

Overview

- solve a problem using multiple distributed devices
 - data might not fit onto a single device
 - * disk cluster
 - maybe All-to-all communication
 - * no locality restrictions
 - * bandwidth restrictions
- output may be just a part of the solution
- time complexity wrt communication rounds instead of computation time
 - efficient = $\text{polylog}(n) \ll n$
 - $\log^* n$

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

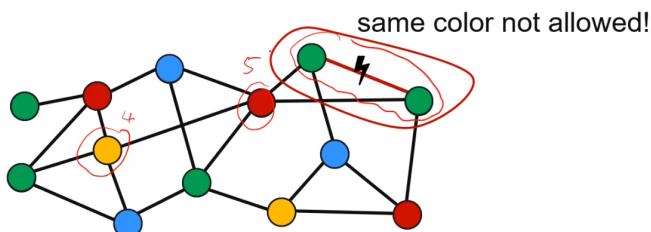
$$\log^{(1)} n = \log n$$

$$\log^{(i+1)} n = \log(\log^{(i)} n)$$

*

Vertex Coloring

- symmetry breaking
 - mutual exclusive access to shared resource
 - colors = time slots => resource scheduling



Goal: Color the graph

- adjacent nodes **different colors**
- prime symmetry breaking problem
- NP-complete

Use $\Delta + 1$ colors!

- Δ is the graph's maximum degree
- sequential greedy algorithm

- color local minima vertices first
 - minima if no neighbor has a higher IP
 - extremely slow
 - linear time
- random coloring

Repeat until colored

- pick a random **free color** c
- keep it, if no *conflict*

Randomized: $O(\log n)$ rounds for 2Δ -coloring, whp.

Randomized: $O(\log n)$ rounds for $(\Delta + 1)$ -coloring, whp.

Simple Color Reduction

- reduce one color per round

– cannot reduce below $\delta + 1$ colors

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

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

- from $O(\delta^2)$ to $\delta + 1$ colors in $O(\delta^2)$ rounds

Theorem: $(\Delta + 1)$ -coloring can be done in $O(\Delta^2 + \log^* n)$ rounds.

Linial's Algorithm

- theorems

~~Harder Colorings~~

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

Linial's algorithm: $O(\log^* n)$ rounds for $O(\Delta^2)$ -coloring.

* Δ = highest degree

- Linial's color reduction

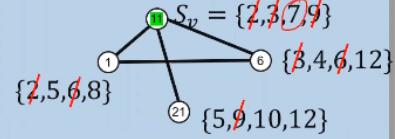
* Given an m -coloring, we can compute a $(100\Delta^2 \cdot \log m)$ -coloring in 1 round.

* Given a Δ^3 -coloring, we can compute a $(100\Delta^2)$ -coloring in 1 round.

Algorithm (1 round):

- node v selects a candidate color set $S_v \subseteq [100\Delta^2]$
- v picks color in

$$S_v \setminus \bigcup_{w \in N(v)} S_w = \emptyset$$



Posture's Postulate (x 2x)

Fix prime $q \in [10 \cdot \Delta, 20\Delta]$. Polynomials p_0, p_1, p_2, \dots of degree 3 over \mathbb{F}_q
(we assume just $q = 10\Delta$) $p: \mathbb{F}_q \rightarrow \mathbb{F}_q, x \mapsto p(x) \quad \begin{matrix} \approx \{0, 1, 2, 3, 4, 5, 6\} \\ (+, \cdot) \text{ modulo } q \end{matrix}$

Node v with input color $i \in \Delta^3$ picks polynomial p_i ($\# \text{polynomials} = q^{d+1} \geq \Delta^3$)

Low intersecting sets

$$S_v = S_i = \{(x, p_i(x)) \mid x \in \mathbb{F}_q\} \subseteq \mathbb{F}_q \times \mathbb{F}_q$$

$$\begin{aligned} p(x) &= 5x^3 + 2x^2 + 3x + 1 & (10\Delta)^4 &\geq \Delta^3 \\ p(x) &= ax^3 + bx^2 + cx + d & a, b, c, d &\in \{0, \dots, q-1\} \\ & q^4 & q^4 \end{aligned}$$

Example:

prime $q = 7$.

$$p_v(x) = 6x^3 + 4x^2 + 3 \in \mathbb{F}_q$$

Low intersecting sets

$$S_v = S_i = \{(x, p_i(x)) \mid x \in \mathbb{F}_q\} \subseteq \mathbb{F}_q \times \mathbb{F}_q$$

$$\begin{aligned} S_v &= \{(0, p_v(0)), (1, p_v(1)), (2, p_v(2)), (3, p_v(3)), (4, p_v(4)), (5, p_v(5)), (6, p_v(6))\} \\ S_v &= \{(0,3), (1,13), (2,67), (3,201), (4,451), (5,853), (6,1443)\} \quad \text{mod } q = \text{mod } 7 \\ S_v &= \{(0,3), (1,6), (2,4), (3,5), (4,3), (5,6), (6,1)\} \end{aligned}$$

- Interpret each tuple (x, y) as an output color
- $|S_v| = q = 7$

◆ small intersection set

■ Two polynomials of degree d over a prime field can intersect in at most d points.



Input coloring provides enough symmetry breaking to assign distinct polynomials to neighboring nodes

Input color $i \mapsto$ polynomial $p_i \mapsto$ set S_i

(adjacent nodes do not receive the same set)

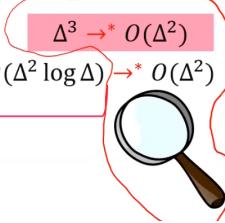


No communication!

◆ ID-assignment is a coloring $|\text{ID-SPACE}| = N$ colors

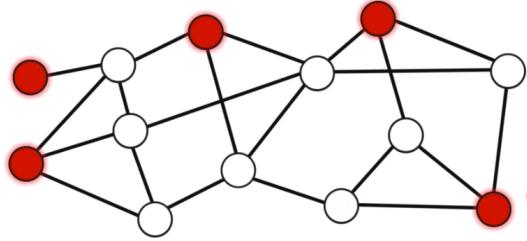


N -coloring $\rightarrow O(\Delta^2 \log N) \rightarrow O(\Delta^2 (\log \Delta + \log \log N) \rightarrow \dots \rightarrow O(\Delta^2 \log \Delta) \rightarrow O(\Delta^2)$



$O(\log^* N)$ iterations/rounds

Maximal Independent Set



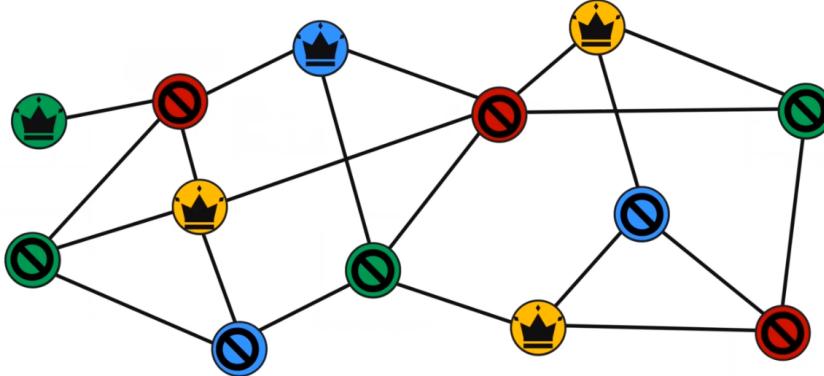
A **maximal independent set (MIS)** of a graph $G = (V, E)$ is a subset $S \subseteq V$ of the nodes such that

- S is an **independent set** in G (no two vertices in S are adjacent)
- S is **maximal**, that is, for any $v \in V \setminus S$, the set $S \cup \{v\}$ is **not** an independent set
- at the end each node knows whether it is in the independent set
- different from the maximum independent set problem
 - independent set need not be of maximum size
- naive centralized greedy algorithm
 - $S = \emptyset$.
 - iterate through the vertices in arbitrary order
 - add a node to S , if none of its neighbors is already in S .
- distributed version
 - **Input:** Graph $G = (V, E)$ with **C-vertex coloring** ϕ
 - executed by each node v in parallel/synchronous
 - $S = \emptyset$
 - For** $i=1$ to C
 - If** $\phi(v) = i$ **then**
 - if** $N(v) \cap S = \emptyset$, **join** S and **notify** neighbors
 - Output** whether $v \in S$.
 - coloring acts as scheduler
 - notifying/asking neighbors
 - * implementation dependent
 - * may take 2 additional rounds
 - * does not change time complexity

```

For i=1 to C
  If  $\phi(v) = i$  then    
    if  $N(v) \cap S = \emptyset$ , join S and notify neighbors

```



There is a deterministic distributed algorithm to compute an MIS in $O(C)$ rounds, given a C -vertex coloring.

- Proof (correctness, independent set):**

Notation: $S_i, i = 0, \dots, C$, the set S after iteration i

Induction: S_i is an independent set.

Induction start: Initially $S_0 = \emptyset$ is an IS

Induction Step ($S_i \rightarrow S_{i+1}$):

Assume for contradiction that there exists adjacent nodes $u, v \in S_{i+1}$:

As S_i is an IS, u or v needs to have input color $i + 1$. Wlog $\phi(v) = i + 1$.

Case 1 ($\phi(u) = i + 1$): Contradiction, as u, v , are adjacent

Case 2 ($\phi(u) < i + 1$): Contradiction, as then $u \in S_i$, and v would not have joined S_i , as the test $N(v) \cap S = \emptyset$ would have been negative.

- Proof (correctness, maximality):**

Assume for contradiction there exists some node $v \in V \setminus S$ such that

$S \cup \{v\}$ is an IS.

As S only keeps growing, this implies that v had no neighbor in $S_{\phi(v)-1}$ when v was processed, and v would have joined $S_{\phi(v)} \subseteq S$, a contradiction.

- Proof (runtime):**

The previous algorithm requires one round of communication per color class to inform neighbors whether one has joined the MIS or not

Corollary:

There is a deterministic distributed algorithm to compute an MIS in

$O(\Delta^2 + \log^* n)$ rounds.

Proof:

Use Linial's algorithm to compute an $O(\Delta^2)$ -coloring in $O(\log^* n)$ rounds. Then use the theorem above.