Randomized Algorithms

Parallelization

- Two main strategies for parallelization
  - Divide & Conquer
  - **Randomization**

Ensure that processors can make local decisions which,
with *high probability*,
add up to good global decisions

Sampling → quicksort

Randomization

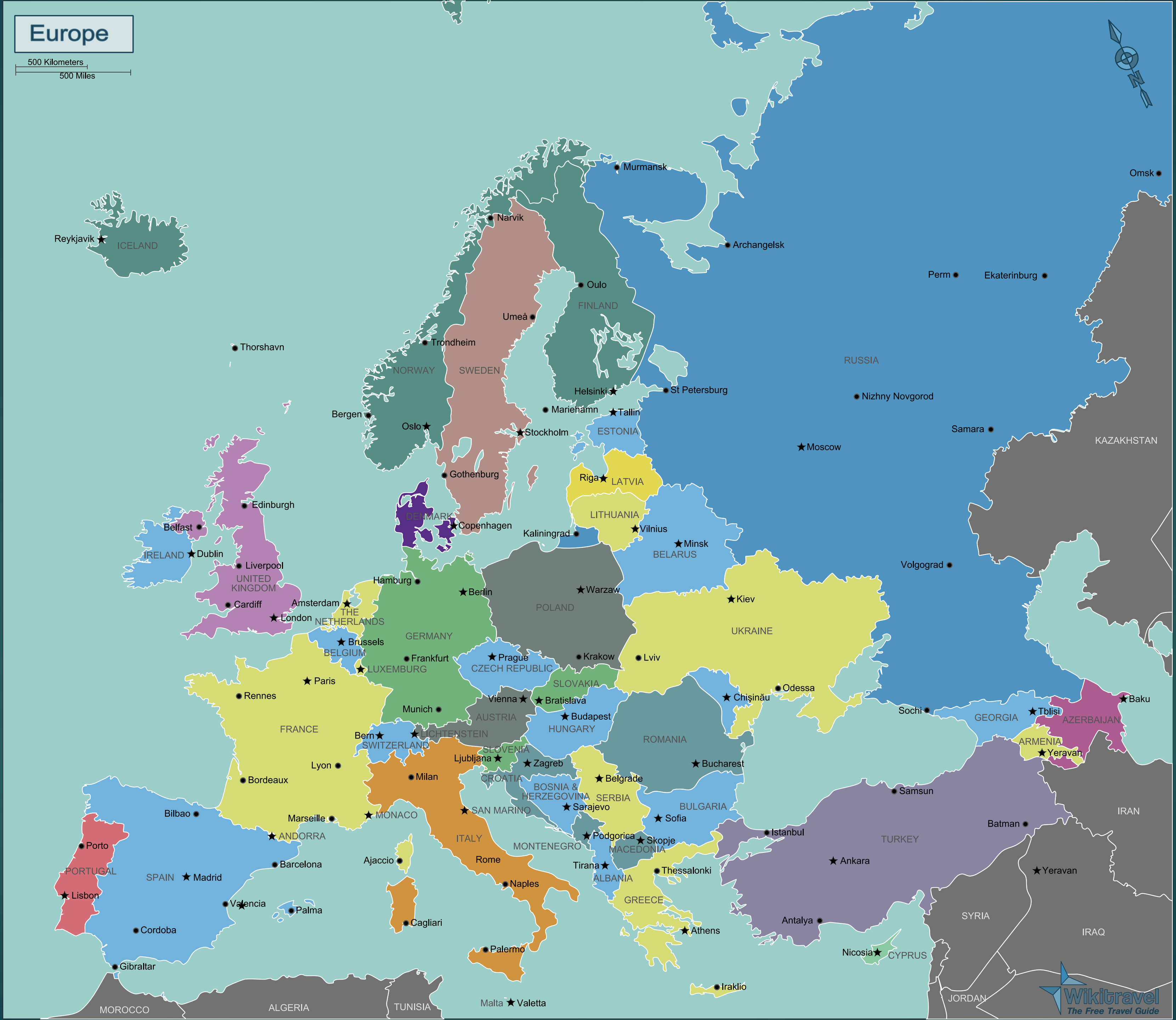
- Sampling
- Symmetry breaking
  - Independent sets → today
- Load balancing

Graph Algorithms

- Will be covered in detail later …
- Graph $\mathcal{G} = (V, E)$, vertices and edges
- Matrices → Graphs
  - Adjacency graph of a matrix A
  - Edge (i, j) exists iff $A_{ij} \neq 0$
    - Edge weight, W_{ij}, can be the A_{ij} value
- Graphs → Matrices
  - Adjacency matrix of a weighted graph
    - Default weight 1, vertex value is in-degree
  - Symmetric → undirected graphs
  - Unsymmetric → directed graphs

Graph Algorithms

- Graph partitioning
  - NP Hard problem
  - We will cover in detail
  - Coloring
- Graph Laplacian & Eigenproblem
- Breadth First Search (BFS)
- Depth First Search (DFS)
- Connected components
  - Spanning Trees



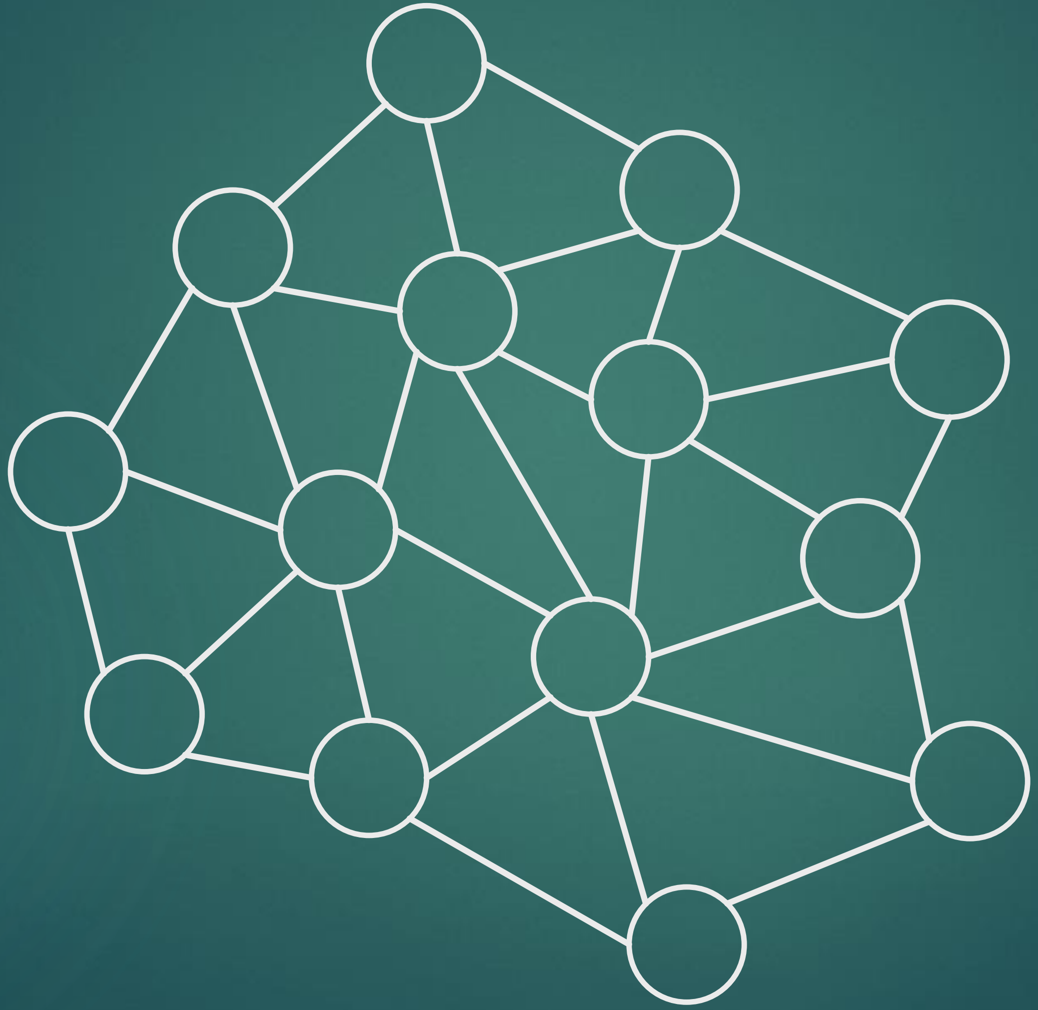
CS 5965/6965 - Big Data Systems - Fall 2014

Sequential Greedy Algorithm

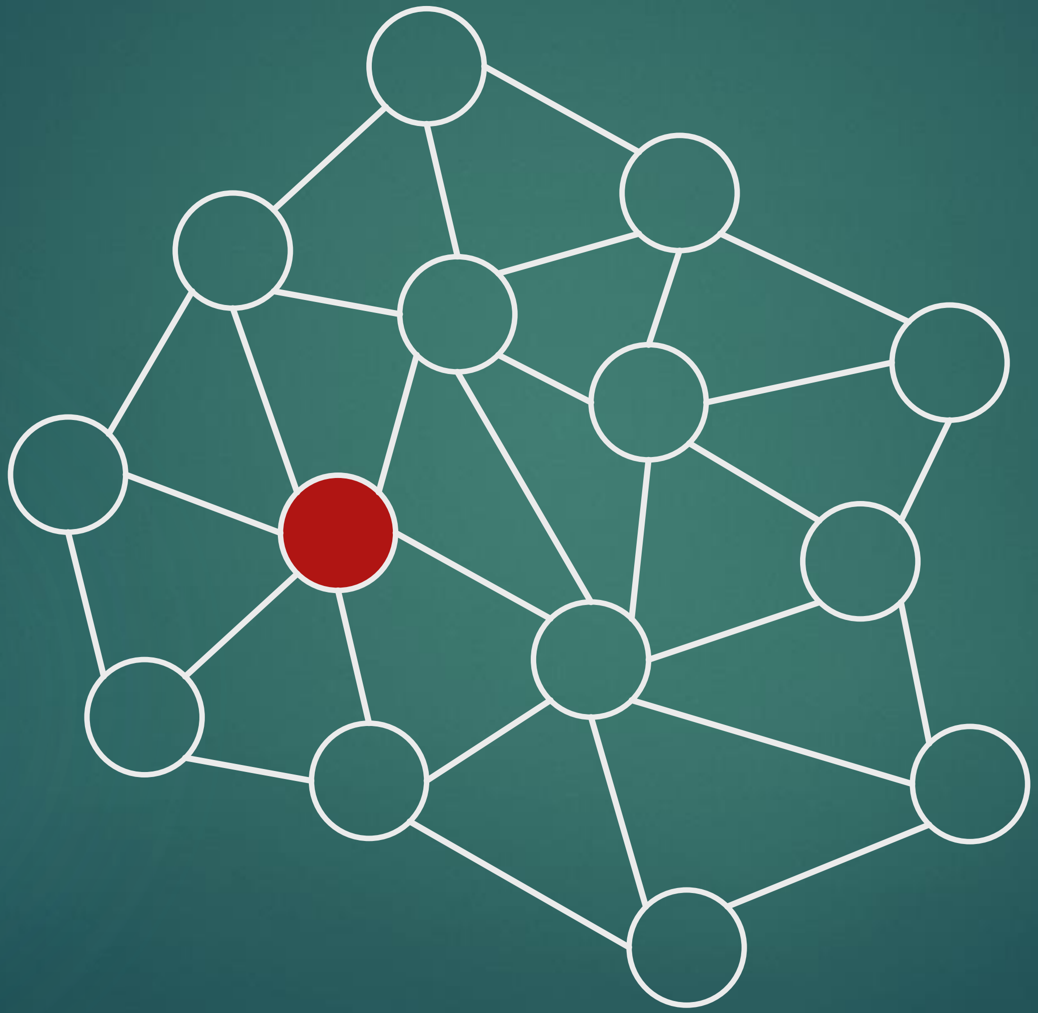
- $n = |V|$
- Choose a random permutation $p(1), \dots, p(n)$ of numbers $1, \dots, n$
- $U \leftarrow V$
- for $i \leftarrow 1$ to n
  - $v \leftarrow p(i)$
  - $S \leftarrow \{\text{colors of all colored neighbors of } v\}$
  - $c(v) \leftarrow \text{smallest color} \notin S$
  - $U \leftarrow U \setminus \{v\}$

Bounds?

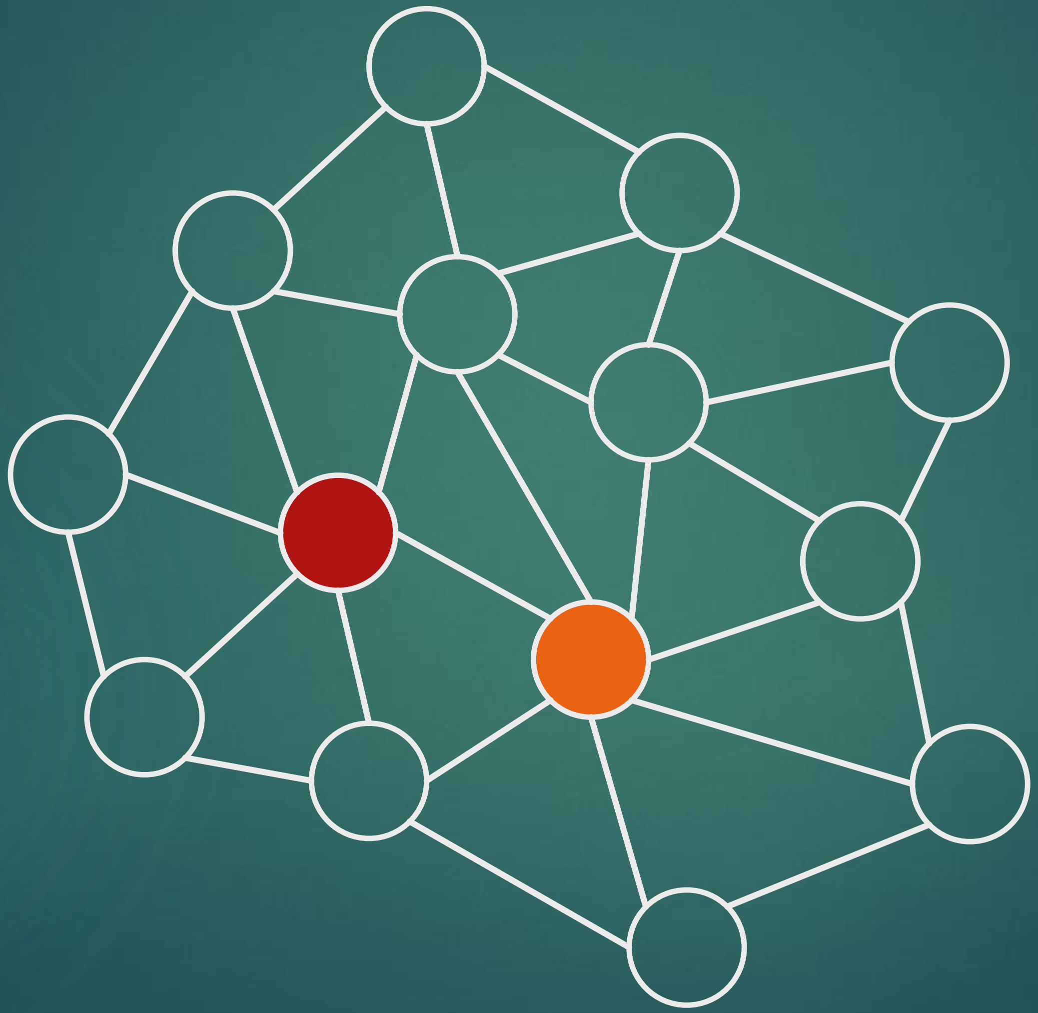
Sequential Greedy Algorithm



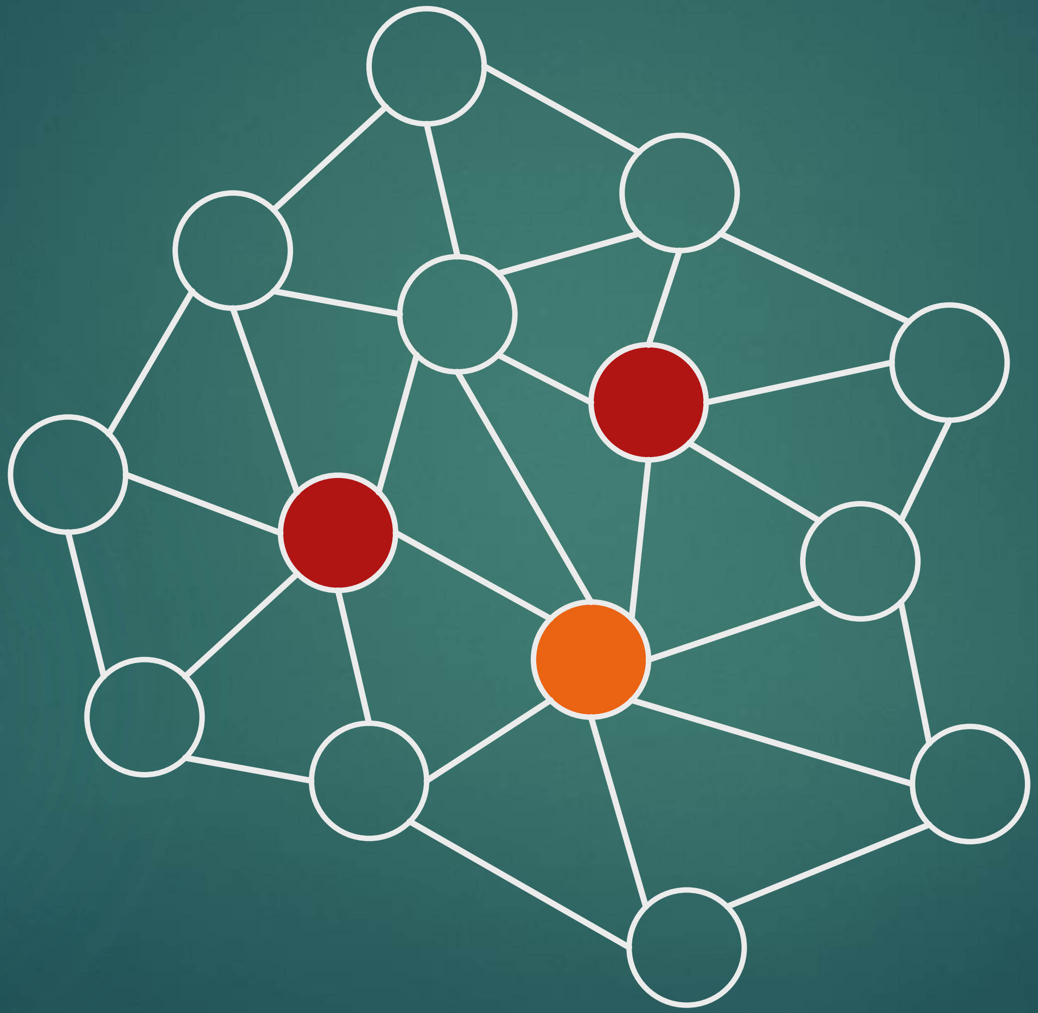
Sequential Greedy Algorithm



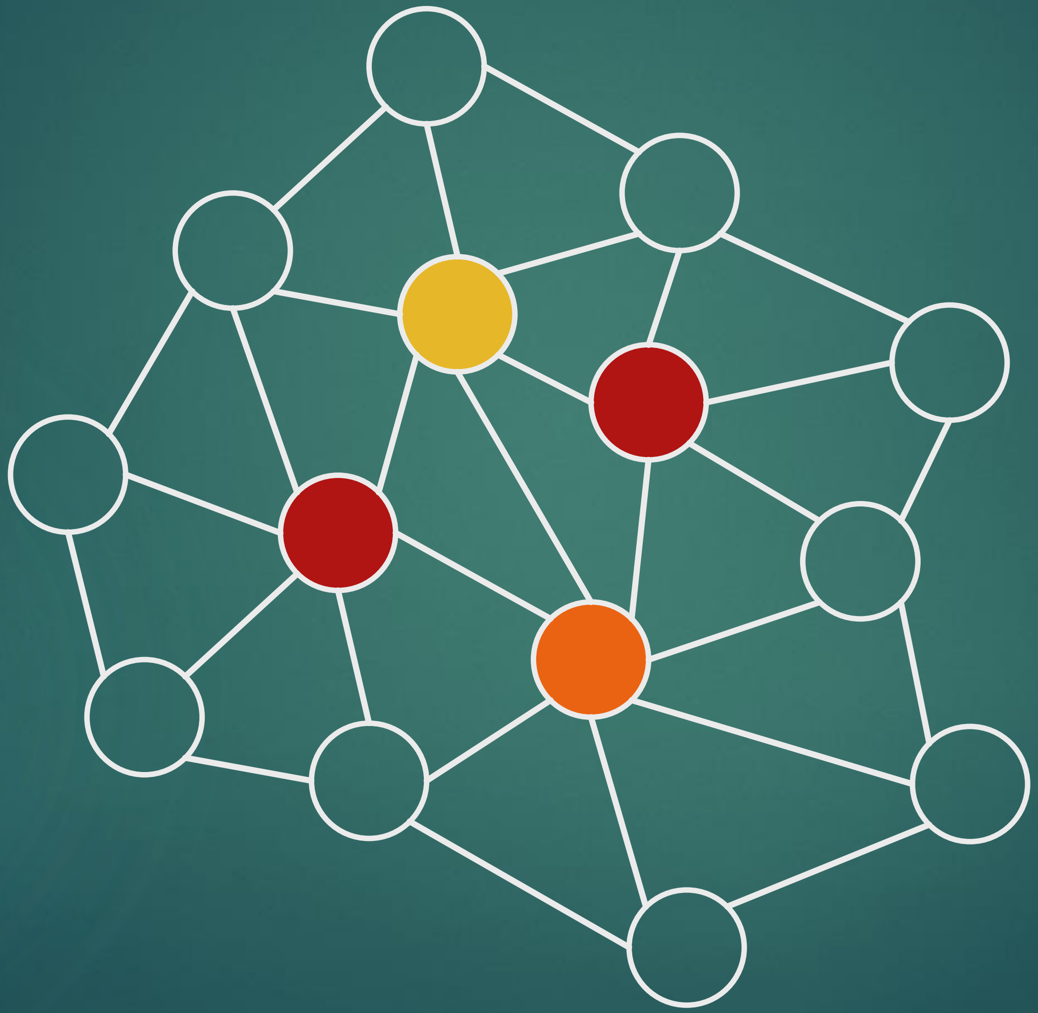
Sequential Greedy Algorithm



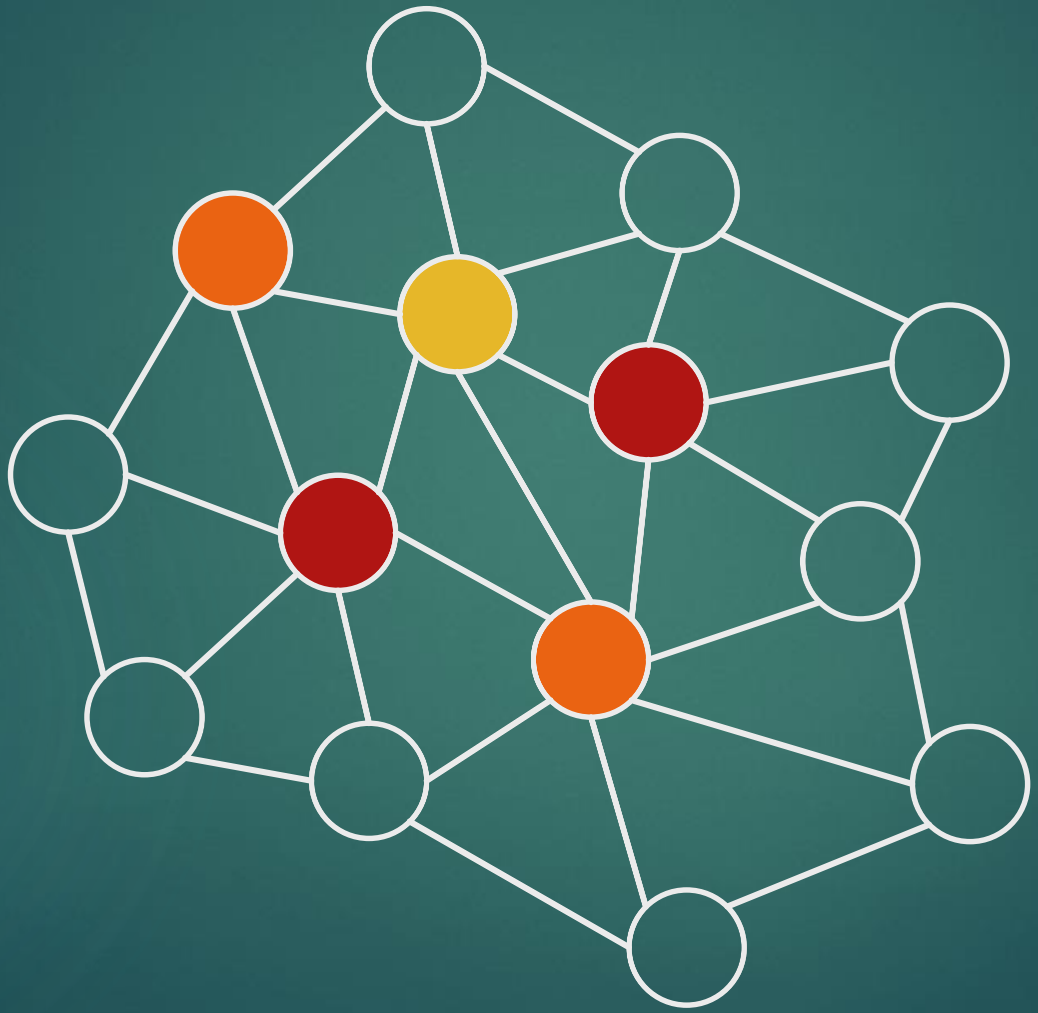
Sequential Greedy Algorithm



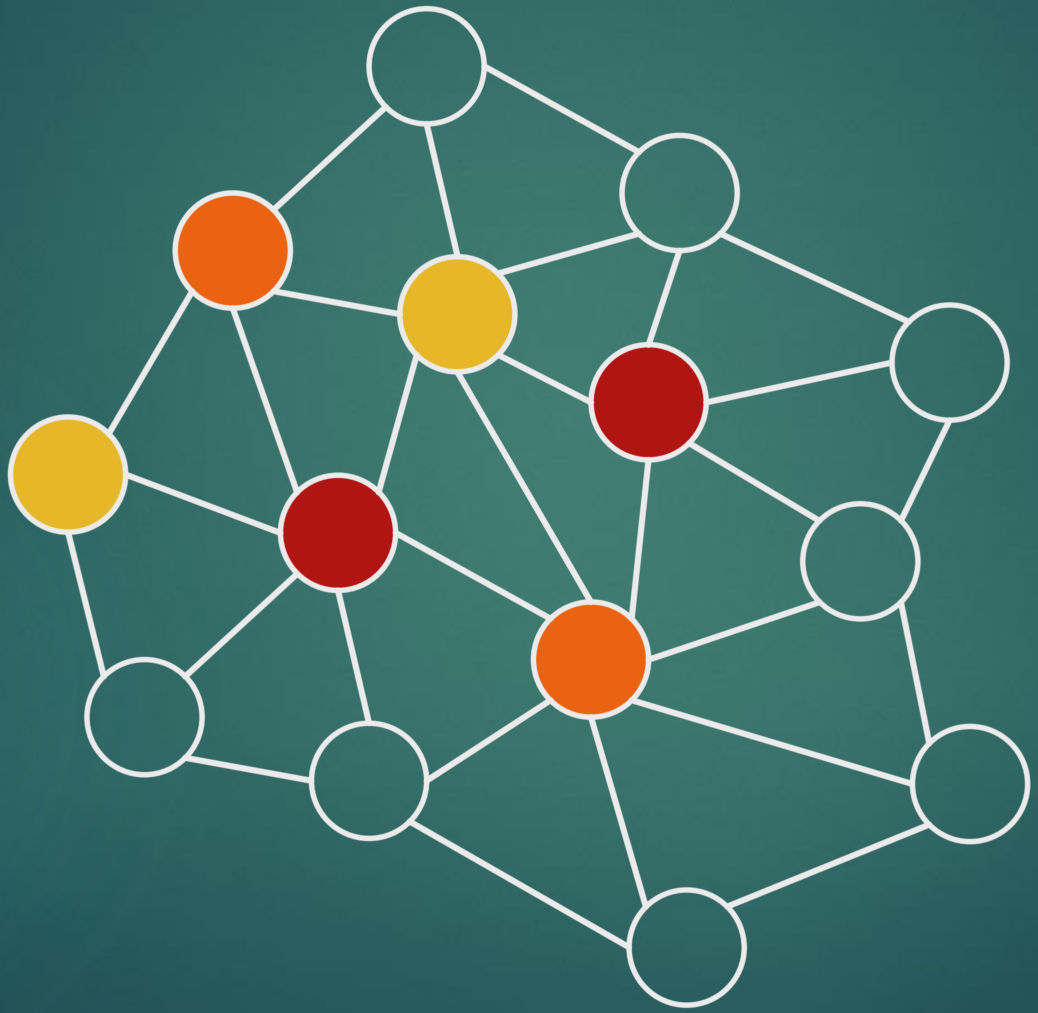
Sequential Greedy Algorithm



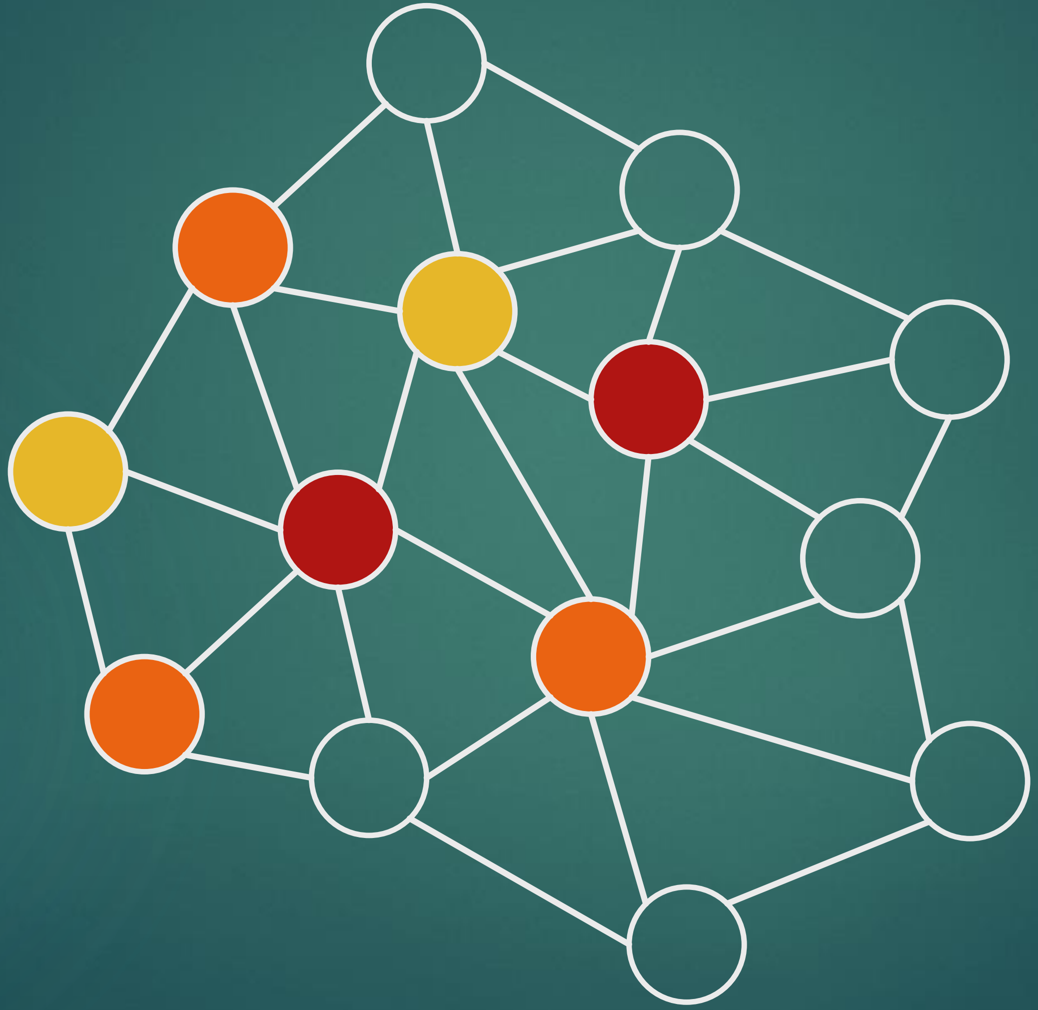
Sequential Greedy Algorithm



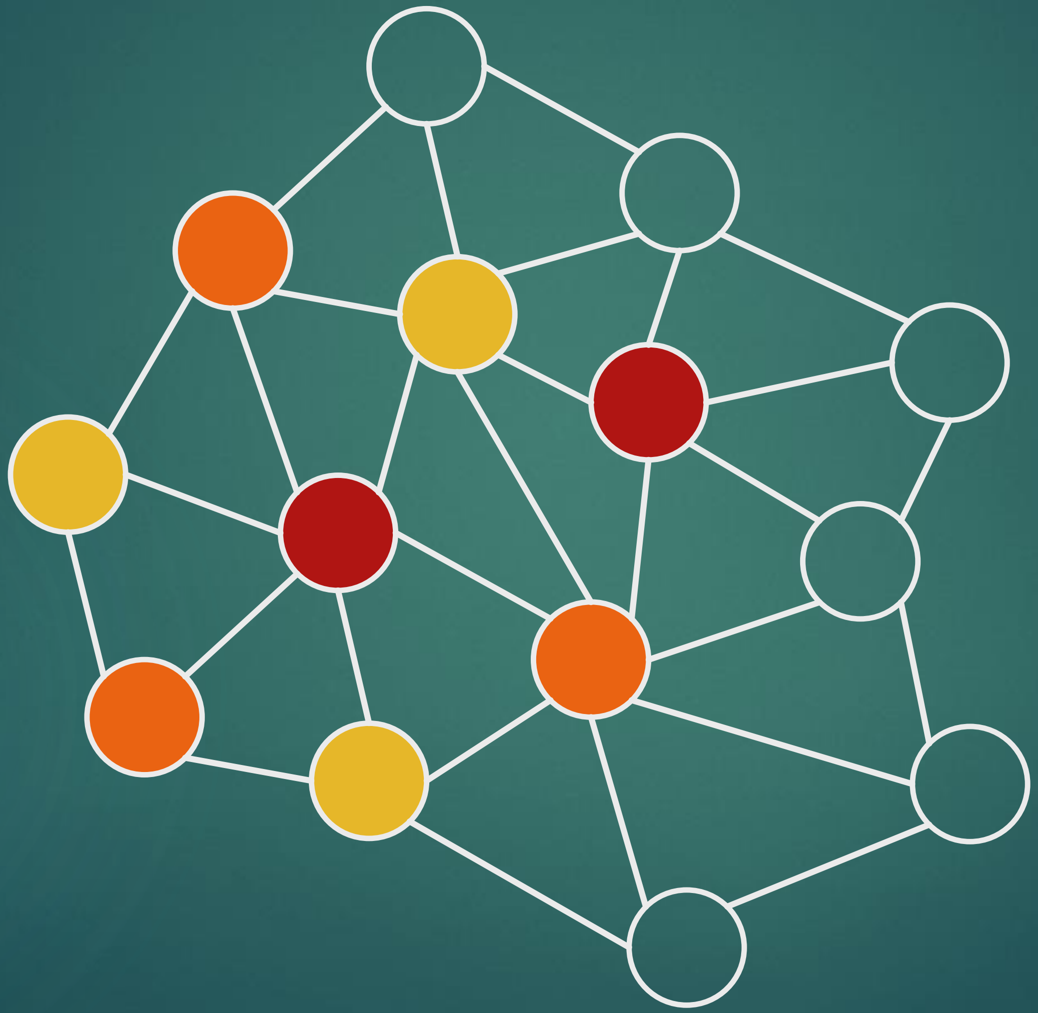
Sequential Greedy Algorithm



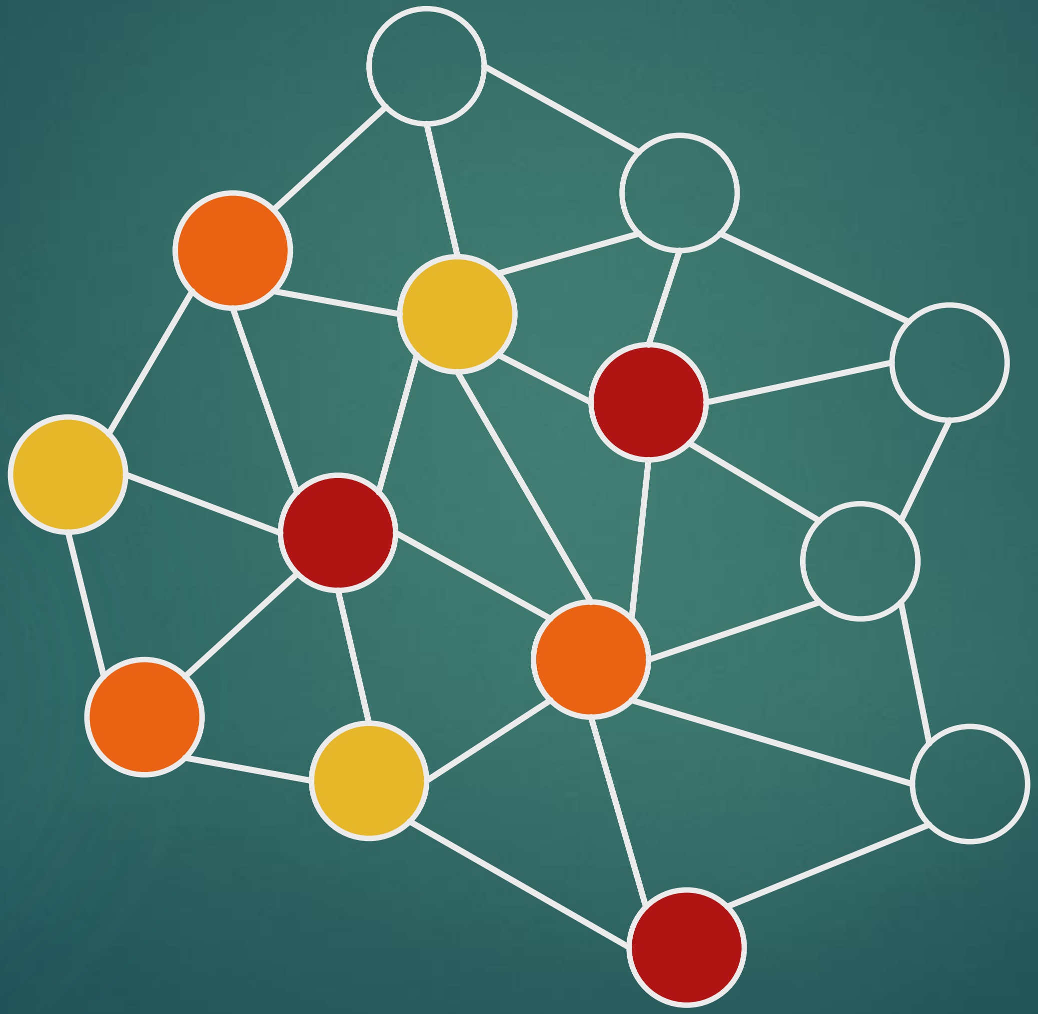
Sequential Greedy Algorithm



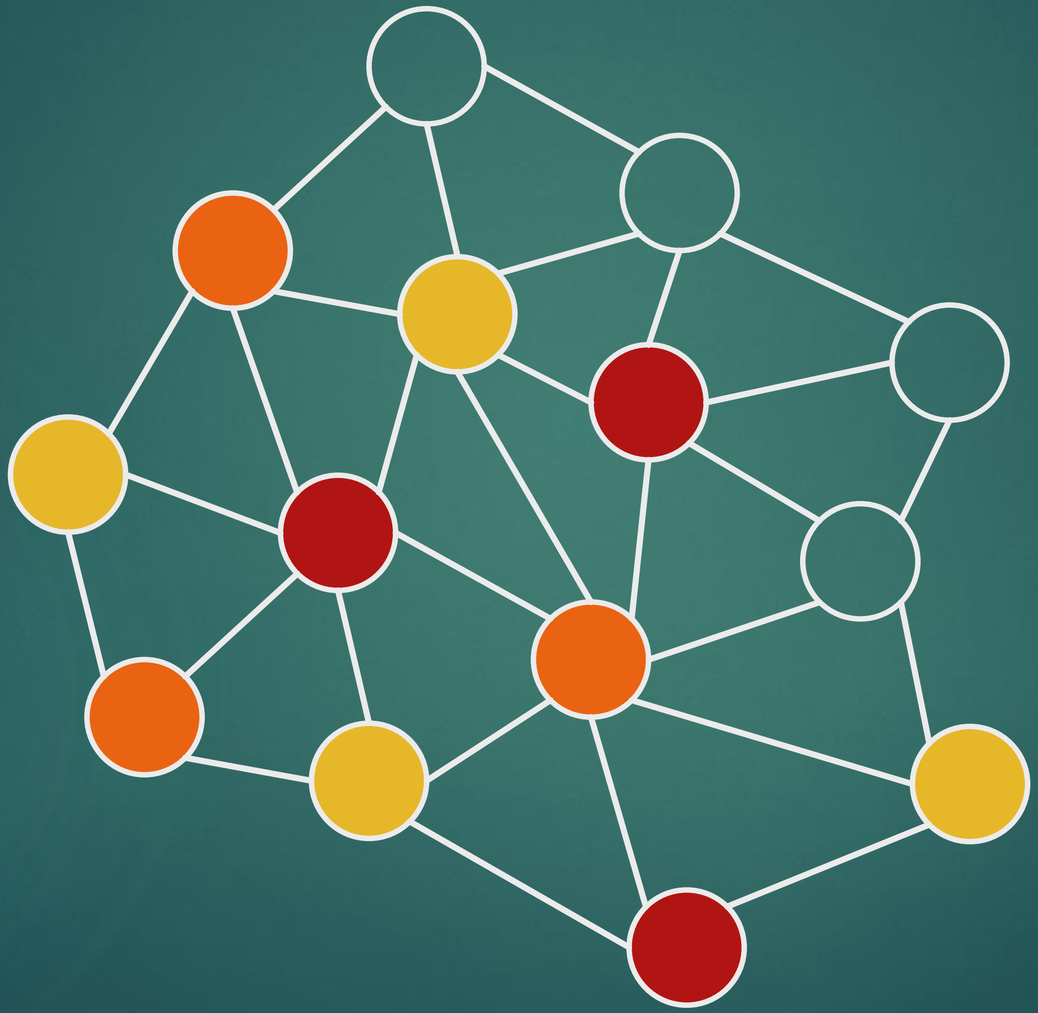
Sequential Greedy Algorithm



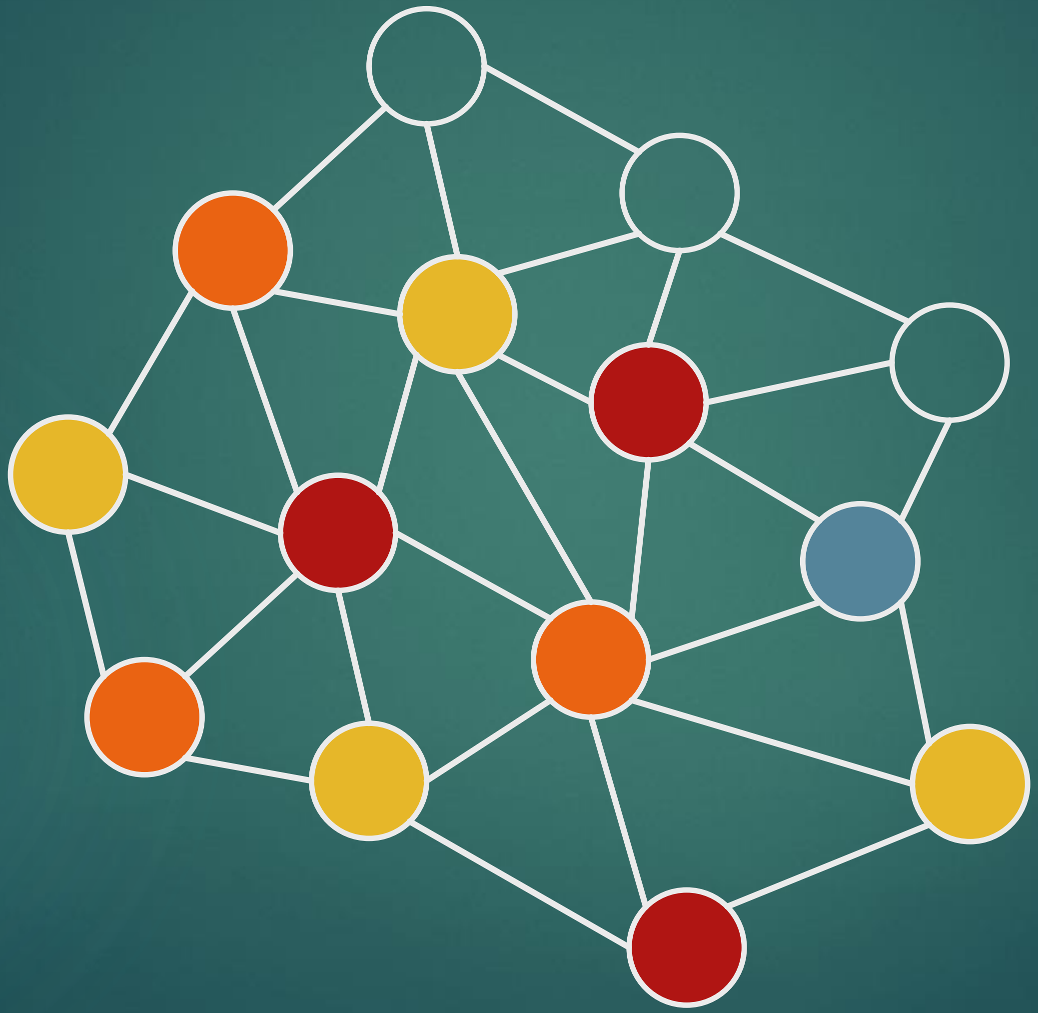
Sequential Greedy Algorithm



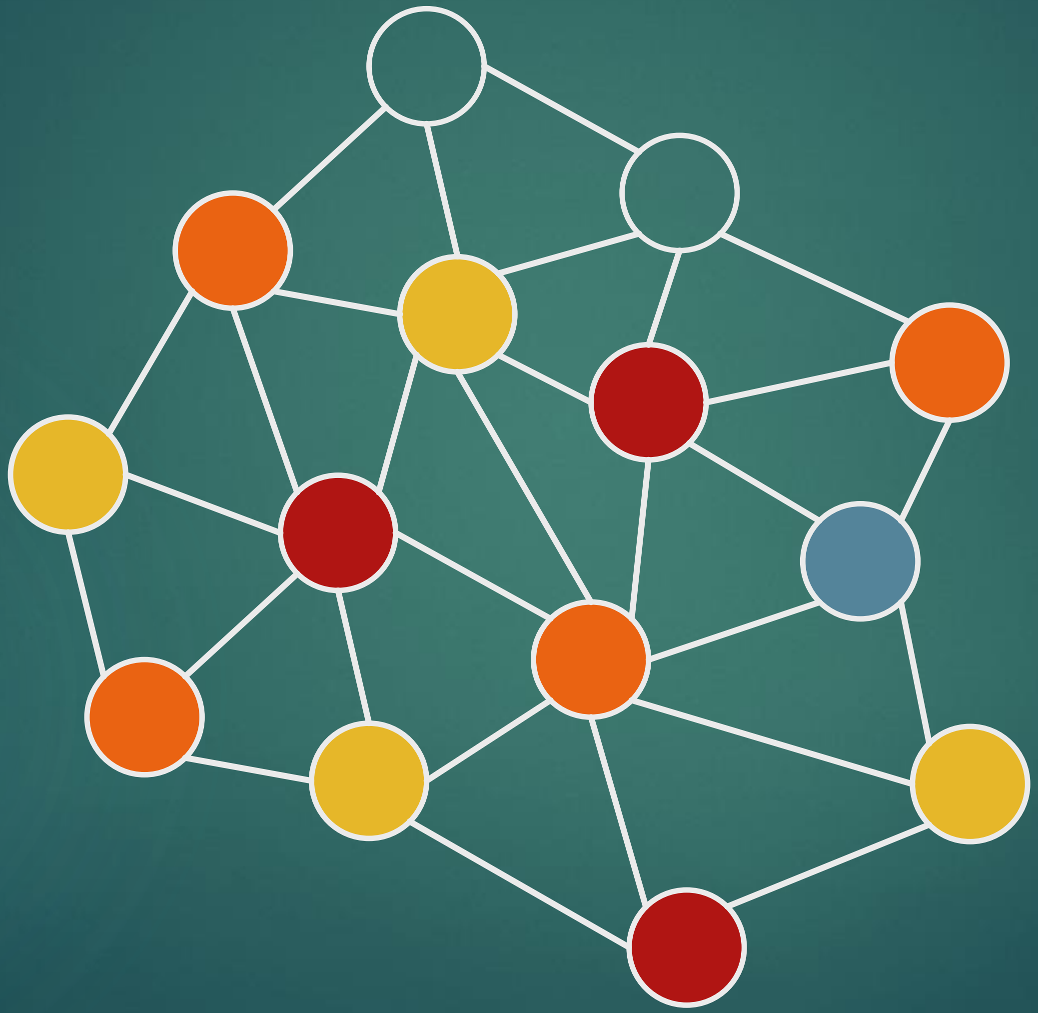
Sequential Greedy Algorithm



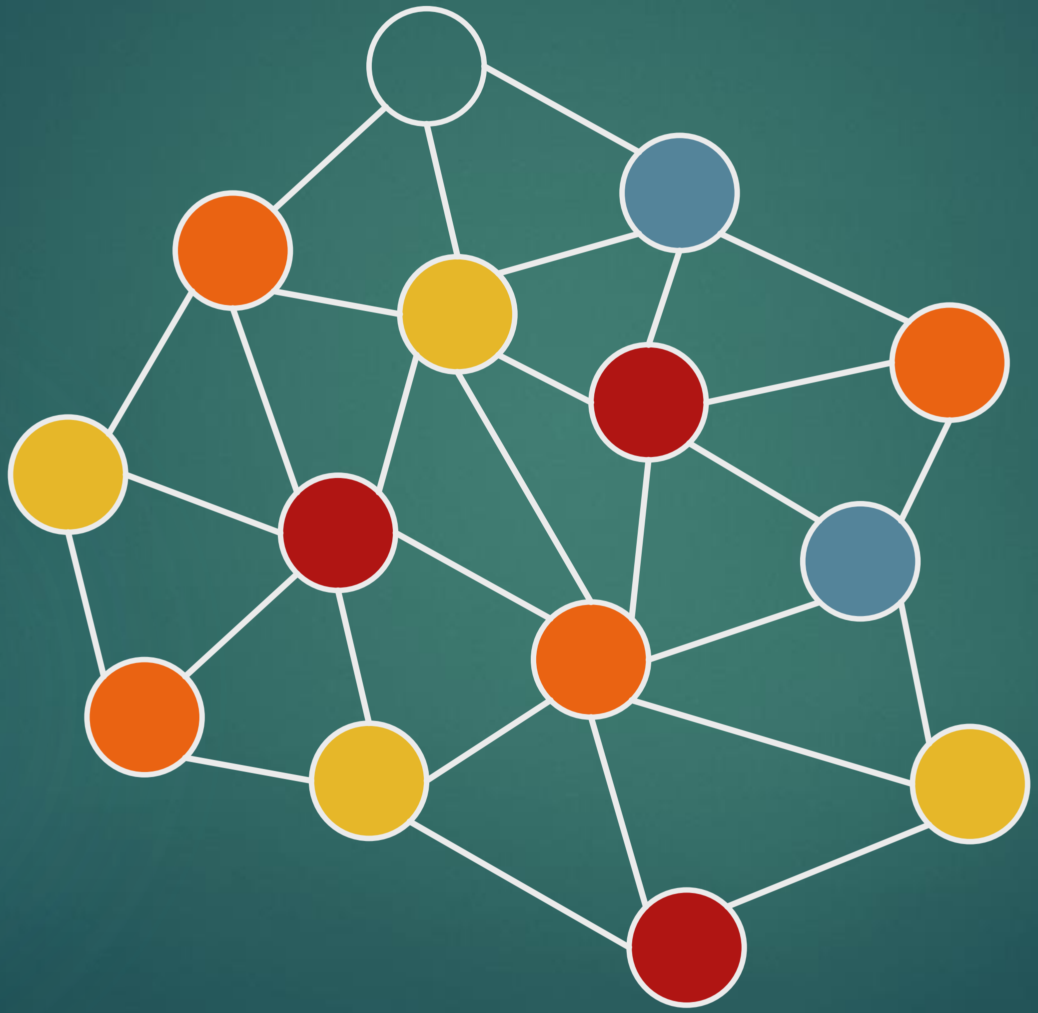
Sequential Greedy Algorithm



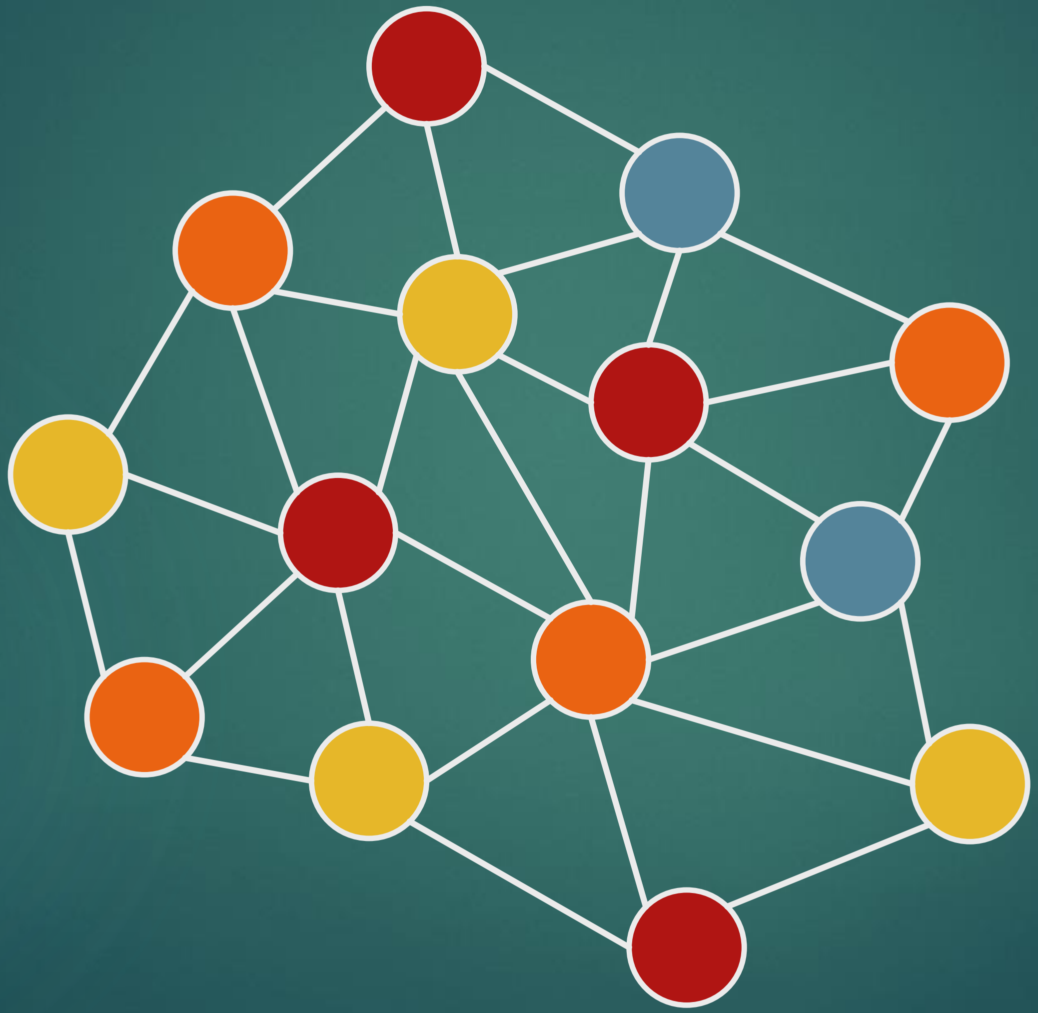
Sequential Greedy Algorithm



Sequential Greedy Algorithm

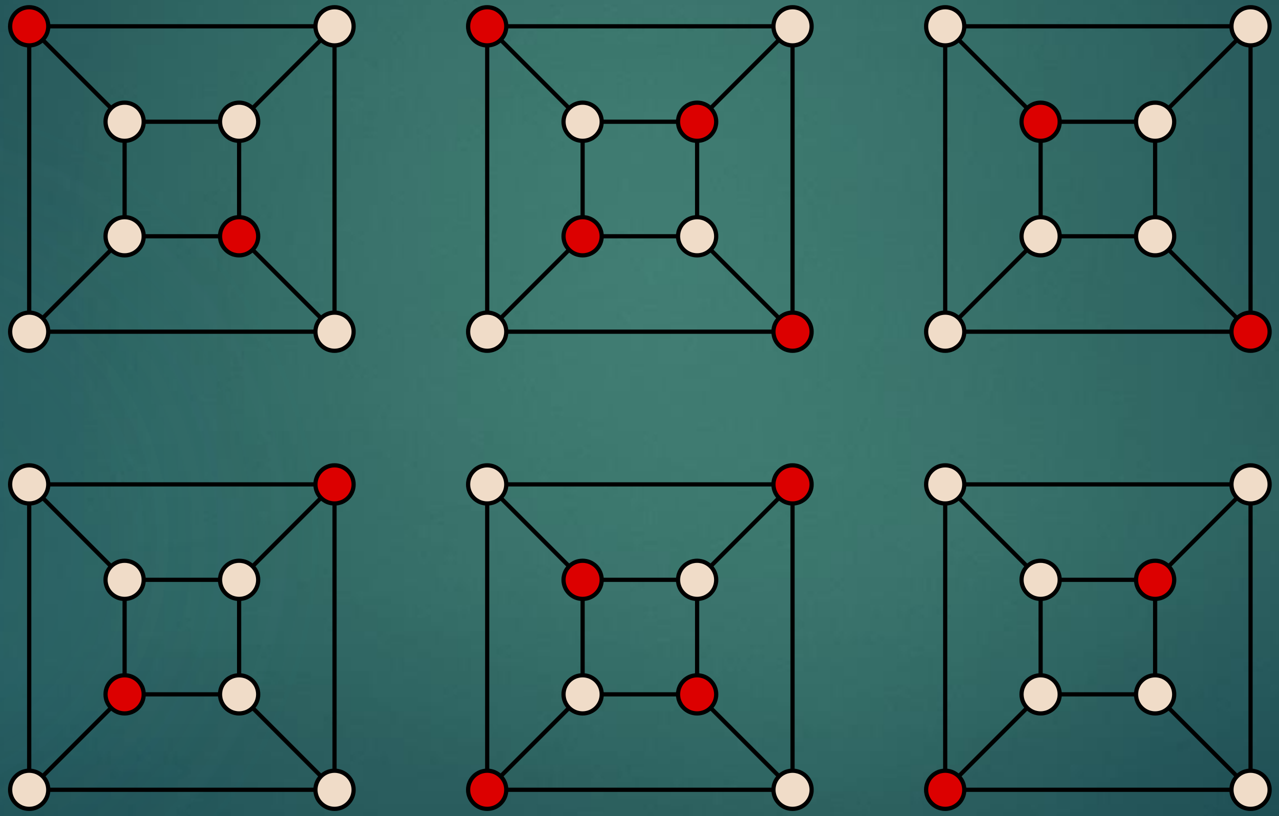


Sequential Greedy Algorithm



(Maximal) Independent Set

Independent Set: no two vertices share a common edge



Parallel Graph Coloring

- Any independent set can be colored in parallel

- $U \leftarrow V$
- while $|U| > 0$ do in parallel
  - Choose an independent set I from U
  - Color all vertices in I
  - $U \leftarrow U \setminus I$

- Optimal Coloring → color using smallest color
- Balanced Coloring → use all colors equally

Maximal Independent Set (Luby)

- find largest MIS from graph
- Color all with the same color and remove from graph
- Recurse

$I \leftarrow \emptyset$

$V' \leftarrow V$

while $|V'| > 0$ **do**

choose and independent set I' from V'

$I \leftarrow I + I'$

$X \leftarrow I' + N(I')$

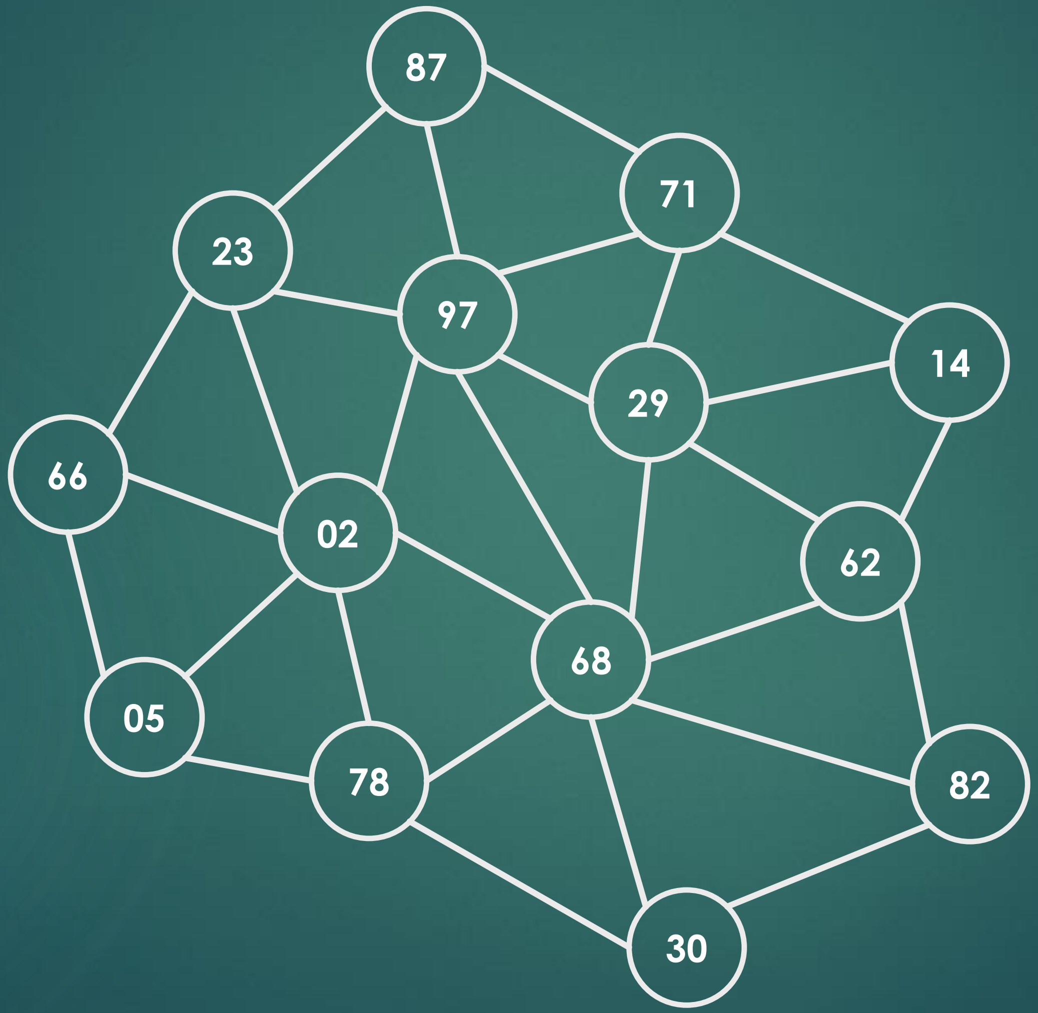
$V' \leftarrow V' - X$

How to choose independent sets in parallel?

- Assign a random weight to each vertex
- Choose vertices that are a local maxima

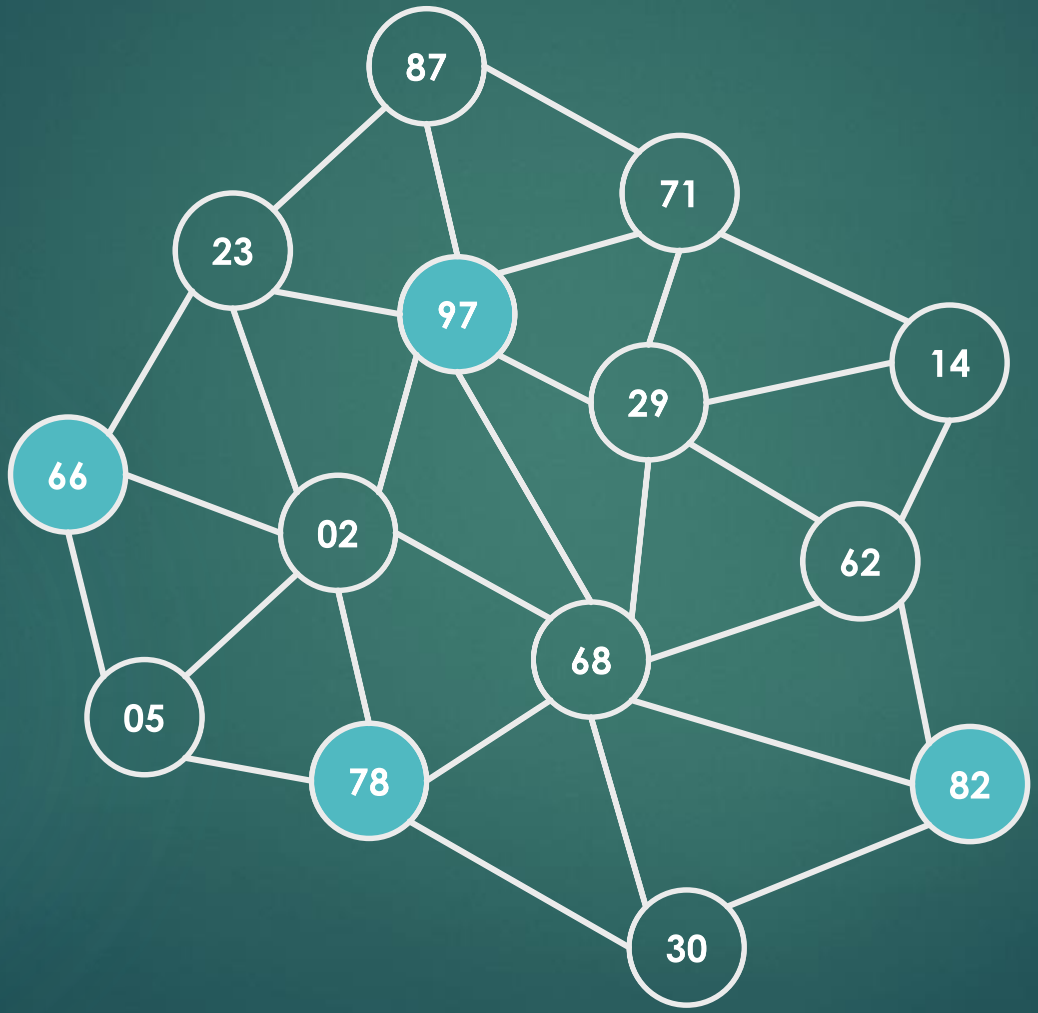
- $\mathcal{O}\left((c+1)\log|V|\right)$ algorithm
  - for sparse graphs

Luby's Algorithm



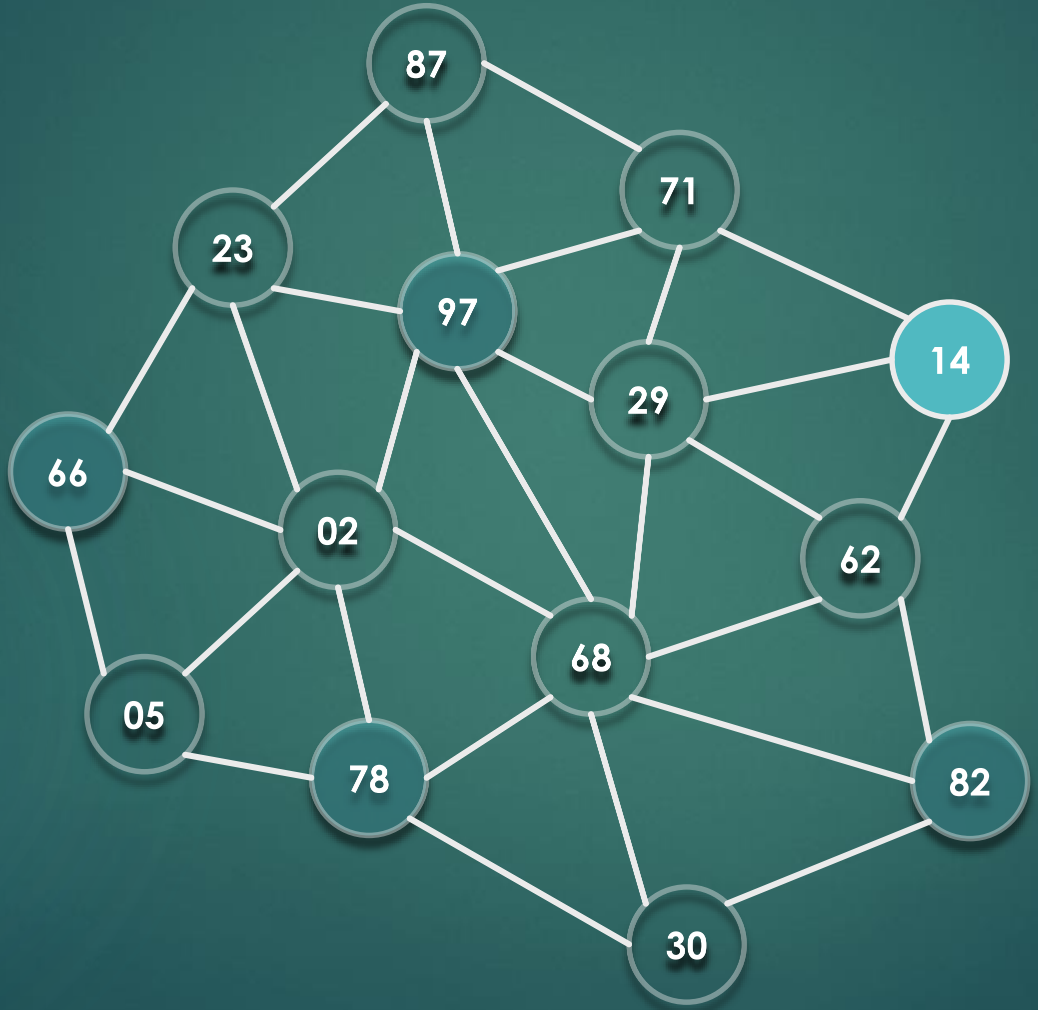
initially assigned random numbers

Luby's Algorithm



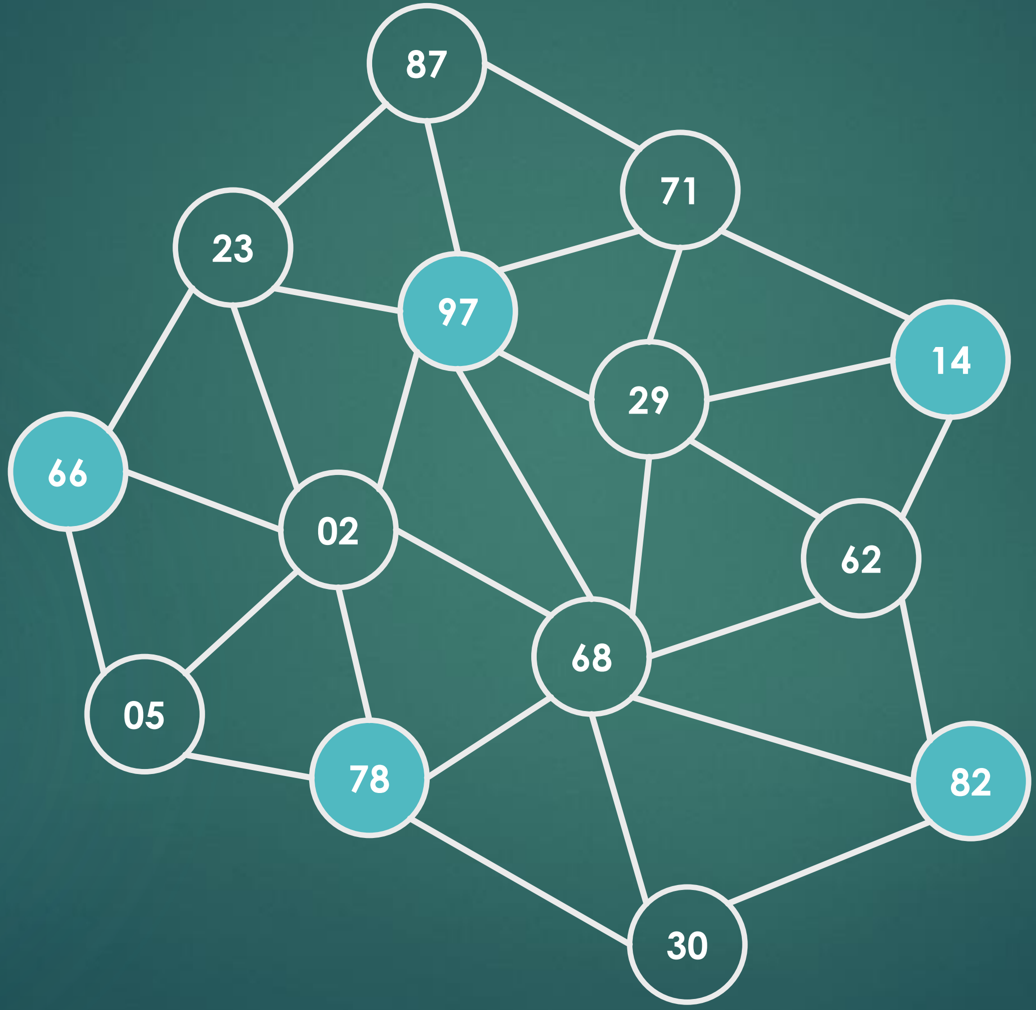
Find MIS

Luby's Algorithm



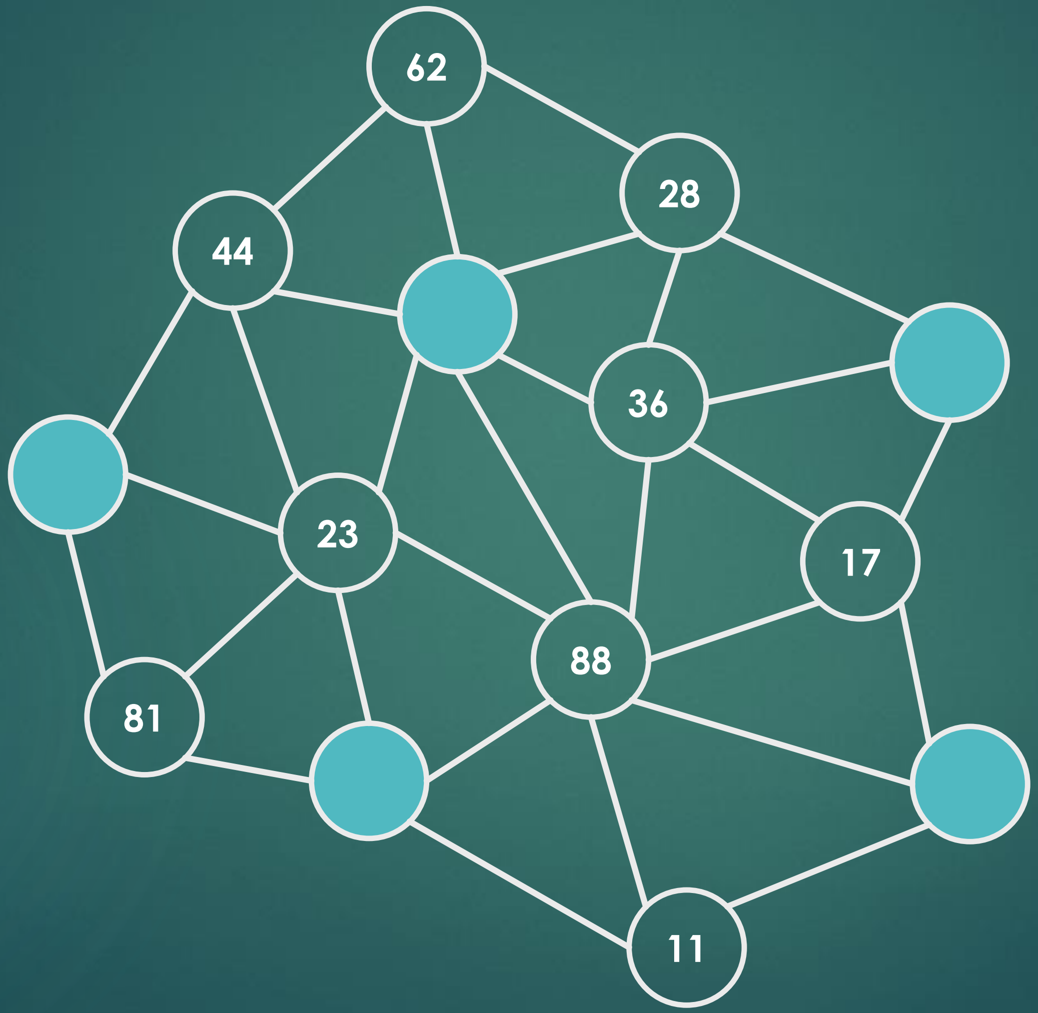
Find MIS

Luby's Algorithm



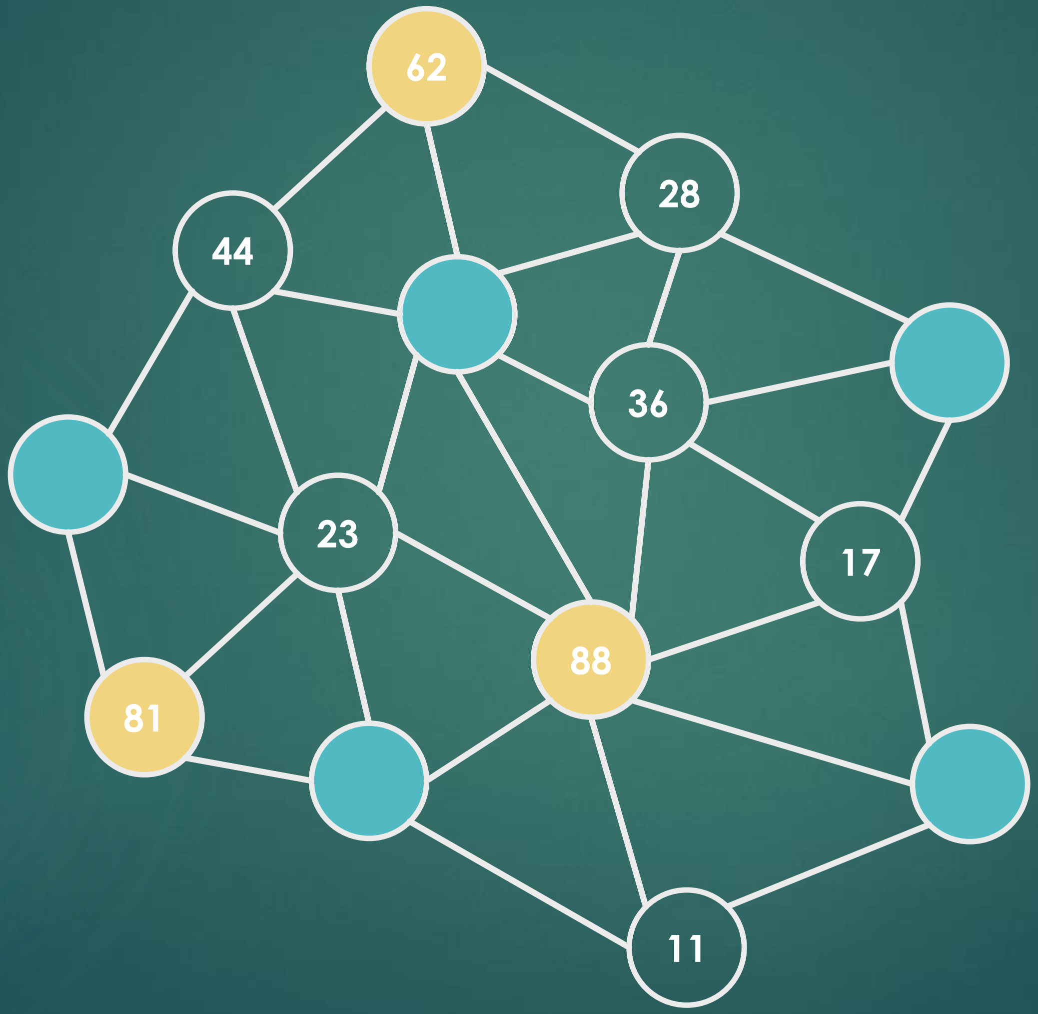
Find MIS, color all the same color

Luby’s Algorithm



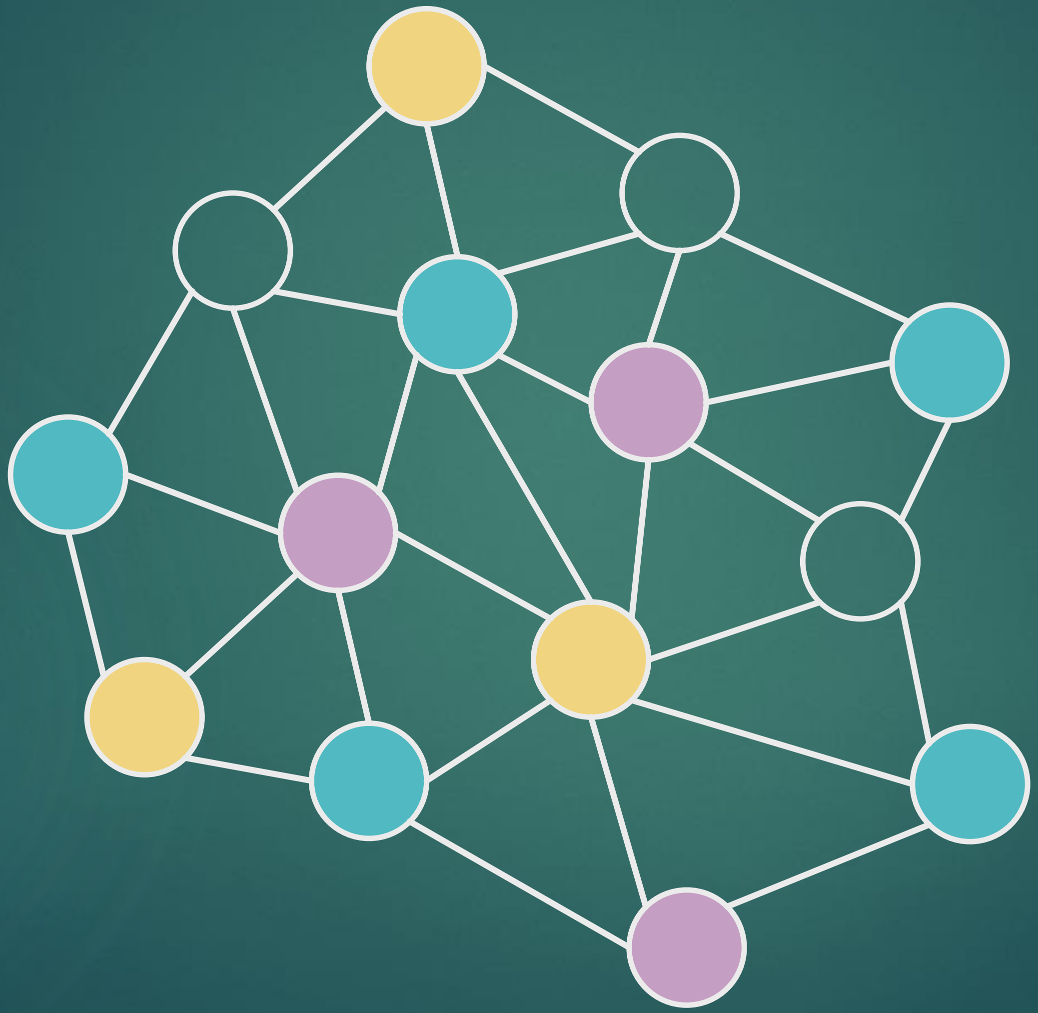
repeat

Luby's Algorithm



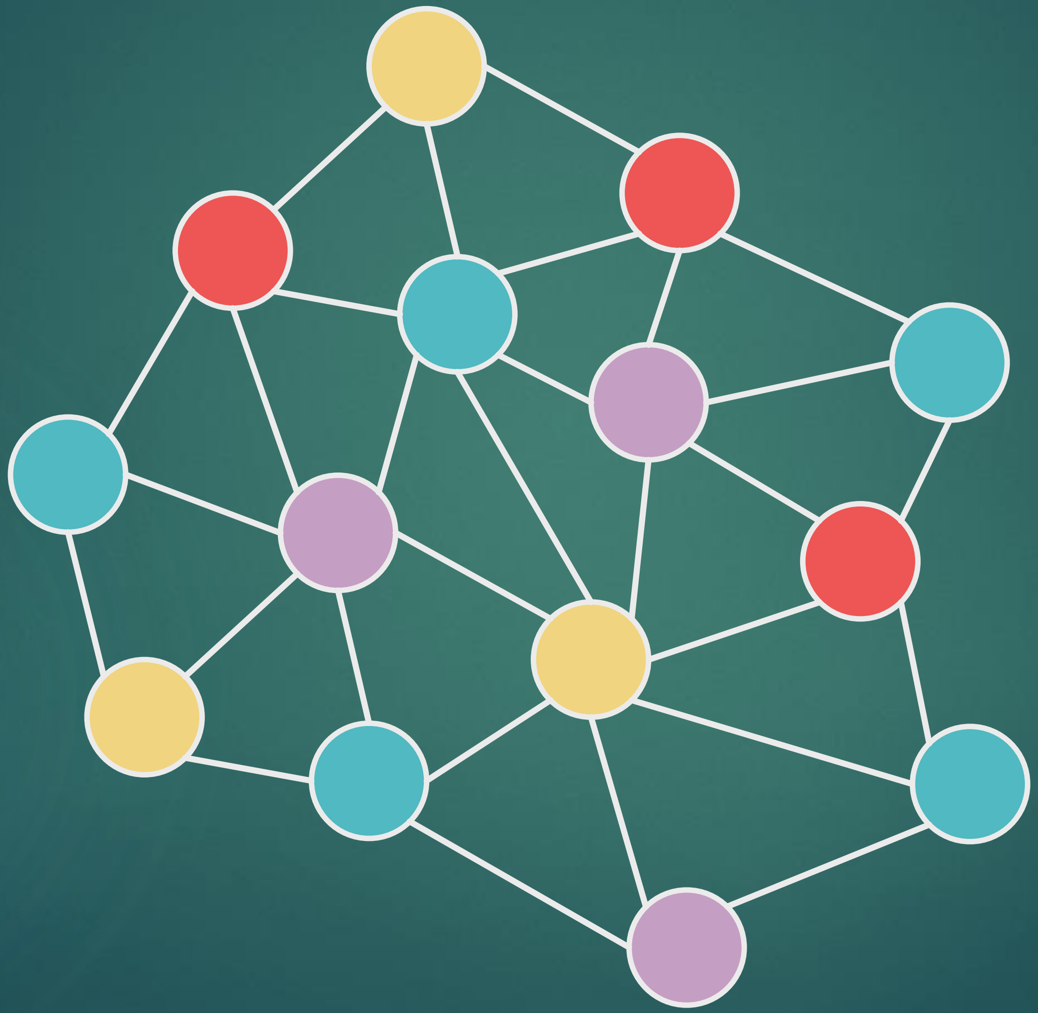
repeat

Luby's Algorithm



repeat

Luby’s Algorithm



repeat

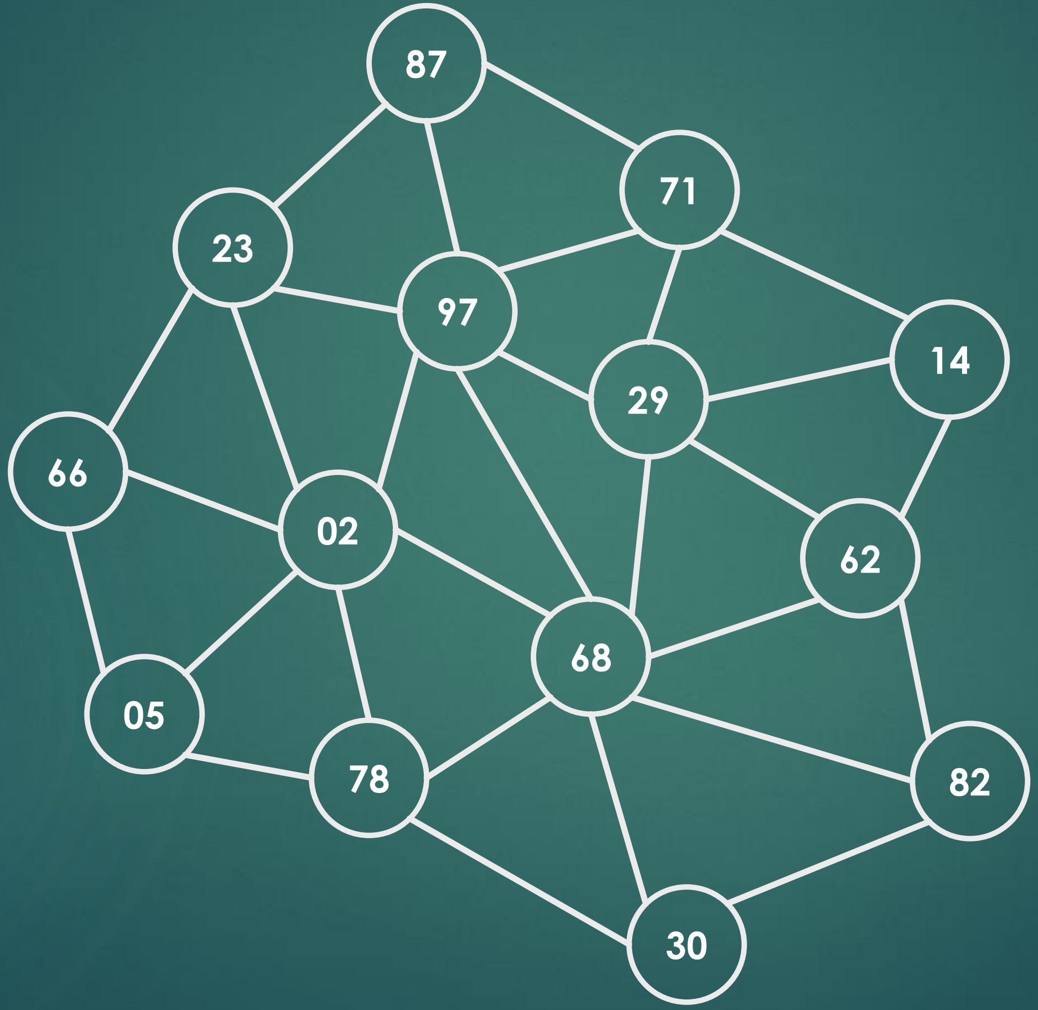
Jones-Plassmann Coloring

- Not necessary to create a new random permutation of vertices every time
- Use vertex number to resolve conflicts
- Does not find a MIS at each step
- Instead,
  - Find independent set
  - Not assigned the same color
  - Color individually using smallest available color

Jones-Plassmann Coloring

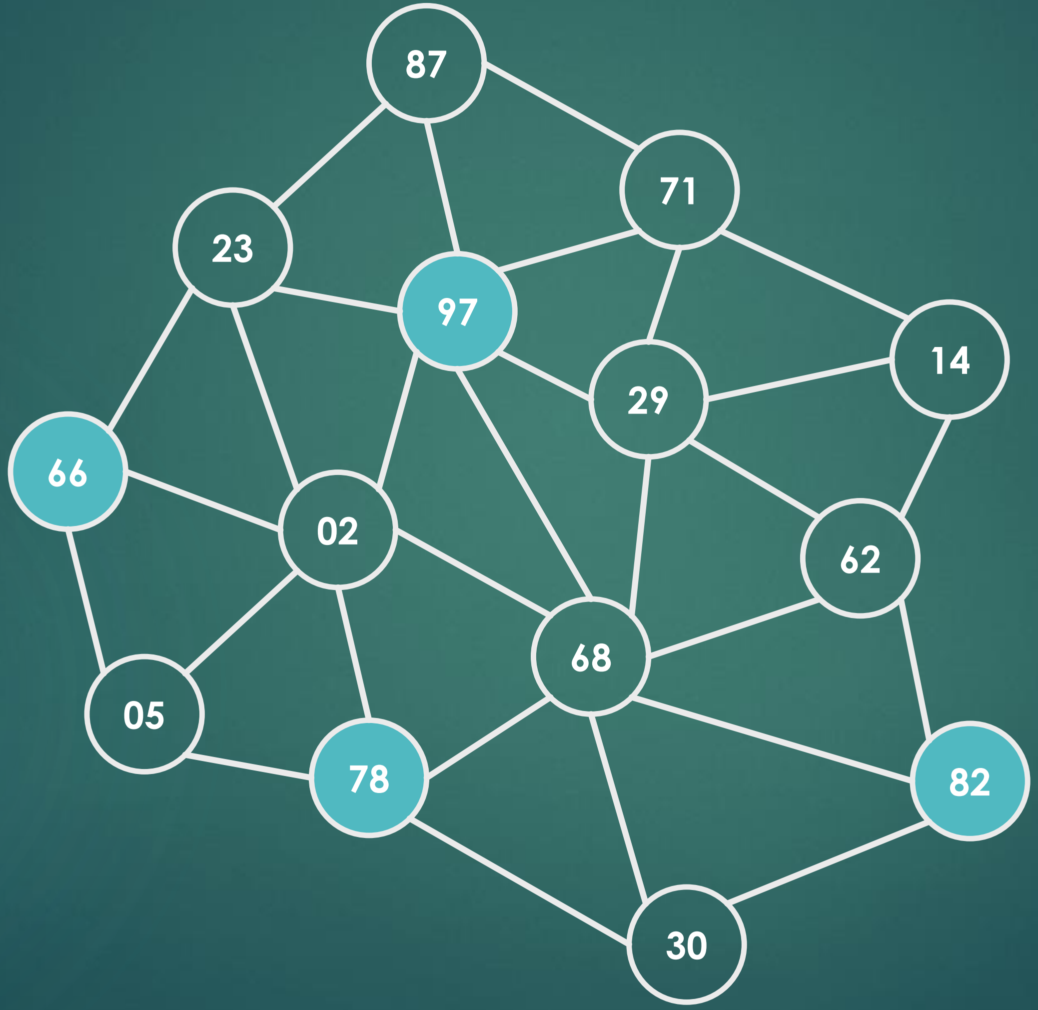
- $U \leftarrow V$
- **while** $|U| > 0$ **do**
  - **for all** vertices $v \in U$ **do in parallel**
    - $I \leftarrow \{v \mid w(v) > w(u)\ \forall\, u \in N(v)\}$
    - **for all** vertices $v' \in I$ **do in parallel**
      - $S \leftarrow \{\text{colors of } N(v')\}$
      - $c(v') \leftarrow \text{minimum color} \notin S$
  - $U \leftarrow U \setminus I$

Jones-Plassmann Coloring



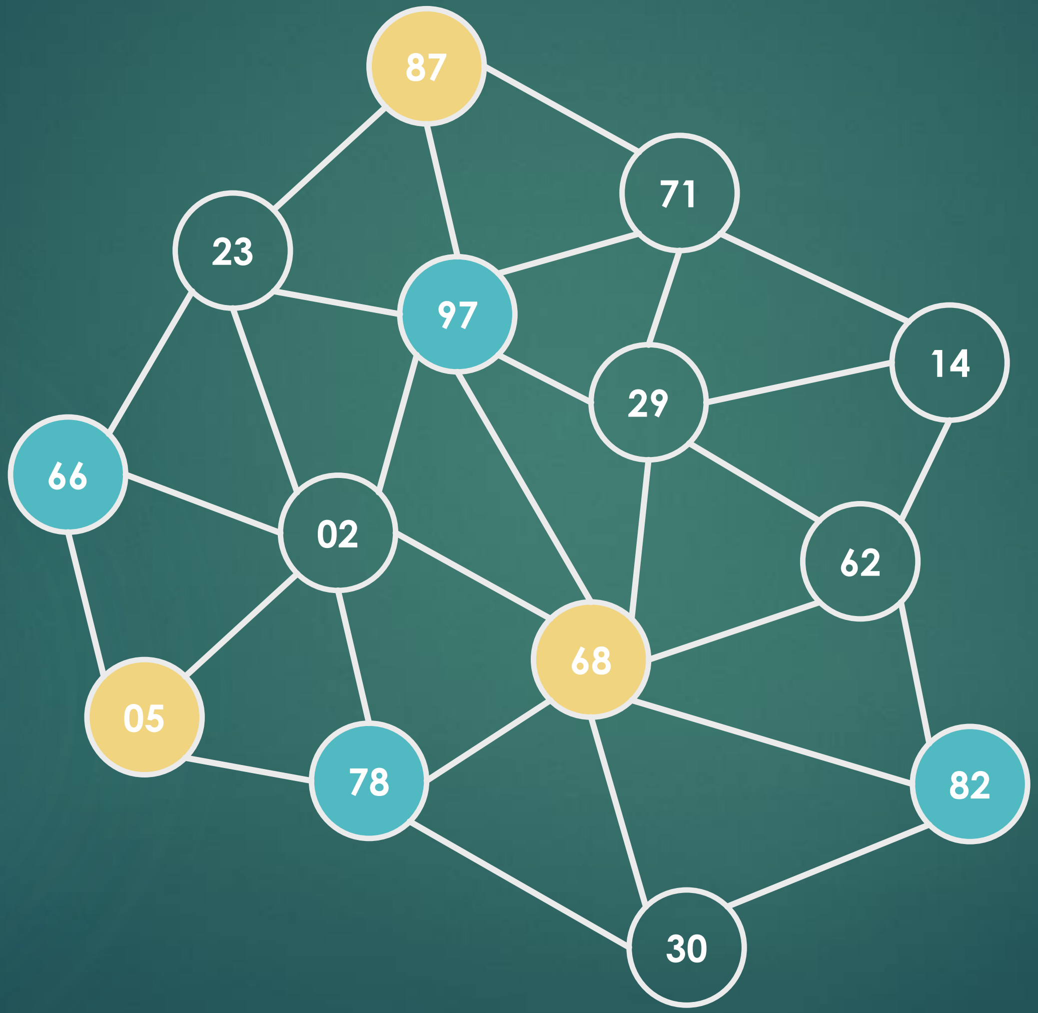
initially assigned random numbers

Jones-Plassmann Coloring



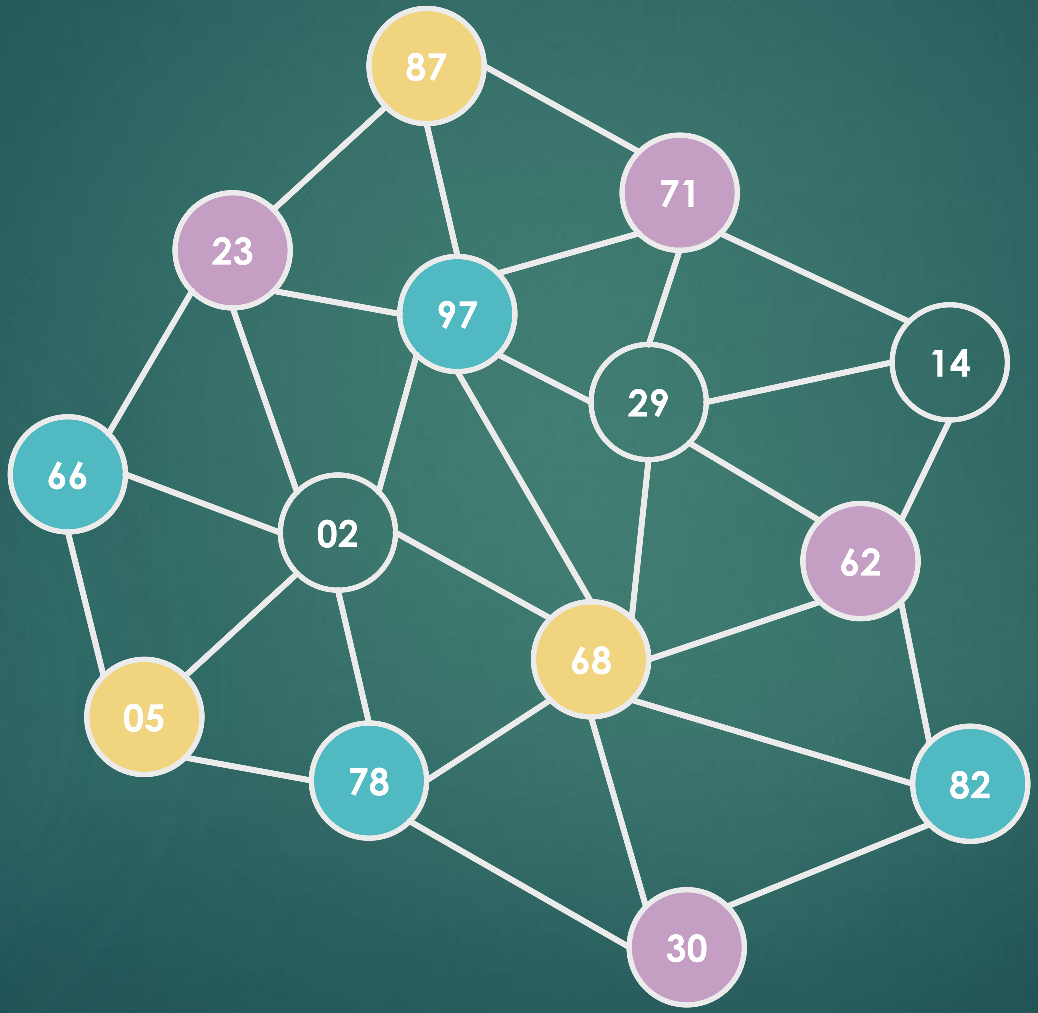
If local maxima, assign lowest color

Jones-Plassmann Coloring



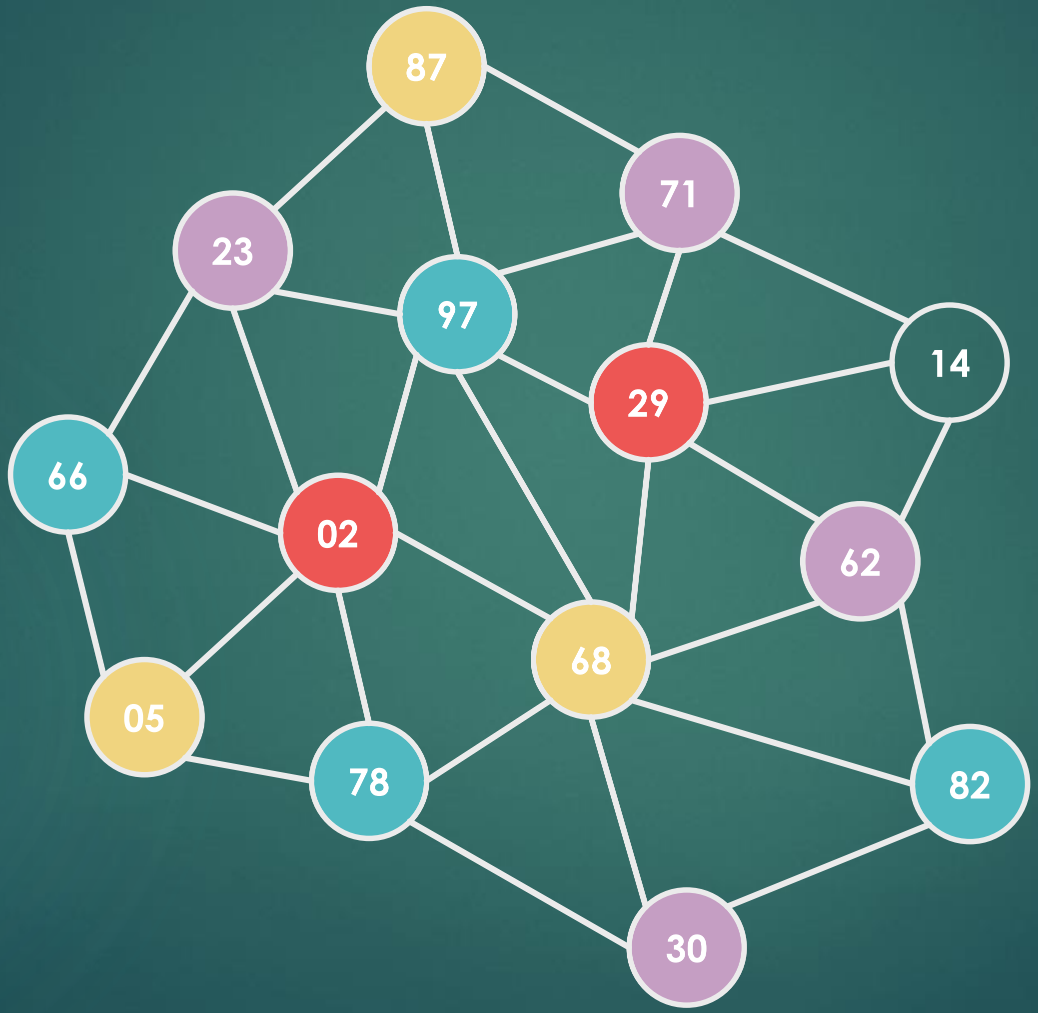
repeat, considering only uncolored vertices

Jones-Plassmann Coloring



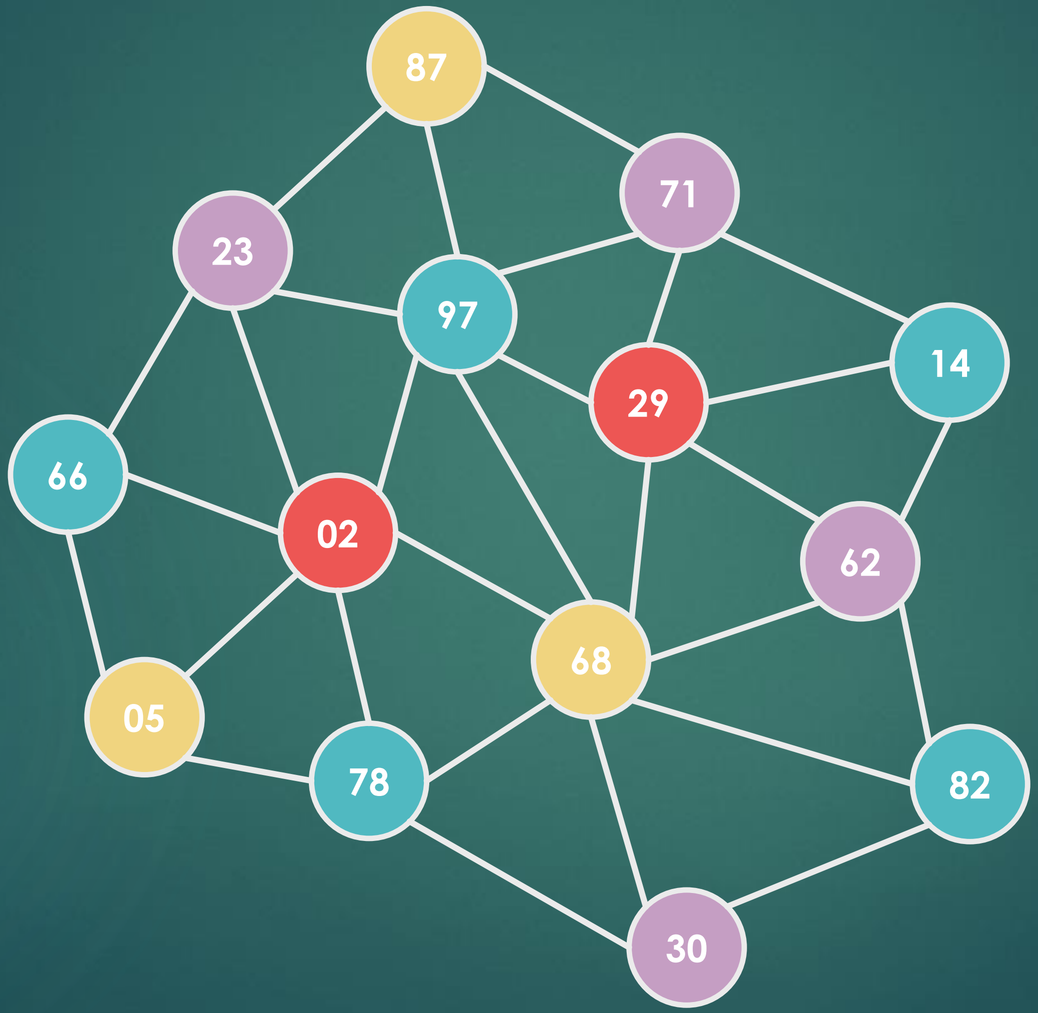
repeat, considering only uncolored vertices

Jones-Plassmann Coloring



repeat, considering only uncolored vertices

Jones-Plassmann Coloring



repeat, considering only uncolored vertices