$\Delta + 1$-*coloring is very "local"*

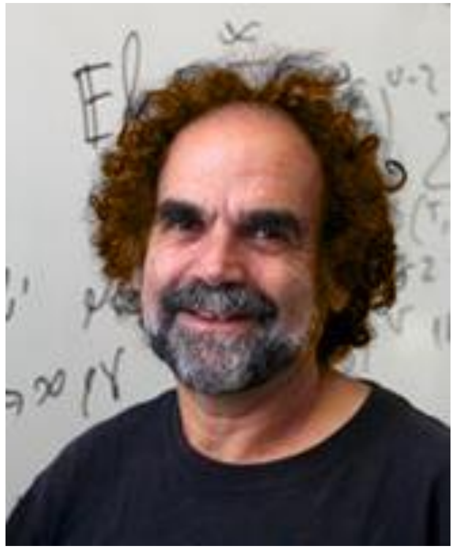
Can we solve it in $O(1)$ rounds?

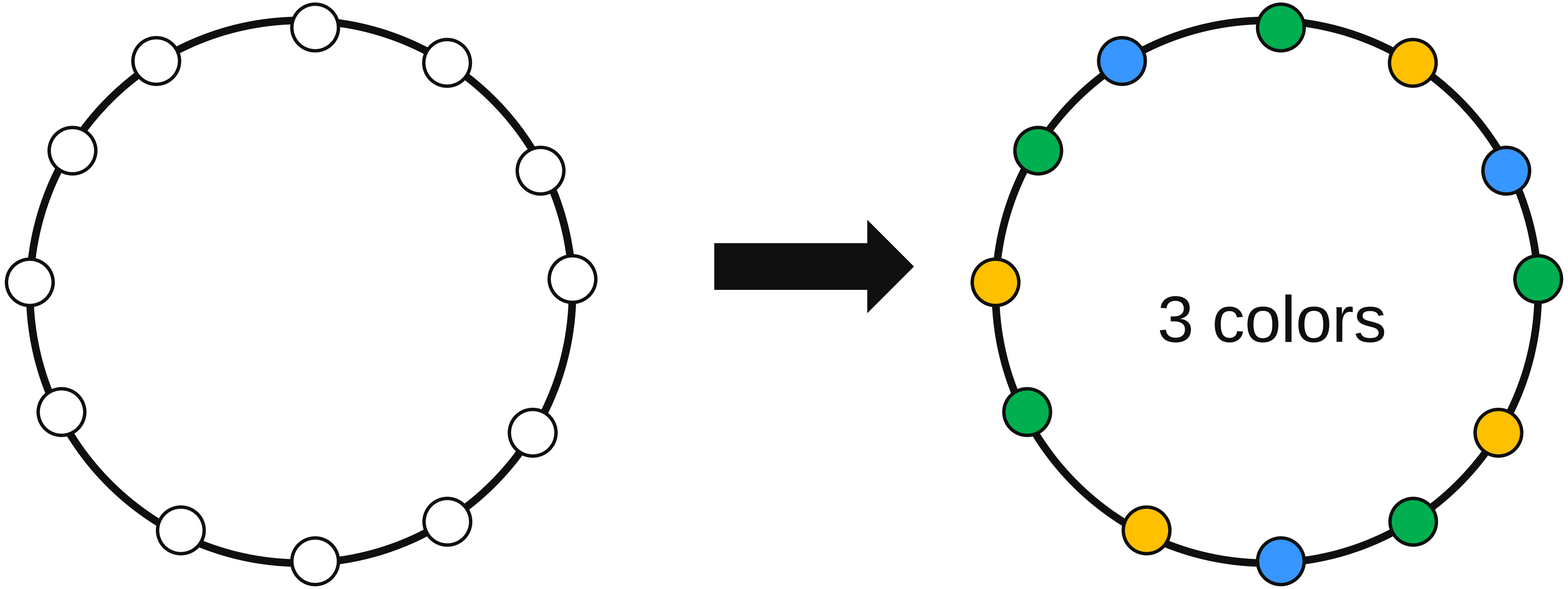
Linial's Seminal Results

Linial's LB: Coloring rings with $\Delta + 1$ colors requires $\Omega(\log^* n)$ rounds

[Linial; FOCS '87]

- taught in every distributed graph algorithms (master) course (1 hour)





Linial’s LB: Coloring rings with $\Delta + 1$ colors requires $\Omega(\log^* n)$ rounds

[Linial; FOCS ‘87]

- taught in every distributed graph algorithms (master) course (1 hour)

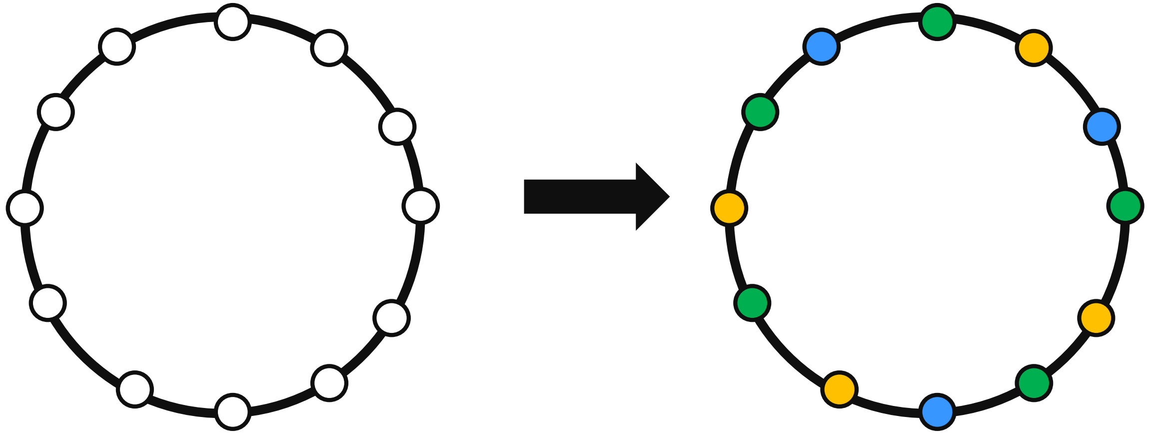
---

extremely small

$\log^* n$: Iterated logarithm

$\log^*($#atoms universe$) = 5$

$$\log^* n = \min\{i \mid \log^{(i)} n \leq 2\}$$
$$\log^{(1)} n = \log n$$
$$\log^{(i+1)} n = \log(\log^{(i)} n)$$



Linial's Seminal Results

Linial's LB: Coloring rings with $\Delta + 1$ colors requires $\Omega(\log^* n)$ rounds

[Linial; FOCS '87]

- taught in every distributed graph algorithms (master) course (1 hour)

---

extremely small

$\log^* n$: Iterated logarithm

$\log^*($#atoms universe$) = 5$

$$\log^* n = \min\{i \mid \log^{(i)} n \le 2\}$$
$$\log^{(1)} n = \log n$$
$$\log^{(i+1)} n = \log\left(\log^{(i)} n\right)$$

Linial's algorithm: $O(\log^* n)$ rounds for $9\boldsymbol{\Delta^2}$**-coloring** (any max degree Δ graph).

[Linial; FOCS '87]

Pro: Tight in terms of the LB

Downside: Many colors: ($\Delta^2 \gg \Delta + 1$),

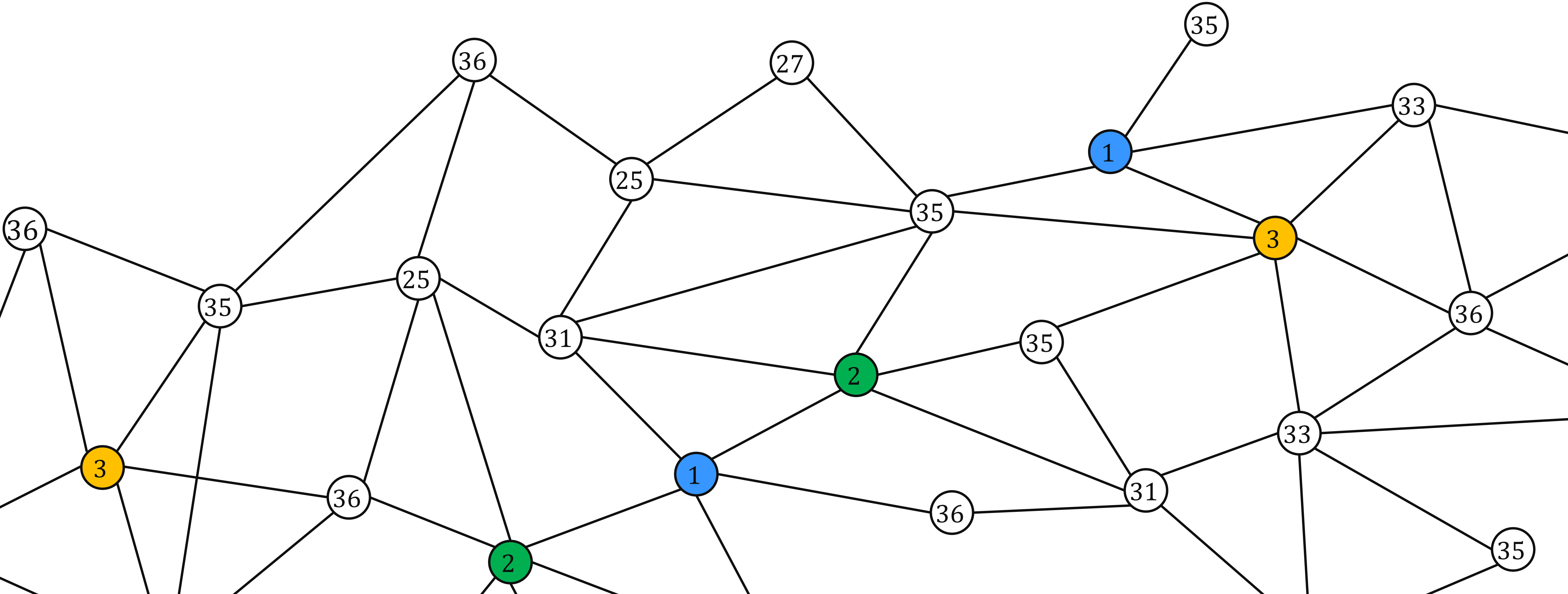
Goal (next few slides):
Color reduction $O(\Delta^2) \rightarrow \Delta + 1$

Simple Color Reduction

Input: 36-coloring

Goal to use colors: 1 2 3 4 5 6 7 ($\Delta = 6$)

Algorithm: reduce one color per round

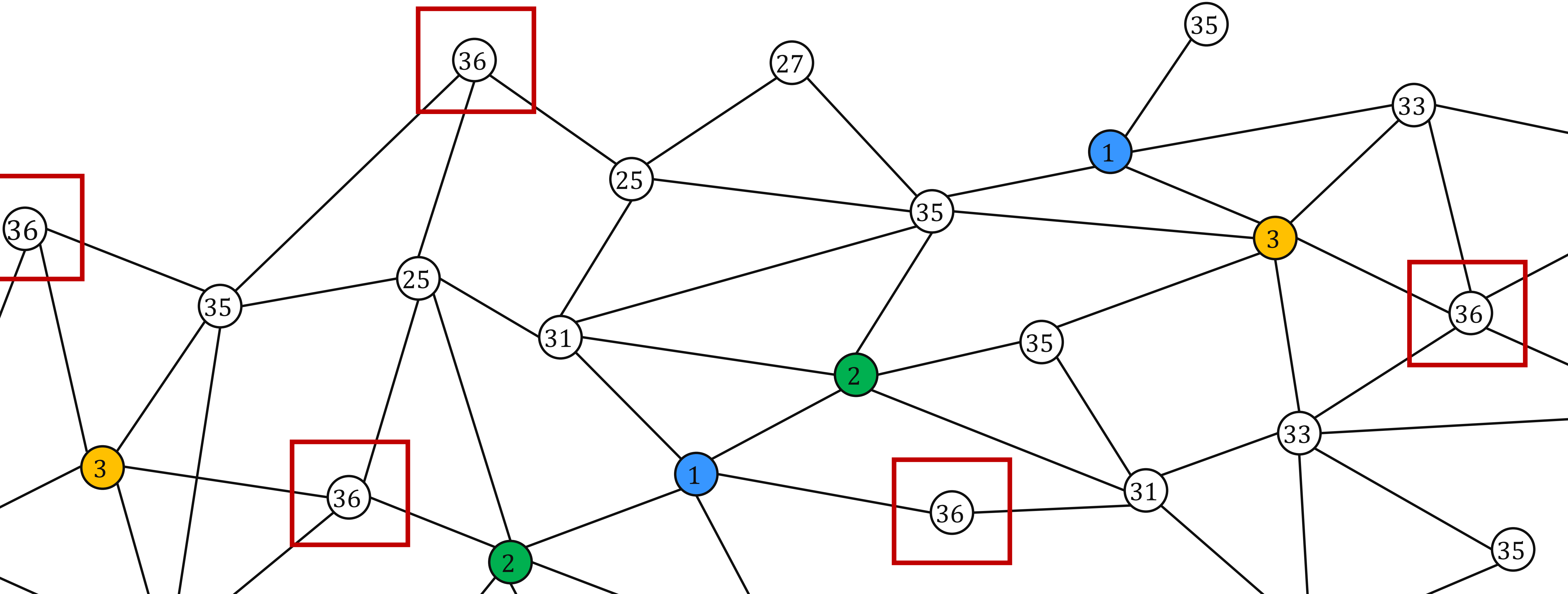


Simple Color Reduction

Input: 36-coloring

Goal to use colors: 1 2 3 4 5 6 7 ($\Delta = 6$)

Algorithm: reduce one color per round

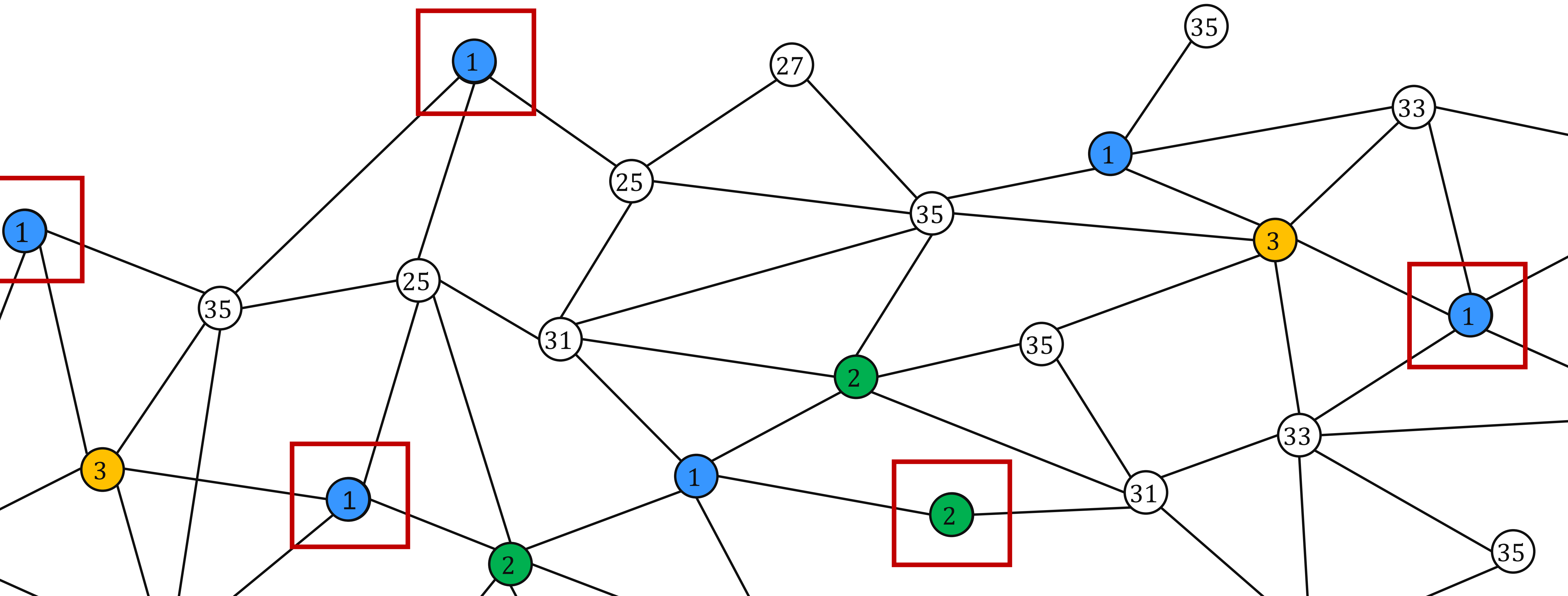


Simple Color Reduction

Input: 36-coloring

Goal to use colors: 1 2 3 4 5 6 7 ($\Delta = 6$)

Algorithm: reduce one color per round

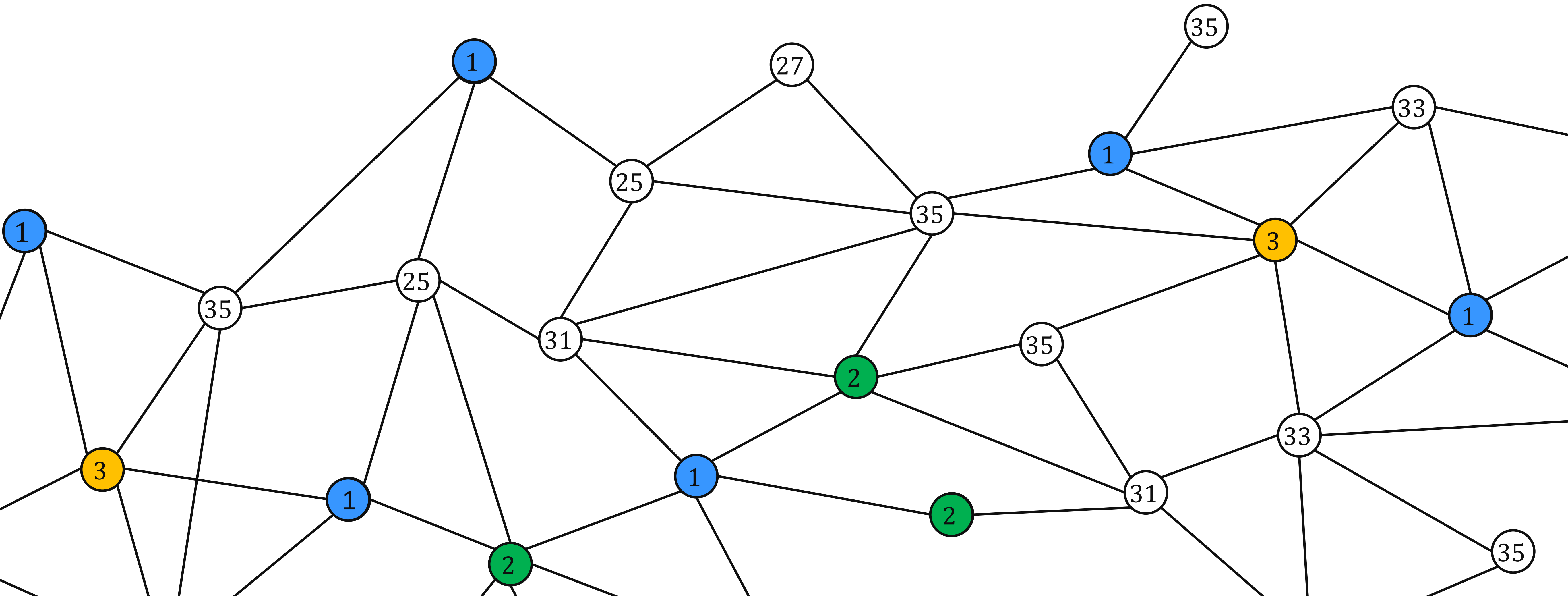


Simple Color Reduction

Input: 36-coloring

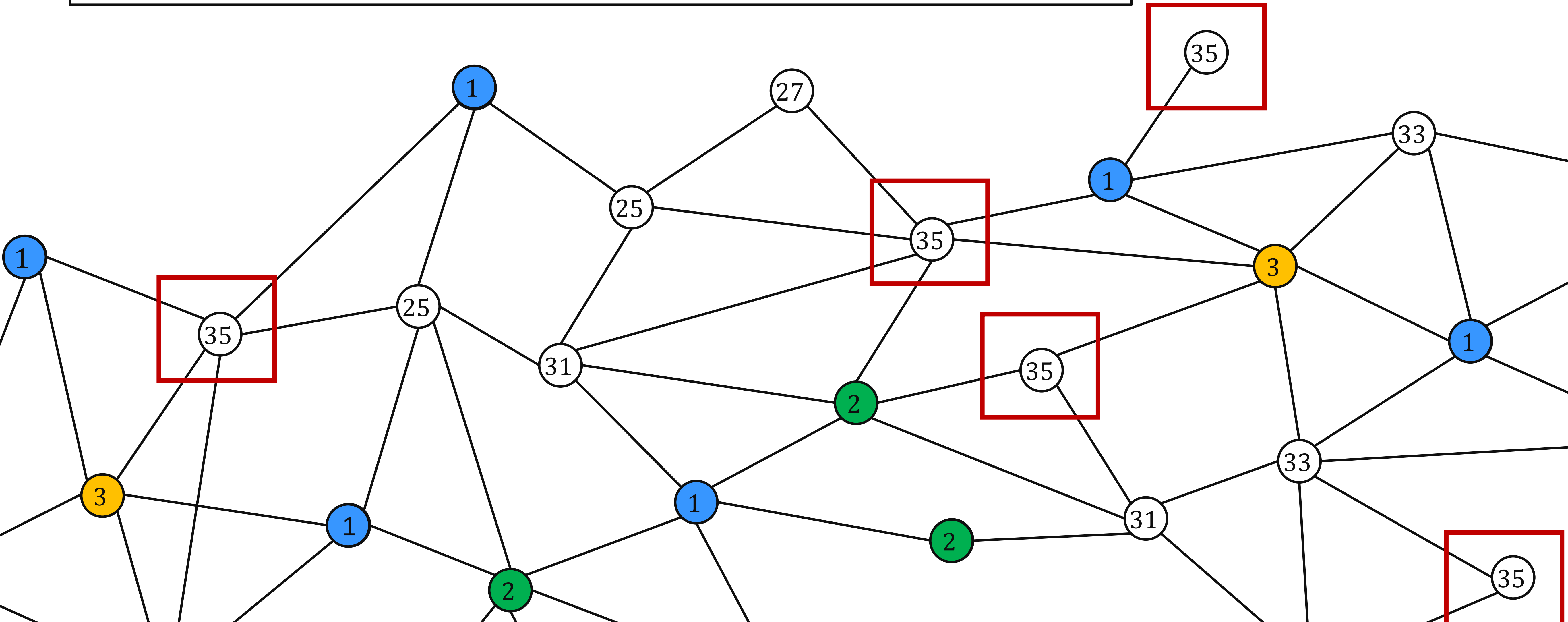
Goal to use colors: 1 2 3 4 5 6 7 ($\Delta = 6$)

Algorithm: reduce one color per round



Simple Color Reduction

Input: 36-coloring
Goal to use colors: 1 2 3 4 5 6 7 $(\Delta = 6)$
Algorithm: reduce one color per round

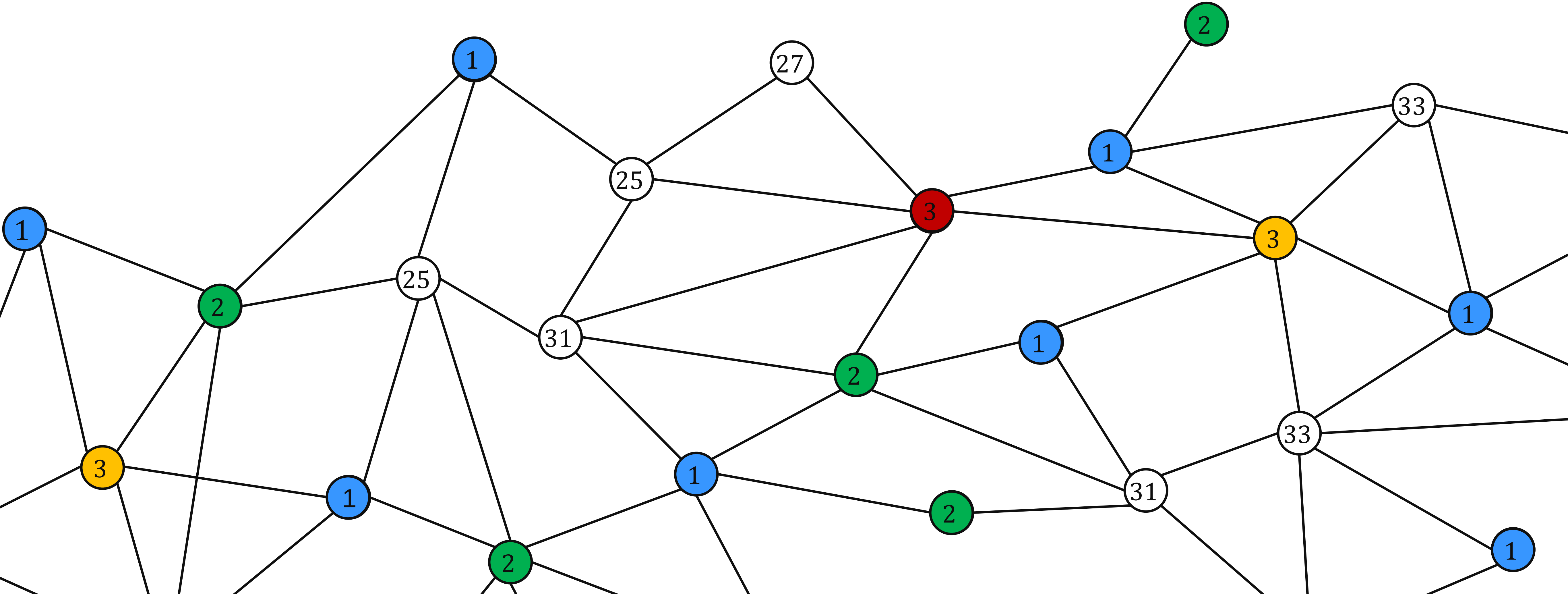


Simple Color Reduction

Input: 36-coloring

Goal to use colors: 1 2 3 4 5 6 7 ($\Delta = 6$)

Algorithm: reduce one color per round



Simple Color Reduction

Input: Graph $G = (V, E)$ with C-vertex input coloring ϕ

Output: Coloring with $\max\{C - 1, \Delta + 1\}$ colors

Each node v executes the following code in parallel

If $\phi(v) \neq C$ **then**

output $\phi(v)$

else

output $min\,\{1, \ldots, \Delta + 1\} \setminus \{\phi(u) \mid u \in N(v)\}$

Messages (2 rounds):

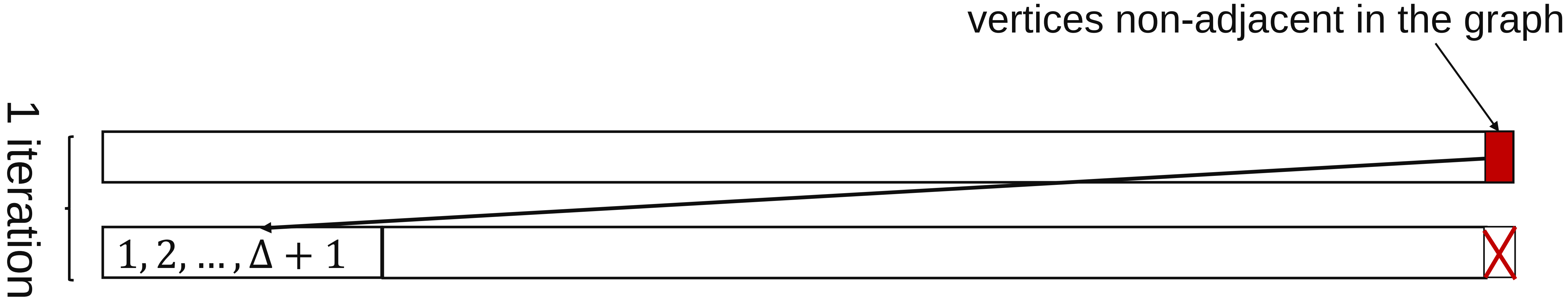
when node v is processed, ask its neighbors $u \in N(v)$ for their color $\phi(u)$.

Messages (1 round, alternative implementation):

every node sends its current color

Simple Color Reduction (SCR)

Simple Color Reduction: There is a 1-round algorithm to compute a $\max\{C-1, \Delta+1\}$-coloring, given a C-coloring.



Why does it work?

- Everyone that recolors itself can always pick a **free color** from $\{1, \dots, \Delta+1\}$, as at most Δ colors are already taken by neighbors
- Nodes that color themselves at the same time cannot be adjacent (proper input coloring)

$(\Delta + 1)$-Coloring

Theorem: $(\boldsymbol{\Delta} + \mathbf{1})$**-coloring can be done in** $\mathbf{O}\left(\boldsymbol{\Delta}^{\mathbf{2}} + \log^* n\right)$ rounds.

Proof: $\text{Linial} \rightarrow 9\Delta^2 \underbrace{\rightarrow (9\Delta^2-1) \rightarrow (9\Delta^2-2) \rightarrow \cdots \rightarrow (\Delta+2) \rightarrow (\Delta+1)}$

$9\Delta^2 - (\Delta + 1) = O(\Delta^2)$ iterations of Simple Color Reduction

Homework assignment: You will develop a faster algorithm for reducing to $\Delta + 1$ colors.