is a coloring of the clusters
  - adjacent clusters have different colors



Theorem ([Ghaffari and Grunau, FOCS '24]):

There exists a deterministic LOCAL algorithm that computes an $(O(\log n), O(\log n))$*-colored decomposition of a graph* $G = (V,E)$ *in time* $\tilde{O}(\log^2 n)$.

(c,d)-**colored decomposition** of power graph G^r:

- $\text{diam}(C_i) \le rd$ for all i
- if C_i, C_j same color, $\text{dist}(C_i, C_j) \ge r$ *# guarantee that clusters of the same color are far enough*
- running time multiplied by r

Ingredient 2: improving sets

Improving set in a 2-colored graph $G = (V, E)$

- Subset $A \subseteq V$ such that by flipping the colors of nodes in A the potential decreases

Ingredient 2: improving sets

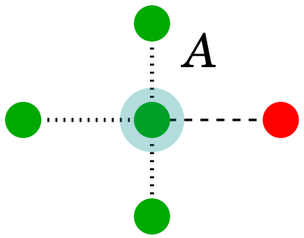
Improving set in a 2-colored graph $G = (V, E)$

- Subset $A \subseteq V$ such that by flipping the colors of nodes in A the potential decreases
- $\mathrm{Imp}(A)$ = improvement in the potential
- Improving ratio: $\mathrm{IR}(A) = \mathrm{Imp}(A)/|A|$

Ingredient 2: improving sets

Improving set in a 2-colored graph $G = (V, E)$

- Subset $A \subseteq V$ such that by flipping the colors of nodes in A the potential decreases
- $\text{Imp}(A)$ = improvement in the potential
- Improving ratio: $\text{IR}(A) = \text{Imp}(A)/|A|$



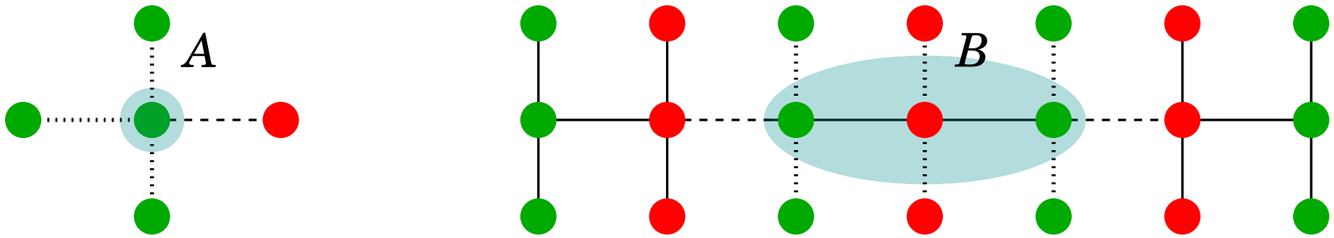
$\text{Imp}(A) = 3 - 1 = 2$

$\text{IR}(A) = 2/1 = 2$

Ingredient 2: improving sets

Improving set in a 2-colored graph $G = (V, E)$

- Subset $A \subseteq V$ such that by flipping the colors of nodes in A the potential decreases
- $\text{Imp}(A)$ = improvement in the potential
- Improving ratio: $\text{IR}(A) = \text{Imp}(A)/|A|$



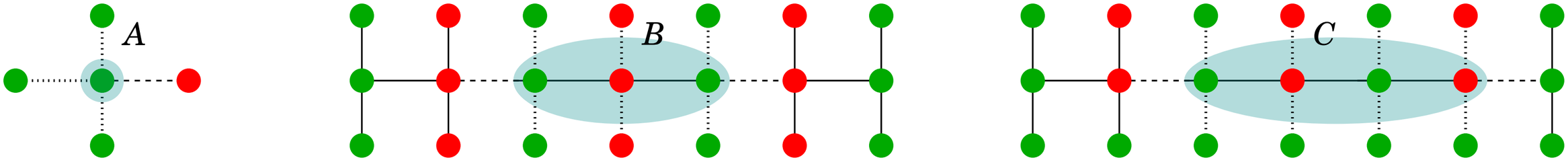
$\text{Imp}(A) = 3 - 1 = 2$ $\quad$ $\text{Imp}(B) = 4$

$\text{IR}(A) = 2/1 = 2$ $\quad$ $\text{IR}(B) = 4/3$

Ingredient 2: improving sets

Improving set in a 2-colored graph $G = (V, E)$

- Subset $A \subseteq V$ such that by flipping the colors of nodes in A the potential decreases
- $\text{Imp}(A)$ = improvement in the potential
- Improving ratio: $\text{IR}(A) = \text{Imp}(A)/|A|$



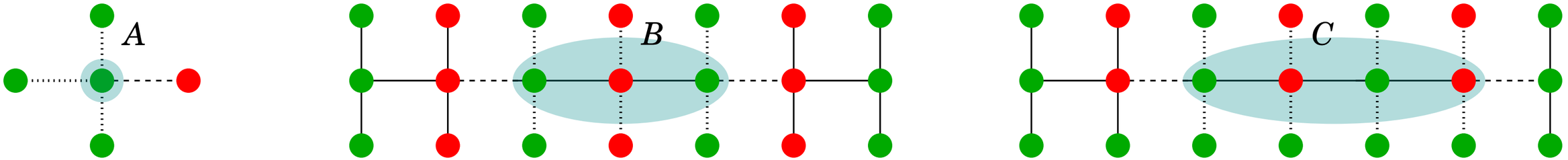
$\text{Imp}(A) = 3 - 1 = 2$ $\quad$ $\text{Imp}(B) = 4$ $\quad$ $\text{Imp}(C) = 2$

$\text{IR}(A) = 2/1 = 2$ $\quad$ $\text{IR}(B) = 4/3$ $\quad$ $\text{IR}(C) = 1/2$

Ingredient 2: improving sets

Improving set in a 2-colored graph $G = (V, E)$

- Subset $A \subseteq V$ such that by flipping the colors of nodes in A the potential decreases
- $\text{Imp}(A)$ = improvement in the potential
- Improving ratio: $\text{IR}(A) = \text{Imp}(A)/|A|$



$\text{Imp}(A) = 3 - 1 = 2$ $\quad$ $\text{Imp}(B) = 4$ $\quad$ $\text{Imp}(C) = 2$

$\text{IR}(A) = 2/1 = 2$ $\quad$ $\text{IR}(B) = 4/3$ $\quad$ $\text{IR}(C) = 1/2$

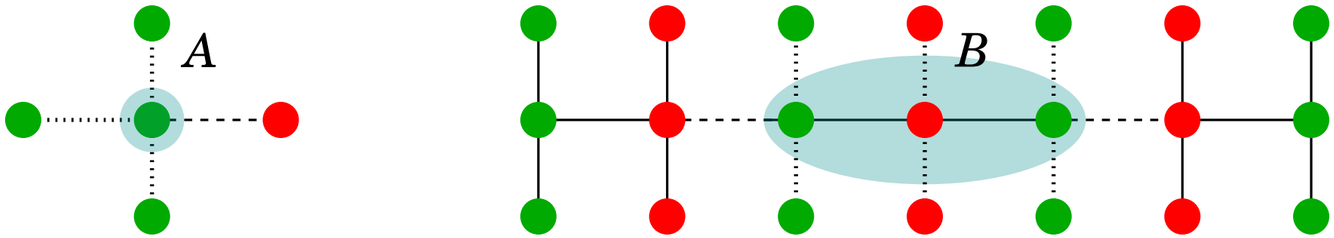
Minimal improving set: improving set A such that

- There is no subset $A' \subseteq A$ with $\text{IR}(A') > \text{IR}(A)$ ⟶ "quality": is there "useless stuff"?

Ingredient 2: improving sets

Improving set in a 2-colored graph $G = (V, E)$

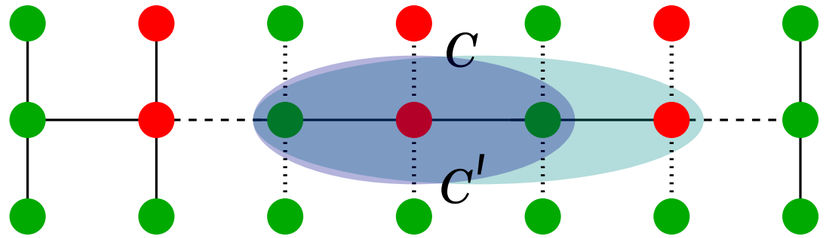
- Subset $A \subseteq V$ such that by flipping the colors of nodes in A the potential decreases
- $\text{Imp}(A)$ = improvement in the potential
- Improving ratio: $\text{IR}(A) = \text{Imp}(A)/|A|$



$\text{Imp}(A) = 3 - 1 = 2$ $\quad$ $\text{Imp}(B) = 4$

$\text{IR}(A) = 2/1 = 2$ $\quad$ $\text{IR}(B) = 4/3$

C not minimal



$\text{Imp}(C) = 2$ $\quad$ $\text{Imp}(C') = 2$

$\text{IR}(C) = 1/2$ $\quad$ $\text{IR}(C') = 2/3$

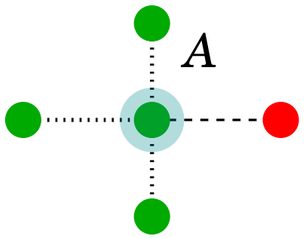
Minimal improving set: improving set A such that

- There is no subset $A' \subseteq A$ with $\text{IR}(A') > \text{IR}(A)$ ⟶ "quality": is there "useless stuff"?

Ingredient 2: improving sets

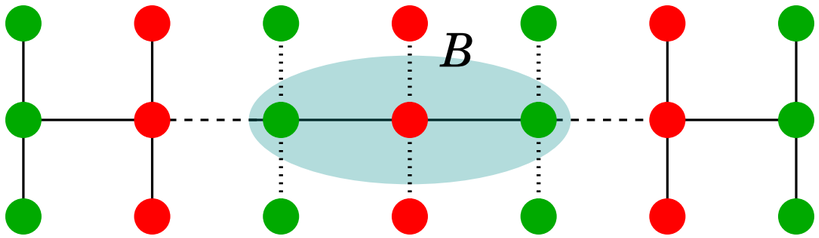
Improving set in a 2-colored graph $G = (V, E)$

- Subset $A \subseteq V$ such that by flipping the colors of nodes in A the potential decreases
- $\text{Imp}(A)$ = improvement in the potential
- Improving ratio: $\text{IR}(A) = \text{Imp}(A)/|A|$



$\text{Imp}(A) = 3 - 1 = 2$

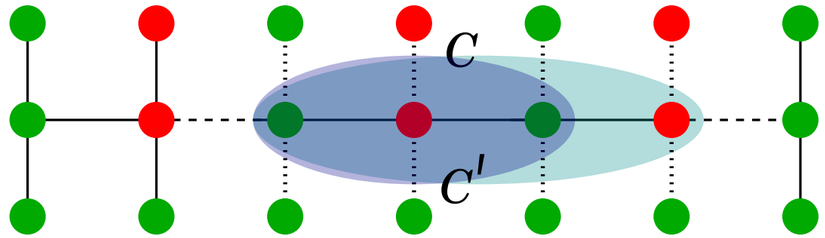
$\text{IR}(A) = 2/1 = 2$



$\text{Imp}(B) = 4$

$\text{IR}(B) = 4/3$

C not minimal



$\text{Imp}(C) = 2$

$\text{IR}(C) = 1/2$

$\text{Imp}(C') = 2$

$\text{IR}(C') = 2/3$

Minimal improving set: improving set A such that

- There is no subset $A' \subseteq A$ with $\text{IR}(A') > \text{IR}(A)$ ⟶ "quality": is there "useless stuff"?

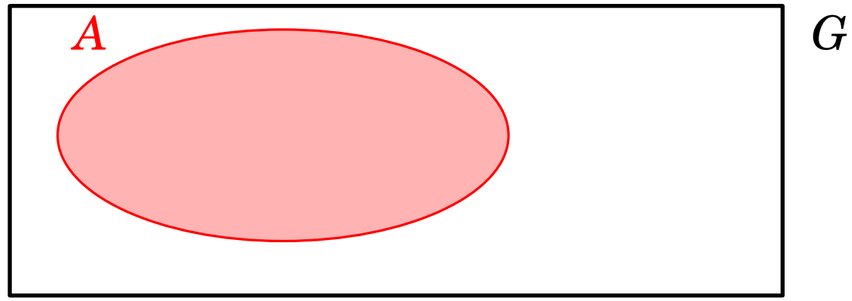
Error: minimal improving set of $\text{IR} \geq 1$

Ingredient 3: properties of improving sets

Property 1: **local witness** of **minimal improving sets**

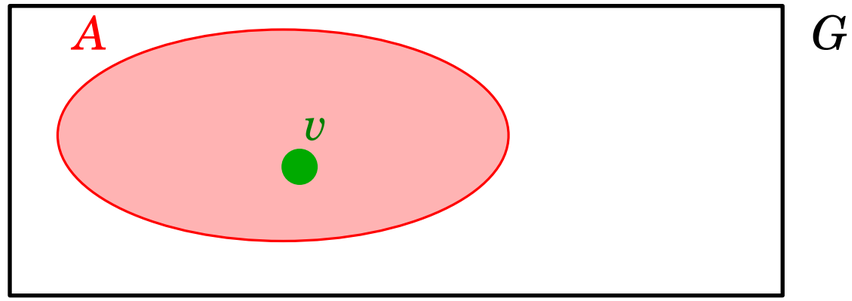
Ingredient 3: properties of improving sets

Property 1: **local witness** of **minimal improving sets**



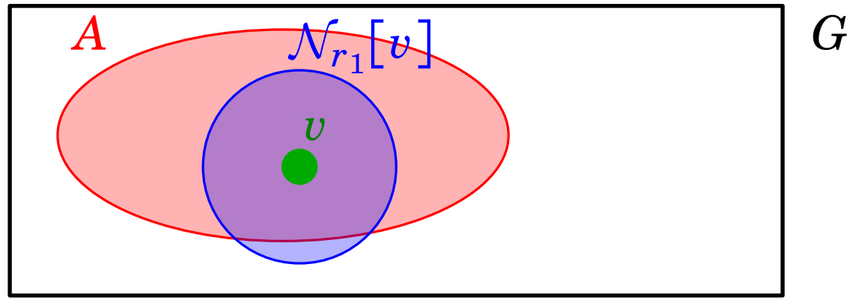
Ingredient 3: properties of improving sets

Property 1: **local witness** of **minimal improving sets**



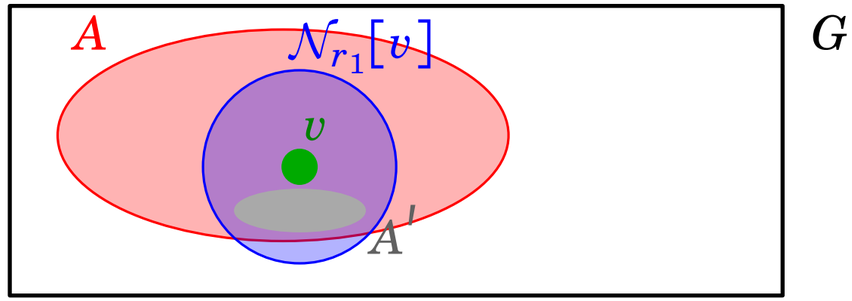
Ingredient 3: properties of improving sets

Property 1: **local witness** of **minimal improving sets**



Ingredient 3: properties of improving sets

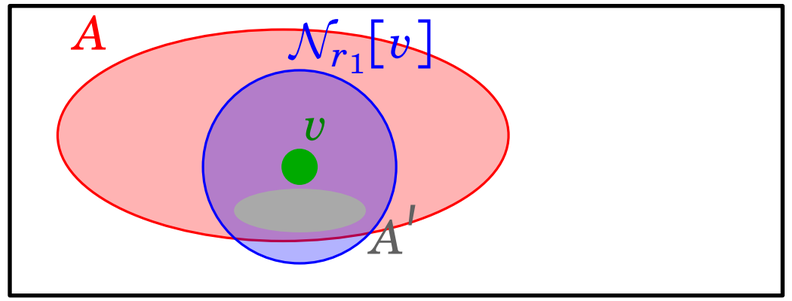
Property 1: **local witness** of **minimal improving sets**



Ingredient 3: properties of improving sets

Property 1: **local witness** of **minimal improving sets**

- $A \subseteq V$ minimal improving set
- $\mathrm{IR}(A) \geq x$
- $\varepsilon < x$

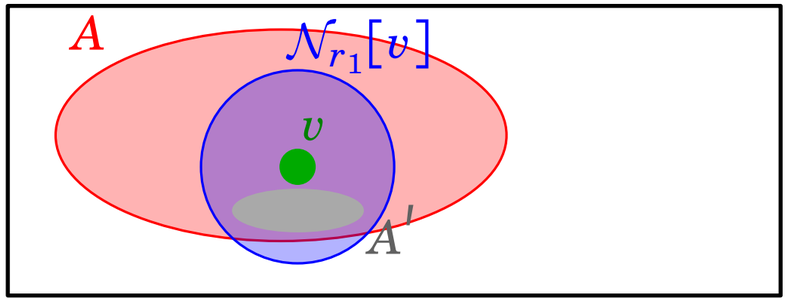


$\Longrightarrow$ for all $v \in A$, $\exists r_1 = O(\Delta \log n/\varepsilon)$ and minimal improving set $A' \subseteq \mathcal{N}_{r_1}[v] \cap A$ such that $\mathrm{IR}(A') \geq x - \varepsilon$

Ingredient 3: properties of improving sets

Property 1: **local witness** of **minimal improving sets**

- $A \subseteq V$ minimal improving set
- $\mathrm{IR}(A) \geq x$
- $\varepsilon < x$



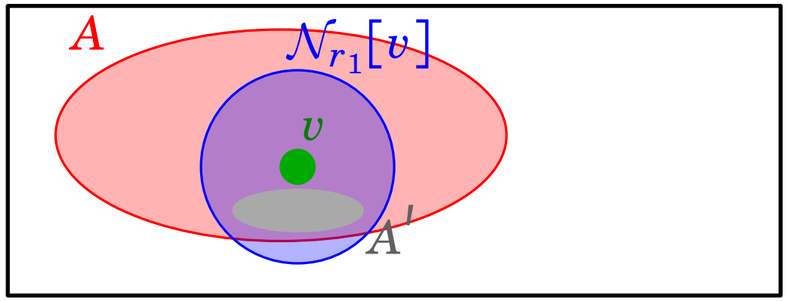
$\Longrightarrow$ for all $v \in A$, $\exists r_1 = O(\Delta \log n/\varepsilon)$ and minimal improving set $A' \subseteq \mathcal{N}_{r_1}[v] \cap A$ such that $\mathrm{IR}(A') \geq x - \varepsilon$

Sequence of x-improving sets:

Ingredient 3: properties of improving sets

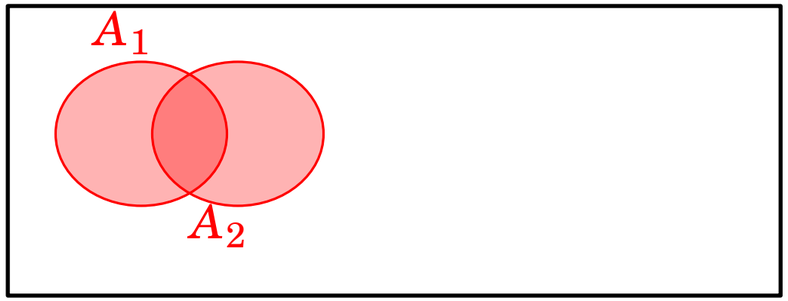
Property 1: **local witness** of **minimal improving sets**

- $A \subseteq V$ minimal improving set
- $\mathrm{IR}(A) \geq x$
- $\varepsilon < x$



$\Longrightarrow$ for all $v \in A$, $\exists r_1 = O(\Delta \log n/\varepsilon)$ and minimal improving set $A' \subseteq \mathcal{N}_{r_1}[v] \cap A$ such that $\mathrm{IR}(A') \geq x - \varepsilon$

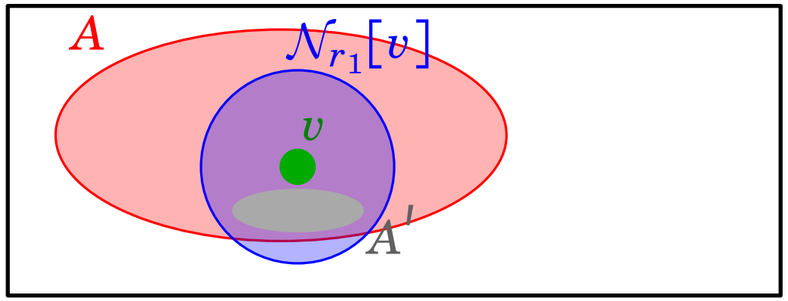
Sequence of x-improving sets:



Ingredient 3: properties of improving sets

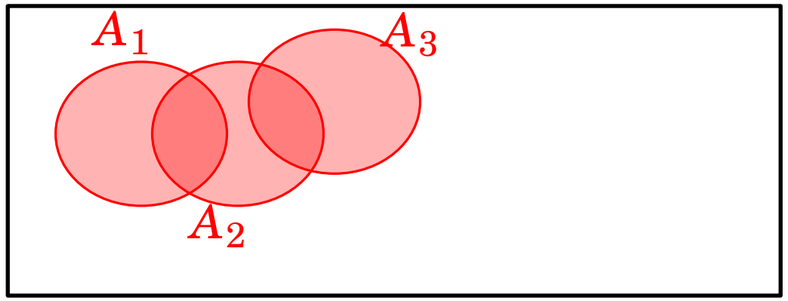
Property 1: **local witness** of **minimal improving sets**

- $A \subseteq V$ minimal improving set
- $\mathrm{IR}(A) \geq x$
- $\varepsilon < x$



$\Longrightarrow$ for all $v \in A$, $\exists r_1 = O(\Delta \log n / \varepsilon)$ and minimal improving set $A' \subseteq \mathcal{N}_{r_1}[v] \cap A$ such that $\mathrm{IR}(A') \geq x - \varepsilon$

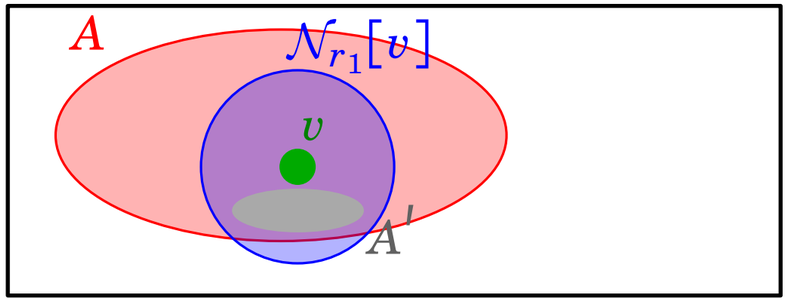
Sequence of x-improving sets:



Ingredient 3: properties of improving sets

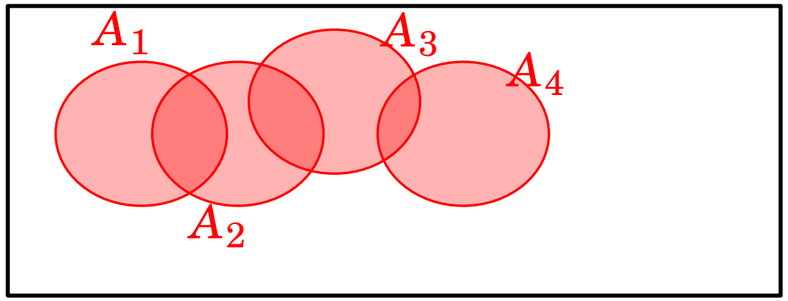
Property 1: **local witness** of **minimal improving sets**

- $A \subseteq V$ minimal improving set
- $\mathrm{IR}(A) \geq x$
- $\varepsilon < x$



$\Longrightarrow$ for all $v \in A$, $\exists r_1 = O(\Delta \log n/\varepsilon)$ and minimal improving set $A' \subseteq \mathcal{N}_{r_1}[v] \cap A$ such that $\mathrm{IR}(A') \geq x - \varepsilon$

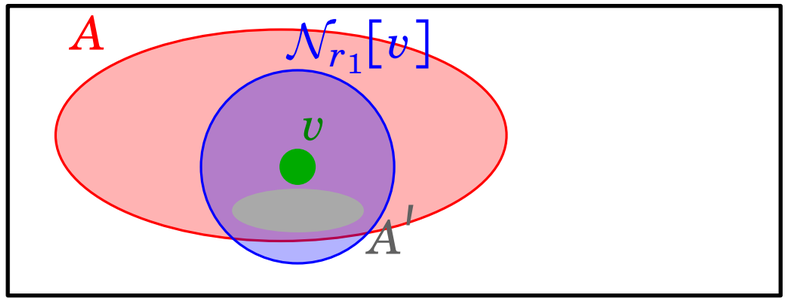
Sequence of x-improving sets:



Ingredient 3: properties of improving sets

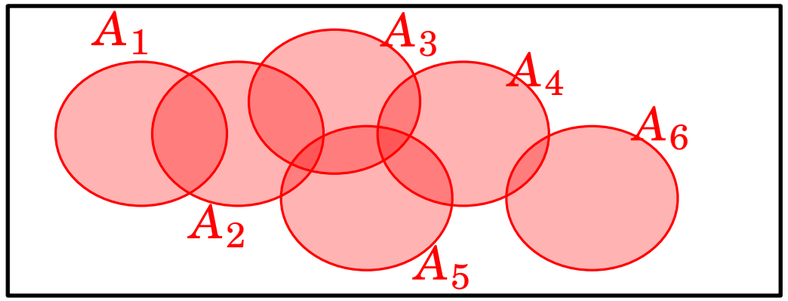
Property 1: **local witness** of **minimal improving sets**

- $A \subseteq V$ minimal improving set
- $\mathrm{IR}(A) \geq x$
- $\varepsilon < x$



$\Longrightarrow$ for all $v \in A$, $\exists r_1 = O(\Delta \log n/\varepsilon)$ and minimal improving set $A' \subseteq \mathcal{N}_{r_1}[v] \cap A$ such that $\mathrm{IR}(A') \geq x - \varepsilon$

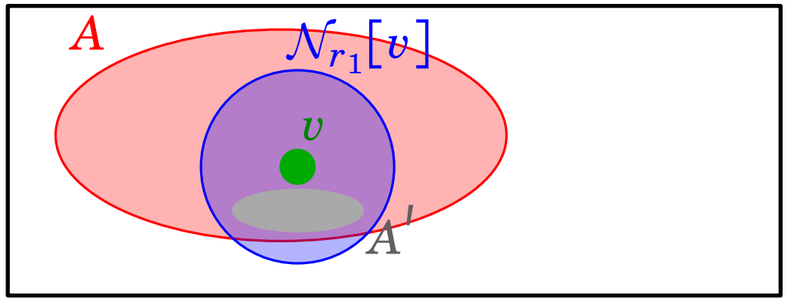
Sequence of x-improving sets:



Ingredient 3: properties of improving sets

Property 1: **local witness** of **minimal improving sets**

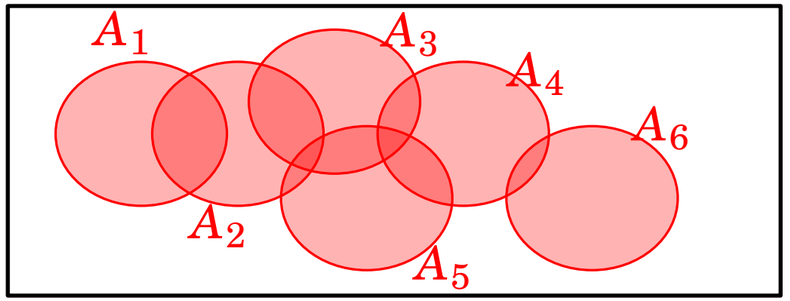
- $A \subseteq V$ minimal improving set
- $\mathrm{IR}(A) \geq x$
- $\varepsilon < x$



$\Longrightarrow$ for all $v \in A$, $\exists r_1 = O(\Delta \log n/\varepsilon)$ and minimal improving set $A' \subseteq \mathcal{N}_{r_1}[v] \cap A$ such that $\mathrm{IR}(A') \geq x - \varepsilon$

Sequence of x-improving sets:

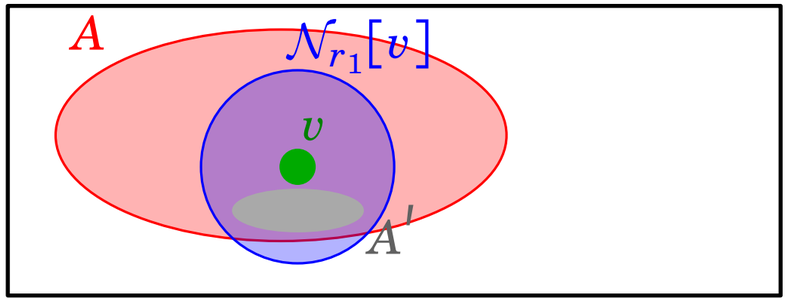
- $A_1, \ldots, A_k \subseteq V$, $\mathrm{diam}(G) \leq r_1$
- A_1 minimal improving set with $\mathrm{IR}(A_1) \geq x$
- for all $i > 1$, A_i minimal improving set with $\mathrm{IR}(A_i) \geq x$ after having flipped $A_1, \ldots, A_{i-1}$



Ingredient 3: properties of improving sets

Property 1: **local witness** of **minimal improving sets**

- $A \subseteq V$ minimal improving set
- $\mathrm{IR}(A) \geq x$
- $\varepsilon < x$

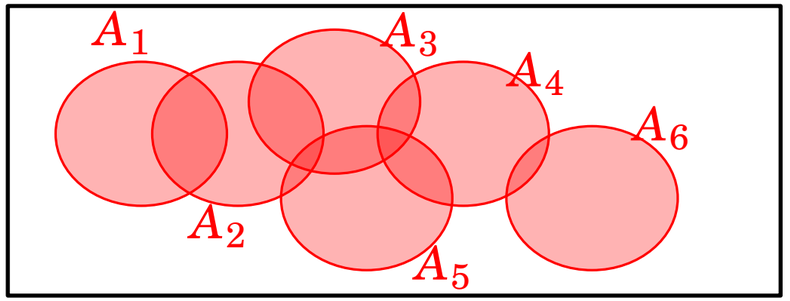


$\Longrightarrow$ for all $v \in A$, $\exists r_1 = O(\Delta \log n/\varepsilon)$ and minimal improving set $A' \subseteq \mathcal{N}_{r_1}[v] \cap A$ such that $\mathrm{IR}(A') \geq x - \varepsilon$

Sequence of x-improving sets:

- $A_1, \ldots, A_k \subseteq V$, $\mathrm{diam}(G) \leq r_1$
- A_1 minimal improving set with $\mathrm{IR}(A_1) \geq x$
- for all $i > 1$, A_i minimal improving set with $\mathrm{IR}(A_i) \geq x$ after having flipped $A_1, \ldots, A_{i-1}$

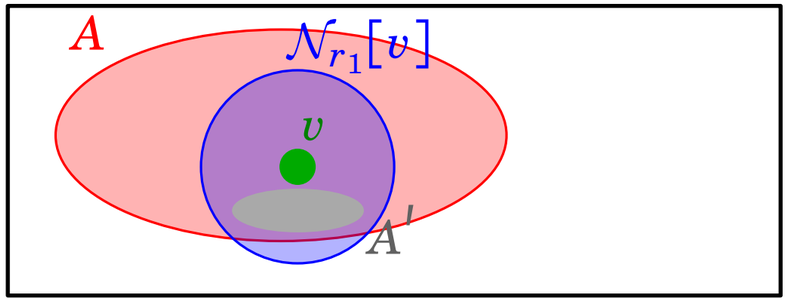
Property 2: **local witness** of **sequences** of x-improving sets



Ingredient 3: properties of improving sets

Property 1: **local witness** of **minimal improving sets**

- $A \subseteq V$ minimal improving set
- $\mathrm{IR}(A) \geq x$
- $\varepsilon < x$

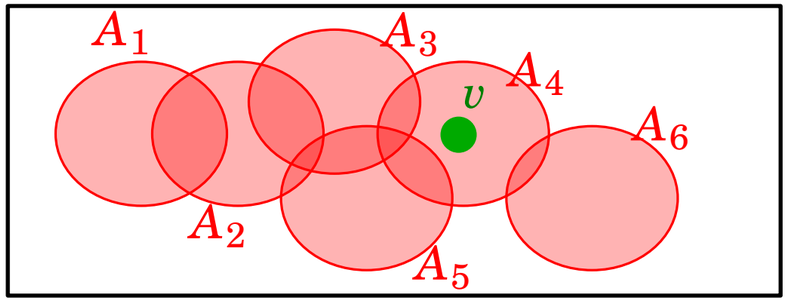


$\Longrightarrow$ for all $v \in A$, $\exists r_1 = O(\Delta \log n / \varepsilon)$ and minimal improving set $A' \subseteq \mathcal{N}_{r_1}[v] \cap A$ such that $\mathrm{IR}(A') \geq x - \varepsilon$

Sequence of x-improving sets:

- $A_1, \ldots, A_k \subseteq V$, $\mathrm{diam}(G) \leq r_1$
- A_1 minimal improving set with $\mathrm{IR}(A_1) \geq x$
- for all $i > 1$, A_i minimal improving set with $\mathrm{IR}(A_i) \geq x$ after having flipped $A_1, \ldots, A_{i-1}$

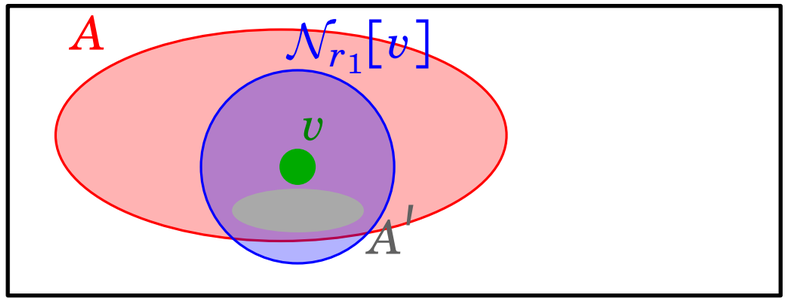
Property 2: **local witness** of **sequences** of x-improving sets



Ingredient 3: properties of improving sets

Property 1: **local witness** of **minimal improving sets**

- $A \subseteq V$ minimal improving set
- $\mathrm{IR}(A) \geq x$
- $\varepsilon < x$

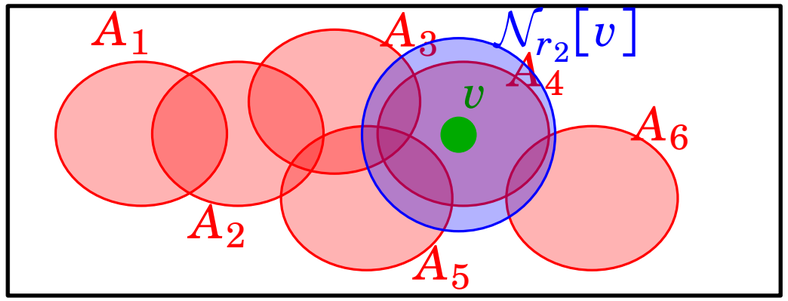


$\Longrightarrow$ for all $v \in A$, $\exists r_1 = O(\Delta \log n / \varepsilon)$ and minimal improving set $A' \subseteq \mathcal{N}_{r_1}[v] \cap A$ such that $\mathrm{IR}(A') \geq x - \varepsilon$

Sequence of x-improving sets:

- $A_1, \ldots, A_k \subseteq V$, $\mathrm{diam}(G) \leq r_1$
- A_1 minimal improving set with $\mathrm{IR}(A_1) \geq x$
- for all $i > 1$, A_i minimal improving set with $\mathrm{IR}(A_i) \geq x$ after having flipped $A_1, \ldots, A_{i-1}$

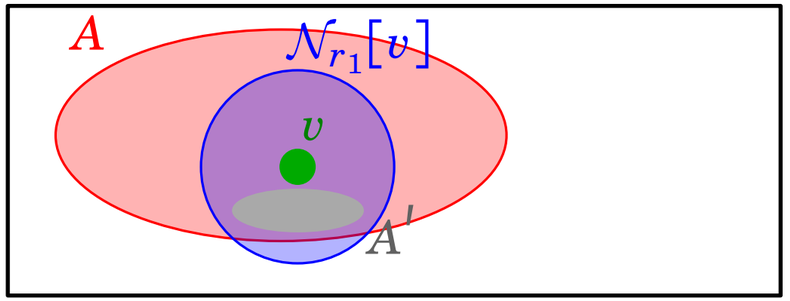
Property 2: **local witness** of **sequences** of x-improving sets



Ingredient 3: properties of improving sets

Property 1: **local witness** of **minimal improving sets**

- $A \subseteq V$ minimal improving set
- $\mathrm{IR}(A) \geq x$
- $\varepsilon < x$

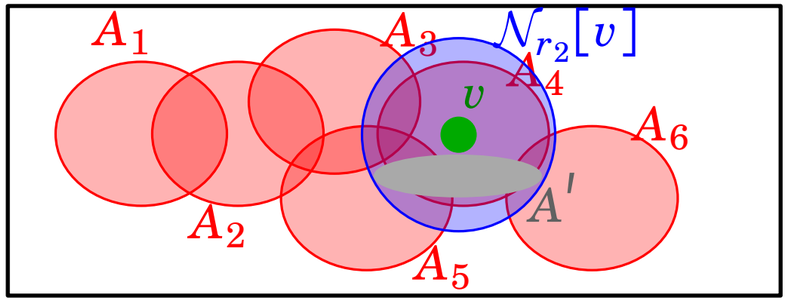


$\Longrightarrow$ for all $v \in A$, $\exists r_1 = O(\Delta \log n/\varepsilon)$ and minimal improving set $A' \subseteq \mathcal{N}_{r_1}[v] \cap A$ such that $\mathrm{IR}(A') \geq x - \varepsilon$

Sequence of x-improving sets:

- $A_1, \ldots, A_k \subseteq V$, $\mathrm{diam}(G) \leq r_1$
- A_1 minimal improving set with $\mathrm{IR}(A_1) \geq x$
- for all $i > 1$, A_i minimal improving set with $\mathrm{IR}(A_i) \geq x$ after having flipped $A_1, \ldots, A_{i-1}$

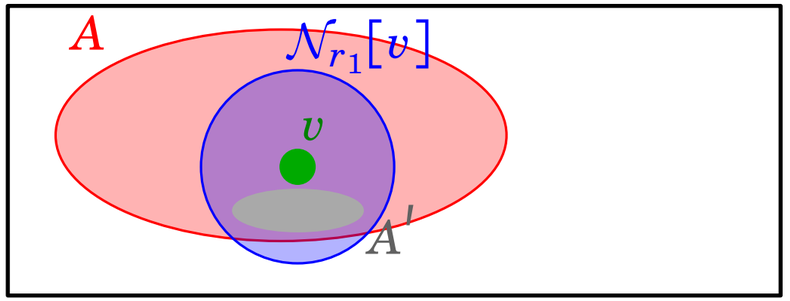
Property 2: **local witness** of **sequences** of x-improving sets



Ingredient 3: properties of improving sets

Property 1: **local witness** of **minimal improving sets**

- $A \subseteq V$ minimal improving set
- $\mathrm{IR}(A) \geq x$
- $\varepsilon < x$



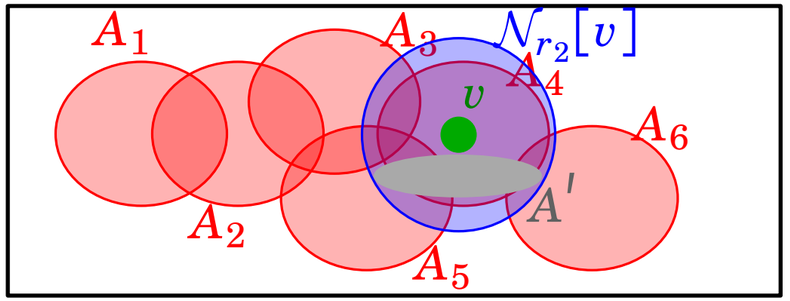
$\Longrightarrow$ for all $v \in A$, $\exists r_1 = O(\Delta \log n/\varepsilon)$ and minimal improving set $A' \subseteq \mathcal{N}_{r_1}[v] \cap A$ such that $\mathrm{IR}(A') \geq x - \varepsilon$

Sequence of x-improving sets:

- $A_1, \ldots, A_k \subseteq V$, $\mathrm{diam}(G) \leq r_1$
- A_1 minimal improving set with $\mathrm{IR}(A_1) \geq x$
- for all $i > 1$, A_i minimal improving set with $\mathrm{IR}(A_i) \geq x$ after having flipped $A_1, \ldots, A_{i-1}$

Property 2: **local witness** of **sequences** of x-improving sets

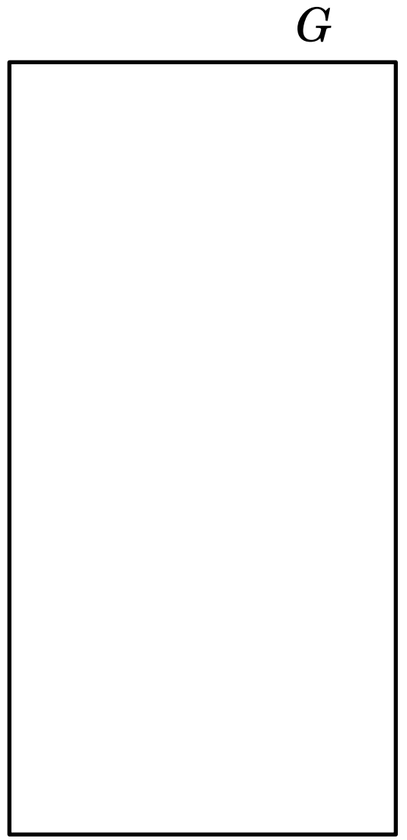
- $A_1, \ldots, A_k \subseteq V$ sequence of x-improving sets
- $\varepsilon < x$
- $A = \cup_i A_i$



$\Longrightarrow$ for all i, for all $v \in A$, $\exists r_2 = O(\Delta^2 \log^2 n/\varepsilon^2)$ and minimal improving set $A' \subseteq \mathcal{N}_{r_2}[v] \cap A$ such that $\mathrm{IR}(A') \geq x - \varepsilon$

Algorithm's idea

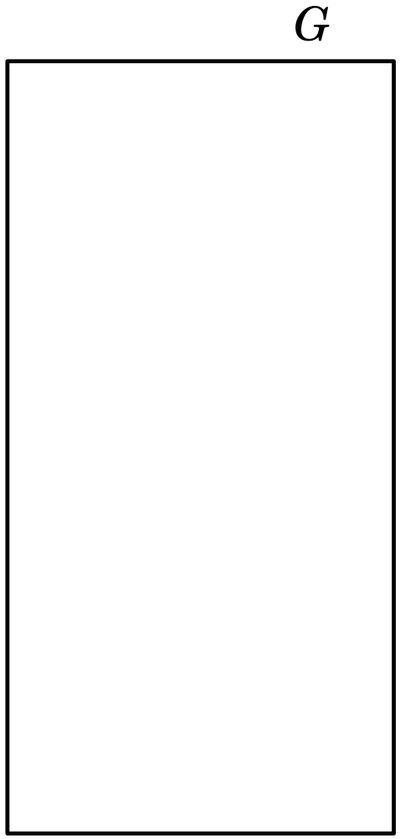
- IR:



Algorithm's idea

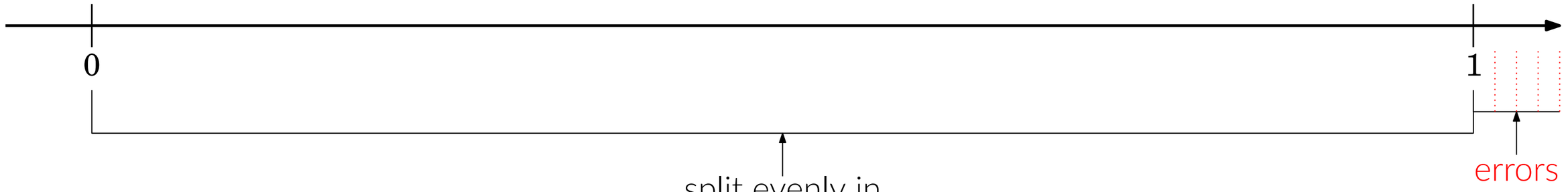
- IR:

- [Ghaffari and Grunau, FOCS '24]:
  $(C_1 \log n, C_2 \log n)$-colored decomposition

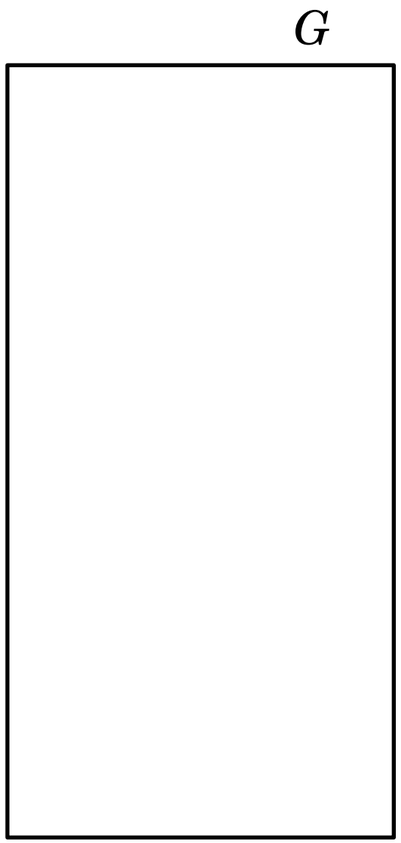


Algorithm's idea

- IR:

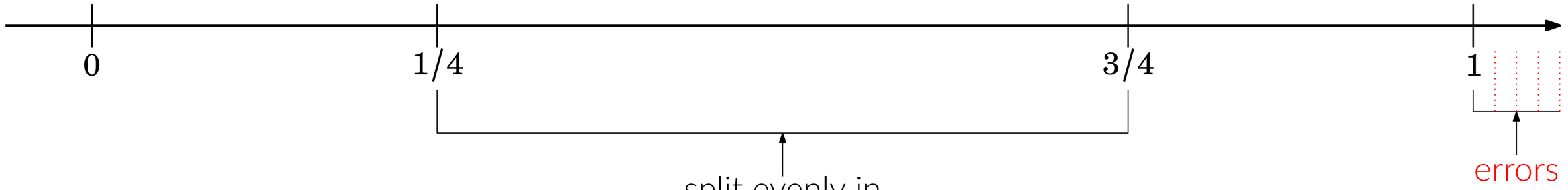


- [Ghaffari and Grunau, FOCS '24]:
  $(C_1 \log n, C_2 \log n)$-colored decomposition

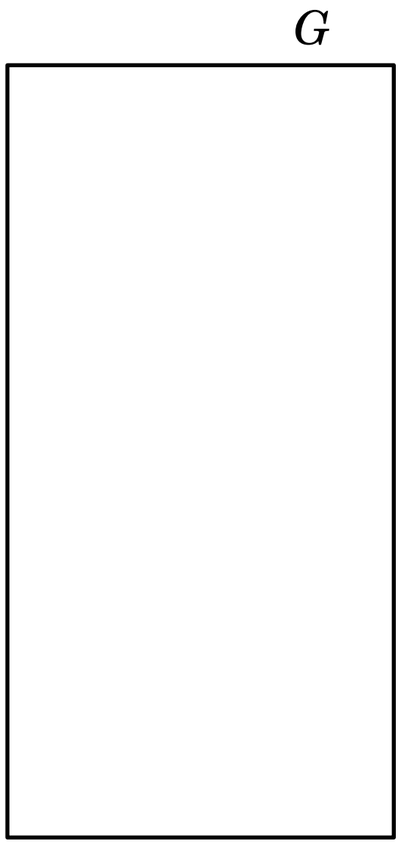


Algorithm's idea

- IR:

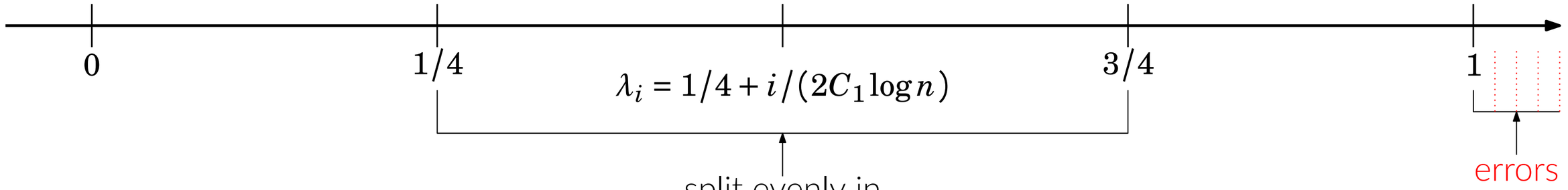


- [Ghaffari and Grunau, FOCS '24]:
  $(C_1 \log n, C_2 \log n)$-colored decomposition

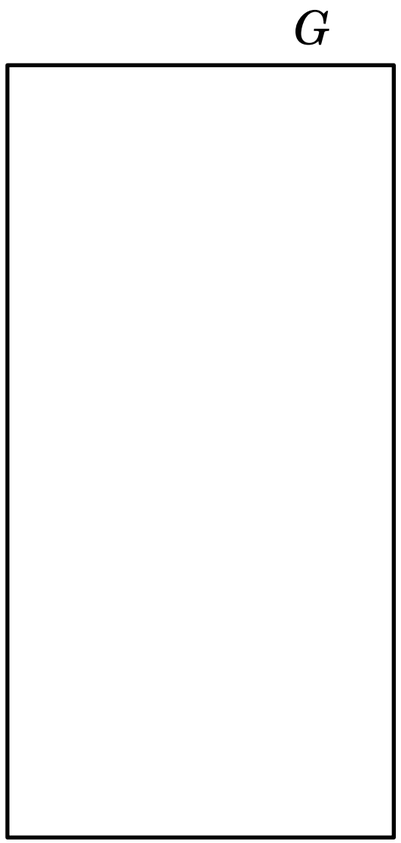


Algorithm's idea

- IR:

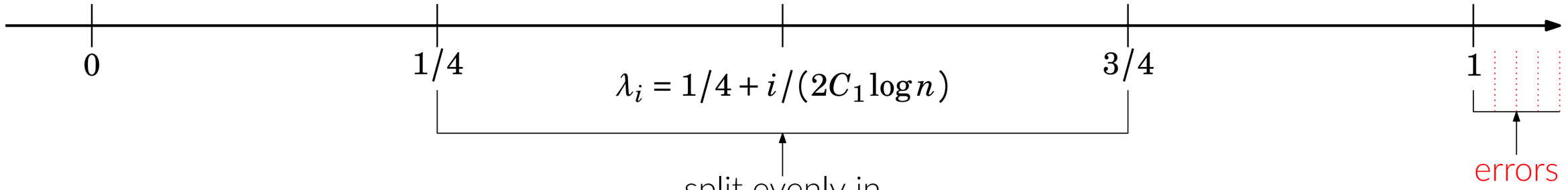


- [Ghaffari and Grunau, FOCS '24]:
  $(C_1 \log n, C_2 \log n)$-colored decomposition



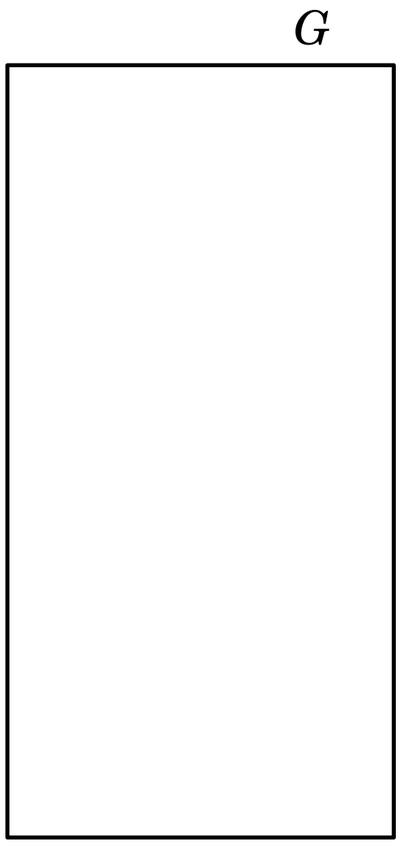
Algorithm's idea

- IR:



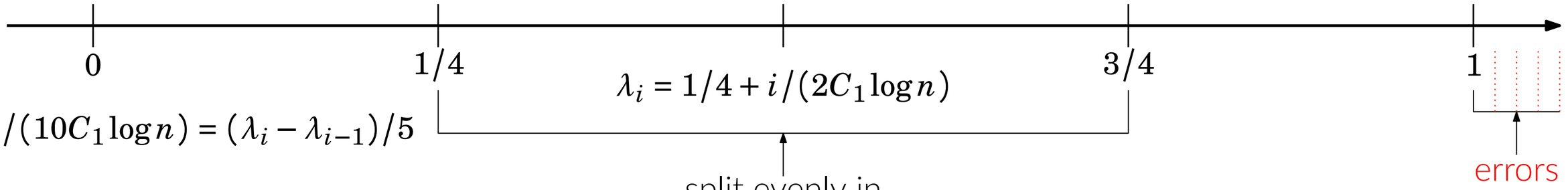
- [Ghaffari and Grunau, FOCS '24]:
  $(C_1 \log n, C_2 \log n)$-colored decomposition

- in phase i, we will **flip improving sets** with IR ≥ λ_i *around* **class color** i



Algorithm's idea

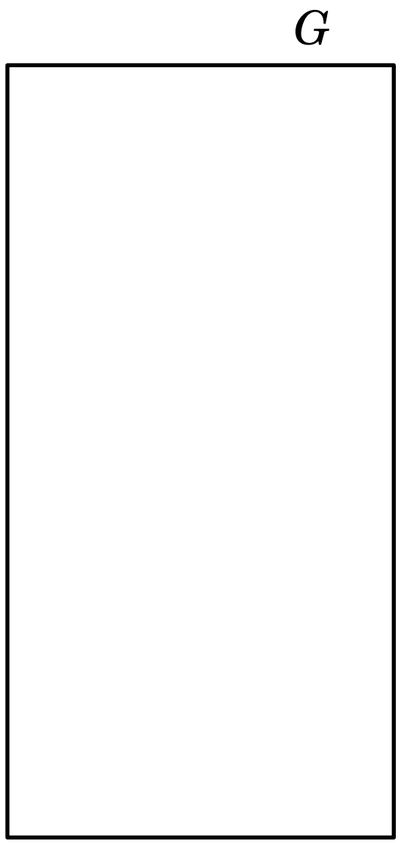
- IR:



- $\varepsilon = 1/(10C_1 \log n) = (\lambda_i - \lambda_{i-1})/5$

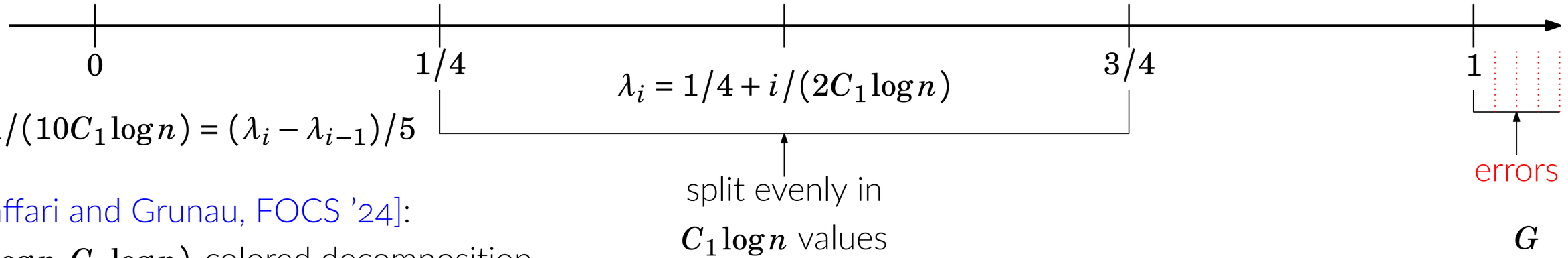
- [Ghaffari and Grunau, FOCS '24]:
  $(C_1 \log n, C_2 \log n)$-colored decomposition

- in phase i, we will **flip improving sets** with IR ≥ λ_i *around* **class color** i



Algorithm's idea

- IR:

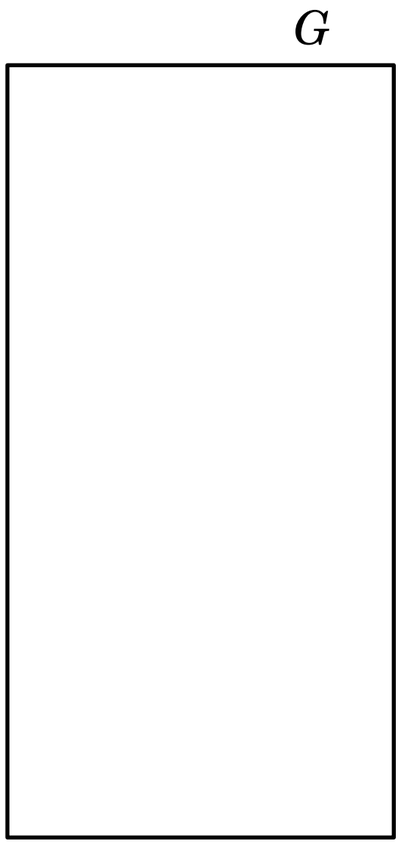


- $\varepsilon = 1/(10C_1 \log n) = (\lambda_i - \lambda_{i-1})/5$

- [Ghaffari and Grunau, FOCS '24]:
  $(C_1 \log n, C_2 \log n)$-colored decomposition

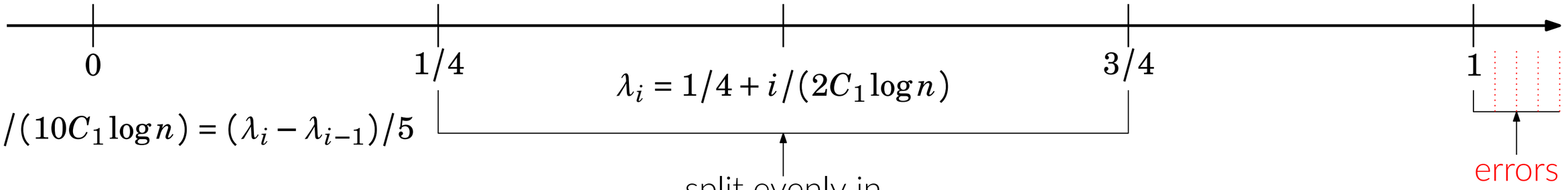
- in phase i, we will **flip improving sets** with IR ≥ λ_i *around* **class color** i

- Set $r_2 = O(\Delta^2 \log^2(n)/\varepsilon^2)$ (as in Property 2: *local witness of sequences*)



Algorithm's idea

- IR:



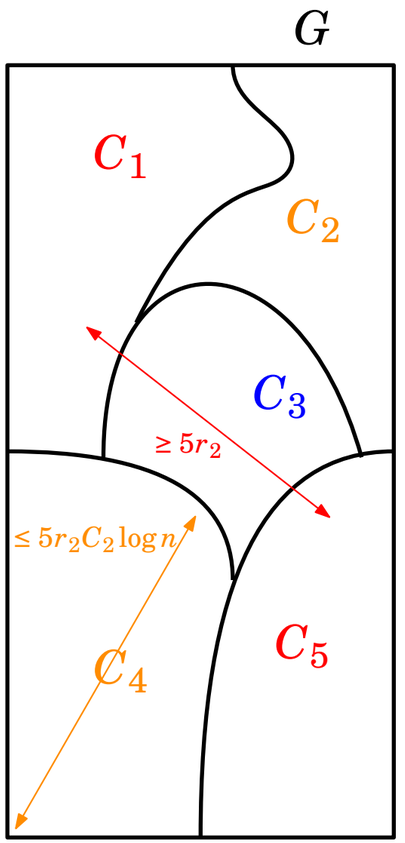
- $\varepsilon = 1/(10C_1 \log n) = (\lambda_i - \lambda_{i-1})/5$

- [Ghaffari and Grunau, FOCS '24]:
  $(C_1 \log n, C_2 \log n)$-colored decomposition

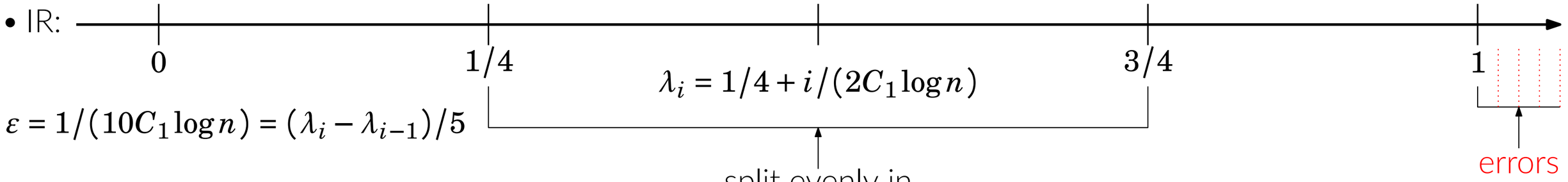
- in phase i, we will **flip improving sets** with IR $\geq \lambda_i$ *around* **class color** i

- Set $r_2 = O(\Delta^2 \log^2(n)/\varepsilon^2)$ (as in Property 2: *local witness of sequences*)

1. **Compute** $(C_1 \log n, C_2 \log n)$-colored decomposition of G^{5r_2}
   - clusters of same color have pairwise distance $\geq 5r_2$



Algorithm's idea



- $\varepsilon = 1/(10C_1 \log n) = (\lambda_i - \lambda_{i-1})/5$

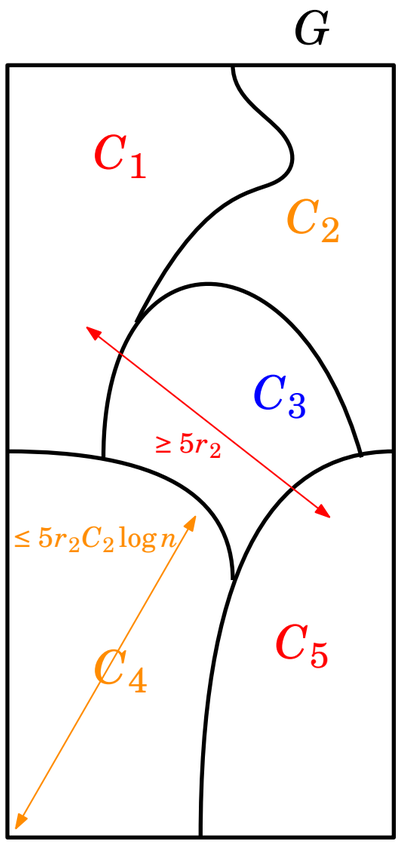
- [Ghaffari and Grunau, FOCS '24]:
  $(C_1 \log n, C_2 \log n)$-colored decomposition

- in phase i, we will **flip improving sets** with IR $\geq \lambda_i$ *around* **class color** i

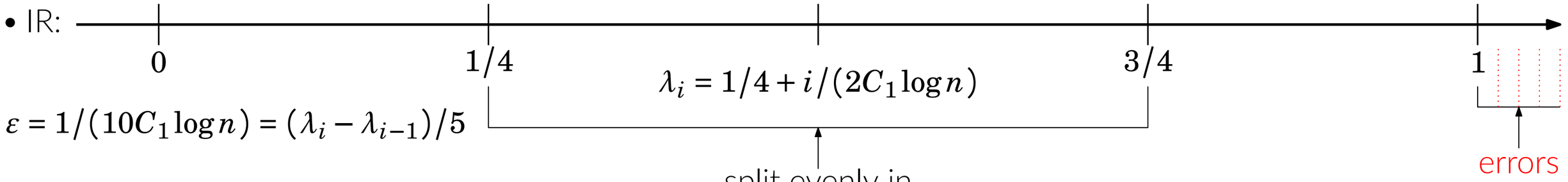
- Set $r_2 = O(\Delta^2 \log^2(n)/\varepsilon^2)$ (as in Property 2: *local witness of sequences*)

1. **Compute** $(C_1 \log n, C_2 \log n)$-colored decomposition of G^{5r_2}
   - clusters of same color have pairwise distance $\geq 5r_2$

- Set $r_1 = O(\Delta \log(n)/\varepsilon)$ (as in Property 1: *local witness of min. imp. set*)



Algorithm's idea



- $\varepsilon = 1/(10C_1 \log n) = (\lambda_i - \lambda_{i-1})/5$

- [Ghaffari and Grunau, FOCS '24]:
  $(C_1 \log n, C_2 \log n)$-colored decomposition

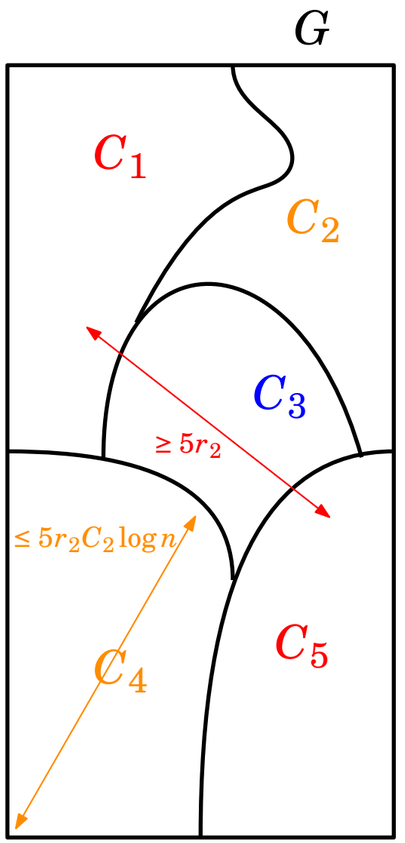
- in phase i, we will **flip improving sets** with IR $\geq \lambda_i$ *around* **class color** i

- Set $r_2 = O(\Delta^2 \log^2(n)/\varepsilon^2)$ (as in Property 2: *local witness of sequences*)

1. **Compute** $(C_1 \log n, C_2 \log n)$-colored decomposition of G^{5r_2}
   - clusters of same color have pairwise distance $\geq 5r_2$

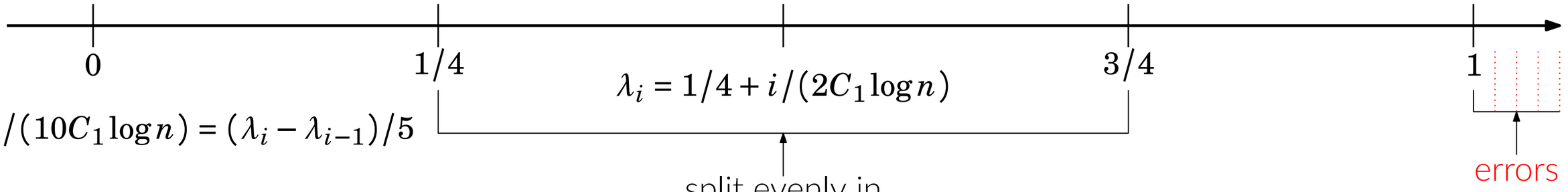
- Set $r_1 = O(\Delta \log(n)/\varepsilon)$ (as in Property 1: *local witness of min. imp. set*)

2. **Phase** i: **for all** clusters $\mathcal{C}$ of color i **do**
   - **flip** any *maximal sequence* of λ_i-improving sets of diameter $\leq r_1$ lying in $\mathcal{N}_{2r_2}[\mathcal{C}]$



Algorithm's idea

- IR:



- $\varepsilon = 1/(10C_1 \log n) = (\lambda_i - \lambda_{i-1})/5$

split evenly in $C_1 \log n$ values

- [Ghaffari and Grunau, FOCS '24]: $(C_1 \log n, C_2 \log n)$-colored decomposition
- in phase i, we will **flip improving sets** with IR $\geq \lambda_i$ *around* **class color** i

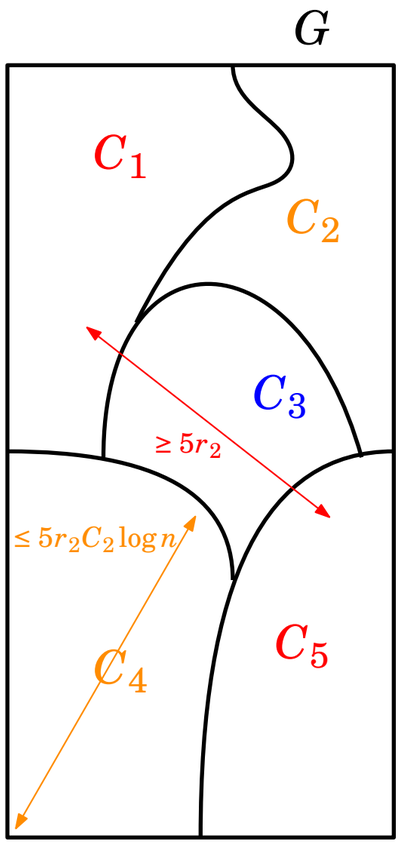
- Set $r_2 = O(\Delta^2 \log^2(n)/\varepsilon^2)$ (as in Property 2: *local witness of sequences*)

1. **Compute** $(C_1 \log n, C_2 \log n)$-colored decomposition of G^{5r_2}
   - clusters of same color have pairwise distance $\geq 5r_2$

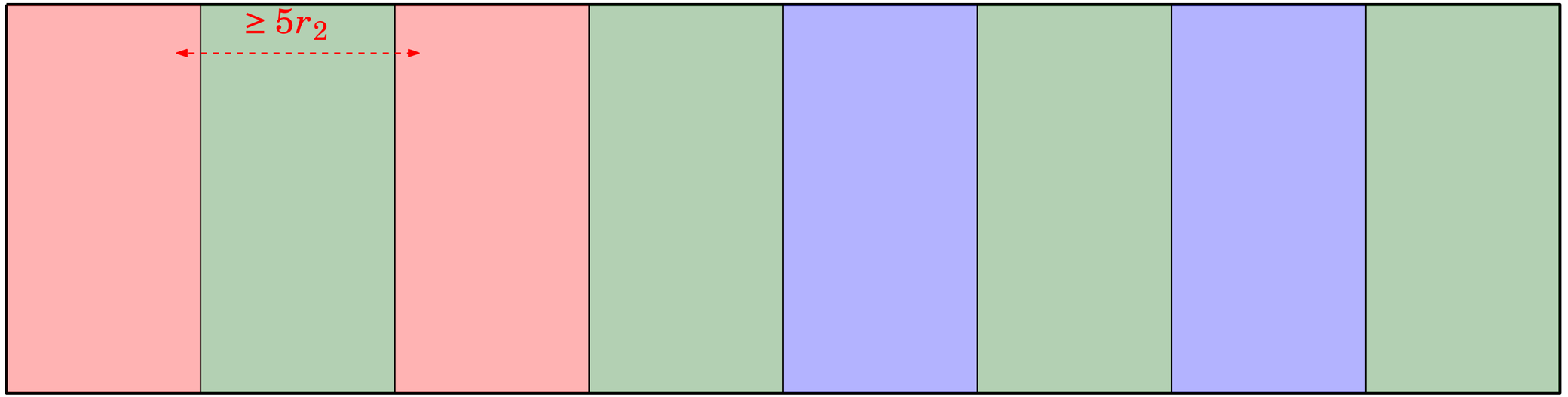
- Set $r_1 = O(\Delta \log(n)/\varepsilon)$ (as in Property 1: *local witness of min. imp. set*)

2. **Phase** i: **for all** clusters $\mathcal{C}$ of color i **do**
   - **flip** any *maximal sequence* of λ_i-improving sets of diameter $\leq r_1$ lying in $\mathcal{N}_{2r_2}[\mathcal{C}]$

1. + 2. $\Longrightarrow$ time $\Delta^2 \tilde{O}(\log^6 n)$



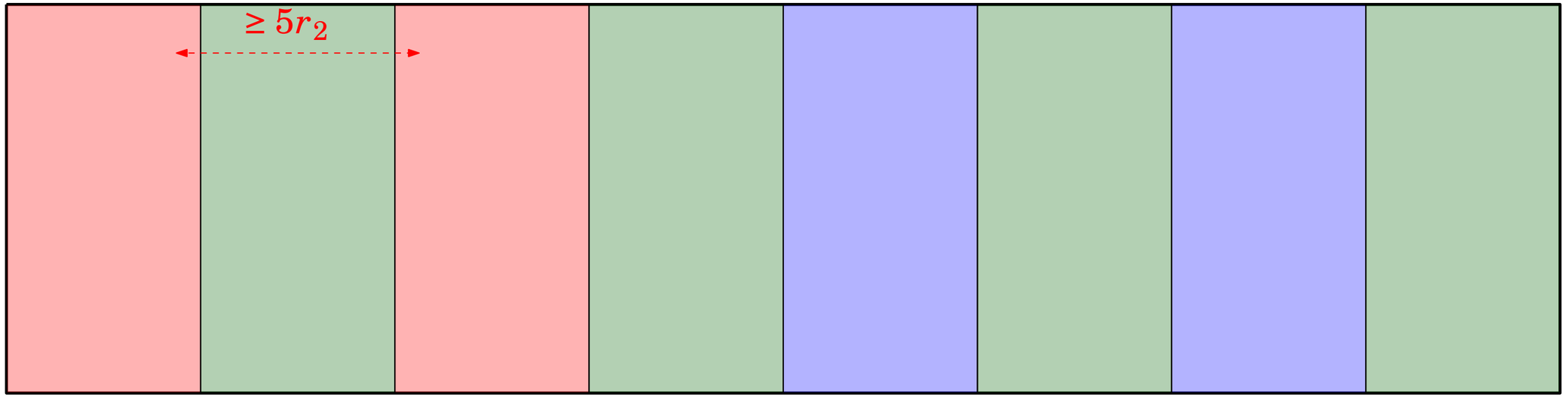
Algorithm's illustration



G

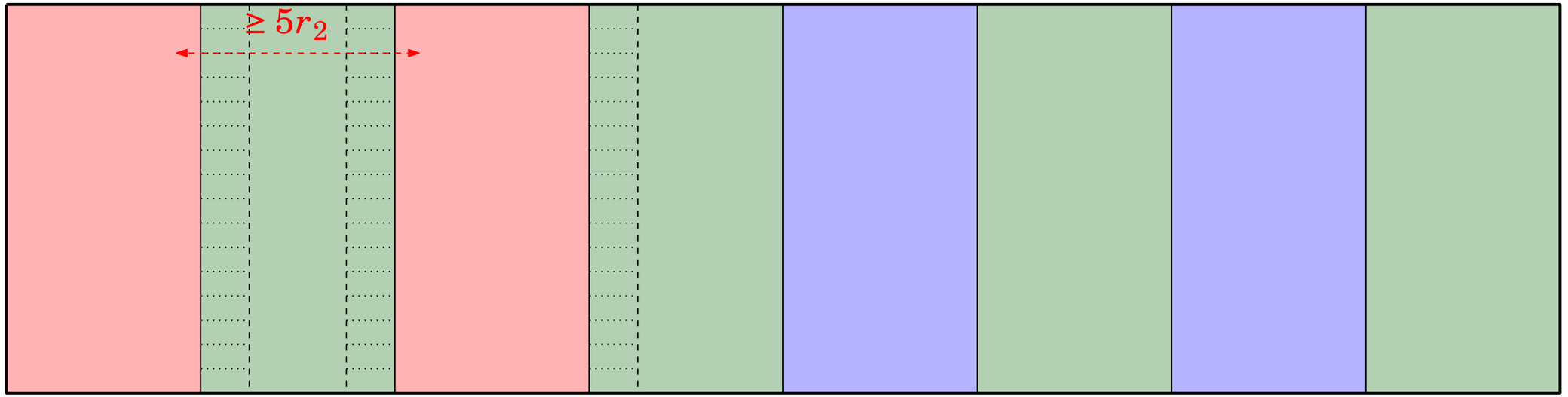
- 3 color classes $\Longrightarrow$ $\lambda_1 = 1/4$, $\lambda_2 = 1/2$, $\lambda_3 = 3/4$

Algorithm's illustration



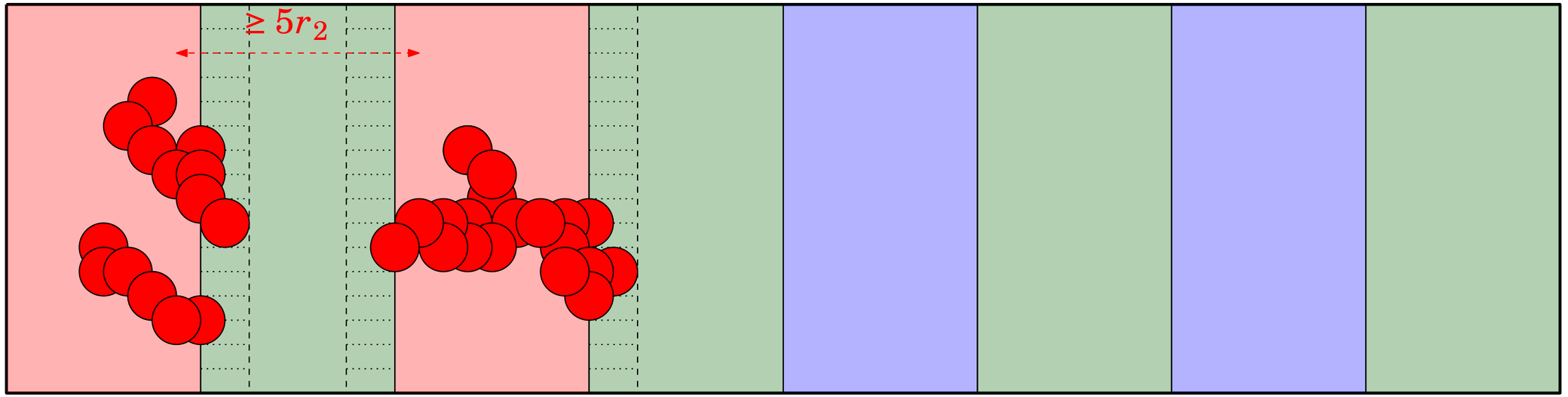
- 3 color classes $\implies$ $\lambda_1 = 1/4$, $\lambda_2 = 1/2$, $\lambda_3 = 3/4$
- **Phase 1**: for each red cluster $\mathcal{C}$ **do**
  - **flip** any *maximal sequence* of $1/4$-improving sets of diameter $\leq r_1$ lying in $\mathcal{N}_{2r_2}[\mathcal{C}]$

Algorithm's illustration



- 3 color classes $\implies$ $\lambda_1 = 1/4$, $\lambda_2 = 1/2$, $\lambda_3 = 3/4$
- **Phase 1**: for each red cluster $\mathcal{C}$ **do**
  - **flip** any *maximal sequence* of $1/4$-improving sets of diameter $\leq r_1$ lying in $\mathcal{N}_{2r_2}[\mathcal{C}]$

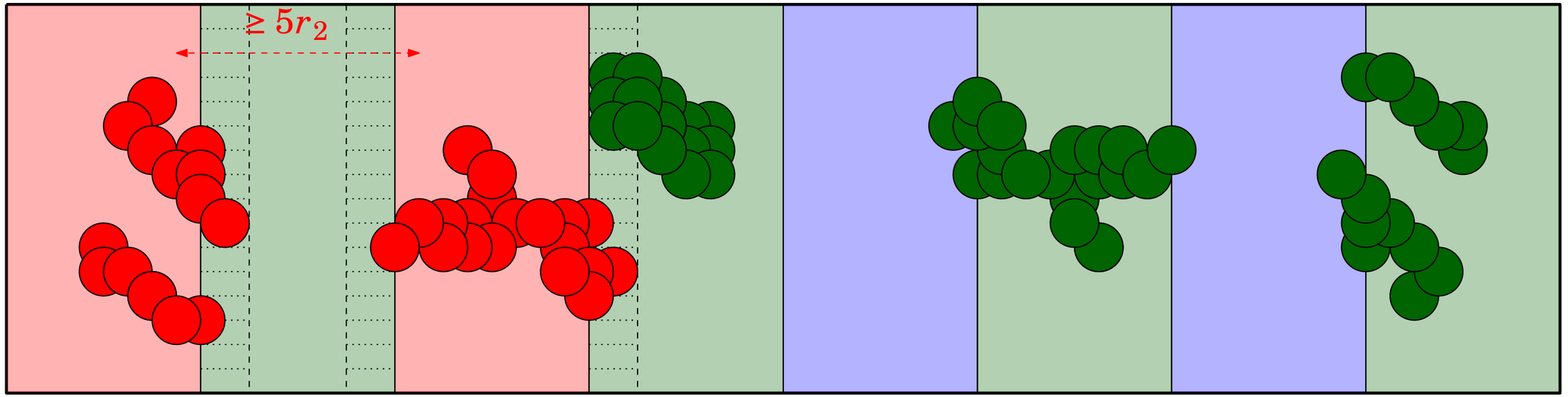
Algorithm's illustration



G

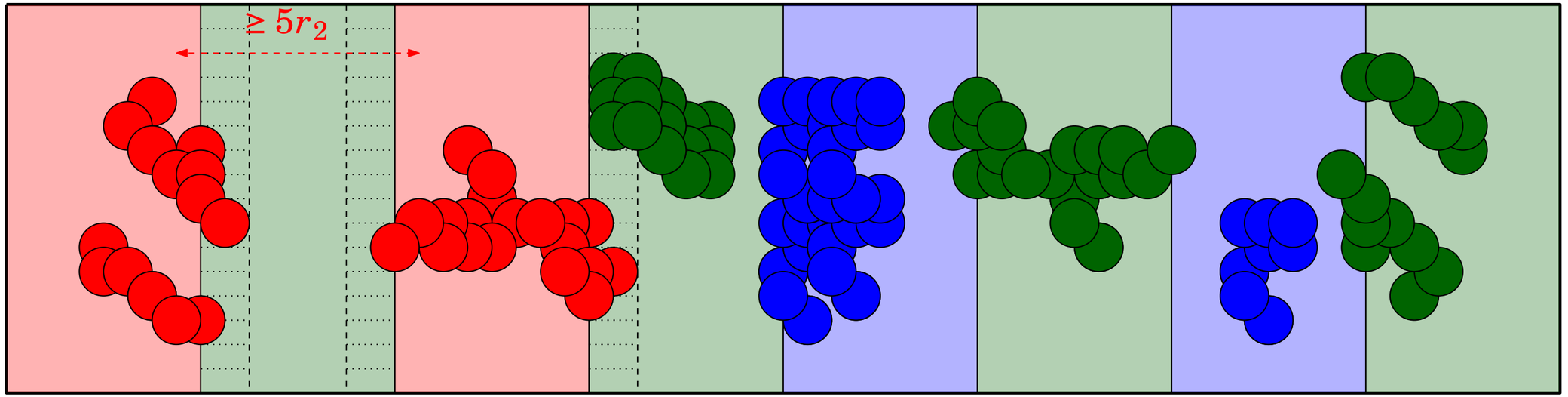
- 3 color classes $\implies$ $\lambda_1 = 1/4$, $\lambda_2 = 1/2$, $\lambda_3 = 3/4$
- **Phase 1**: for each red cluster $\mathcal{C}$ **do**
  - **flip** any *maximal sequence* of $1/4$-improving sets of diameter $\leq r_1$ lying in $\mathcal{N}_{2r_2}[\mathcal{C}]$

Algorithm's illustration



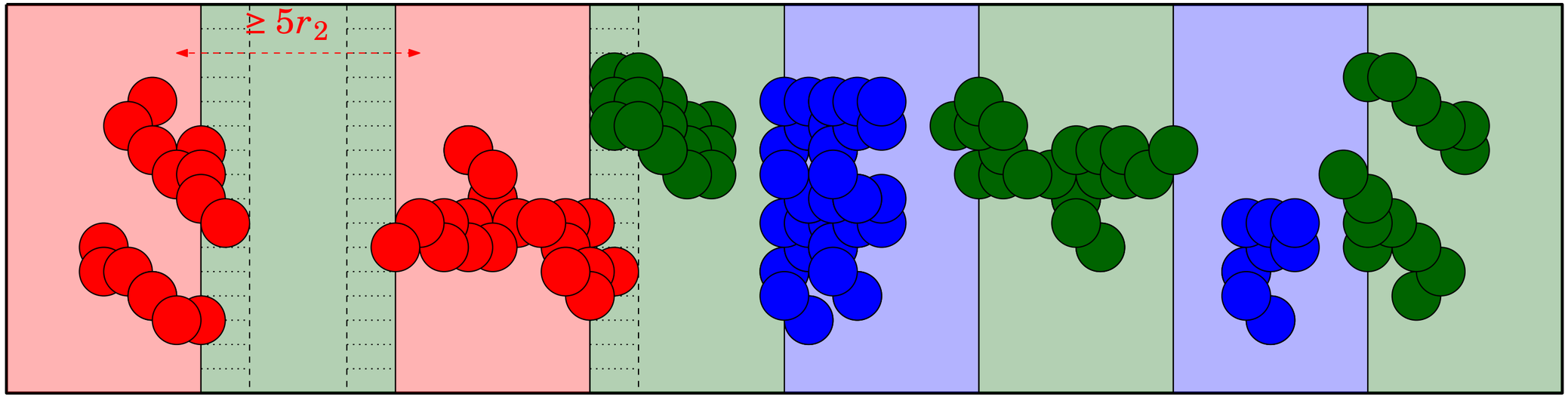
- 3 color classes $\implies$ $\lambda_1 = 1/4$, $\lambda_2 = 1/2$, $\lambda_3 = 3/4$
- **Phase 1**: for each red cluster $\mathcal{C}$ **do**
  - **flip** any *maximal sequence* of $1/4$-improving sets of diameter $\leq r_1$ lying in $\mathcal{N}_{2r_2}[\mathcal{C}]$
- **Phase 2**: for each green cluster $\mathcal{C}$ **do**
  - **flip** any *maximal sequence* of $1/2$-improving sets of diameter $\leq r_1$ lying in $\mathcal{N}_{2r_2}[\mathcal{C}]$

Algorithm's illustration



- 3 color classes $\implies$ $\lambda_1 = 1/4$, $\lambda_2 = 1/2$, $\lambda_3 = 3/4$
- **Phase 1**: for each red cluster $\mathcal{C}$ **do**
  - **flip** any *maximal sequence* of $1/4$-improving sets of diameter $\leq r_1$ lying in $\mathcal{N}_{2r_2}[\mathcal{C}]$
- **Phase 2**: for each green cluster $\mathcal{C}$ **do**
  - **flip** any *maximal sequence* of $1/2$-improving sets of diameter $\leq r_1$ lying in $\mathcal{N}_{2r_2}[\mathcal{C}]$
- **Phase 3**: for each blue cluster $\mathcal{C}$ **do**
  - **flip** any *maximal sequence* of $3/4$-improving sets of diameter $\leq r_1$ lying in $\mathcal{N}_{2r_2}[\mathcal{C}]$

Algorithm's illustration

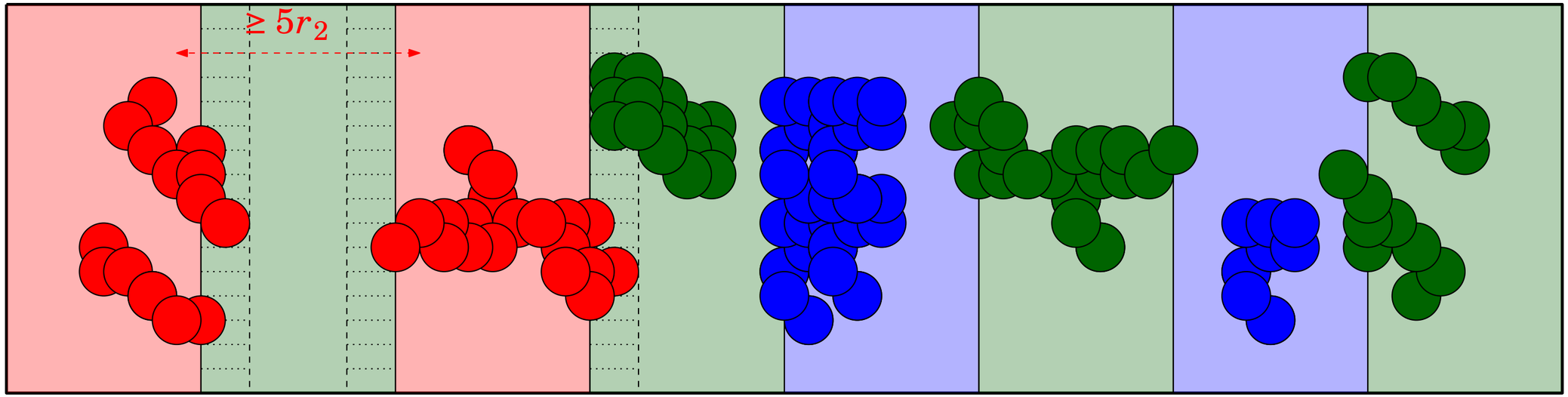


G

- 3 color classes $\implies$ $\lambda_1 = 1/4$, $\lambda_2 = 1/2$, $\lambda_3 = 3/4$
- **Phase 1**: for each red cluster $\mathcal{C}$ **do**
  - **flip** any *maximal sequence* of $1/4$-improving sets of diameter $\leq r_1$ lying in $\mathcal{N}_{2r_2}[\mathcal{C}]$
- **Phase 2**: for each green cluster $\mathcal{C}$ **do**
  - **flip** any *maximal sequence* of $1/2$-improving sets of diameter $\leq r_1$ lying in $\mathcal{N}_{2r_2}[\mathcal{C}]$
- **Phase 3**: for each blue cluster $\mathcal{C}$ **do**
  - **flip** any *maximal sequence* of $3/4$-improving sets of diameter $\leq r_1$ lying in $\mathcal{N}_{2r_2}[\mathcal{C}]$

→ We can define unique sequence of improving sets σ

Algorithm's illustration

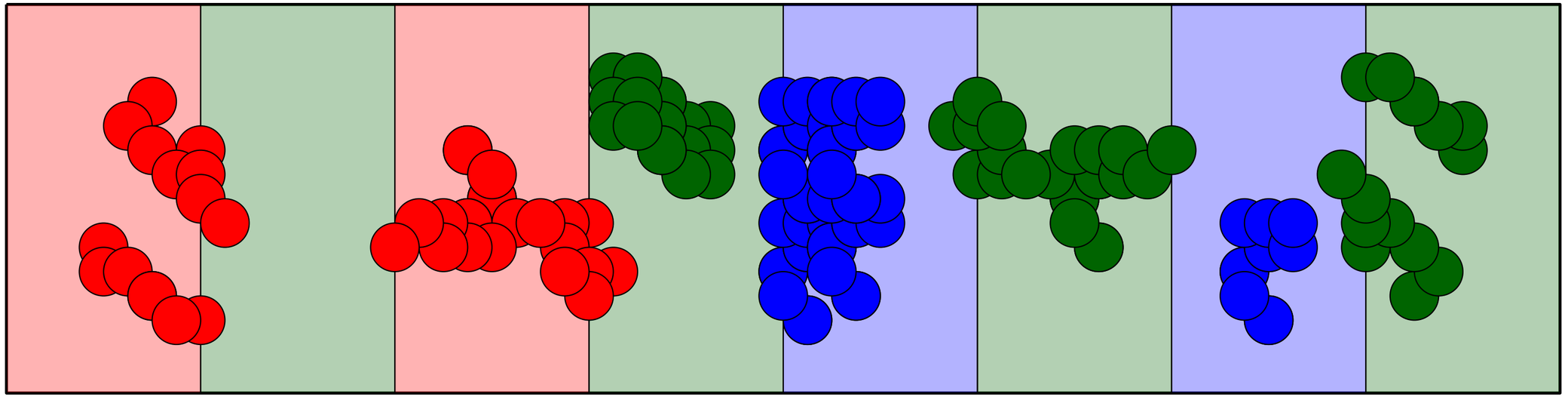


- 3 color classes $\implies$ $\lambda_1 = 1/4$, $\lambda_2 = 1/2$, $\lambda_3 = 3/4$
- **Phase 1**: for each red cluster $\mathcal{C}$ **do**
  - **flip** any *maximal sequence* of $1/4$-improving sets of diameter $\leq r_1$ lying in $\mathcal{N}_{2r_2}[\mathcal{C}]$
- **Phase 2**: for each green cluster $\mathcal{C}$ **do**
  - **flip** any *maximal sequence* of $1/2$-improving sets of diameter $\leq r_1$ lying in $\mathcal{N}_{2r_2}[\mathcal{C}]$
- **Phase 3**: for each blue cluster $\mathcal{C}$ **do**
  - **flip** any *maximal sequence* of $3/4$-improving sets of diameter $\leq r_1$ lying in $\mathcal{N}_{2r_2}[\mathcal{C}]$

→ We can define unique sequence of improving sets σ

- σ_1 for red clusters
- σ_2 for green clusters
- σ_3 for blue clusters
- $\sigma = \sigma_1\sigma_2\sigma_3$

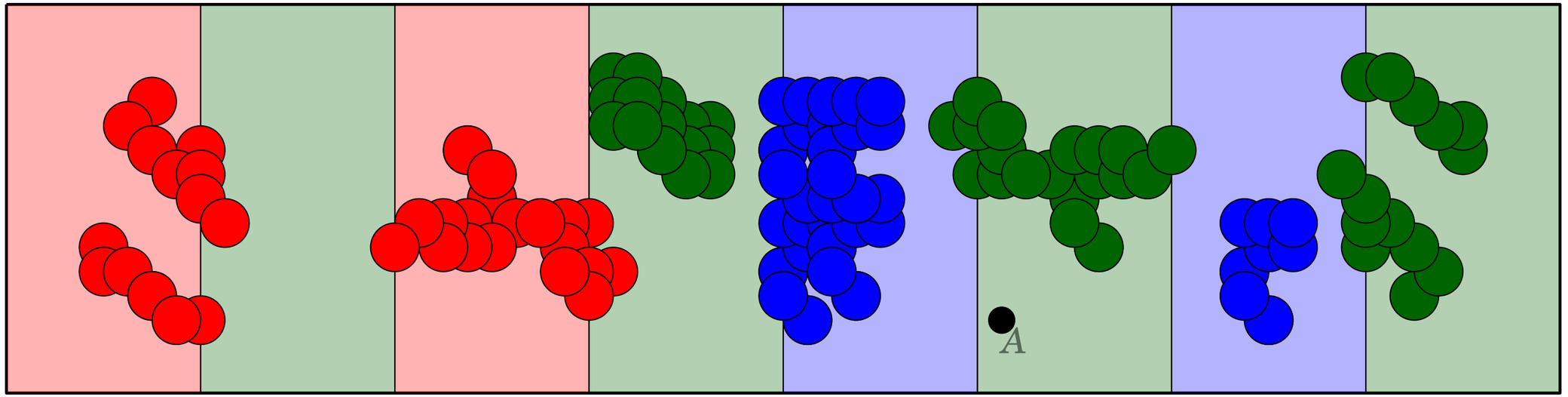
Algorithm's correctness



G

- σ_1 for red clusters (IR $\geq 1/4$)
- σ_2 for green clusters (IR $\geq 1/2$)
- σ_3 for blue clusters (IR $\geq 3/4$)
- $\sigma = \sigma_1\sigma_2\sigma_3$

Algorithm's correctness

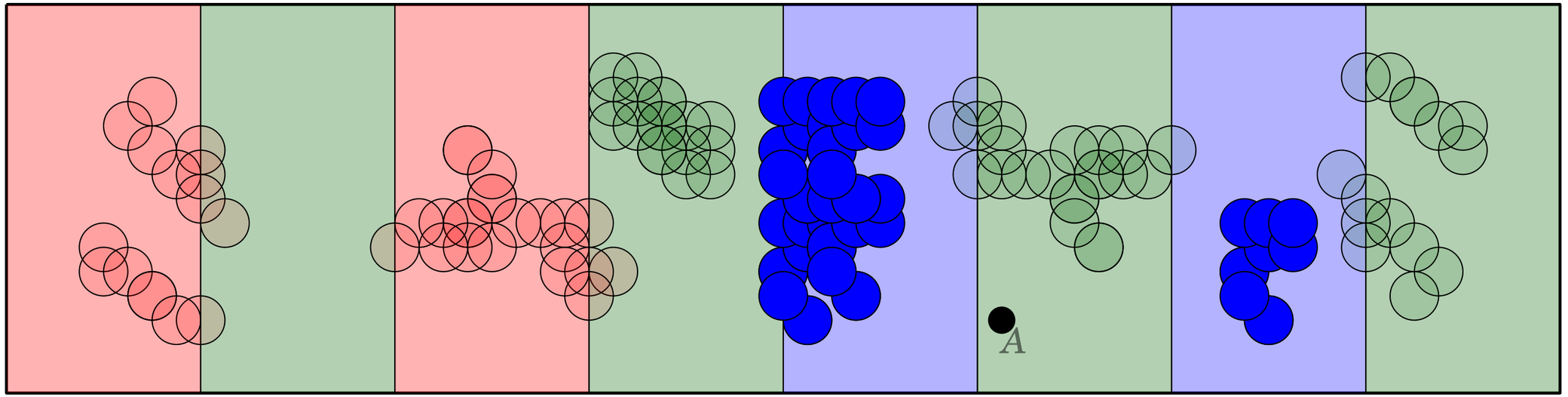


G

- σ_1 for red clusters (IR $\geq 1/4$)
- σ_2 for green clusters (IR $\geq 1/2$)
- σ_3 for blue clusters (IR $\geq 3/4$)
- $\sigma = \sigma_1\sigma_2\sigma_3$

- suppose **error** in (wlog) green $\implies$ imp. set A with IR ≥ 1 of size 1

Algorithm's correctness

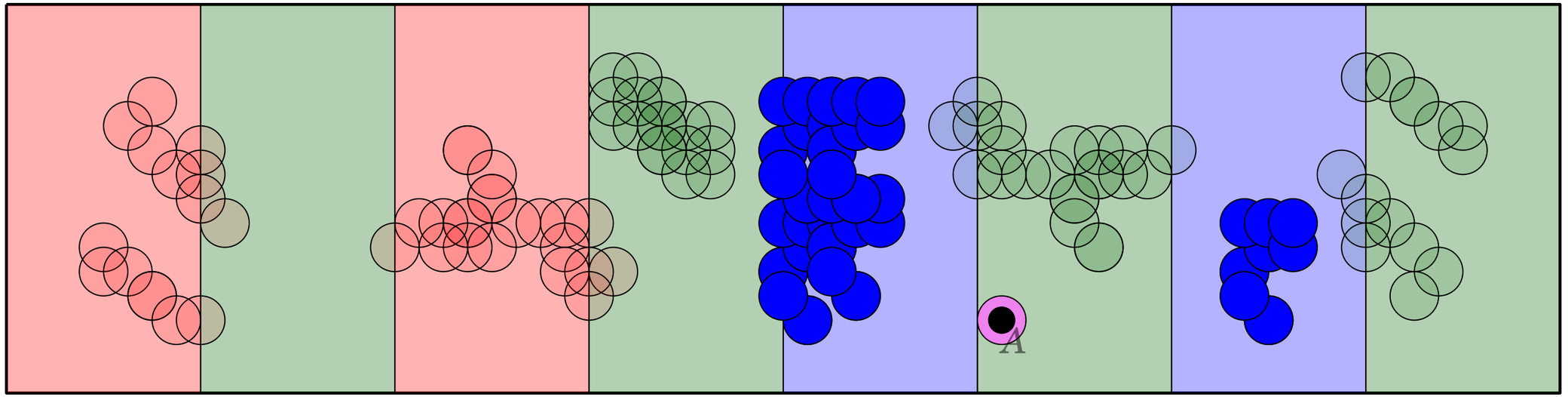


- σ_1 for red clusters (IR $\geq 1/4$)
- σ_2 for green clusters (IR $\geq 1/2$)
- σ_3 for blue clusters (IR $\geq 3/4$)
- $\sigma = \sigma_1\sigma_2\sigma_3$

- suppose **error** in (wlog) green $\implies$ imp. set A with IR ≥ 1 of size 1

- $\sigma_3 + A$ sequence of $3/4$-improving sets of diam. $\leq r_1$ (w.r.t. $\sigma_1 + \sigma_2$)

Algorithm's correctness



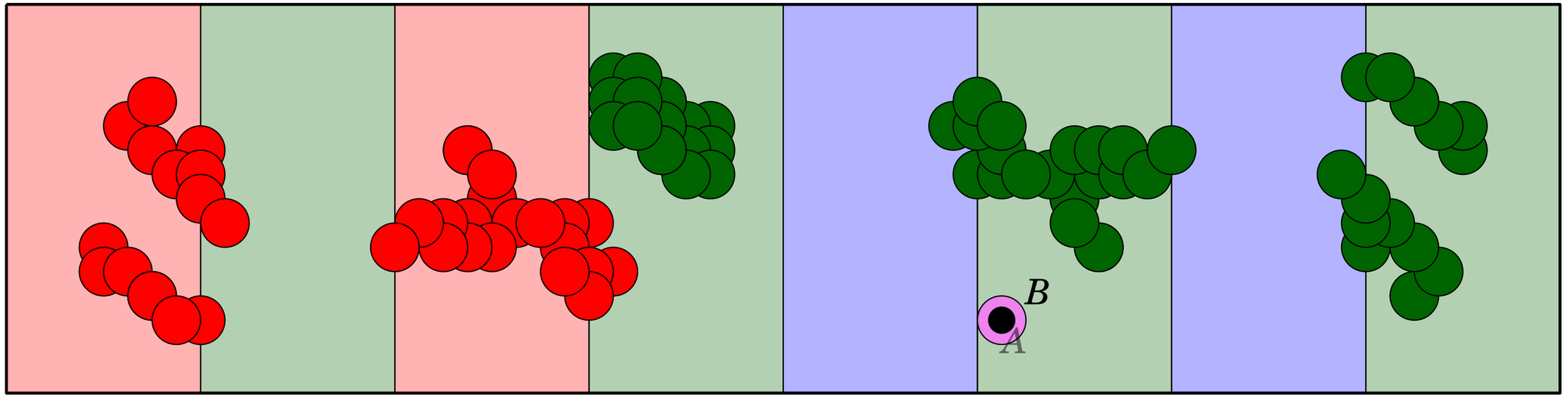
- σ_1 for red clusters (IR $\geq 1/4$)
- σ_2 for green clusters (IR $\geq 1/2$)
- σ_3 for blue clusters (IR $\geq 3/4$)
- $\sigma = \sigma_1\sigma_2\sigma_3$

local witness (properties 1 and 2)

- suppose **error** in (wlog) green $\implies$ imp. set A with IR ≥ 1 of size 1

- $\sigma_3 + A$ sequence of $3/4$-improving sets of diam. $\leq r_1$ (w.r.t. $\sigma_1 + \sigma_2$) $\implies$ B of diam. $\leq r_1$ in $\mathcal{N}_{r_2}[A]$ with IR $\geq 3/4 - \varepsilon$

Algorithm's correctness



- σ_1 for red clusters (IR $\geq 1/4$)
- σ_2 for green clusters (IR $\geq 1/2$)
- σ_3 for blue clusters (IR $\geq 3/4$)
- $\sigma = \sigma_1\sigma_2\sigma_3$

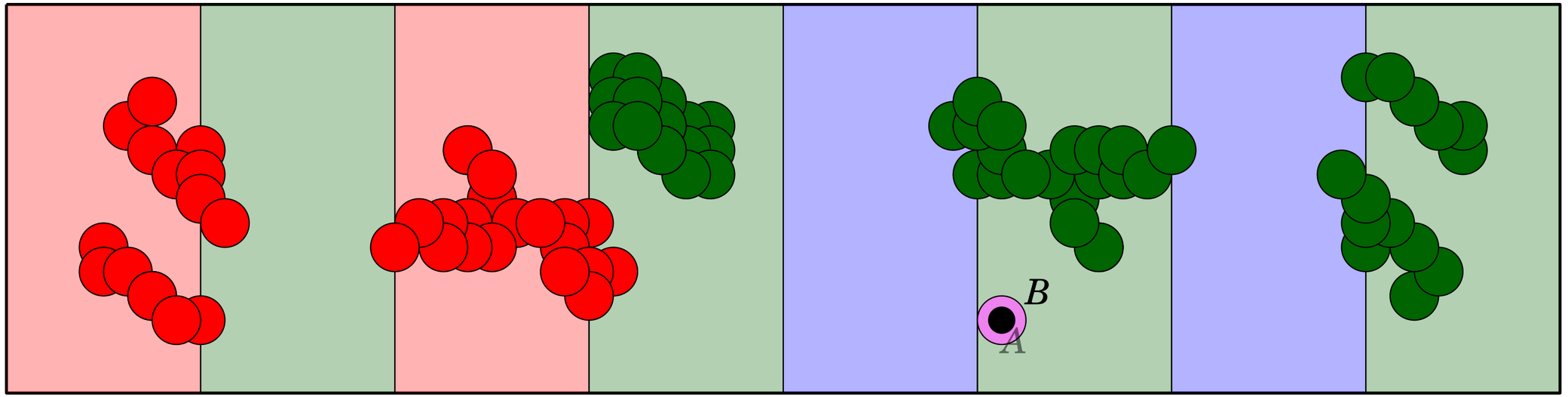
local witness (properties 1 and 2)

- suppose **error** in (wlog) green $\implies$ imp. set A with IR ≥ 1 of size 1

- $\sigma_3 + A$ sequence of $3/4$-improving sets of diam. $\leq r_1$ (w.r.t. $\sigma_1 + \sigma_2$) $\implies$ B of diam. $\leq r_1$ in $\mathcal{N}_{r_2}[A]$ with IR $\geq 3/4 - \varepsilon$

- $\sigma_2 + B$ still sequence of $1/2$-improving sets of diam. $\leq r_1$ (w.r.t. σ_1)

Algorithm's correctness



- σ_1 for red clusters (IR $\geq 1/4$)
- σ_2 for green clusters (IR $\geq 1/2$)
- σ_3 for blue clusters (IR $\geq 3/4$)
- $\sigma = \sigma_1\sigma_2\sigma_3$

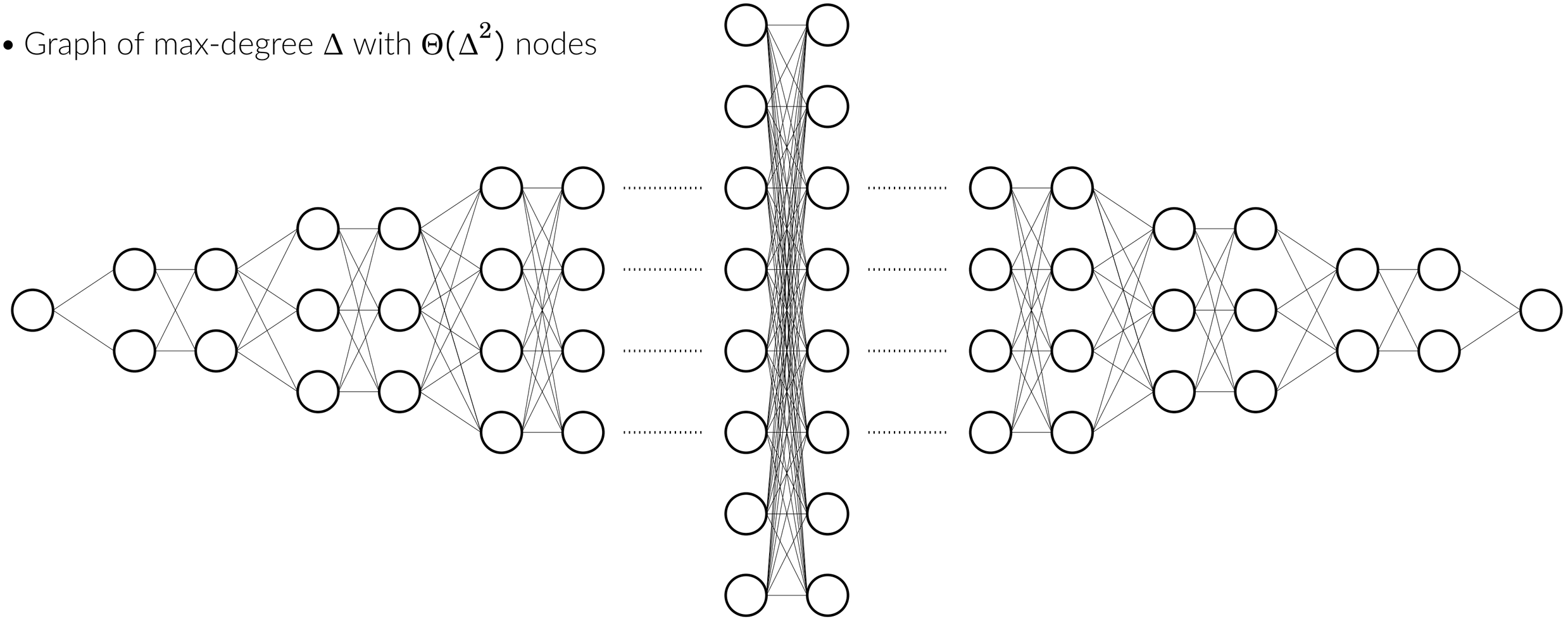
local witness (properties 1 and 2)

- suppose **error** in (wlog) green $\implies$ imp. set A with IR ≥ 1 of size 1
- $\sigma_3 + A$ sequence of $3/4$-improving sets of diam. $\leq r_1$ (w.r.t. $\sigma_1 + \sigma_2$) $\implies$ B of diam. $\leq r_1$ in $\mathcal{N}_{r_2}[A]$ with IR $\geq 3/4 - \varepsilon$
- $\sigma_2 + B$ still sequence of $1/2$-improving sets of diam. $\leq r_1$ (w.r.t. σ_1) $\implies$ σ_2 not maximal (*contradiction*)

Lower bound

Idea: construct a graph that *forces* propagation

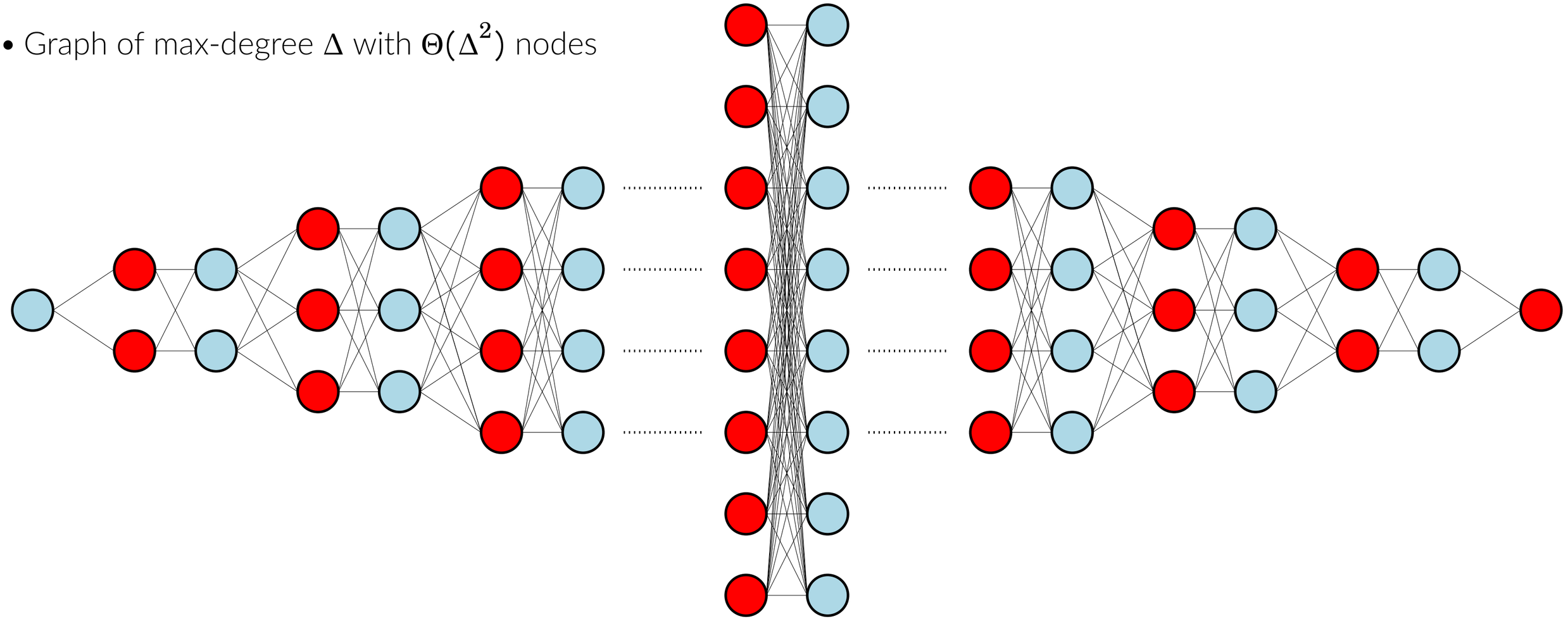
- Graph of max-degree Δ with $\Theta(\Delta^2)$ nodes



Lower bound

Idea: construct a graph that *forces* propagation

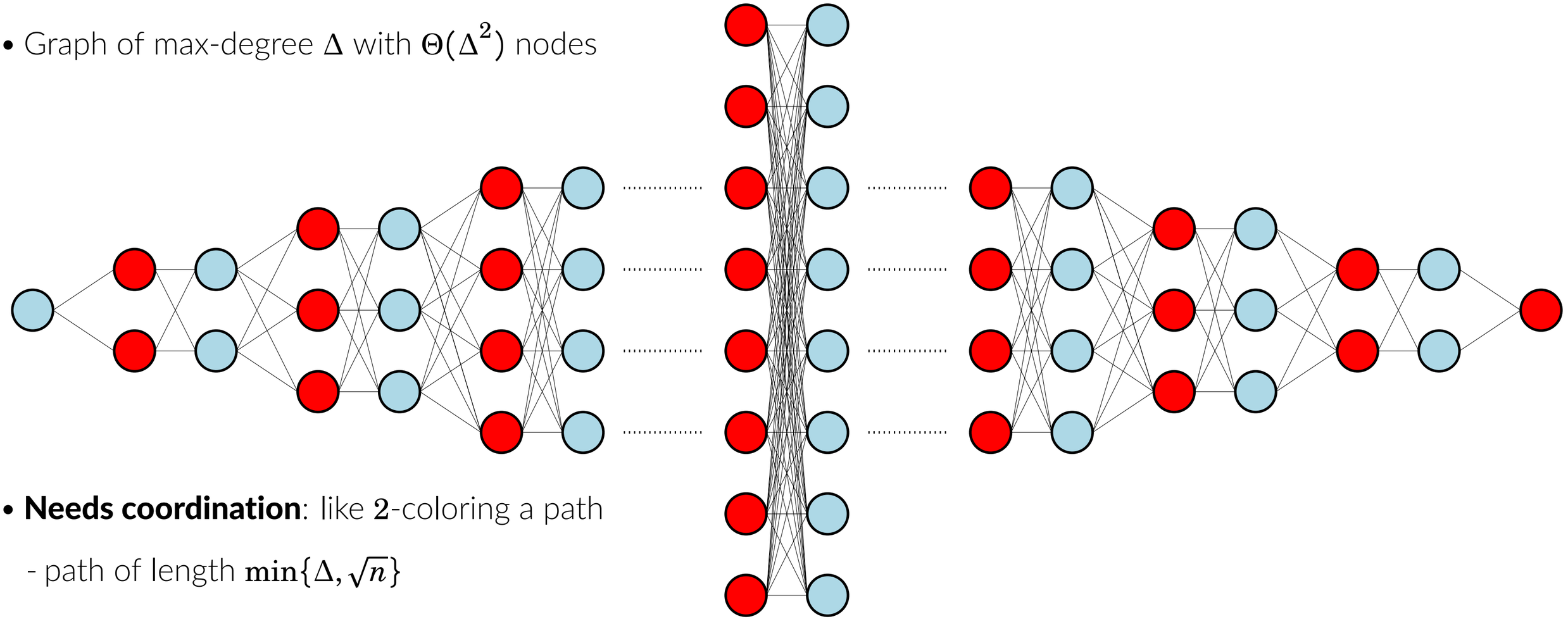
- Graph of max-degree Δ with $\Theta(\Delta^2)$ nodes



Lower bound

Idea: construct a graph that *forces* propagation

- Graph of max-degree Δ with $\Theta(\Delta^2)$ nodes



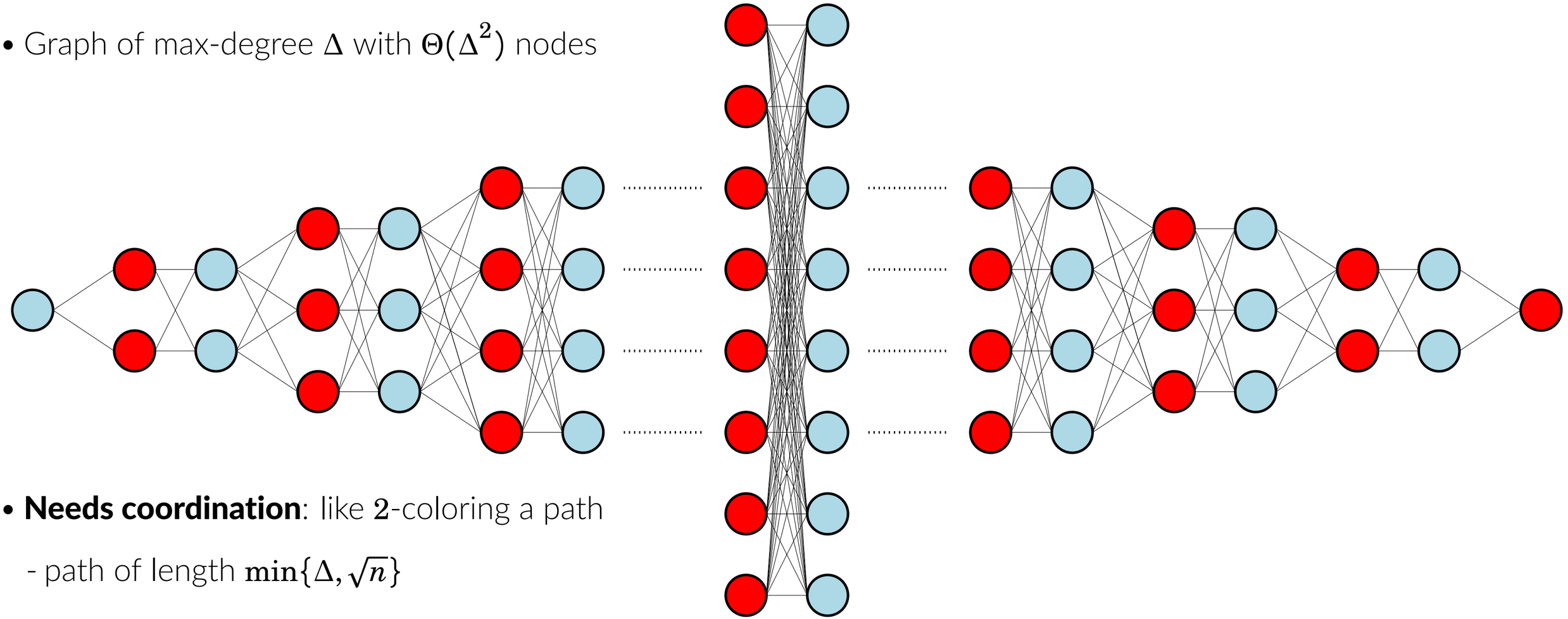
- **Needs coordination**: like 2-coloring a path

  - path of length $\min\{\Delta, \sqrt{n}\}$

Lower bound

Idea: construct a graph that *forces* propagation

- Graph of max-degree Δ with $\Theta(\Delta^2)$ nodes



- **Needs coordination**: like 2-coloring a path

  - path of length $\min\{\Delta, \sqrt{n}\}$

- Lower bound $\Omega(\min\{\Delta, \sqrt{n}\})$

Local potential problems

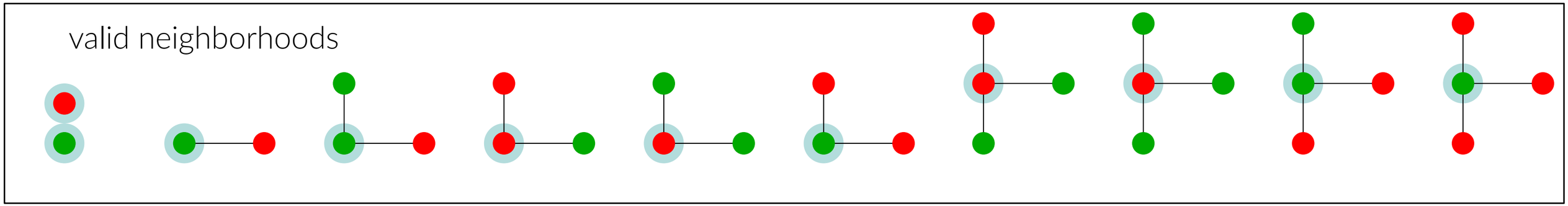
- $r > 0$, $\Delta > 0$
- $\mathcal{C}$ list of valid neighborhoods of radius r and max degree Δ

Local potential problems

- $r > 0$, $\Delta > 0$
- $\mathcal{C}$ list of valid neighborhoods of radius r and max degree Δ

locally optimal cut with

$\Delta = 3$ (r = 1)

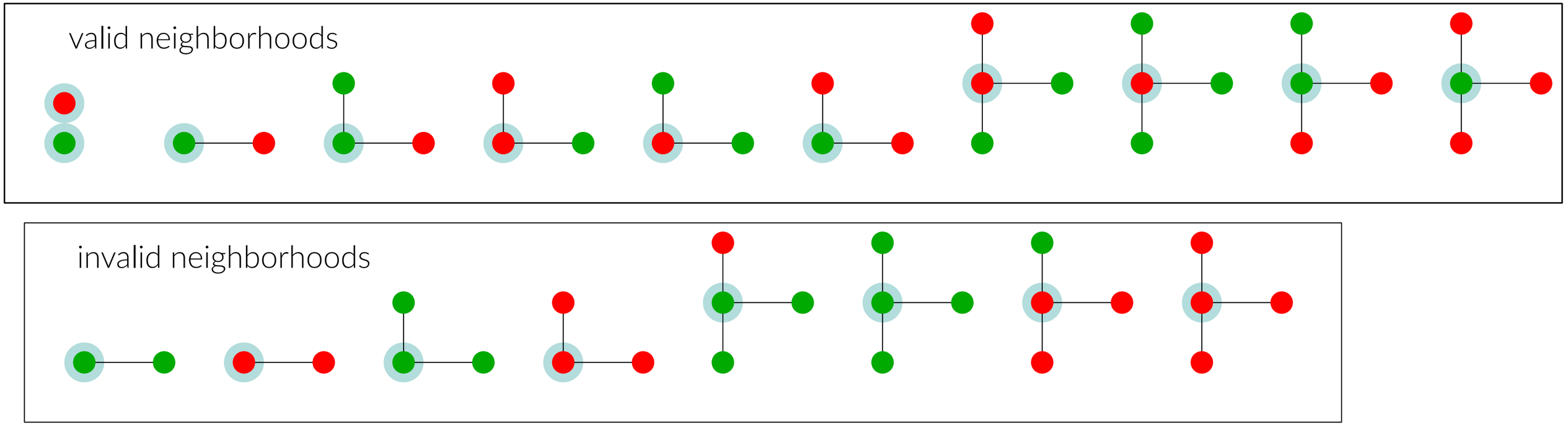


Local potential problems

- $r > 0$, $\Delta > 0$
- $\mathcal{C}$ list of valid neighborhoods of radius r and max degree Δ

locally optimal cut with

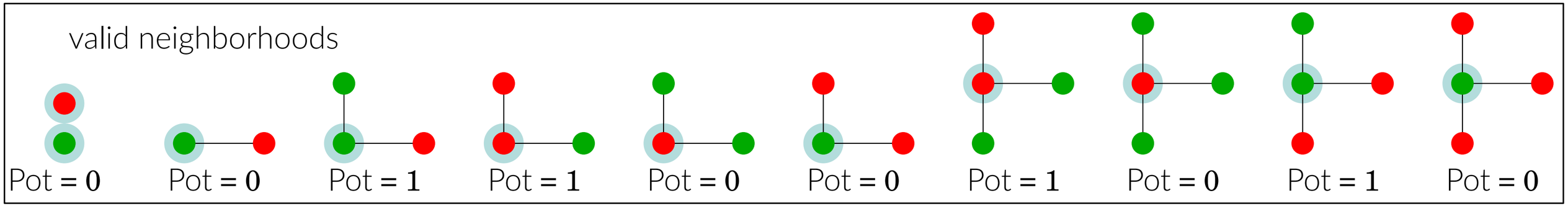
$\Delta = 3$ (r = 1)

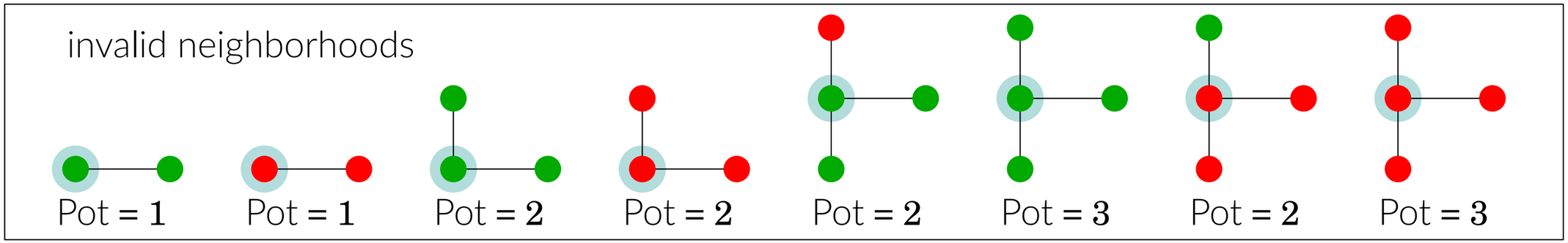


Local potential problems

- $r > 0$, $\Delta > 0$
- $\mathcal{C}$ list of valid neighborhoods of radius r and max degree Δ
- Pot : $\mathcal{G}_{r,\Delta} \to \mathbb{R}_{\geq 0}$ assigns a potential to every labeled neighborhood of radius r and max degree Δ

locally optimal cut with $\Delta = 3$ (r = 1)

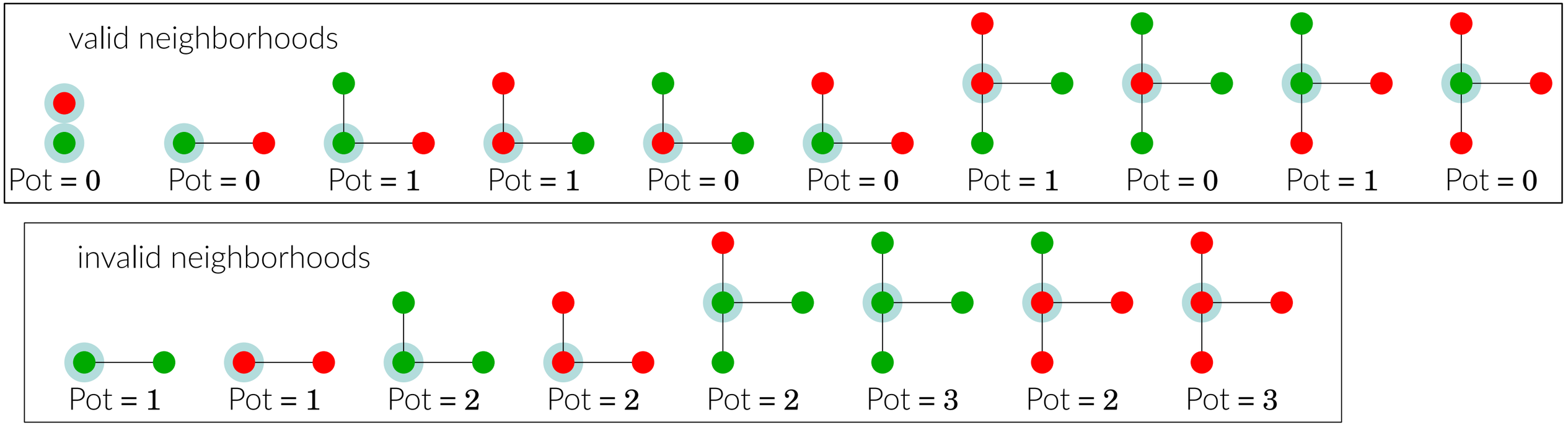




Local potential problems

- $r > 0$, $\Delta > 0$
- $\mathcal{C}$ list of valid neighborhoods of radius r and max degree Δ
- $\text{Pot} : \mathcal{G}_{r,\Delta} \to \mathbb{R}_{\geq 0}$ assigns a potential to every labeled neighborhood of radius r and max degree Δ
- For every *invalid neighborhood*, there is a way to relabel the central node to decrease the local (thus, *global*) potential

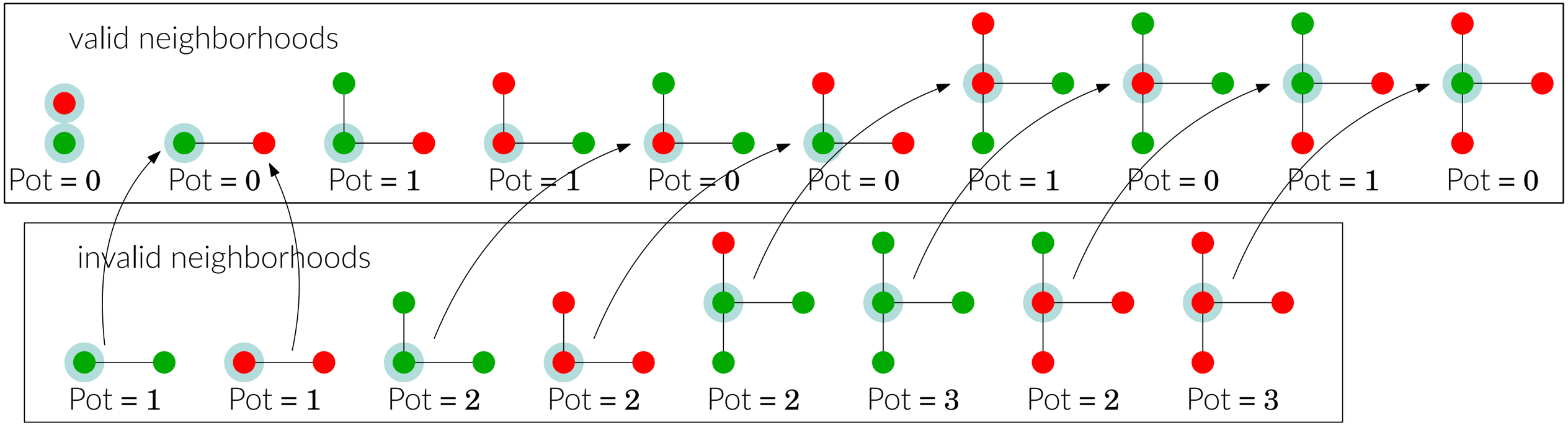
locally optimal cut with $\Delta = 3$ (r = 1)



Local potential problems

- $r > 0$, $\Delta > 0$
- $\mathcal{C}$ list of valid neighborhoods of radius r and max degree Δ
- $\text{Pot} : \mathcal{G}_{r,\Delta} \to \mathbb{R}_{\geq 0}$ assigns a potential to every labeled neighborhood of radius r and max degree Δ
- For every *invalid neighborhood*, there is a way to relabel the central node to decrease the local (thus, *global*) potential

locally optimal cut with $\Delta = 3$ (r = 1)



Recap & conclusions

Theorem:

- *There exists a determinstc LOCAL algorithm that solves Locally Optimal Cut in time* $\Delta^2 \cdot \tilde{O}(\log^6 n)$.

Recap & conclusions

Theorem:

- *There exists a determinstc LOCAL algorithm that solves Locally Optimal Cut in time* $\Delta^2 \cdot \tilde{O}(\log^6 n)$.
- *For any local potential problem* Π, *there exists a determinstc LOCAL algorithm that solves* Π *in time* $\mathbf{poly}(\Delta) \cdot \tilde{O}(\log^6 n)$.

Recap & conclusions

Theorem:

- *There exists a deterministc LOCAL algorithm that solves Locally Optimal Cut in time* $\Delta^2 \cdot \tilde{O}(\log^6 n)$.
- *For any local potential problem* Π, *there exists a deterministc LOCAL algorithm that solves* Π *in time* $\mathbf{poly}(\Delta) \cdot \tilde{O}(\log^6 n)$.

Lower bound: - $\Omega(\min\{\Delta, \sqrt{n}\})$-rounds (even in quantum or non-signaling models)

- **bounded-degree**: $\Omega(\log\log n)$ randomized rounds and $\Omega(\log n)$ deterministic rounds

Recap & conclusions

Theorem:

- *There exists a deterministc LOCAL algorithm that solves Locally Optimal Cut in time* $\Delta^2 \cdot \tilde{O}(\log^6 n)$.
- *For any local potential problem* Π, *there exists a deterministc LOCAL algorithm that solves* Π *in time* $\mathbf{poly}(\Delta) \cdot \tilde{O}(\log^6 n)$.

Lower bound: - $\Omega(\min\{\Delta, \sqrt{n}\})$-rounds (even in quantum or non-signaling models)

- **bounded-degree**: $\Omega(\log\log n)$ randomized rounds and $\Omega(\log n)$ deterministic rounds

Questions: - Deterministic and randomized LCL complexity? Polynomial and exponential gaps, resp.

Recap & conclusions

Theorem:

- *There exists a deterministc LOCAL algorithm that solves Locally Optimal Cut in time* $\Delta^2 \cdot \tilde{O}(\log^6 n)$.
- *For any local potential problem* Π, *there exists a deterministc LOCAL algorithm that solves* Π *in time* $\mathbf{poly}(\Delta) \cdot \tilde{O}(\log^6 n)$.

Lower bound: - $\Omega(\min\{\Delta, \sqrt{n}\})$-rounds (even in quantum or non-signaling models)

- **bounded-degree**: $\Omega(\log\log n)$ randomized rounds and $\Omega(\log n)$ deterministic rounds

Questions: - Deterministic and randomized LCL complexity? Polynomial and exponential gaps, resp.

- Connections with LLL?

Recap & conclusions

Theorem:

- *There exists a deterministc LOCAL algorithm that solves Locally Optimal Cut in time* $\Delta^2 \cdot \tilde{O}(\log^6 n)$.
- *For any local potential problem* Π, *there exists a deterministc LOCAL algorithm that solves* Π *in time* $\mathbf{poly}(\Delta) \cdot \tilde{O}(\log^6 n)$.

Lower bound: - $\Omega(\min\{\Delta, \sqrt{n}\})$-rounds (even in quantum or non-signaling models)

- **bounded-degree**: $\Omega(\log\log n)$ randomized rounds and $\Omega(\log n)$ deterministic rounds

Questions: - Deterministic and randomized LCL complexity? Polynomial and exponential gaps, resp.

- Connections with LLL?

- Right dependence in Δ?

Recap & conclusions

Theorem:

- *There exists a deterministc LOCAL algorithm that solves Locally Optimal Cut in time* $\Delta^2 \cdot \tilde{O}(\log^6 n)$.
- *For any local potential problem* Π, *there exists a deterministc LOCAL algorithm that solves* Π *in time* $\mathbf{poly}(\Delta) \cdot \tilde{O}(\log^6 n)$.

Lower bound: - $\Omega(\min\{\Delta, \sqrt{n}\})$-rounds (even in quantum or non-signaling models)

- **bounded-degree**: $\Omega(\log\log n)$ randomized rounds and $\Omega(\log n)$ deterministic rounds

Questions: - Deterministic and randomized LCL complexity? Polynomial and exponential gaps, resp.

- Connections with LLL?

- Right dependence in Δ?

- Complexity in CONGEST? We require gathering-type subroutines

Recap & conclusions

Theorem:

- *There exists a deterministc LOCAL algorithm that solves Locally Optimal Cut in time* $\Delta^2 \cdot \tilde{O}(\log^6 n)$.
- *For any local potential problem* Π, *there exists a deterministc LOCAL algorithm that solves* Π *in time* $\mathbf{poly}(\Delta) \cdot \tilde{O}(\log^6 n)$.

Lower bound: - $\Omega(\min\{\Delta, \sqrt{n}\})$-rounds (even in quantum or non-signaling models)

- **bounded-degree**: $\Omega(\log\log n)$ randomized rounds and $\Omega(\log n)$ deterministic rounds

Questions: - Deterministic and randomized LCL complexity? Polynomial and exponential gaps, resp.

- Connections with LLL?

- Right dependence in Δ?

- Complexity in CONGEST? We require gathering-type subroutines

THANKS