Networked Computing

Distributed Graph Algorithms

Outline

- Intro
  - What do we try to do
  - What this is not
- Coloring
  - $\Delta + 1$ colors
- The LOCAL model
- Coloring a ring
  - Color reduction
  - Lower bound

Outline

- Intro
  - What do we try to do
  - What this is not
- Coloring
  - $\Delta + 1$ colors
- The LOCAL model
- Coloring a ring
  - Color reduction
  - Lower bound

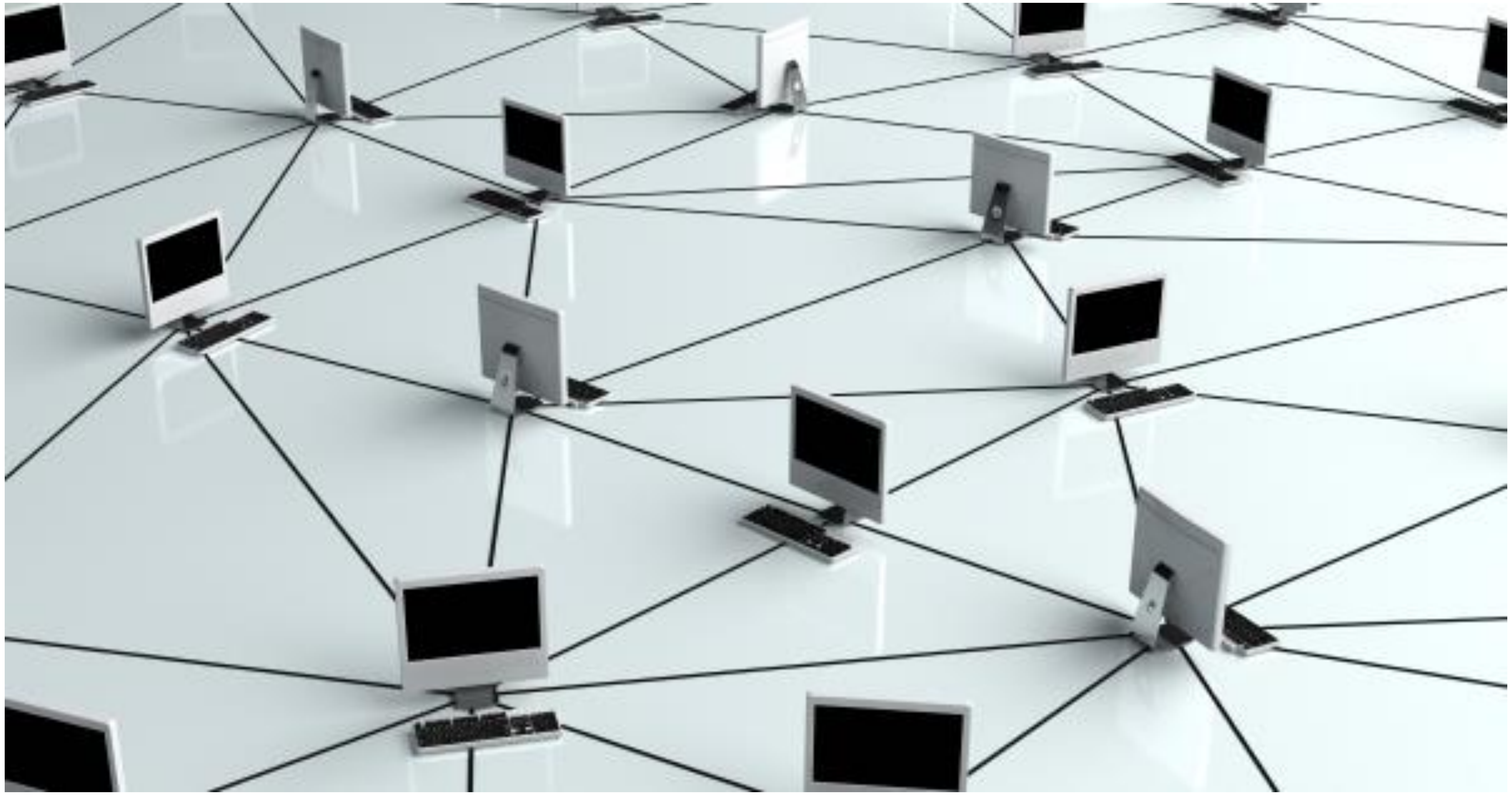
Learning objectives:
You are able to
- describe the LOCAL model of distributed computing
- describe the color reduction algorithm
- analyse the distributed complexity of 2-coloring a ring graph

Message Passing

The Internet

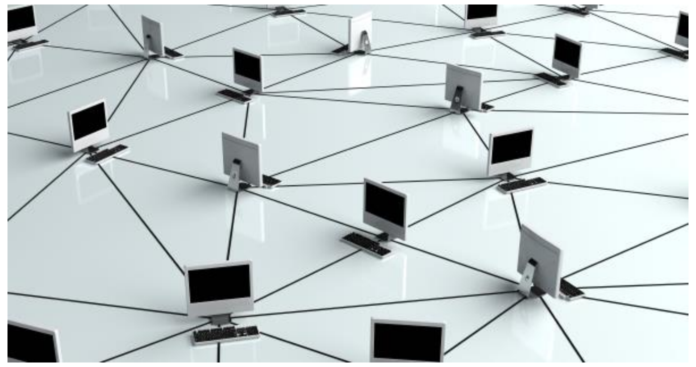
Sensor Networks

Message Passing on a Graph



Message Passing on a Graph

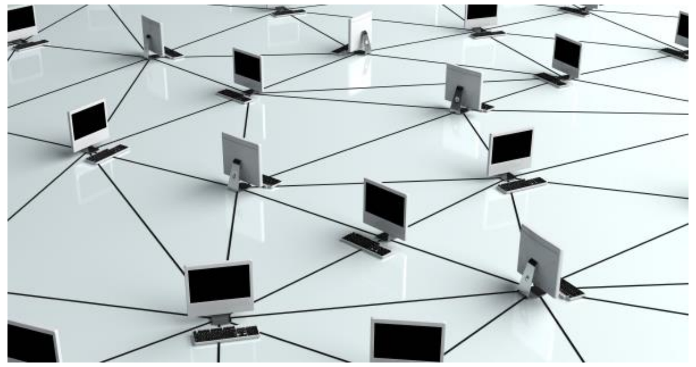
Disclaimer:
We are *modeling* networked computing.



Message Passing on a Graph

Disclaimer:
We are *modeling* networked computing.

Try to understand what is possible (and what is not)

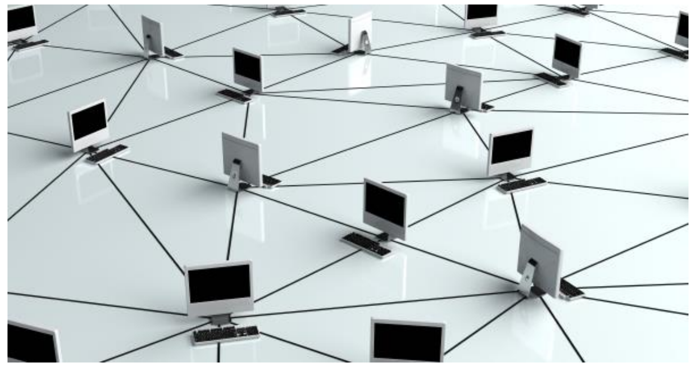


Message Passing on a Graph

Disclaimer:
We are *modeling* networked computing.

Try to understand what is possible (and what is not)

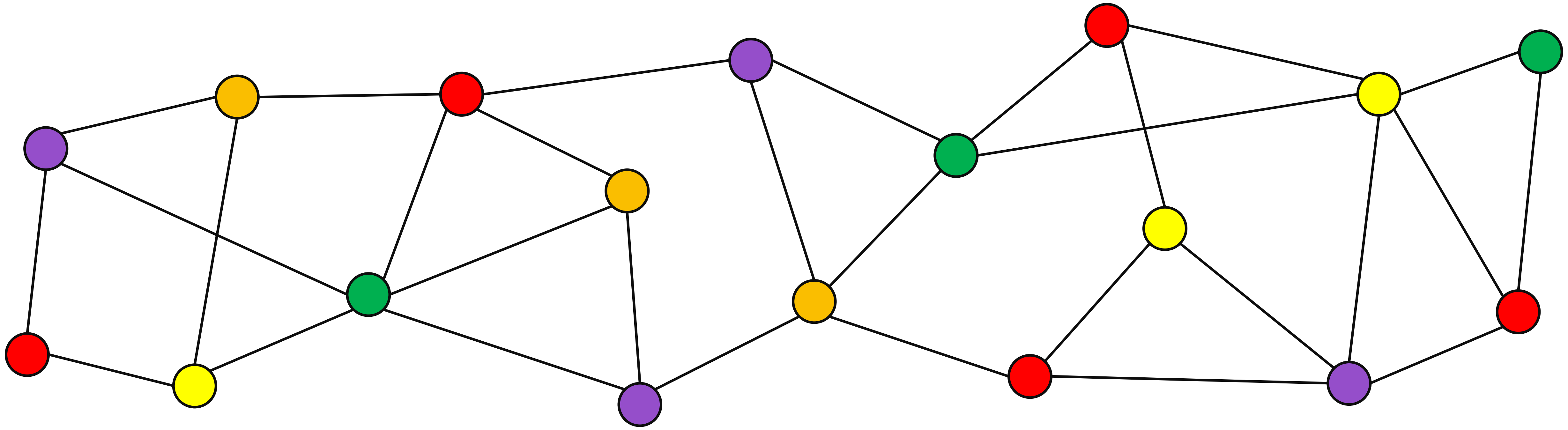
The goal is not to find algorithms for practice. Ideally, such algorithms get inspired by our findings



Outline

- Intro
  - What do we try to do
  - What this is not
- Coloring
  - $\Delta + 1$ colors
- The LOCAL model
- Coloring a ring
  - Color reduction
  - Lower bound

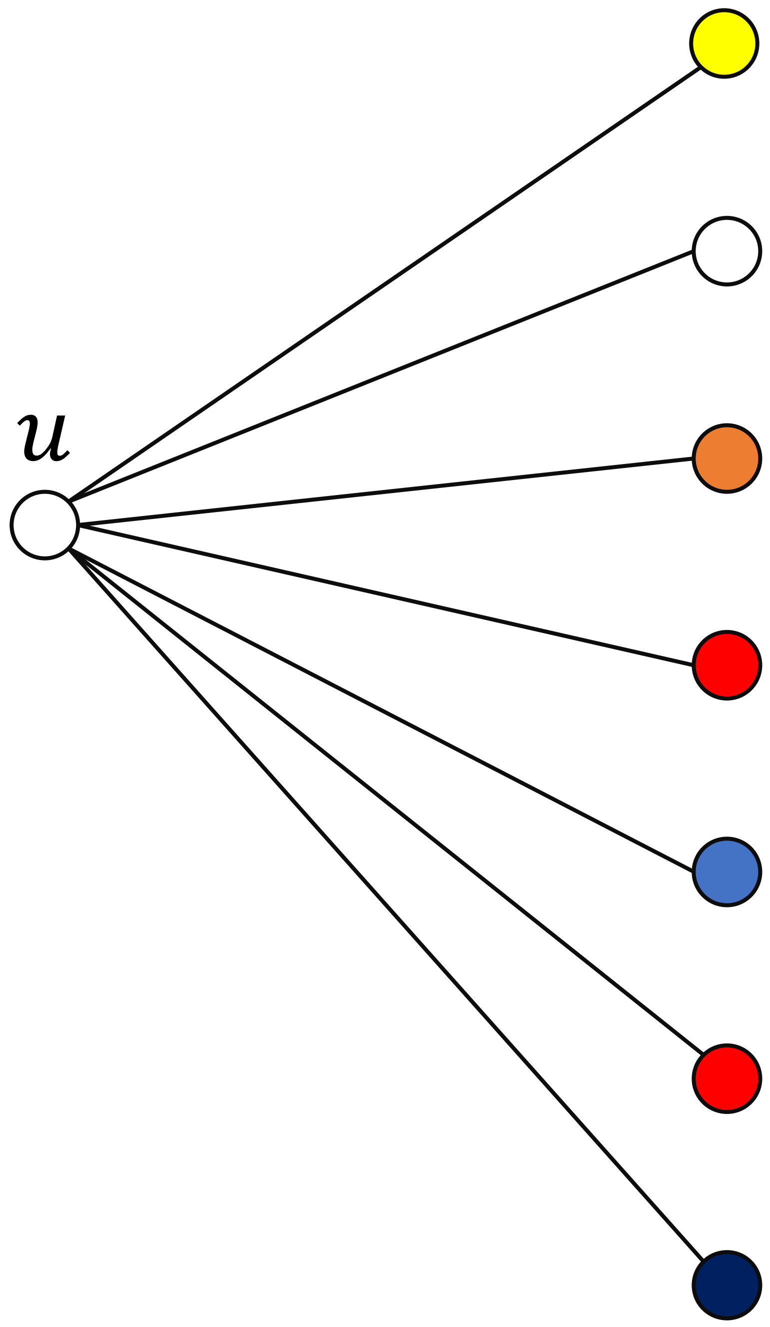
A Classic Problem: $(\Delta + 1)$-Node-Coloring



Goal: Color the nodes with at most $\Delta + 1$ colors.

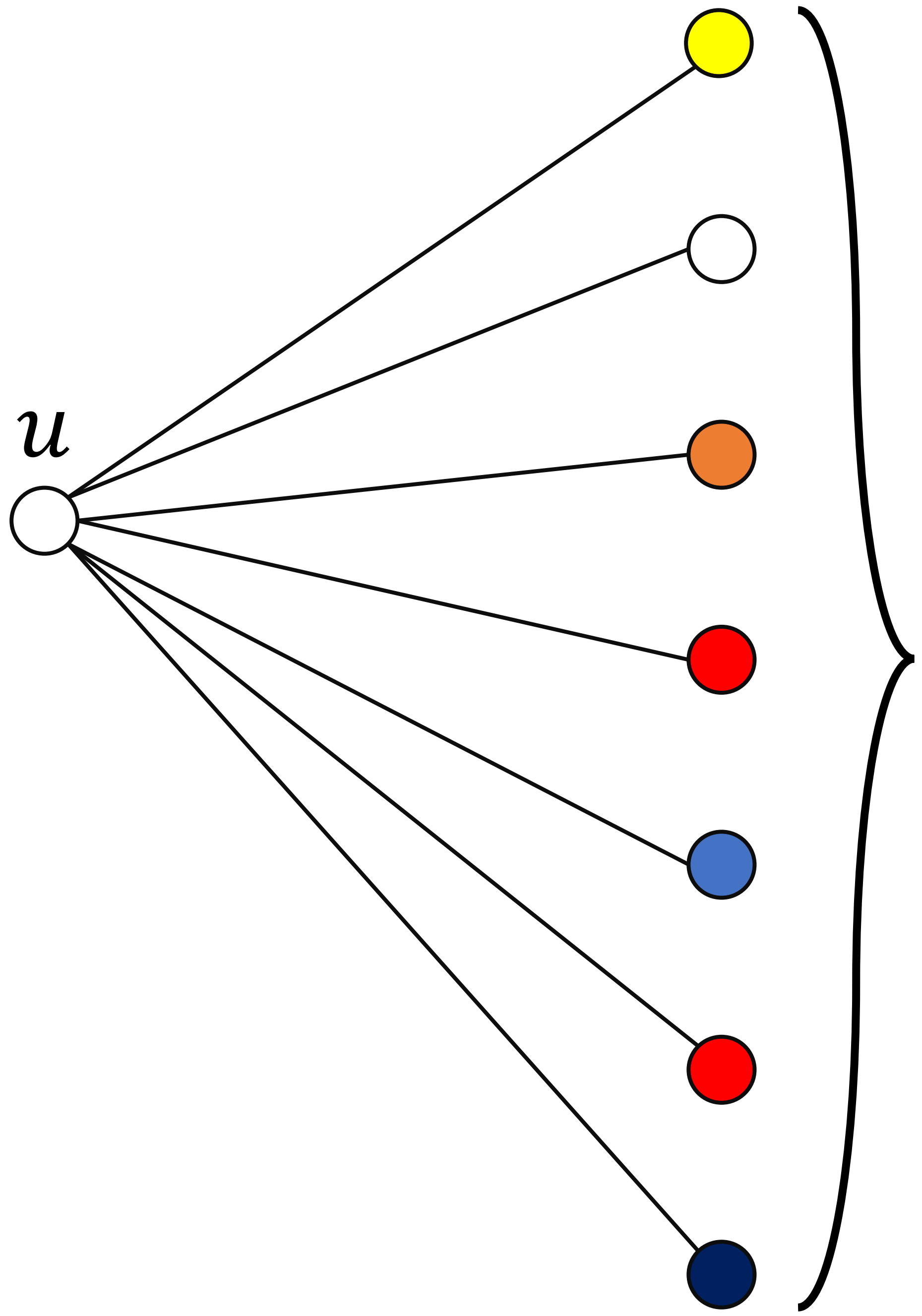
- Adjacent nodes have different colors.
- Δ is the maximum degree of the graph
- A greedy algorithm requires $\Delta + 1$ colors.

A Classic Problem: $(\Delta + 1)$-Node-Coloring



Observation: $\Delta + 1$ colors always suffice

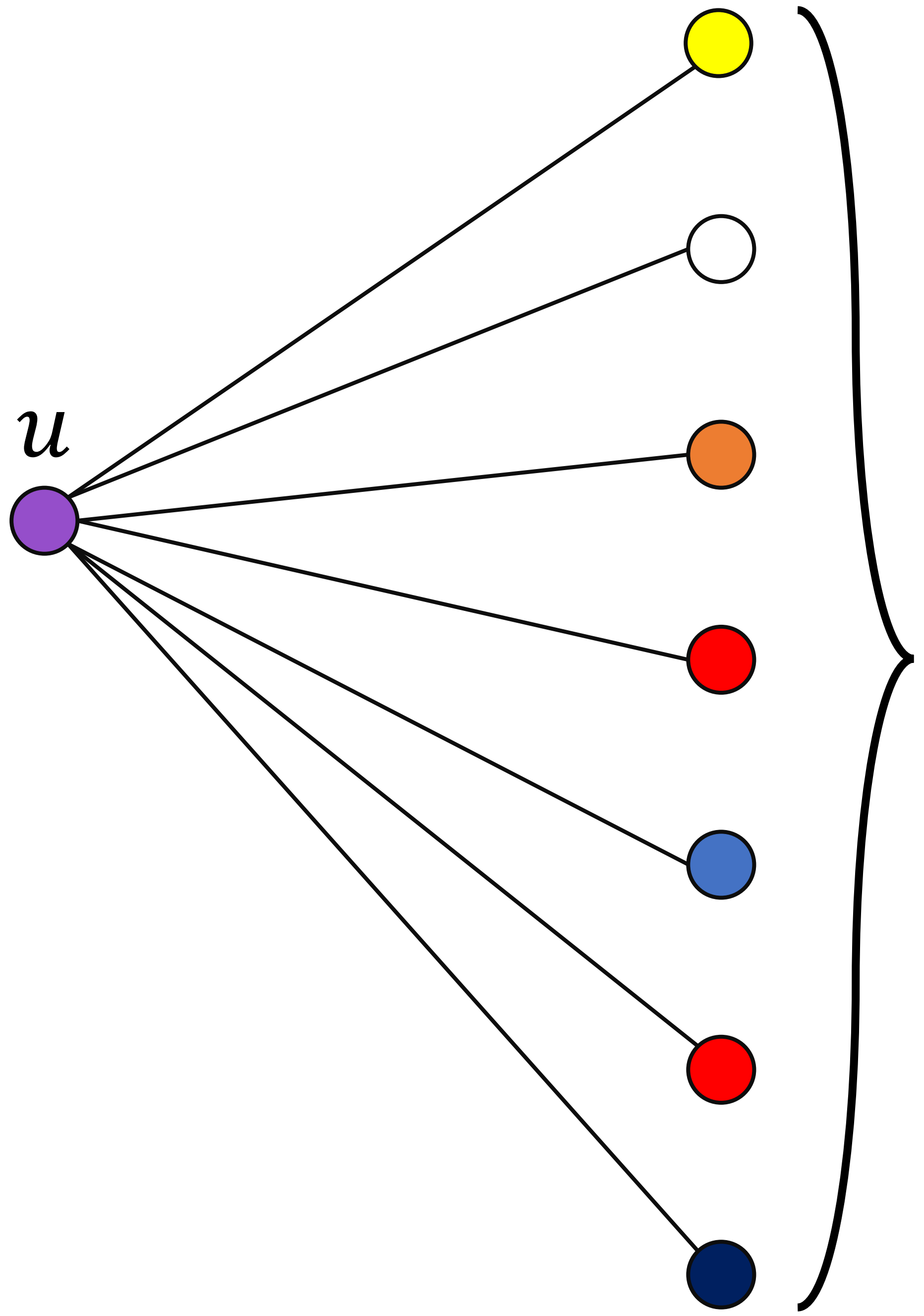
A Classic Problem: $(\Delta + 1)$-Node-Coloring



At most Δ different colors

Observation: $\Delta + 1$ colors always suffice

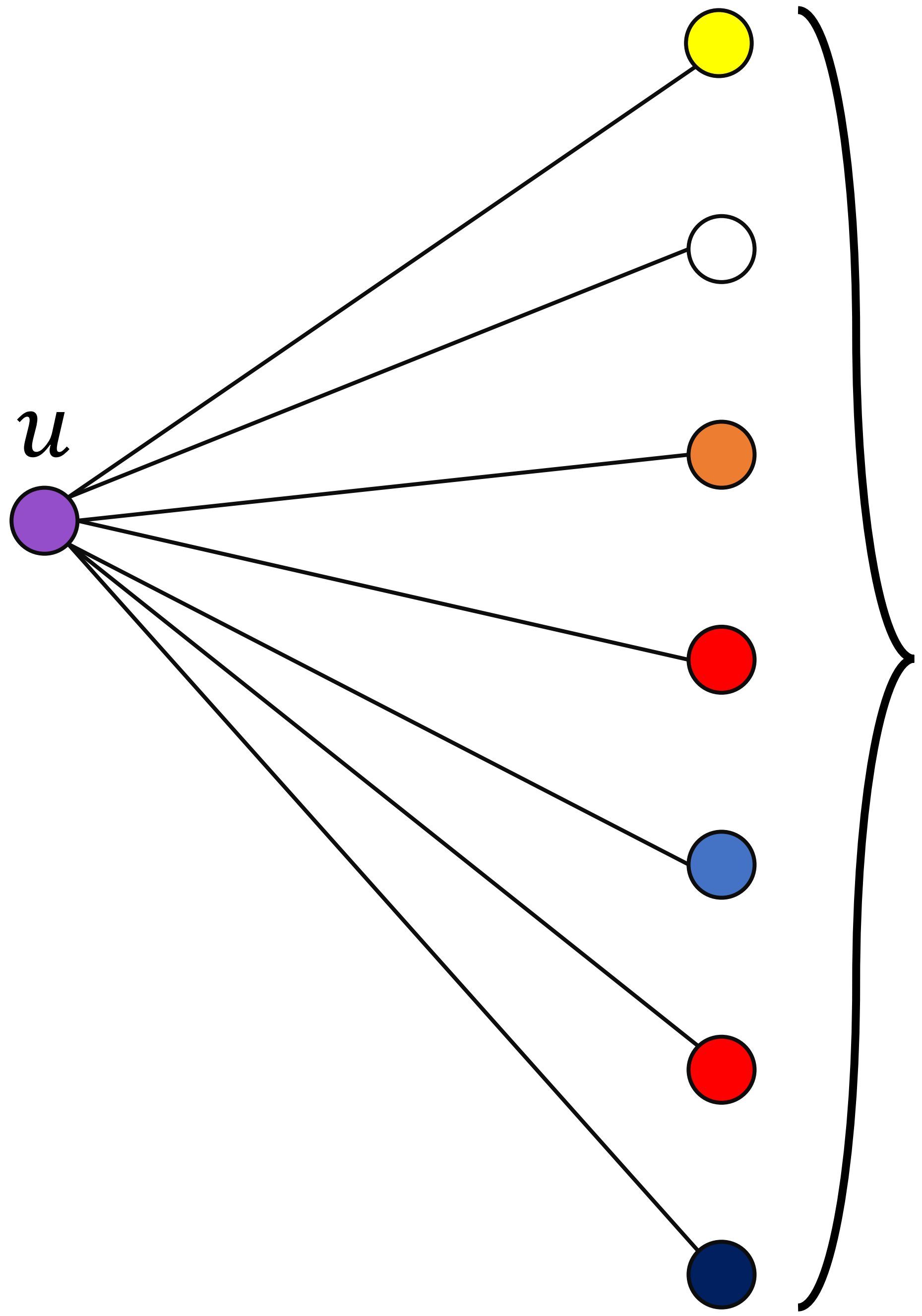
A Classic Problem: $(\Delta + 1)$-Node-Coloring



At most Δ different colors

Observation: $\Delta + 1$ colors always suffice

A Classic Problem: $(\Delta + 1)$-Node-Coloring



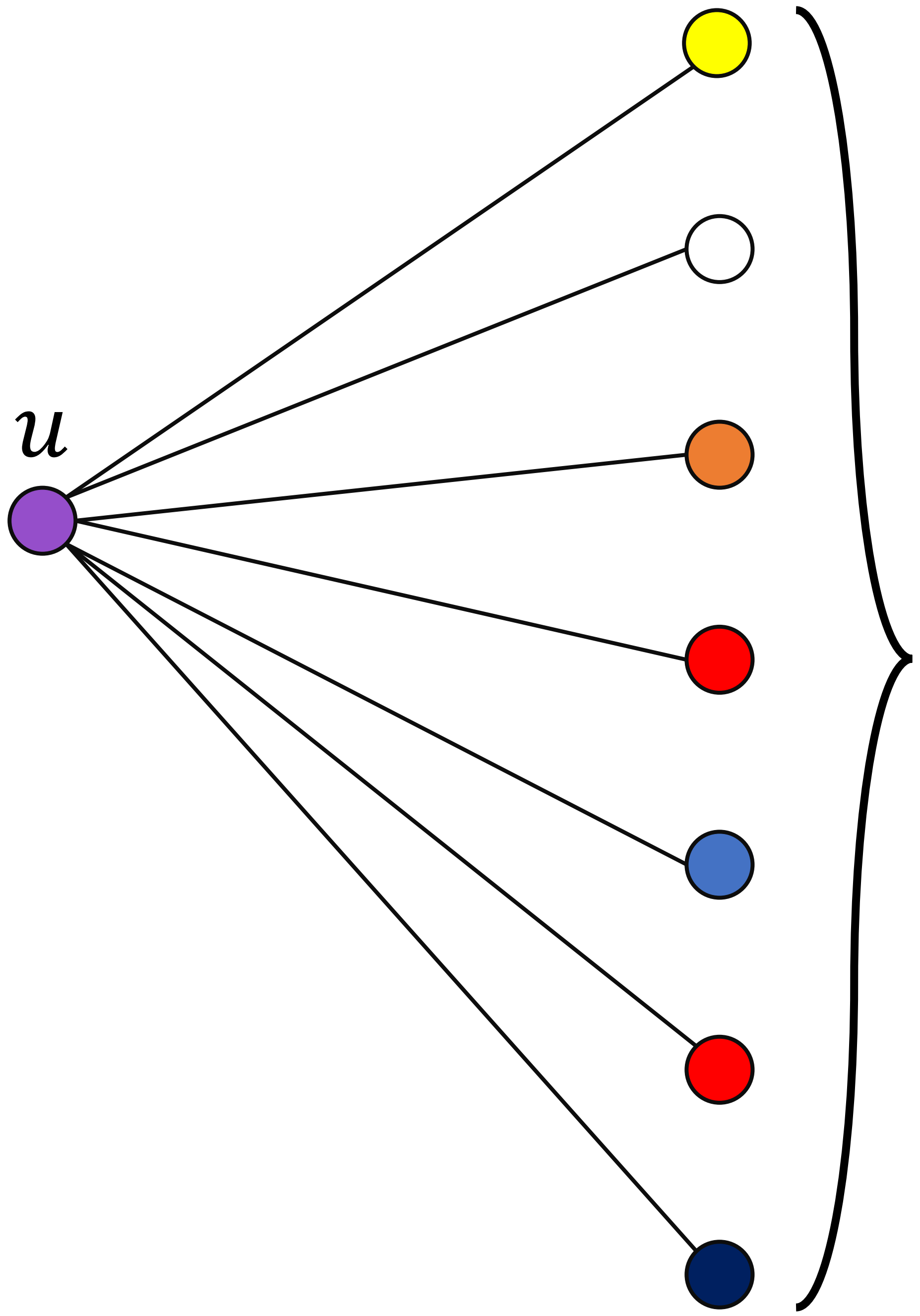
At most Δ different colors

Observation: $\Delta + 1$ colors always suffice

Proof:
Consider any partial $(\Delta + 1)$-coloring of the graph. Choose any node u. There is a free color for u to pick.

You can find a $(\Delta + 1)$-coloring with a greedy algorithm.

A Classic Problem: $(\Delta + 1)$-Node-Coloring



At most Δ different colors

Observation: $\Delta + 1$ colors always suffice

Proof:
Consider any partial $(\Delta + 1)$-coloring of the graph. Choose any node u. There is a free color for u to pick.

You can find a $(\Delta + 1)$-coloring with a greedy algorithm.

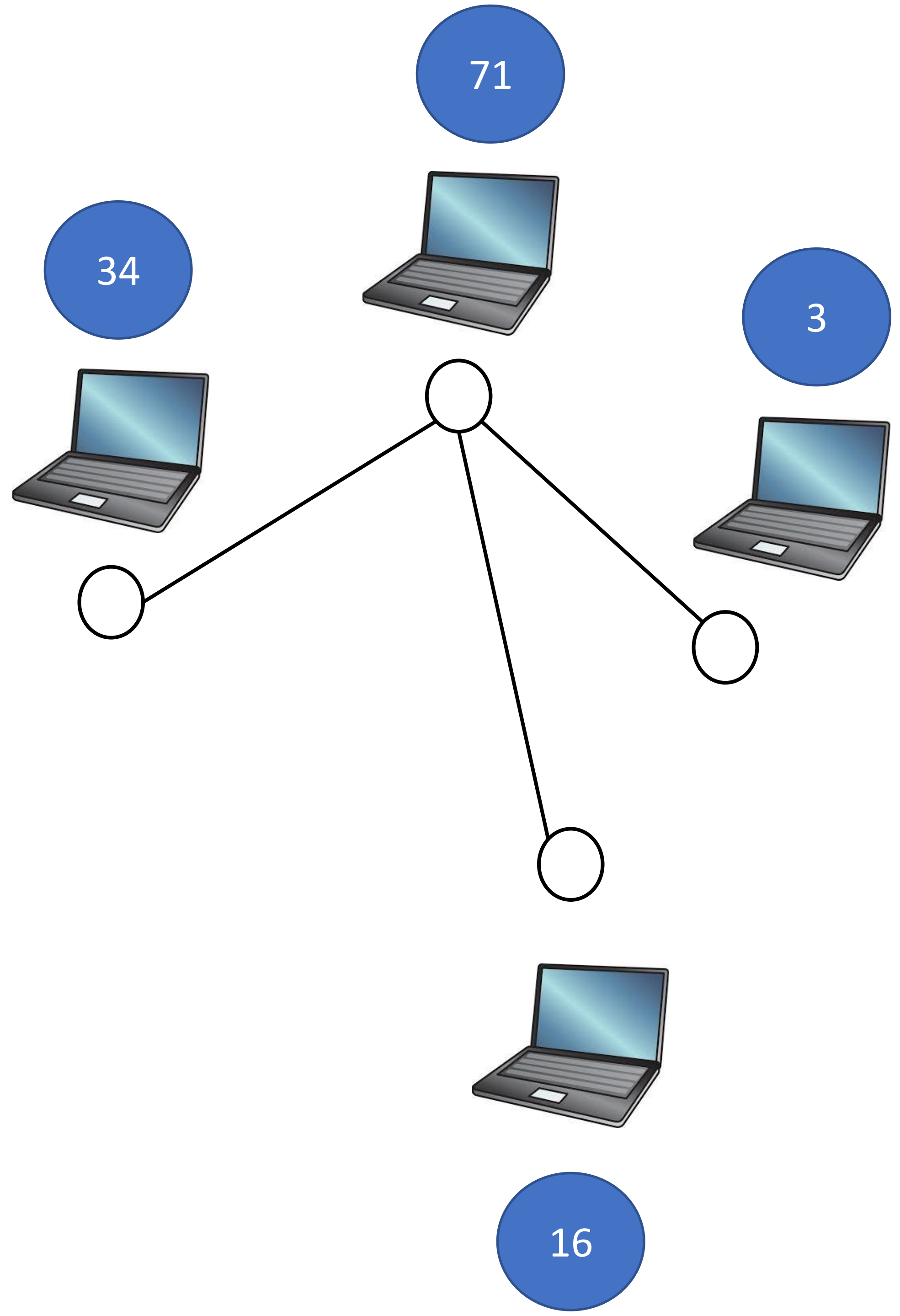
Outline

- Intro
  - What this is not
- Coloring
  - $\Delta + 1$ colors
- The LOCAL model
- Coloring a ring
  - Color reduction
  - Lower bound

$(\Delta + 1)$-Node-Coloring

In the beginning: Node only knows its own name and the names of its neighbors.

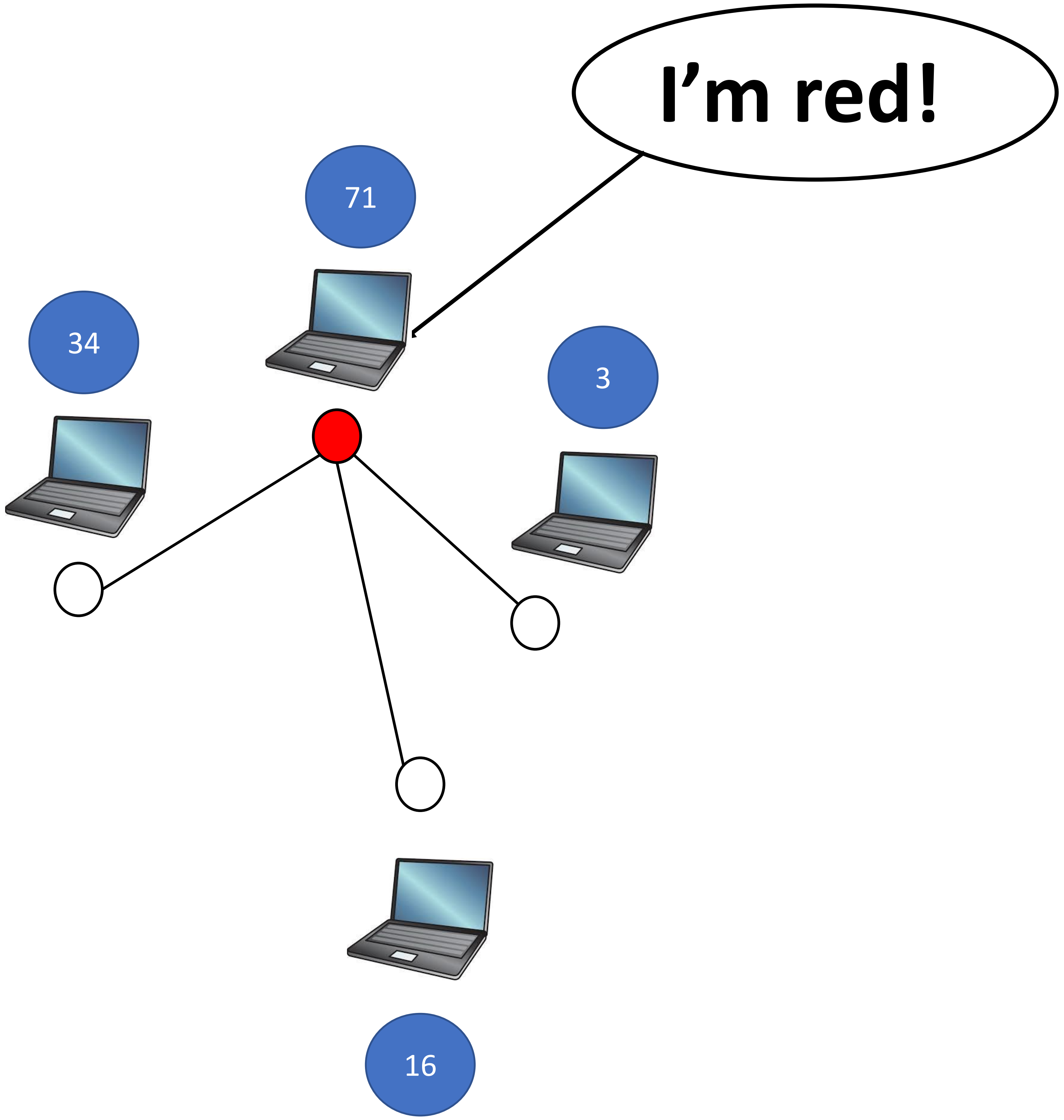
In the end: Node outputs its part of the solution, e.g., its color.



$(\Delta + 1)$-Node-Coloring

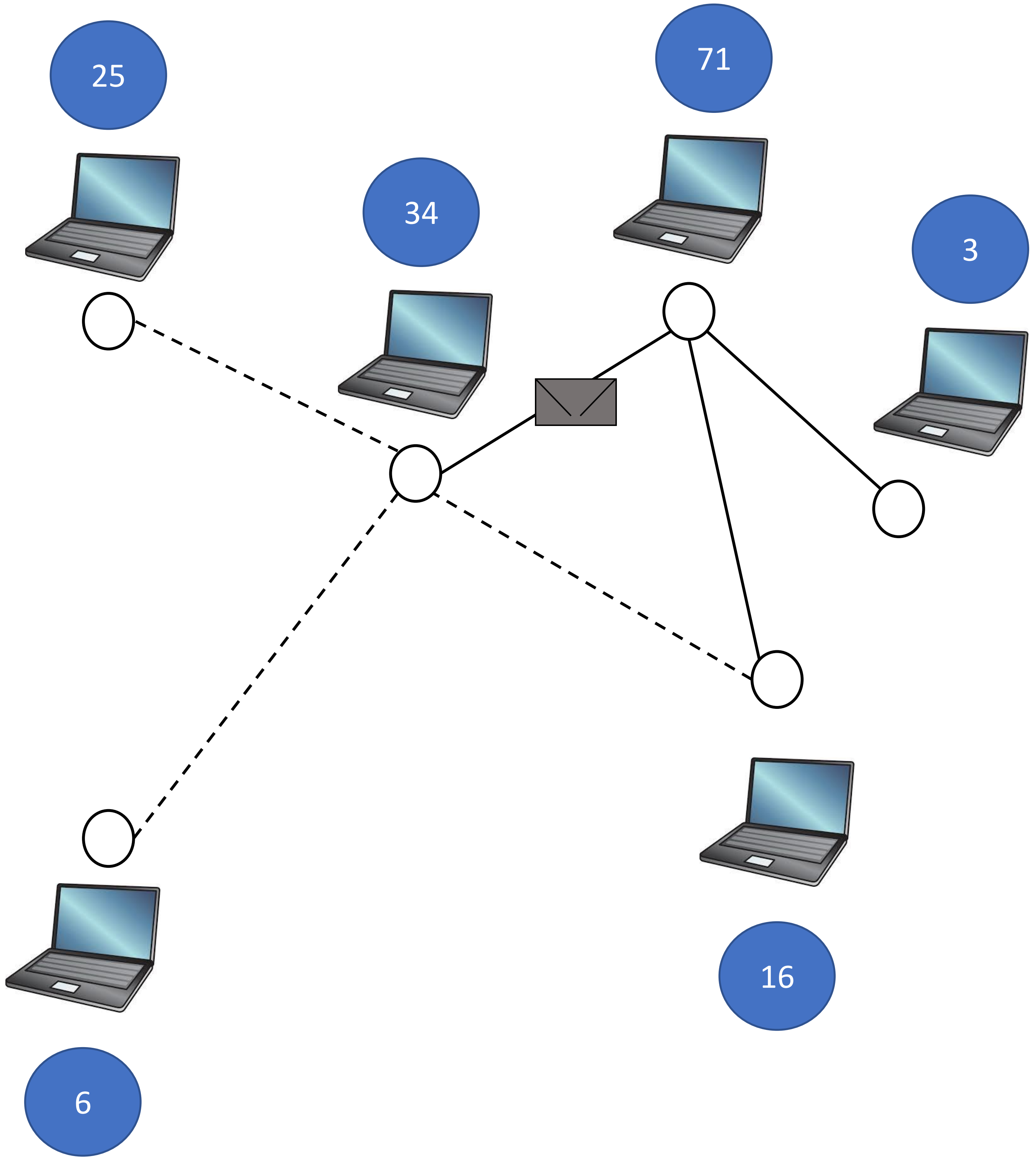
In the beginning: Node only knows its own name and the names of its neighbors.

In the end: Node outputs its part of the solution, e.g., its color.



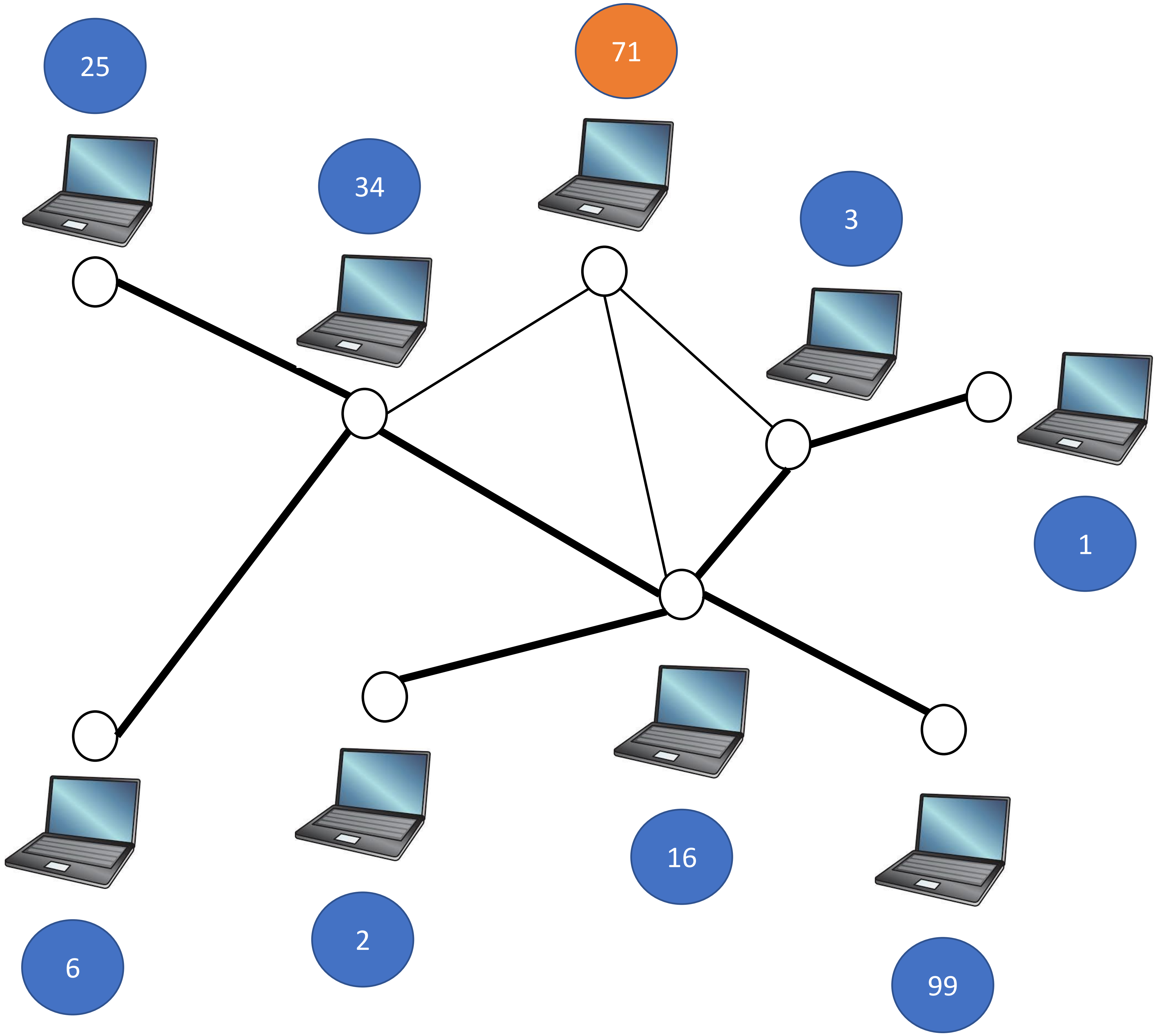
$(\Delta + 1)$-Node-Coloring

A round: Nodes can communicate along edges.



$(\Delta + 1)$-Node-Coloring

Round 2: Node can know the neighbors of neighbors.

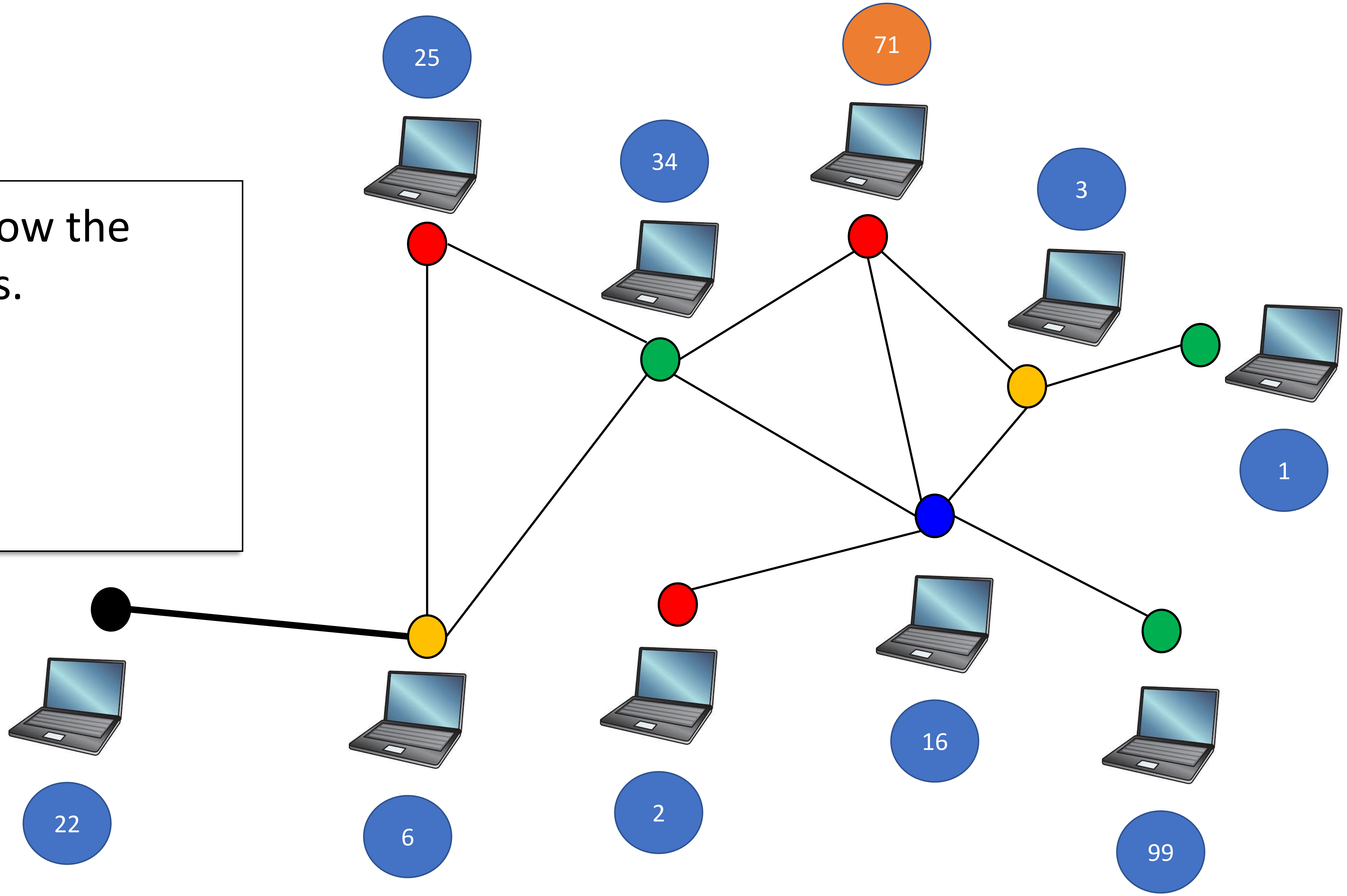


$(\Delta + 1)$-Node-Coloring

Round 2: Node can know the neighbors of neighbors.

Round 3: $N^3(v)$

Round i**:** $N^i(v)$



The **LOCAL** Model of Distributed Computing

[Linial FOCS '87]

We abstract away asynchrony, congestion, and local computations

1. Synchronous rounds
2. Unlimited message sizes
3. Unlimited local computations

Complexity: #rounds

The **LOCAL** Model of Distributed Computing

[Linial FOCS ’87]

We abstract away asynchrony, congestion, and local computations

1. Synchronous rounds
2. Unlimited message sizes
3. Unlimited local computations

Complexity: #rounds

1. Receive message
2. Compute
3. Send messages

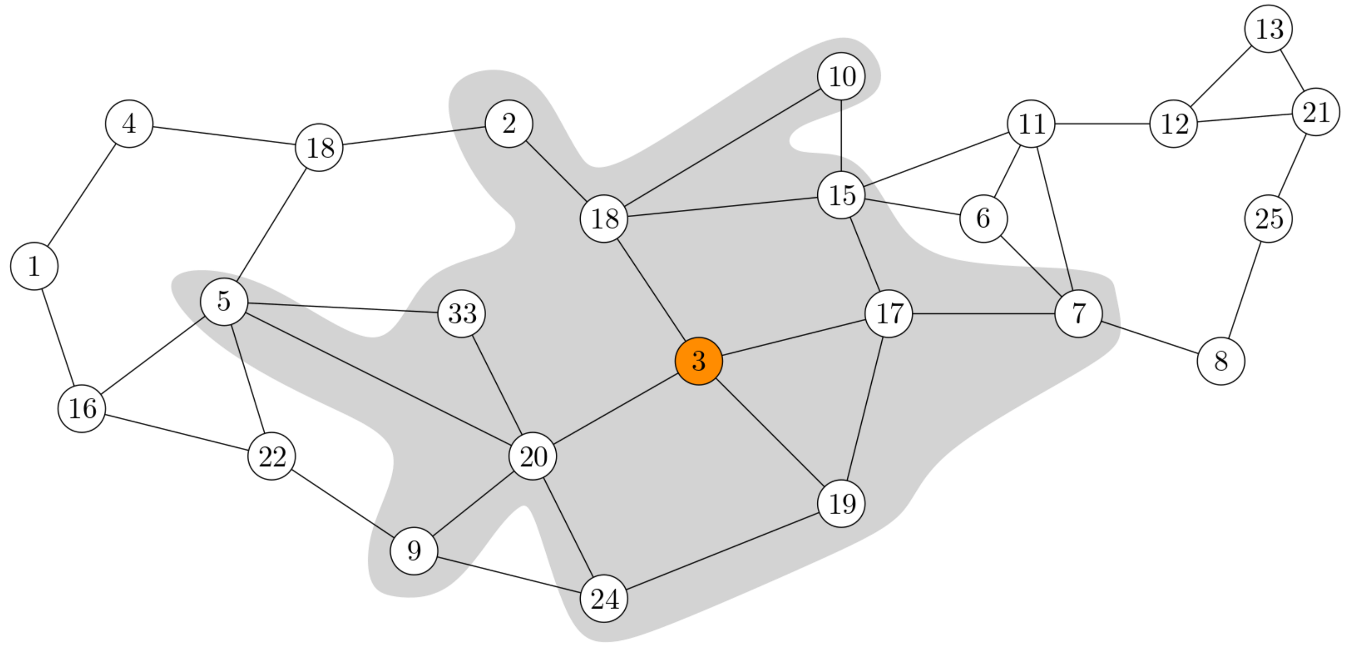
The **LOCAL** Model of Distributed Computing

[Linial FOCS '87]

We abstract away asynchrony, congestion, and local computations

1. Synchronous rounds
2. Unlimited message sizes
3. Unlimited local computations

Complexity: #rounds



$N^t(v)$ = the topology of t-hop neighborhood and the $O(\log n)$ bit unique identifiers.

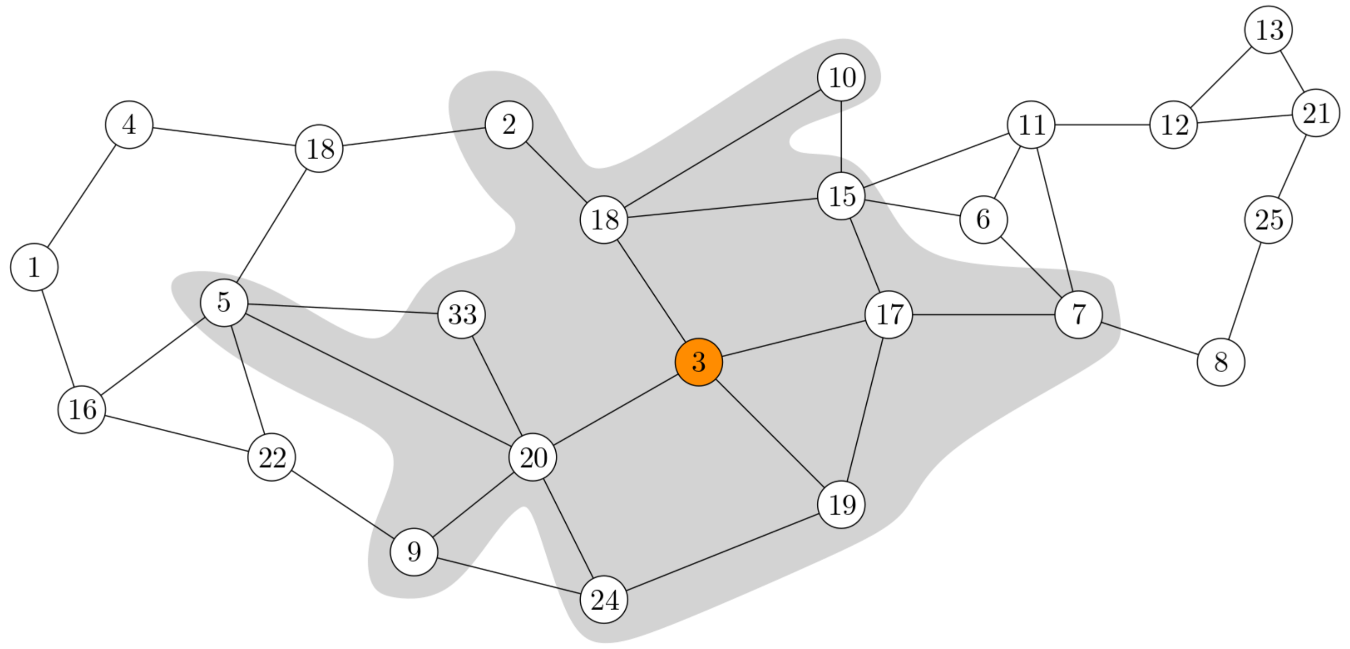
The **LOCAL** Model of Distributed Computing

[Linial FOCS '87]

We abstract away asynchrony, congestion, and local computations

1. Synchronous rounds
2. Unlimited message sizes
3. Unlimited local computations

Complexity: #rounds



A local distributed algorithm:
$N^t(v) \rightarrow$ output

If $t =$ diameter, any problem solvable by brute force.

$N^t(v)$ = the topology of t-hop neighborhood and the unique identifiers.

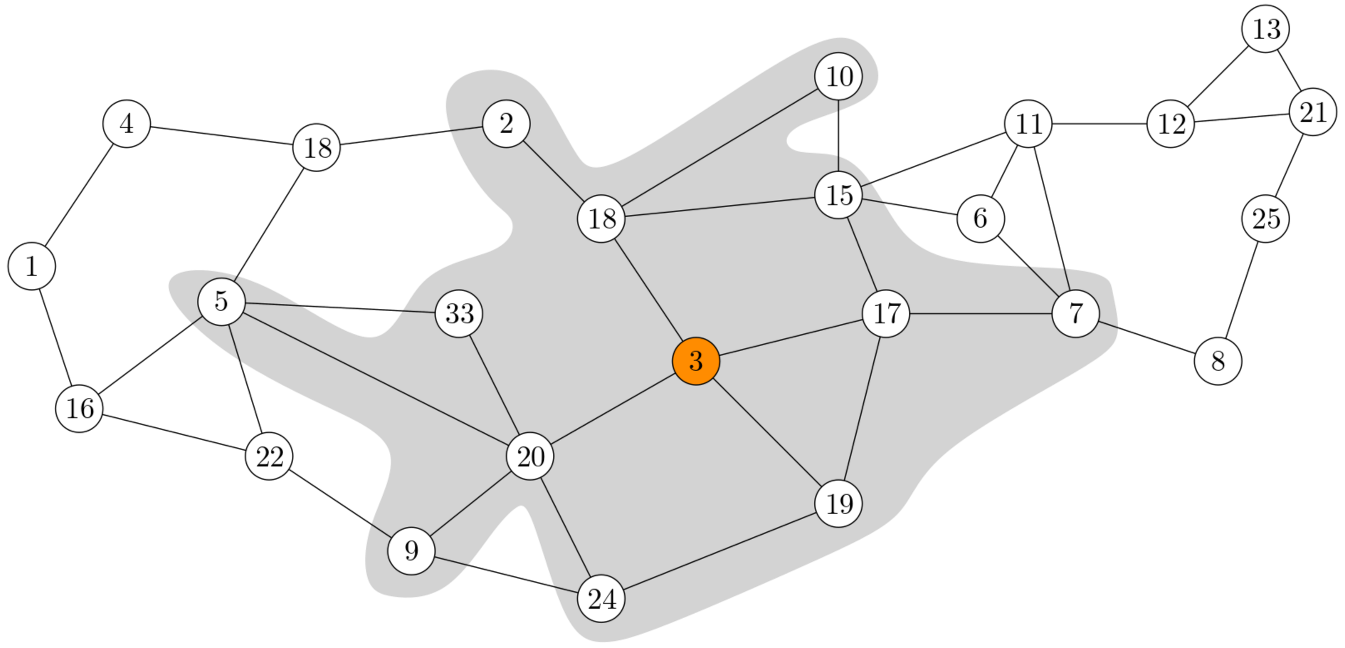
The **LOCAL** Model of Distributed Computing

[Linial FOCS ’87]

We abstract away asynchrony, congestion, and local computations

1. Synchronous rounds
2. Unlimited message sizes
3. Unlimited local computations

Complexity: #rounds



A local distributed algorithm:
$N^t(v) \rightarrow$ output

If $t =$ diameter, any problem solvable by brute force.

Trivial algorithm:
Gather all the information

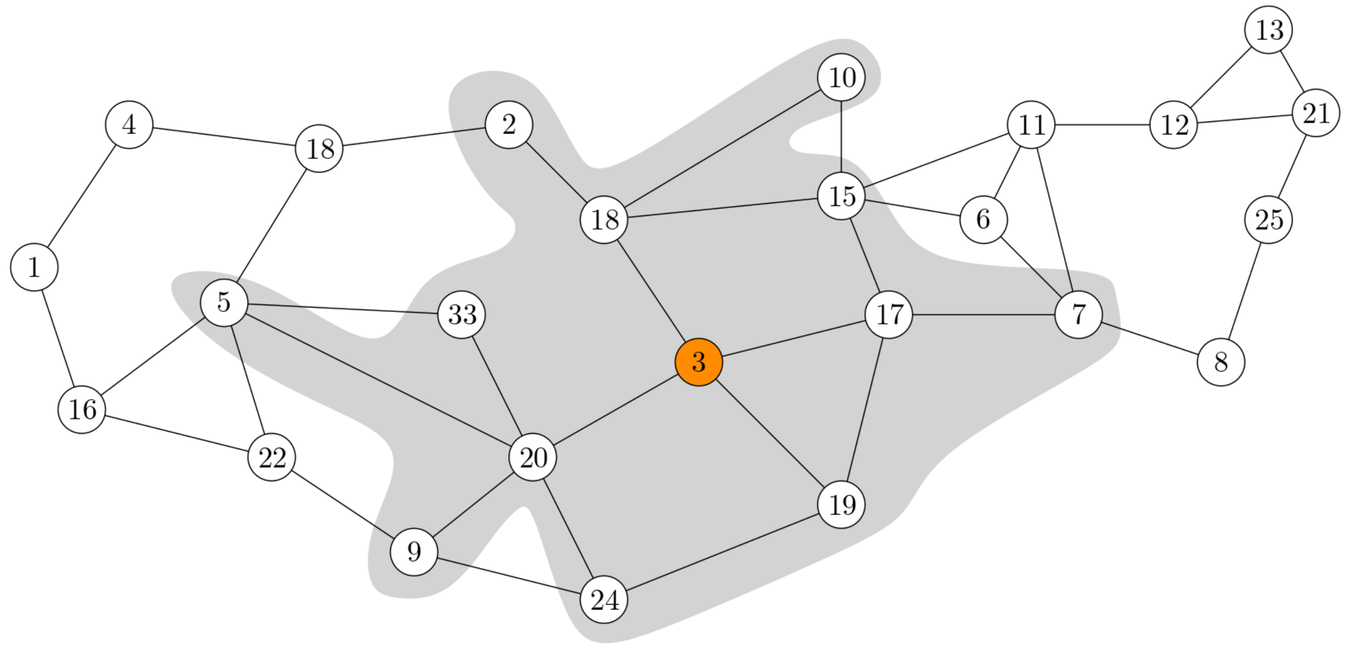
The **LOCAL** Model of Distributed Computing

[Linial FOCS ’87]

We abstract away asynchrony, congestion, and local computations

1. Synchronous rounds
2. Unlimited message sizes
3. Unlimited local computations

Complexity: #rounds



A local distributed algorithm:
$N^t(v) \rightarrow$ output

If $t =$ diameter, any problem solvable by brute force.

Captures Locality:
How far do we have to communicate?

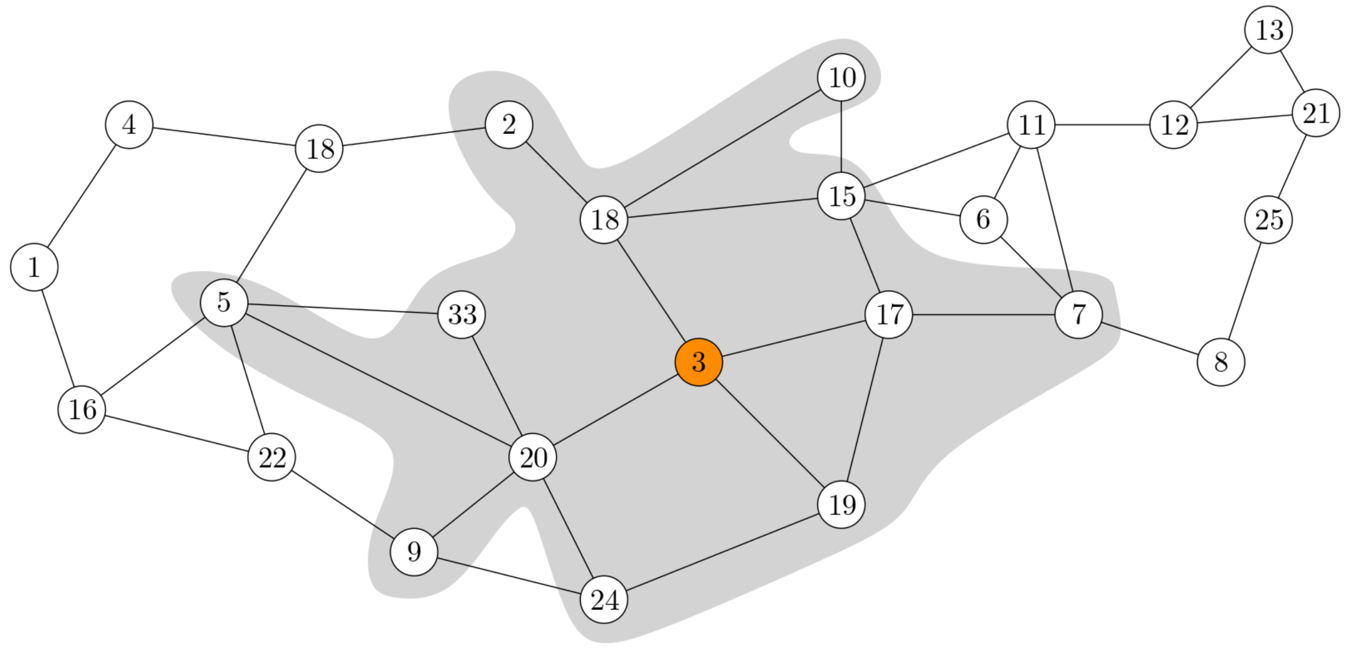
The **LOCAL** Model of Distributed Computing

[Linial FOCS ’87]

We abstract away asynchrony, congestion, and local computations

1. Synchronous rounds
2. Unlimited message sizes
3. Unlimited local computations

Complexity: #rounds



Why?

1. A very clean model
2. Information theoretic lower bounds!

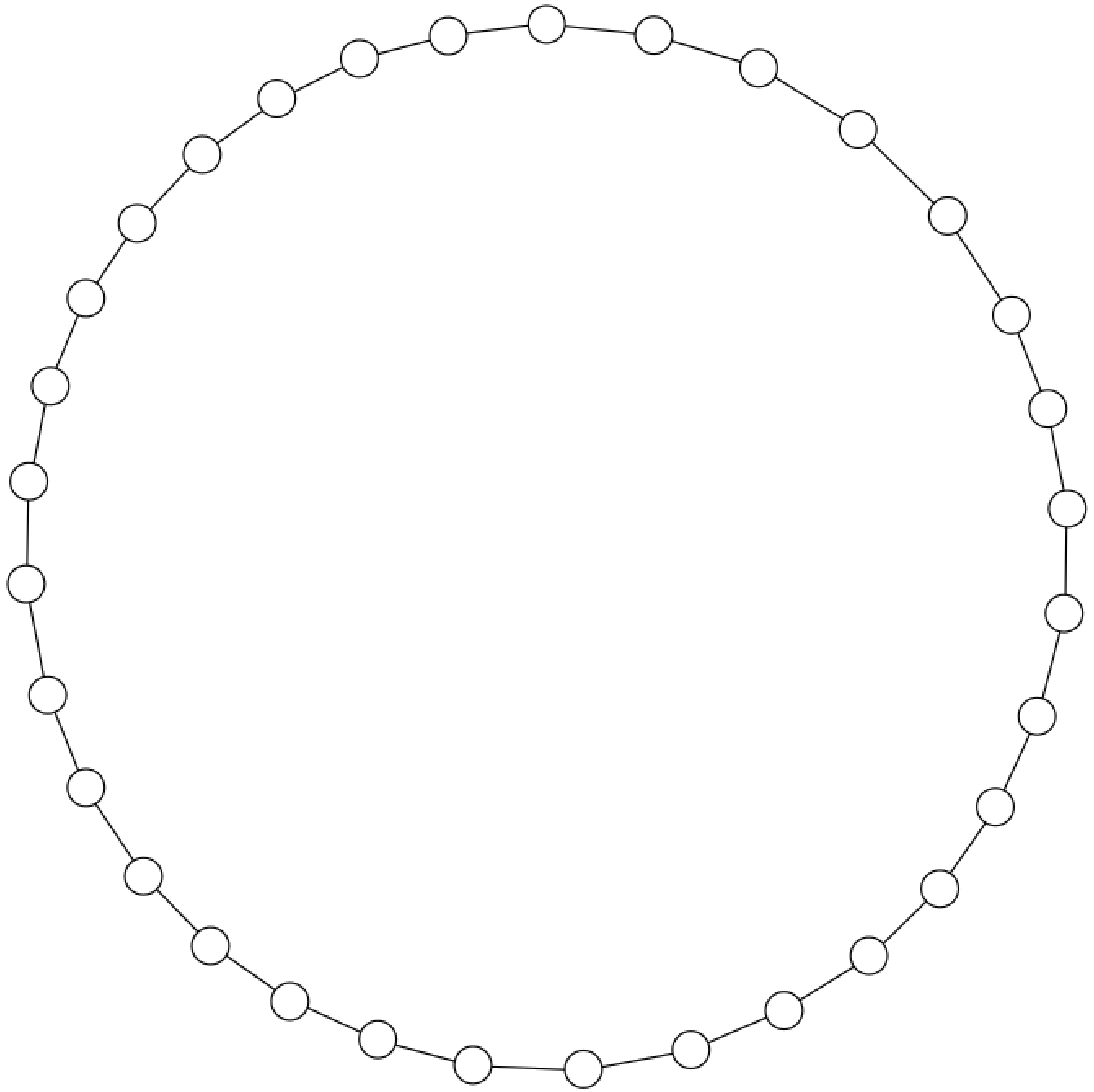
Much more in:
CS-E4510 - Distributed Algorithms
by Jukka Suomela

Outline

- Intro
  - What this is not
- Coloring
  - $\Delta + 1$ colors
- The LOCAL model
- Coloring a ring
  - Color reduction
  - Lower bound

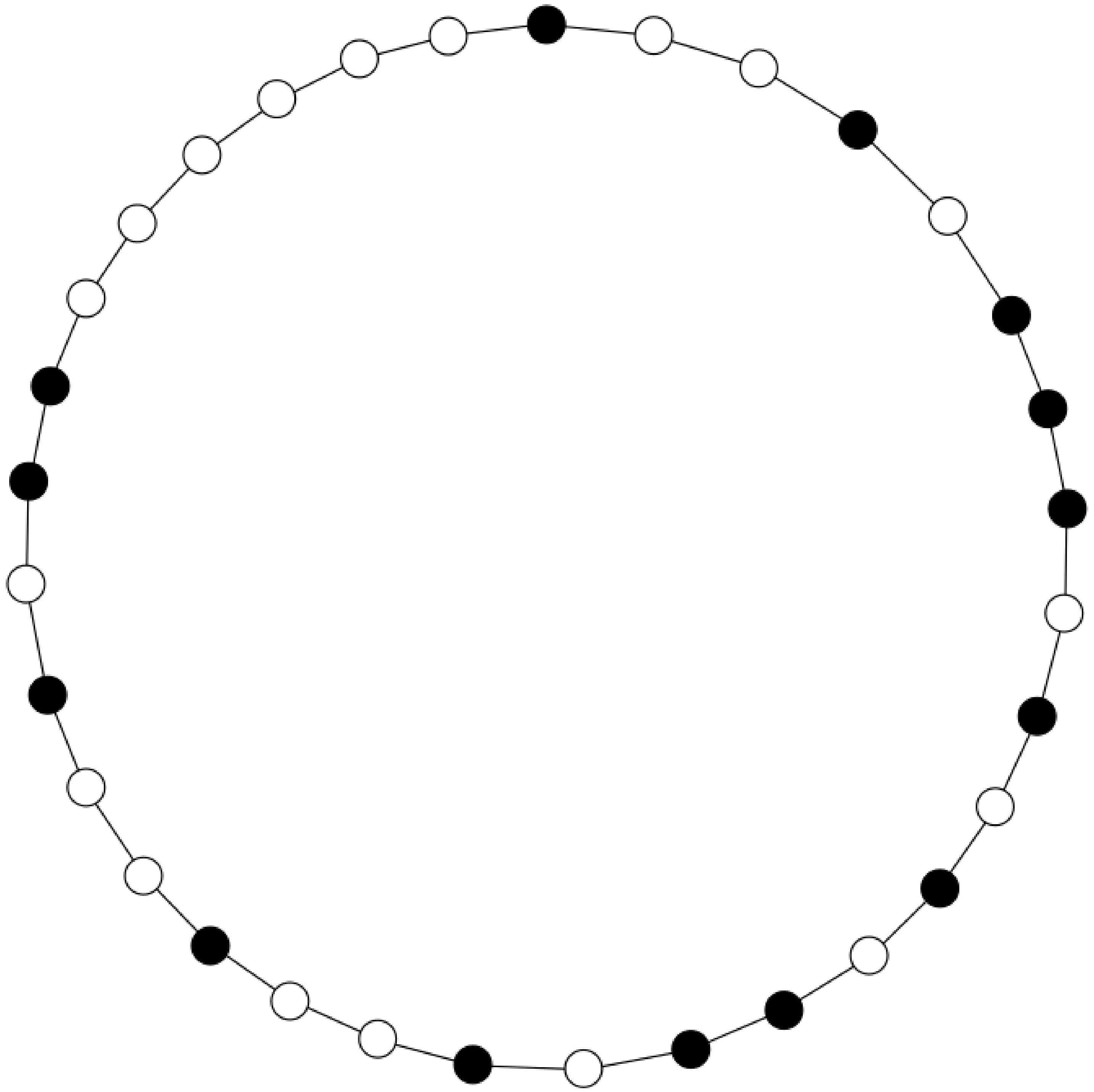
A Warm-Up Problem

A warm-up problem:
An n-node ring
Find a 4-coloring

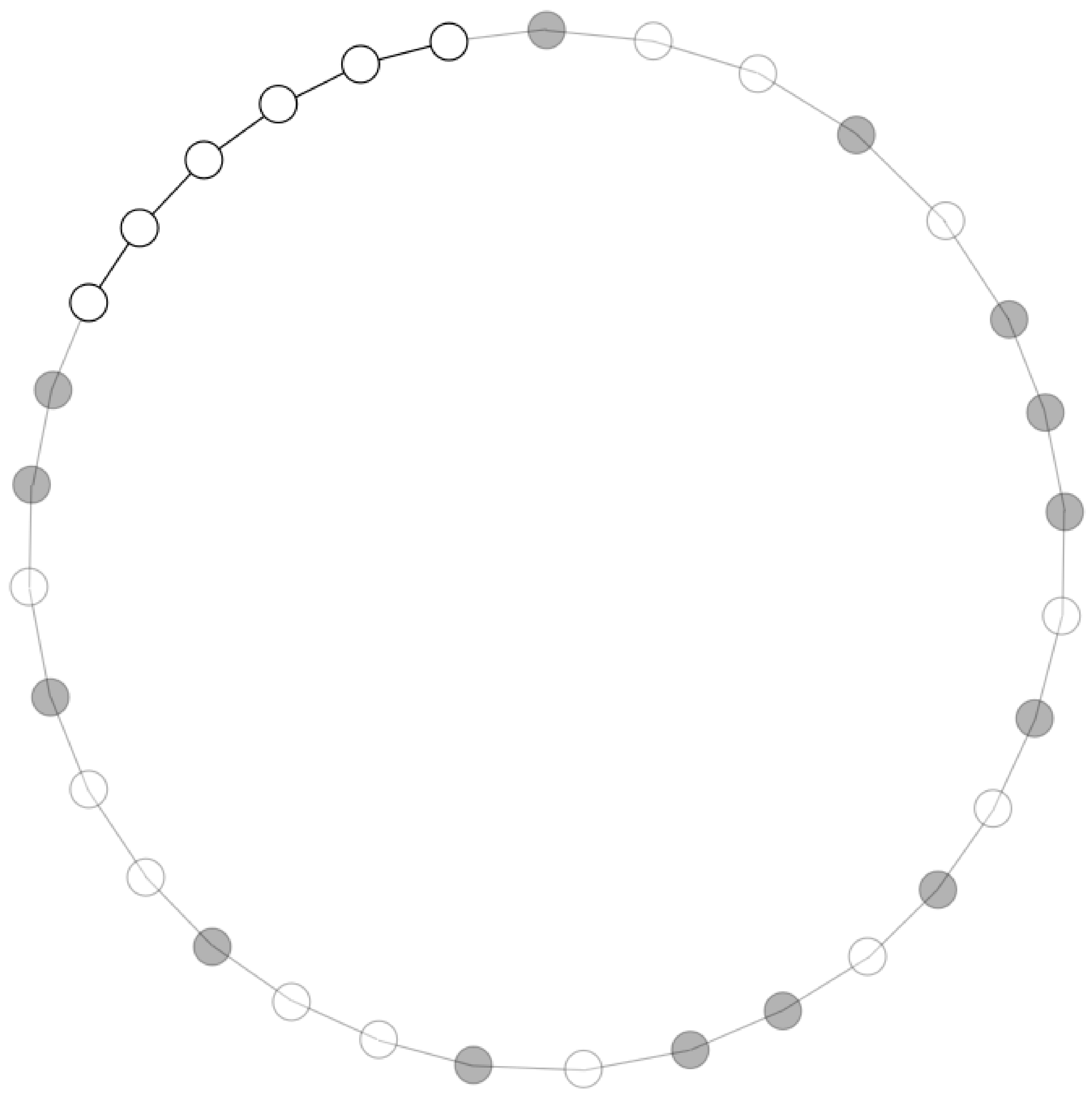


A Warm-Up Problem

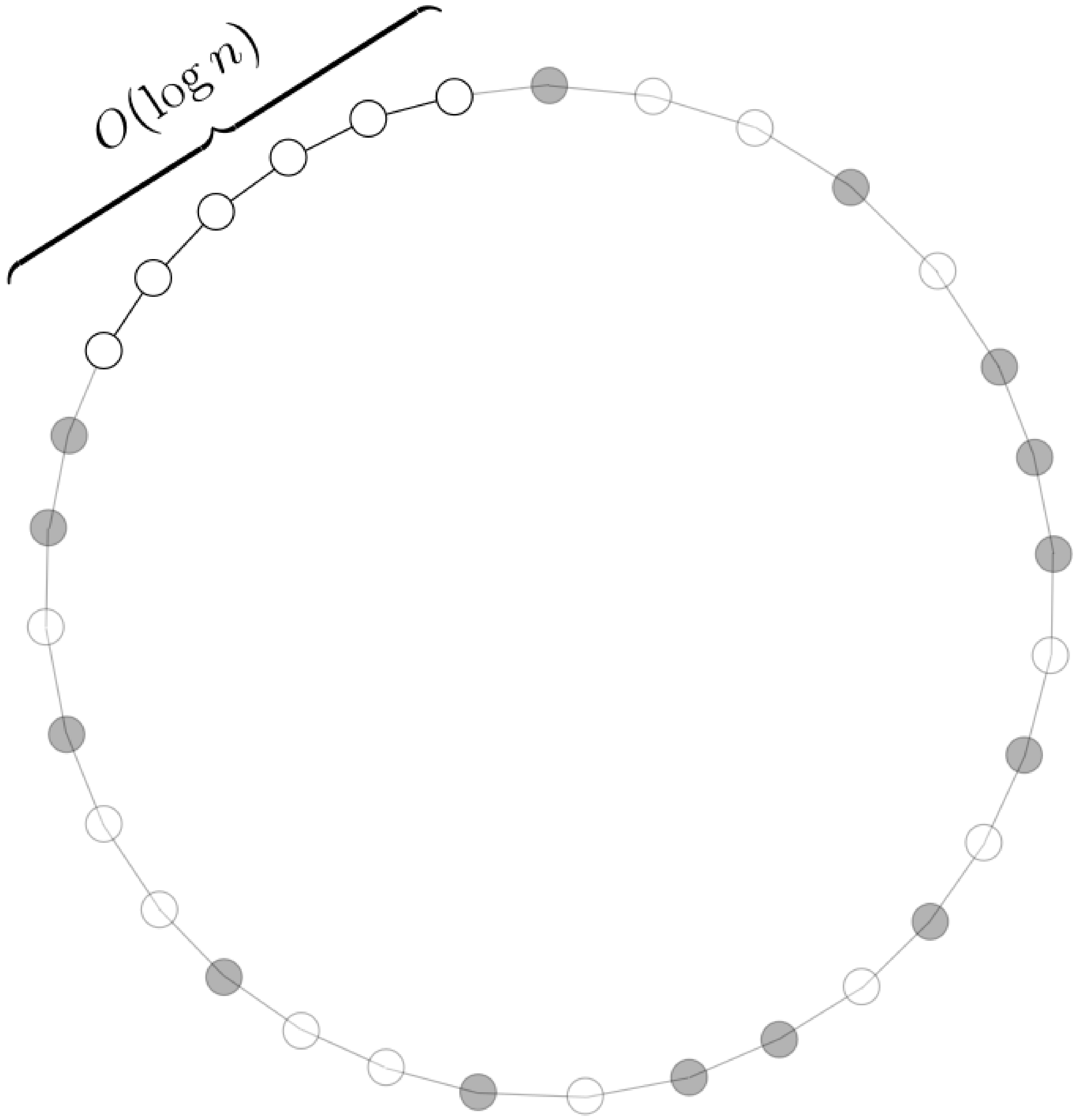
Toss a fair coin.
Split into two parts



A Warm-Up Problem



A Warm-Up Problem

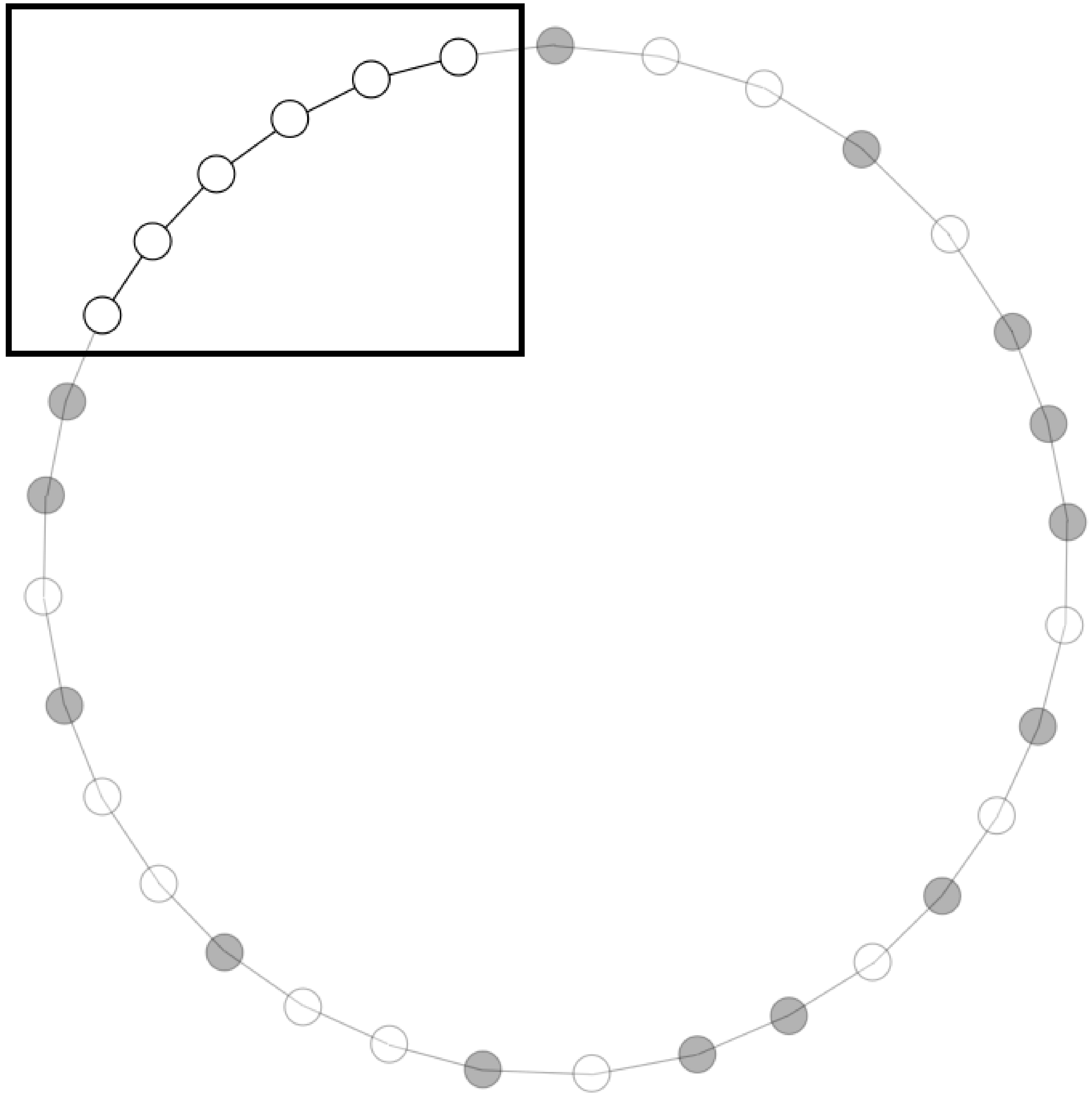


Lemma:
Monochromatic parts are of length $O(\log n)$ with high probability
$1 - 1/n^c$

A Warm-Up Problem

Find a 2-coloring:
For example, by the trivial algorithm

Runtime: $O(\log n)$



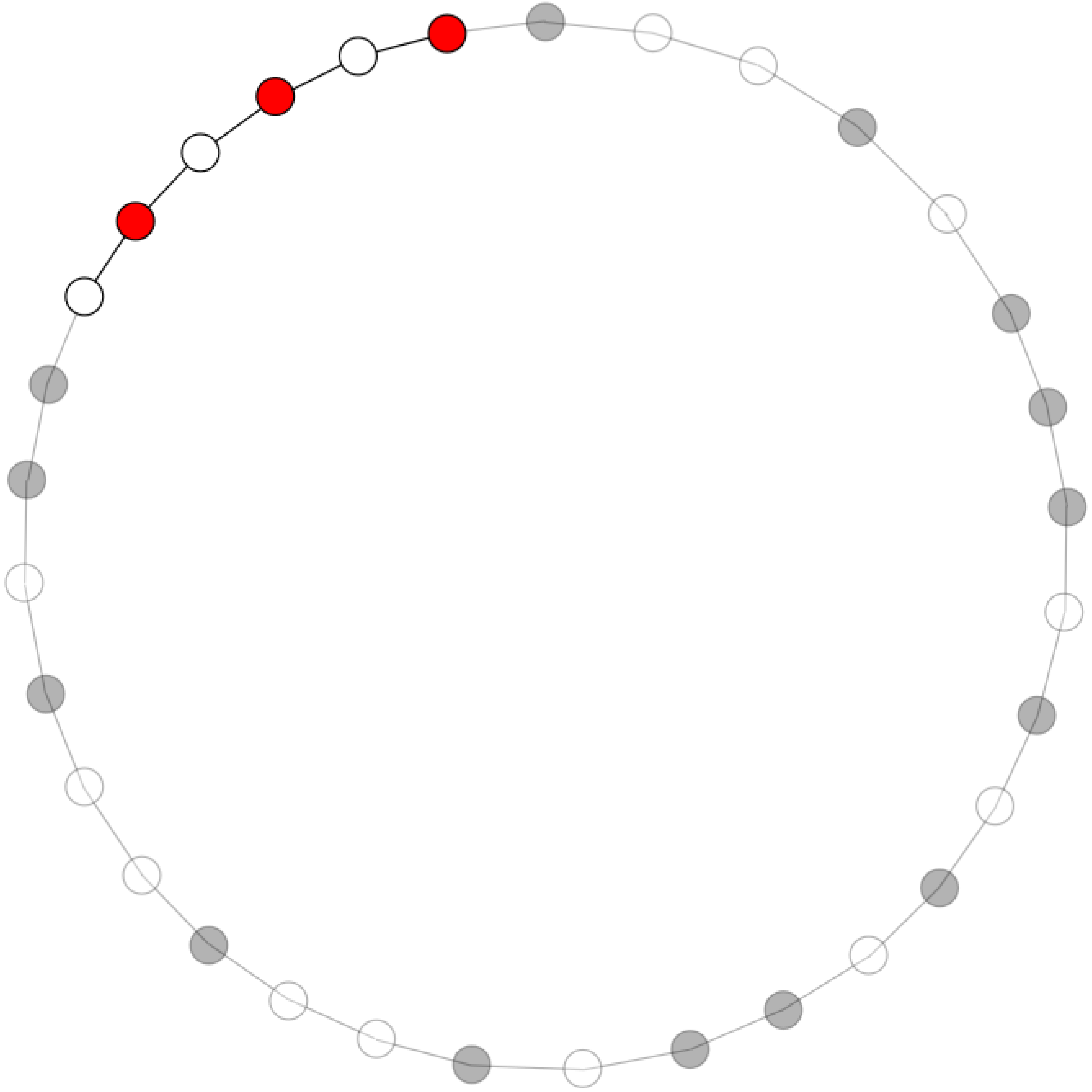
Lemma:
Monochromatic parts are of length $O(\log n)$ with high probability

$$1 - 1/n^c$$

A Warm-Up Problem

White part:
Colors white/red

Black part:
Colors black/blue

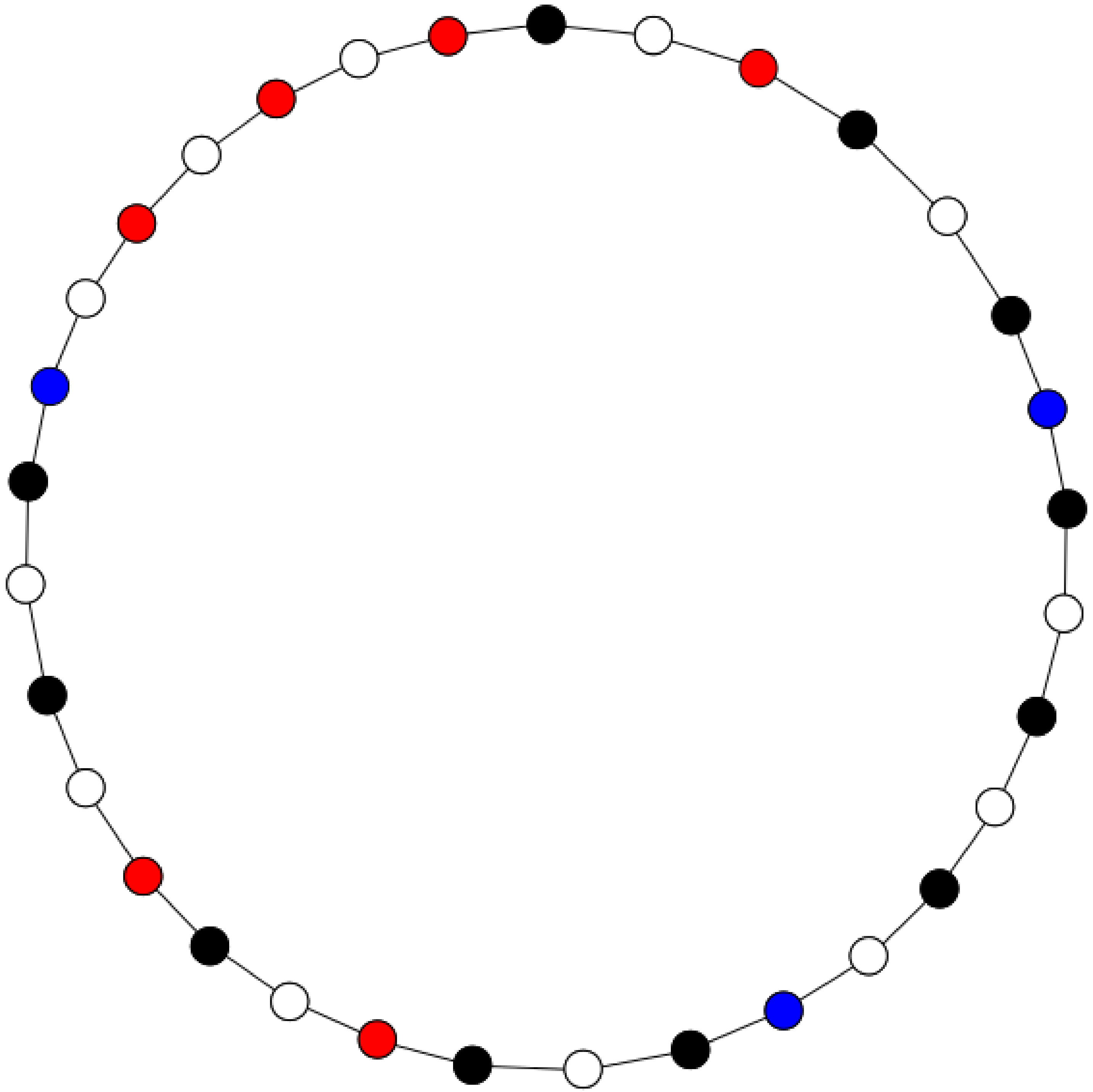


Lemma:
Monochromatic parts are of length $O(\log n)$ with high probability $1 - 1/n^c$

A Warm-Up Problem

White part:
Colors white/red

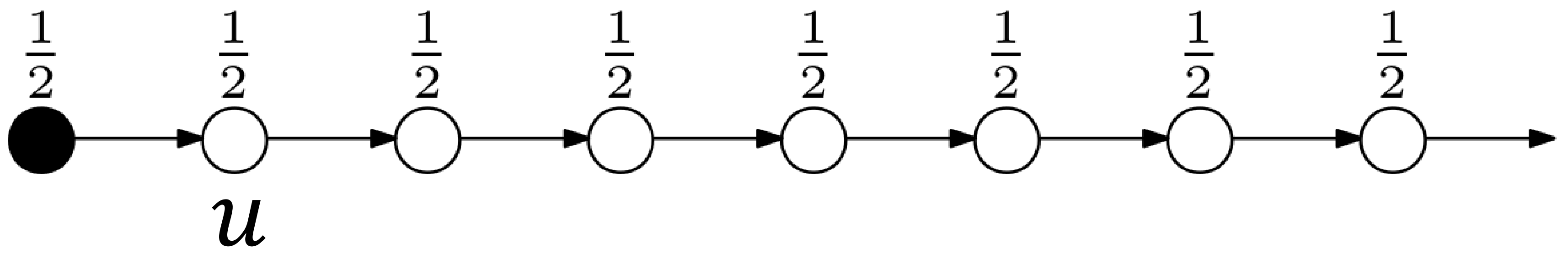
Black part:
Colors black/blue



Lemma:
Monochromatic parts are of length $O(\log n)$ with high probability
$1 - 1/n^c$

Runtime:
$O(\log n)$

A Warm-Up Problem

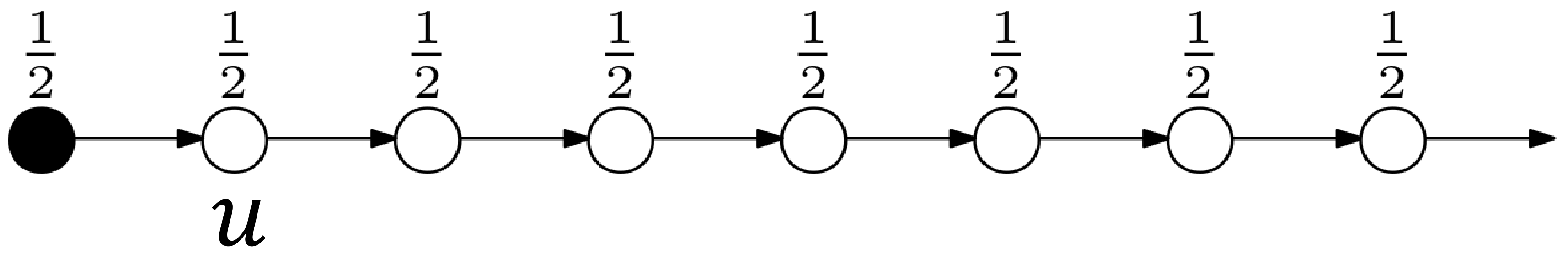


Lemma:
Monochromatic parts are of length $O(\log n)$ with high probability

$$1 - 1/n^c$$

Think of an oriented ring

A Warm-Up Problem



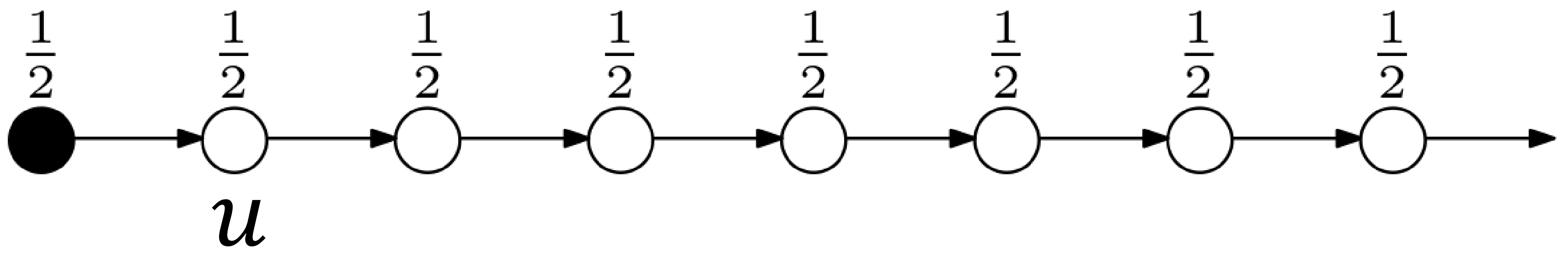
Lemma:
Monochromatic parts are of length $O(\log n)$ with high probability

$$1 - 1/n^c$$

Event $\mathrm{E}(u)$:
A monochromatic path of length at least $2c \log n$ starts from u

Think of an oriented ring

A Warm-Up Problem



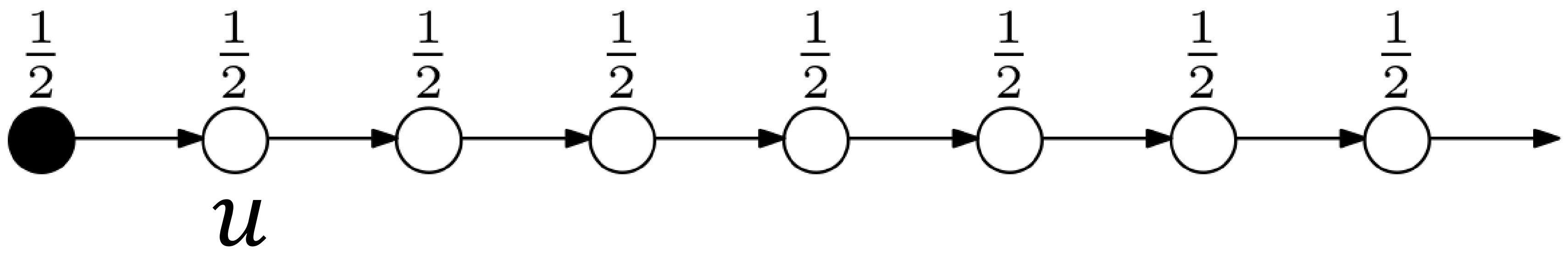
Lemma:
Monochromatic parts are of length $O(\log n)$ with high probability
$$1 - 1/n^c$$

Event $\mathrm{E}(u)$:
A monochromatic path of length at least $2c \log n$ starts from u

Think of an oriented ring

$$P\big(\mathrm{E}(u)\big) \leq 2^{-(2c \log n)} = n^{-2c}$$

A Warm-Up Problem



Lemma:
Monochromatic parts are of length $O(\log n)$ with high probability

$$1 - 1/n^c$$

Event $\mathrm{E}(u)$:
A monochromatic path of length at least $2c \log n$ starts from u

Think of an oriented ring

$$P\big(\mathrm{E}(u)\big) \leq 2^{-(2c \log n)} = n^{-2c}$$

Union bound:

$$\Sigma_{u \in V} P\big(\mathrm{E}(u)\big) \ll n^{-c}$$

4-Coloring a Ring

Correctness:
The graph is split into two disjoint parts.

The disjoint parts are colored with disjoint color palettes.

Runtime (w.h.p.):
Monochromatic parts are of length $D = O(\log n)$
with high probability

$$1 - 1/n^c$$

Monochromatic parts are colored in parallel in time $O(D)$.

Outline

- Intro
  - What this is not
- Coloring
  - $\Delta + 1$ colors
- The LOCAL model
- Coloring a ring
  - Color reduction
  - Lower bound

3-Coloring a Ring

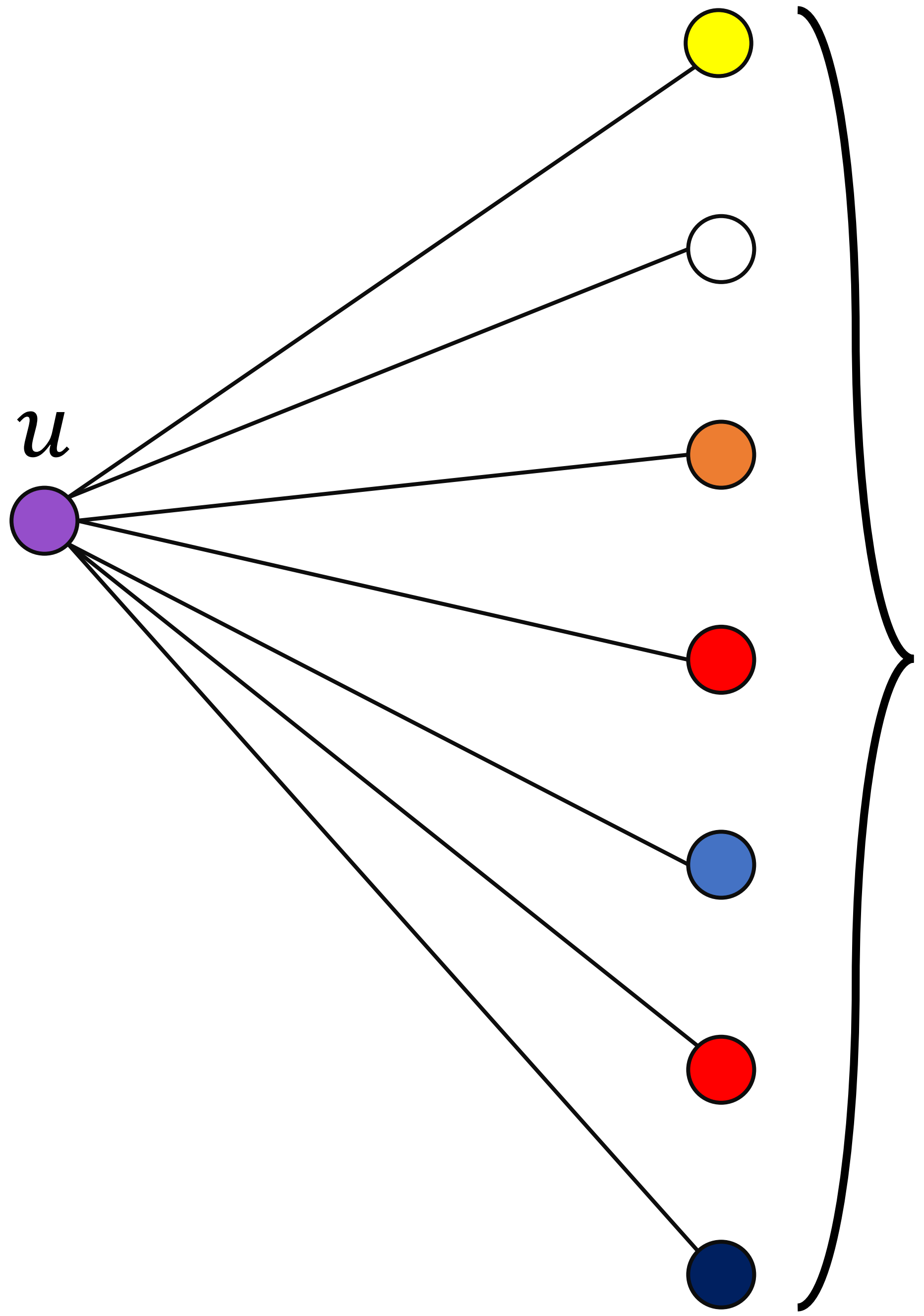
Color Reduction:
Suppose you are *given* a coloring C with $q > \Delta + 1$ colors.

We can turn coloring C into a $(\Delta + 1)$-coloring in $O(q)$ rounds.

Algorithm:
Iterate over the q colors, from highest to lowest.

Always pick the smallest free color.

3-Coloring a Ring



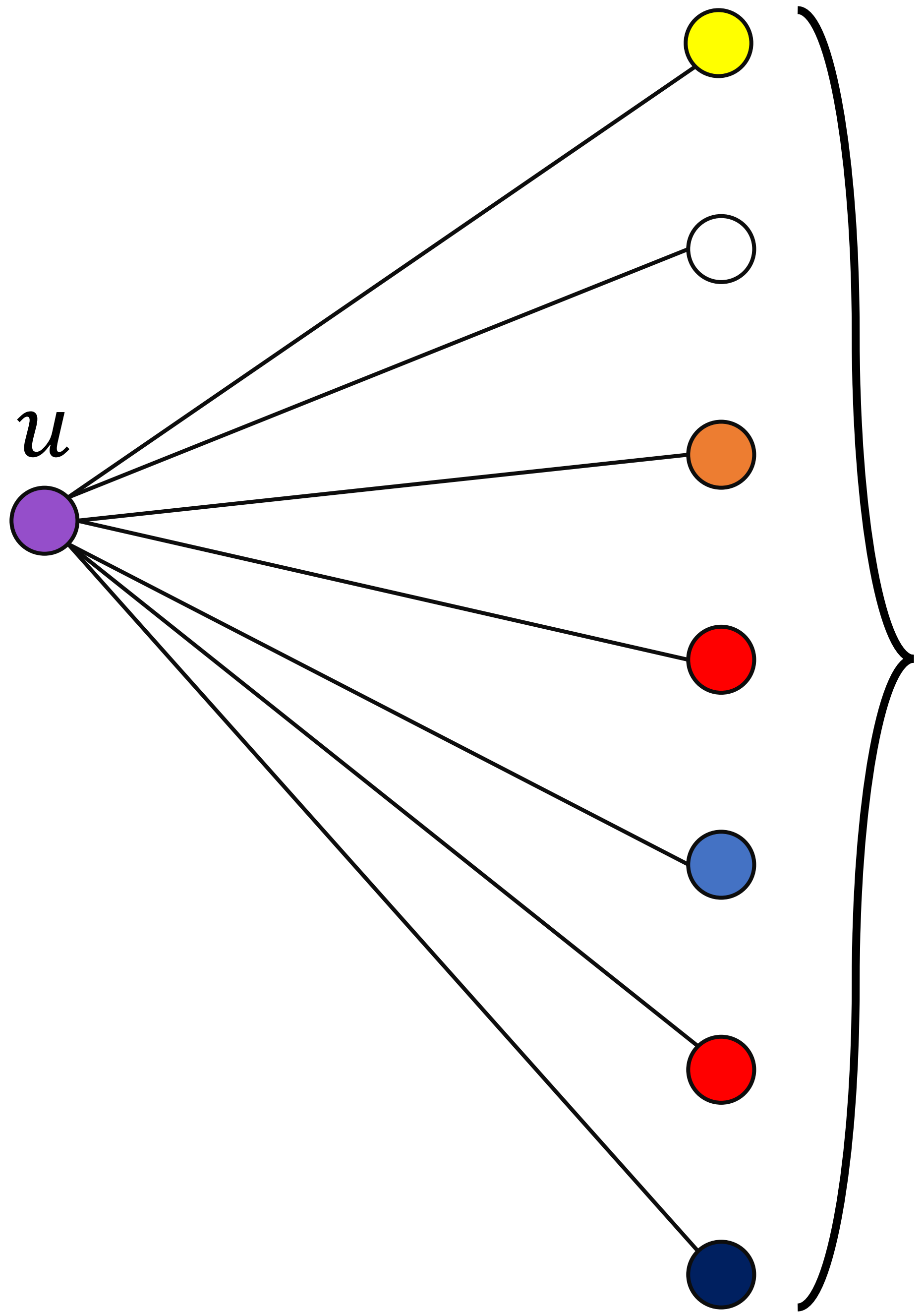
At most Δ different colors.

One out of $\Delta + 1$ is free.

Algorithm:
Iterate over the q colors, from highest to lowest.

Always pick the smallest free color.

3-Coloring a Ring



At most Δ different colors.

One out of $\Delta + 1$ is free.

Algorithm:
Iterate over the q colors, from highest to lowest.

Always pick the smallest free color.

Analysis:
Since C is a coloring, no neighbors pick at the same time.

Runtime clearly $O(q)$.

3-Coloring a Ring

Theorem:
The is a distributed algorithm that finds a 3-coloring of a ring in time $O(\log n)$

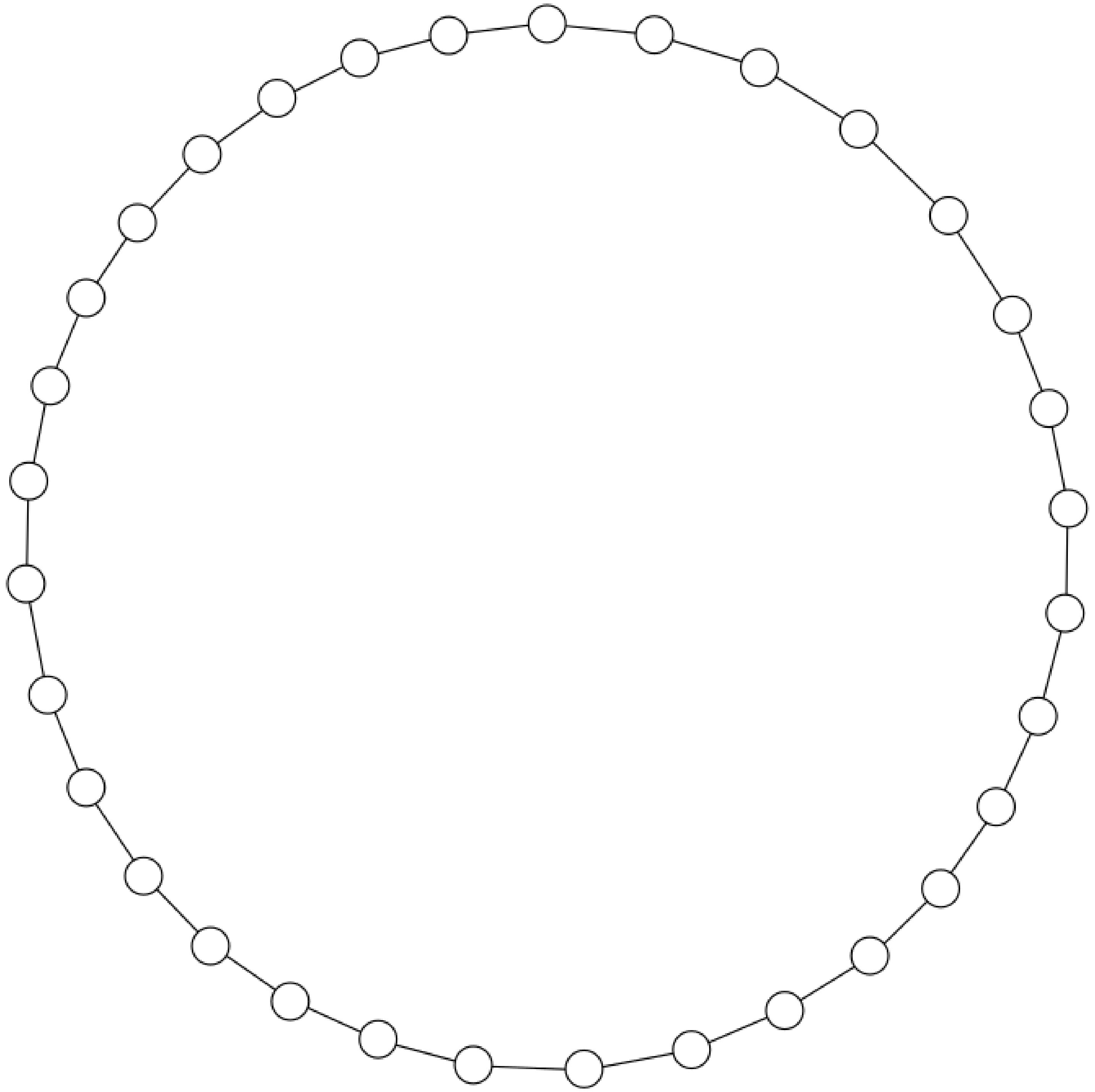
Proof:
The 4-coloring algorithm combined with color reduction.

Outline

- Intro
  - What this is not
- Coloring
  - $\Delta + 1$ colors
- The LOCAL model
- Coloring a ring
  - Color reduction
  - Lower bound

2-coloring

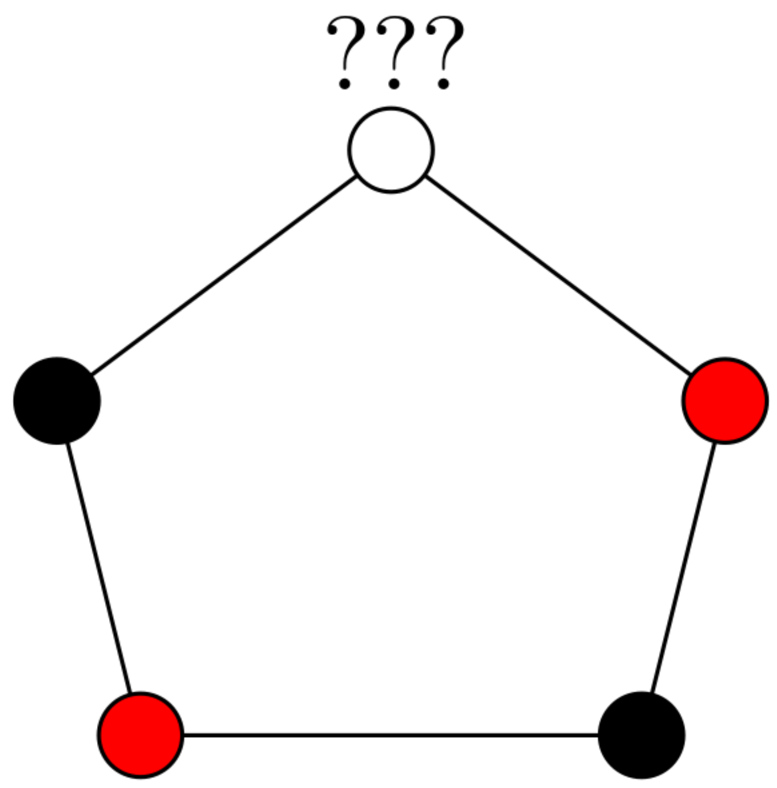
So, why not color with 2-colors?



2-coloring

So, why not color with 2-colors?

Answer 1:
It might not be possible at all.



2-coloring

So, why not color even paths with 2-colors?

2-coloring

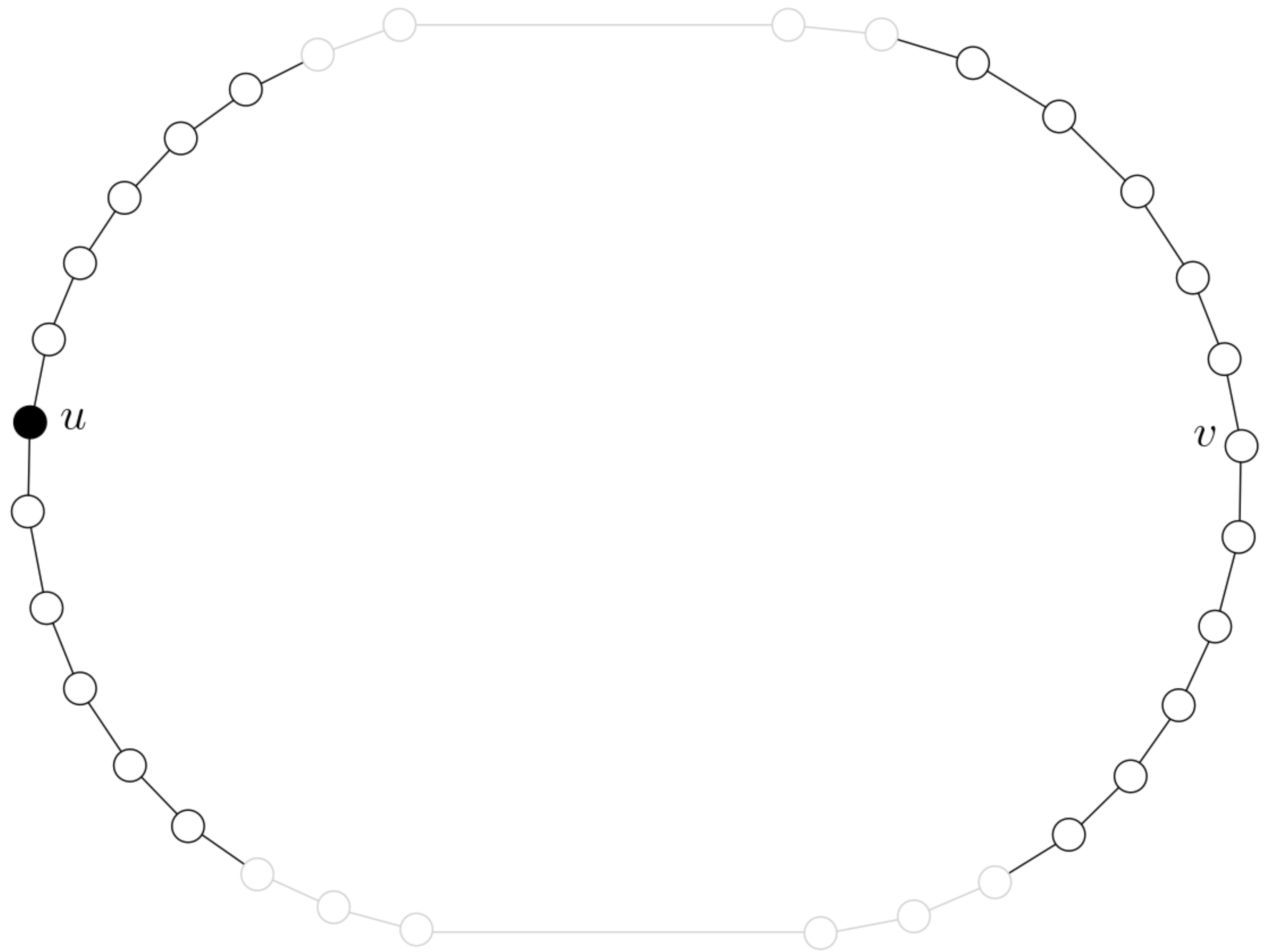
So, why not color even paths with 2-colors?

Suppose we have an algorithm that runs in time $\left(\frac{n}{4}\right)$

2-coloring

So, why not color even paths with 2-colors?

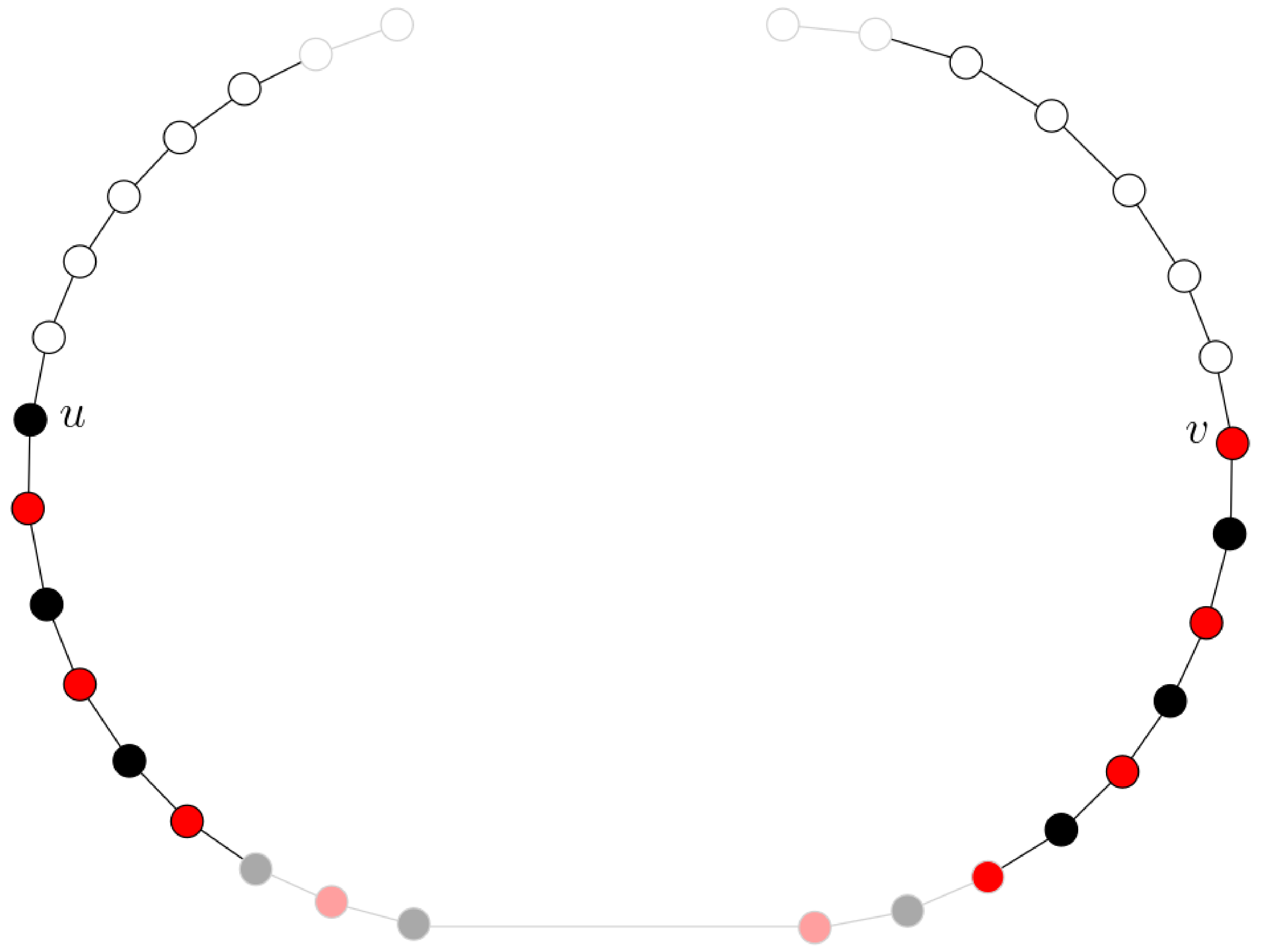
Suppose we have an algorithm that runs in time $\left(\frac{n}{4}\right)$



2-coloring

Color choice of u determines the choice of v.

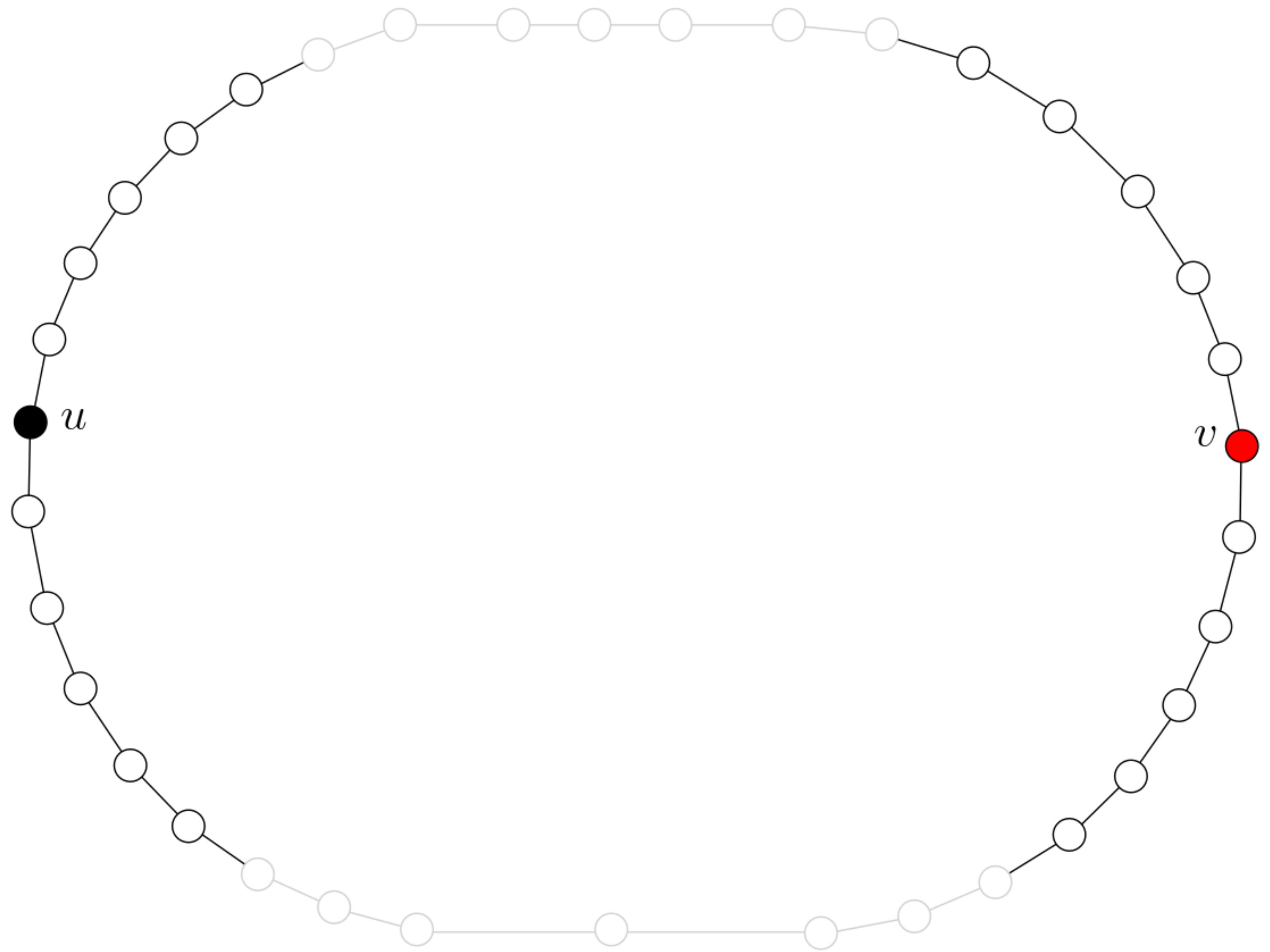
Suppose we have an algorithm that runs in time $\left(\frac{n}{4}\right)$.



2-coloring

Let's change the parity of their distance. The output of v cannot depend on this change.

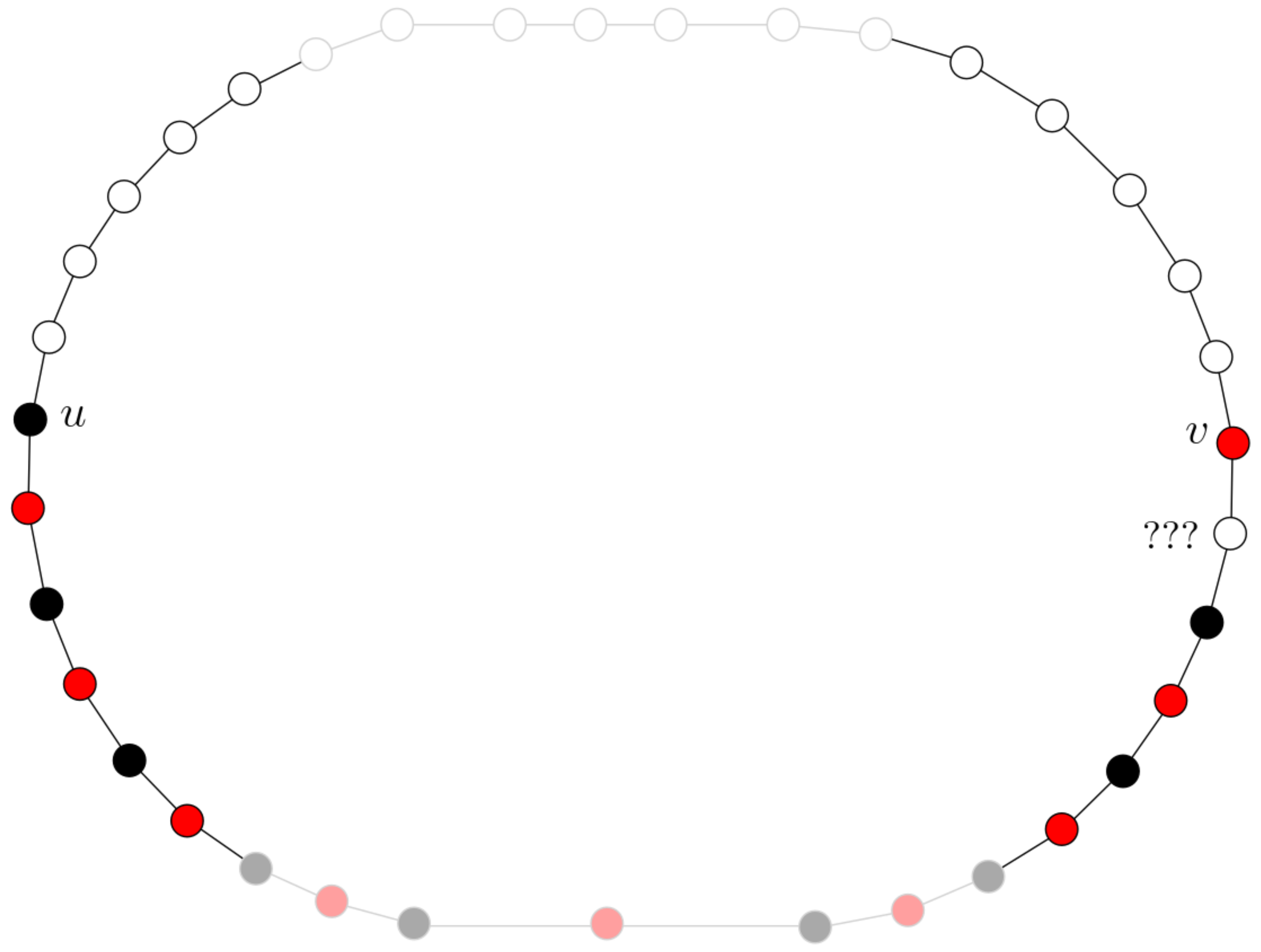
Suppose we have an algorithm that runs in time $\left(\frac{n}{4}\right)$.



2-coloring

Let's change the parity of their distance. The output of v cannot depend on this change.

Suppose we have an algorithm that runs in time $\left(\frac{n}{4}\right)$.



Contradiction. Any algorithm takes $\Omega(n)$ time.

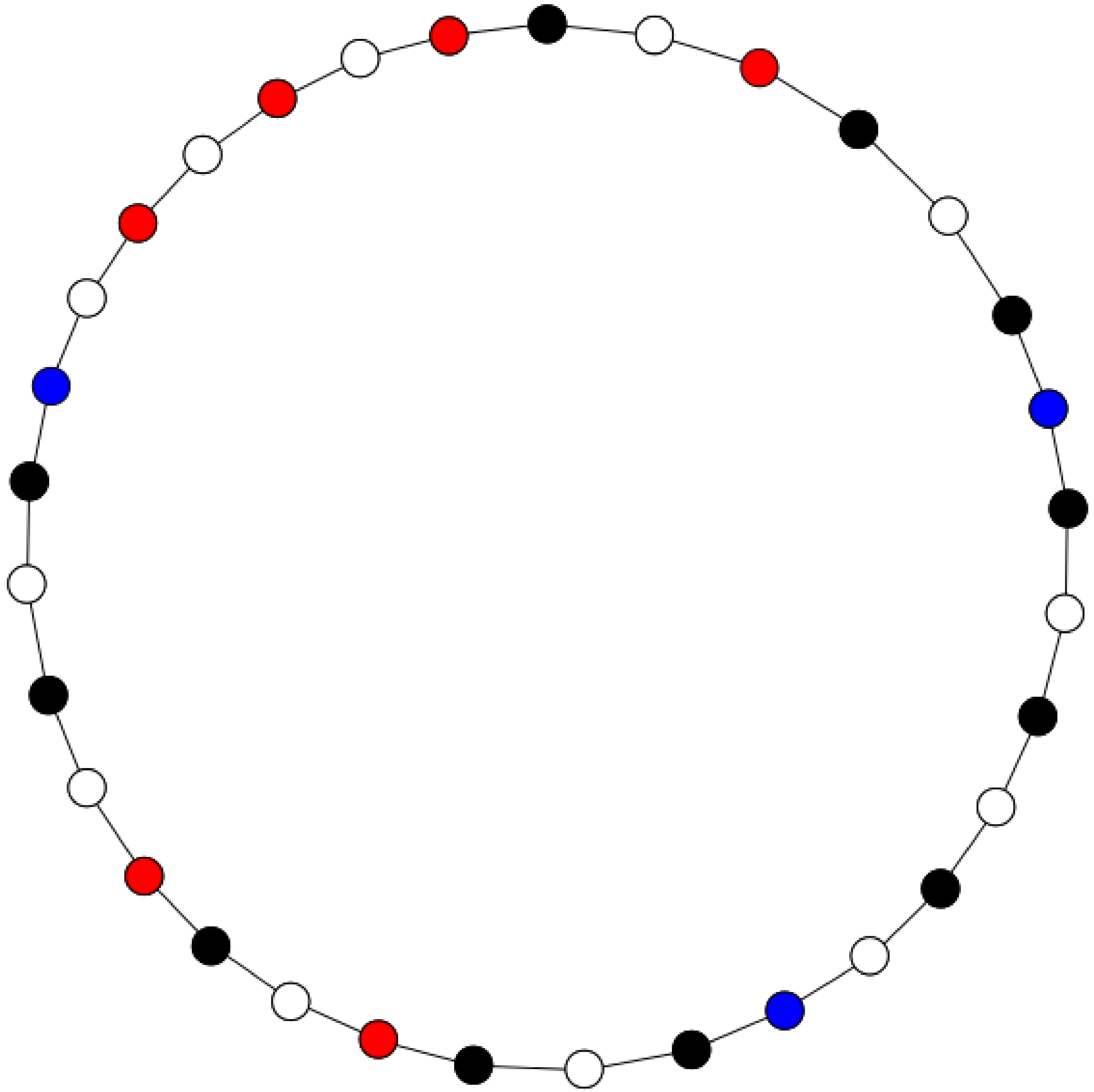
2-Coloring

Theorem:
The complexity of distributed 2-coloring an even ring is $\Theta(n)$.

Literature

3 Coloring a ring
Runtime:
$\Theta(\log^* n)$

[Cole-Vishkin Inf. Contr.'86]
[Linial FOCS '87]



Color Reduction

$(\Delta + 1)$-coloring **in general graphs:**

$$\log^2 \Delta \cdot \log(n)$$

[Ghaffari, Kuhn Nov 2020]

$$\sqrt{\Delta \log \Delta} + \log^* n$$

[Maus, Tonoyan 2020]

Color Reduction

$(\Delta + 1)$-coloring **in general graphs:**

$$\log^2 \Delta \cdot \log(n)$$

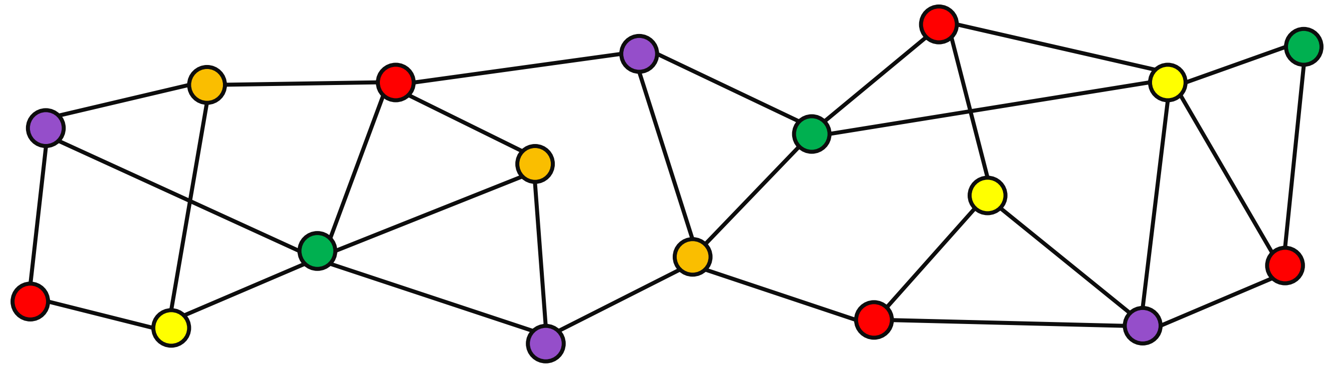
[Ghaffari, Kuhn Nov 2020]

$$\sqrt{\Delta \log \Delta} + \log^* n$$

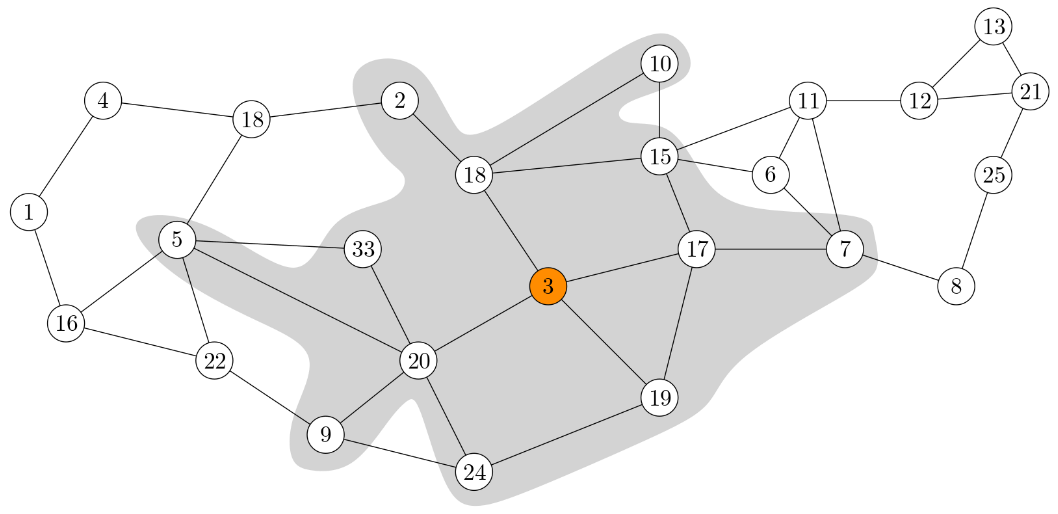
[Maus, Tonoyan 2020]

Huge difference for small values of Δ.

Wrap-up



$(\Delta + 1)$-coloring



The LOCAL model:
How far do we need to communicate?

Coloring a ring:
3 colors: $\Theta(\log n)$
2 colors: $\Theta(n)$