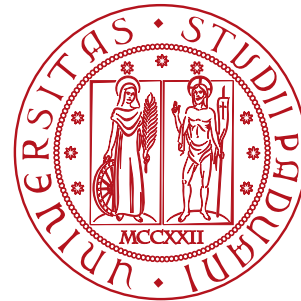


# Efficient Low Diameter Clustering

with strong diameter in the CONGEST model

Christian Micheletti



UNIVERSITÀ  
DEGLI STUDI  
DI PADOVA

1. Distributed Algorithms
2. Network Decomposition
3. Low Diameter Clustering

- We want to solve graph problems on **networks**
  - Computers are like nodes in a graph

Distribution  $\Rightarrow$  Multiple processors

- We want to solve graph problems on **networks**
  - Computers are like nodes in a graph

Distribution  $\Rightarrow$  Multiple processors

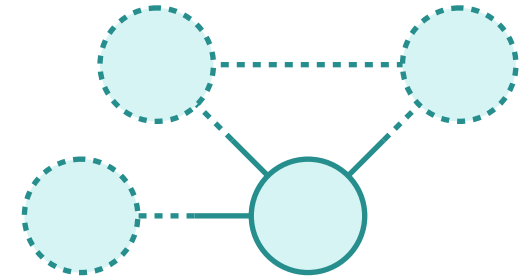
- Nodes can run code
  - Should be **the same for all nodes**



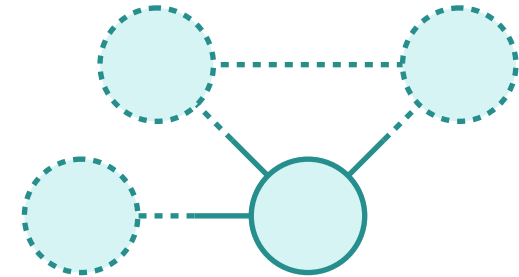
Each node gives a partial solution

- Arcs are **communication links** between computers

- In the **PN-Network** a node only knows its *neighbours*
  - And how to “contact” them
- There are no **self loops**
- Connection is two-way
- There is  $\leq 1$  arc between two nodes

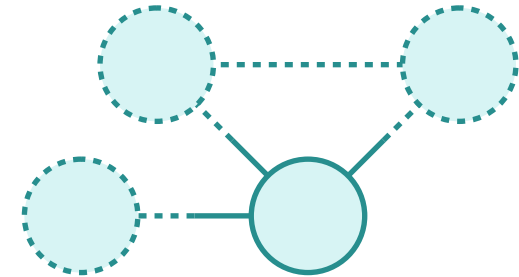


- In the **PN-Network** a node only knows its *neighbours*
  - And how to “contact” them
- There are no **self loops**
- Connection is two-way
- There is  $\leq 1$  arc between two nodes



A node can't see the whole topology

- In the **PN-Network** a node only knows its *neighbours*
  - And how to “contact” them
- There are no **self loops**
- Connection is two-way
- There is  $\leq 1$  arc between two nodes



A node can't see the whole topology



All nodes appear identical



We add **unique identifiers** to the nodes

$$id : V \rightarrow \mathbb{N}$$

where  $\forall v \in V : id(v) \leq n^c$  for some  $c \geq 1$

We choose  $n^c$  so we need  $O(\log n)$  bits to  
represent an identifier,  
(identifiers are reasonably “*small*”)



- Collaboration requires **exchanging messages**  
...on a medium that is **slow** and **unreliable**



⇒ Communication is the main pitfall

- Too many messages congest the network
- We **quantify** the number of messages that an algorithm requires
  - An ***“efficient”*** algorithm will need few messages

W.l.o.g.<sup>1</sup> we adopt a model of ***synchronous communication***

Each round, a node  $v \in V$  performs these actions:

1.  $v$  ***sends*** a message  $msg \in \mathbb{N}$  to its neighbours
2.  $v$  ***receives*** messages from its neighbours
3. ...

---

<sup>1</sup>Without loss of generality.

W.l.o.g.<sup>2</sup> we adopt a model of ***synchronous communication***

Each round, a node  $v \in V$  performs these actions:

1.  $v$  ***sends*** a message  $msg \in \mathbb{N}$  to its neighbours
  2.  $v$  ***receives*** messages from its neighbours
  3. ...
- (1.) and (2.) establish a ***communication round***
    - **Measure unit of complexity**
    - Few communication rounds  $\Rightarrow$  few messages

---

<sup>2</sup>Without loss of generality.

3.  $v$  ***executes locally*** some algorithm (same for each node).
  - A node may ***stop*** in this phase
    - Its local result is **final**



(3.) doesn't affect the algorithm's complexity

3.  $v$  **executes locally** some algorithm (same for each node).
  - A node may **stop** in this phase
    - Its local result is **final**



(3.) doesn't affect the algorithm's complexity

- When all nodes **stopped** the algorithm terminates

An algorithm is “*efficient*” when it stops in a number of rounds **polylogarithmic** in  $|V|$



- The node with  $\text{id}(v) = 0$  “waves *hello*” to neighbours
  - ...sending them a message



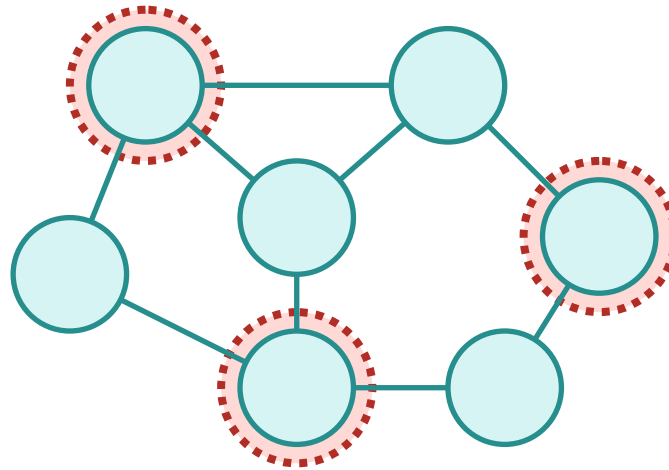
- The node with  $\text{id}(v) = 0$  “waves *hello*” to neighbours
  - ...sending them a message
- When a node receives the message, **forwards** it to its neighbours
  - And then **stops**



- The node with  $\text{id}(v) = 0$  “waves *hello*” to neighbours
  - ...sending them a message
- When a node receives the message, **forwards** it to its neighbours
  - And then **stops**
- The running time of this algorithm on a graph  $G$  is  $O(\text{diam}(G))$



### Example: **Maximal Independent Set (MIS)**



- Solving it **centralized** is easy
- How can we solve it **distributed**?



Let's leverage  $\text{id}(v)$  to select the next MIS node

- At round  $\#i$ , node  $v : \text{id}(v) = i$  executes
  - If no neighbour is in the MIS, add the node
    - And inform the neighbours
  - Otherwise, the node is outside the MIS



Let's leverage  $\text{id}(v)$  to select the next MIS node

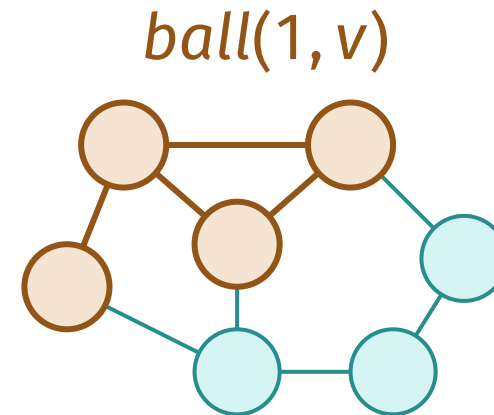
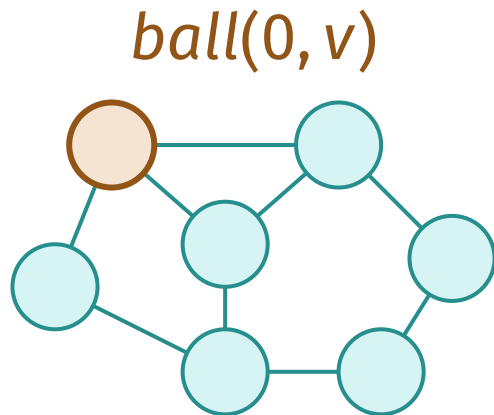
- At round  $\#i$ , node  $v : \text{id}(v) = i$  executes
  - If no neighbour is in the MIS, add the node
    - And inform the neighbours
  - Otherwise, the node is outside the MIS
- It is correct since no node has the same id
- This algorithm runs in  $O(n^c)$  (the maximum id)
  - **Very bad**



Running a centralized algorithm on a single node would take  $O(1)$  rounds

- We'd like to run a MIS algorithm on each node
  - Each must have a **local copy** of the **entire** graph
  - The algorithm must be deterministic
    - When a node stops it checks if it is included in MIS

- The algorithm GATHER-ALL makes all nodes build a local copy of the whole graph
  - At round  $i$ , each node  $v$  knows  $ball(i, v)$



- All nodes will know the whole graph after  $O(diam(G))$  rounds

- GATHER-ALL assumes that messages size is **unbounded**



Requires to send the whole graph in one message

- It is not always possible to send arbitrary large messages
  - Heavy ones may be “sharded”
- We provide an upper bound for message size
  - Messages need to be reasonably “*small*”
  - Large messages will require more rounds to be sent

In the CONGEST model, messages size has to be  $O(\log n)$

- Sending  $k$  identifiers takes  $O(1)$  rounds
- Sending a set of identifiers can take up to  $O(n)$
- Sending the whole graph requires  $O(n^2)$  rounds:
  - The adjacency matrix suffices...



⇒ We can't use **GATHER-ALL** in the CONGEST model

# Solving problems in CONGEST



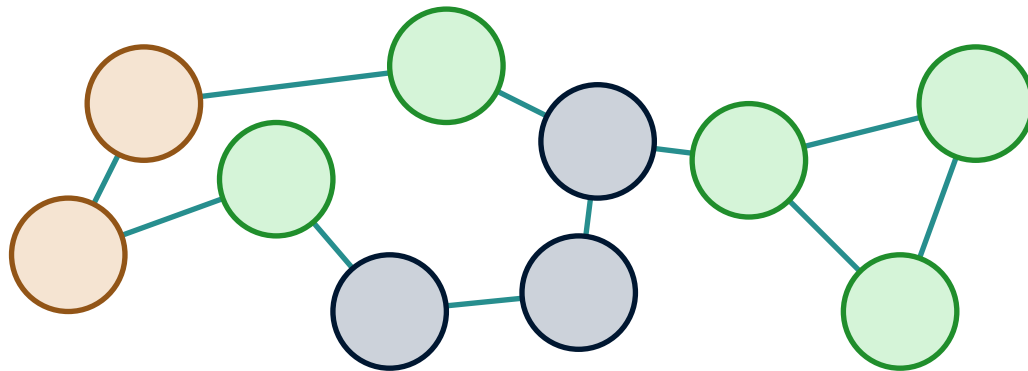
- Censor-Hillel et al. [1] provided an algorithm that solves MIS in  $O(\text{diam}(G) \log^2 n)$  in CONGEST



The diameter can be very large

- Worst case:  $\text{diam}(G) = n$
- How can we improve it?

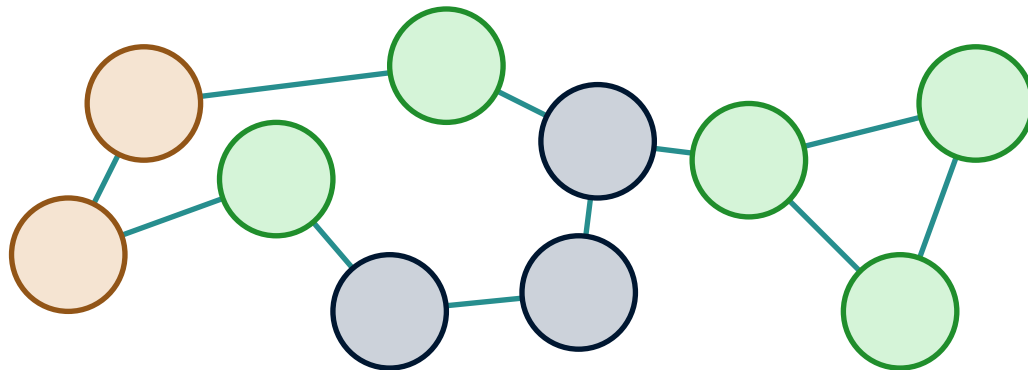
- A **Network Decomposition** groups nodes in **colored clusters**
  - Clusters with the same color are not adjacent
  - We say it to **have diameter**  $d$  if each cluster has diameter at most  $d$
  - It has  $c$  colors





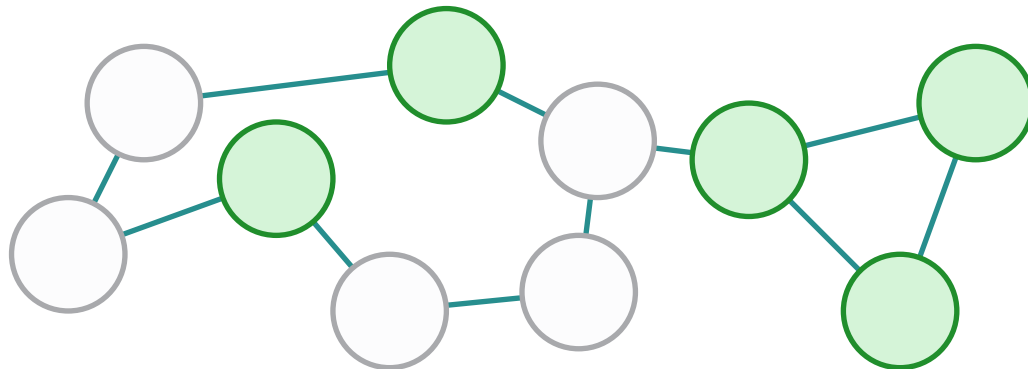
### Solving MIS in a color gives a partial solution

- We can apply [1] for all colors
  - (dropping MIS neighbours after each iter)
  - This has complexity  $O(c \cdot d \log^2 n)$ 
    - If  $c = O(\log n) = d$  it would be **“efficient”**



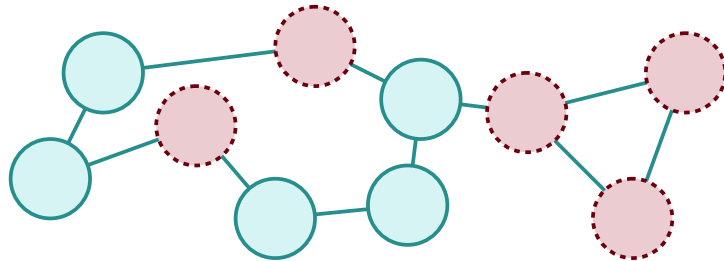
A **low diameter clustering**  $\mathcal{C} \subseteq 2^V$  for a graph  $G$  with diameter  $d$  is such:

1.  $\forall C_1 \neq C_2 \in \mathcal{C} : \text{dist}_G(C_1, C_2) \geq 2$ 
  - ***“There are no adjacent clusters”***
2.  $\forall C \in \mathcal{C} : \text{diam}(G[C]) \leq d$ 
  - ***“Any cluster has diameter at most  $d$ ”***



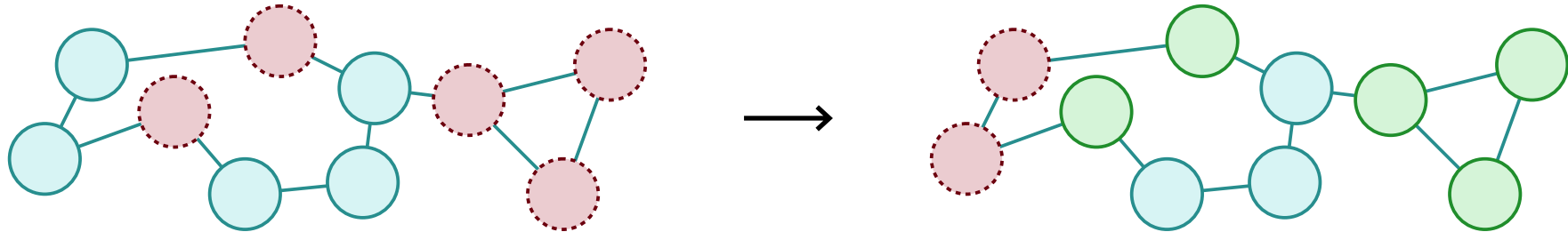
Main iteration:

1. Find a low diameter clustering
2. Assign a free color to its nodes
3. Repeat to discarded nodes until there are no more left



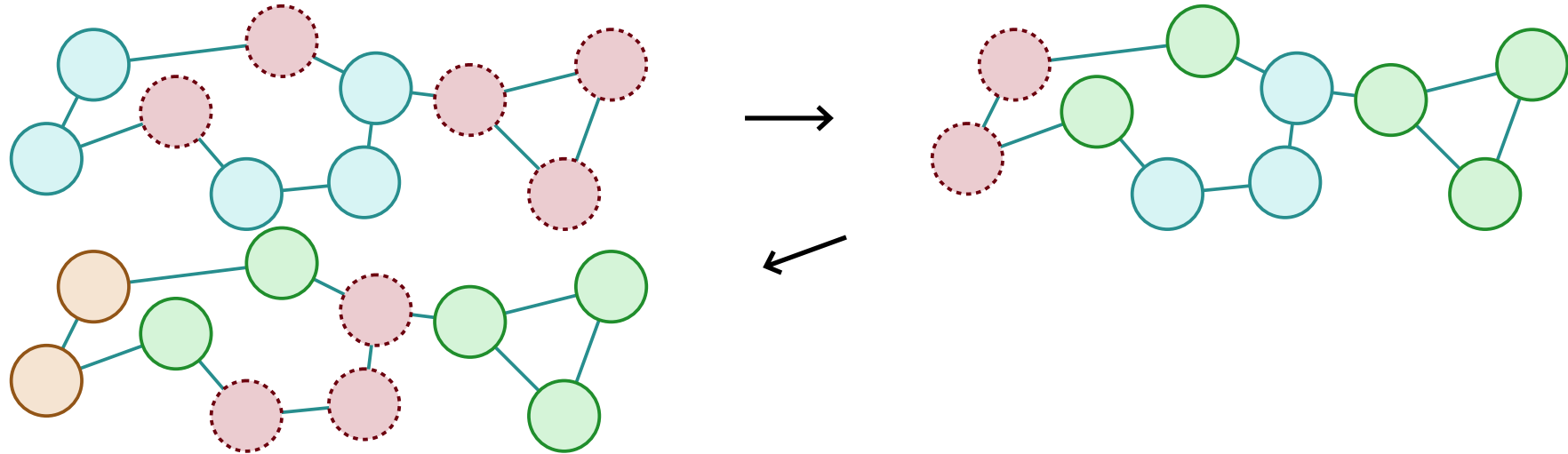
Main iteration:

1. Find a low diameter clustering
2. Assign a free color to its nodes
3. Repeat to discarded nodes until there are no more left



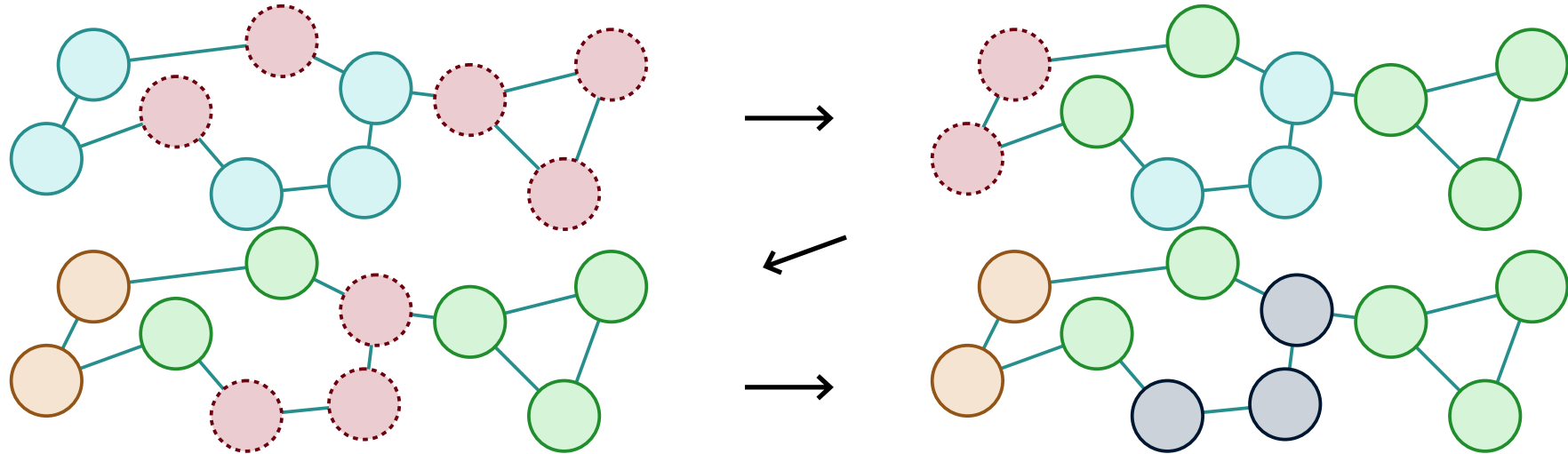
Main iteration:

1. Find a low diameter clustering
2. Assign a free color to its nodes
3. Repeat to discarded nodes until there are no more left



Main iteration:

1. Find a low diameter clustering
2. Assign a free color to its nodes
3. Repeat to discarded nodes until there are no more left





- Our previous definition of diameter is also called **strong** diameter

We say a clustering has **weak** diameter when:

1. (unchanged) “There are no adjacent clusters”
2. Any cluster has “**diameter in  $G$** ” at most  $d$

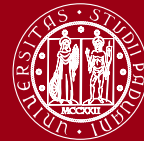
How to compute a low  
diameter clustering



- Main accomplishments of [2]:
  - Terminates in  $O(\log^6 n)$  rounds in the CONGEST model
  - Outputs a clustering with  $O(\log^3 n)$  colors
  - Directly strong diameter



- Main accomplishments of [2]:
  - Terminates in  $O(\log^6 n)$  rounds in the CONGEST model
  - Outputs a clustering with  $O(\log^3 n)$  colors
  - Directly strong diameter
- Previously [3] provided a l.d.c with **weak** diameter
  - $O(\log^7 n)$  rounds with  $O(\log^3 n)$  colors
  - It's possible to turn it into strong diameter



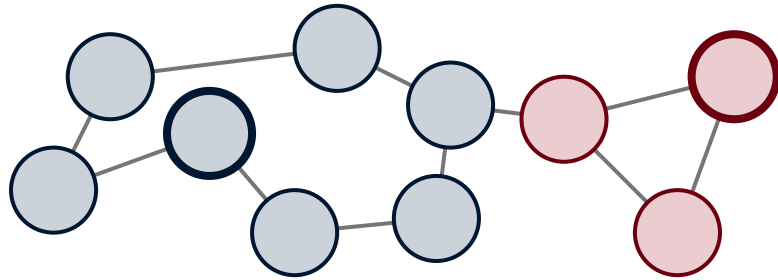
- Main accomplishments of [2]:
  - Terminates in  $O(\log^6 n)$  rounds in the CONGEST model
  - Outputs a clustering with  $O(\log^3 n)$  colors
  - Directly strong diameter
- Previously [3] provided a l.d.c with **weak** diameter
  - $O(\log^7 n)$  rounds with  $O(\log^3 n)$  colors
  - It's possible to turn it into strong diameter
- [4] did it in  $O(\log^4 n)$  rounds with  $O(\log^3 n)$  colors
  - Has to pass by a weak d. intermediate solution

### Objectives:

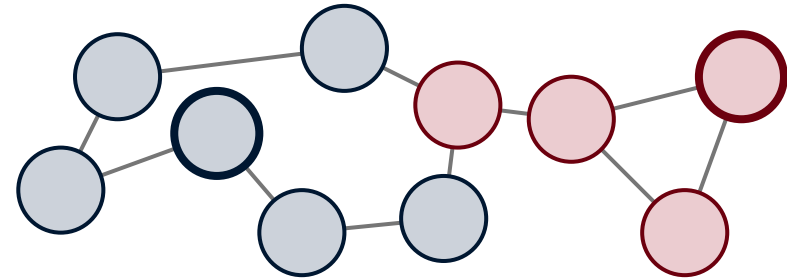
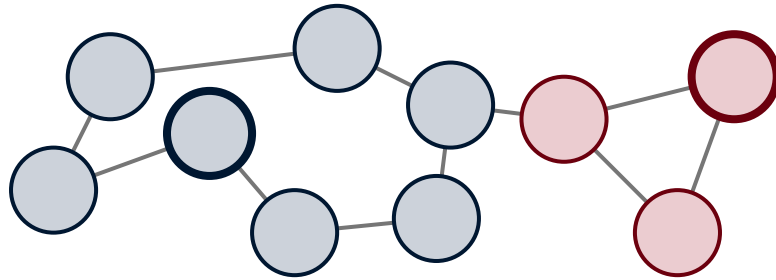
1. Creating connected components with “**low**” diameters
  - Keep track of the “center” of the c.c.
    - A.k.a. **Terminal**
  - We will merge c.c.s
    - Only one terminal is going to be the new center
  - Remove c.c.s with nodes “too far away”

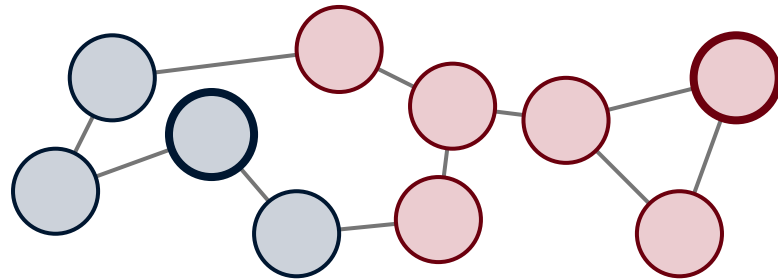
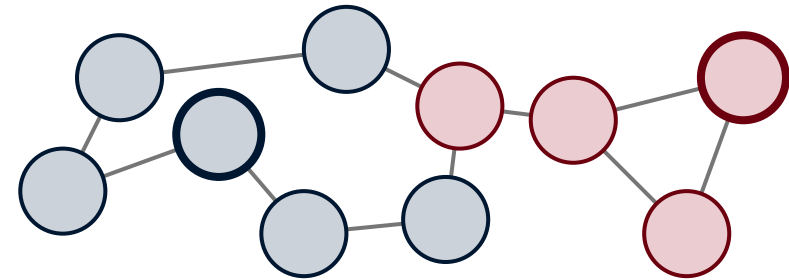
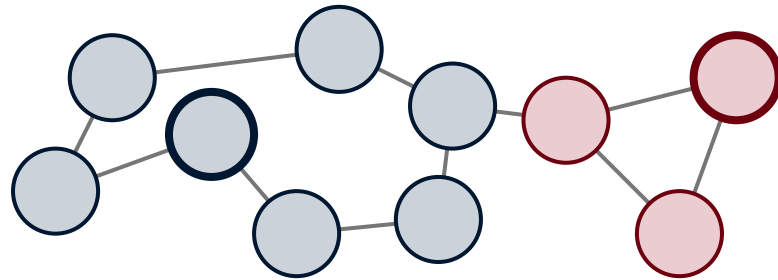
### Objectives:

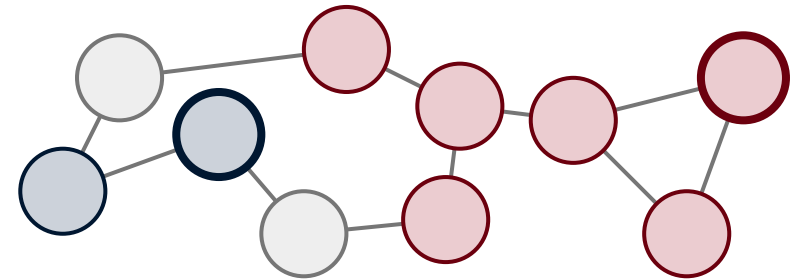
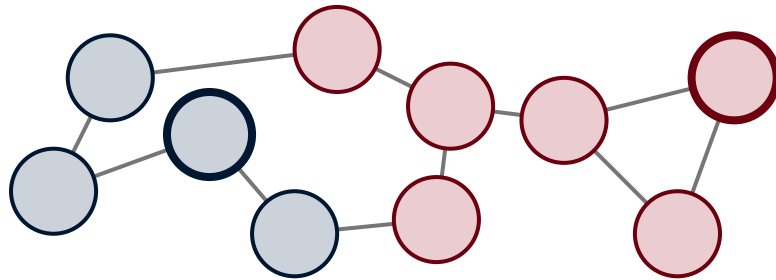
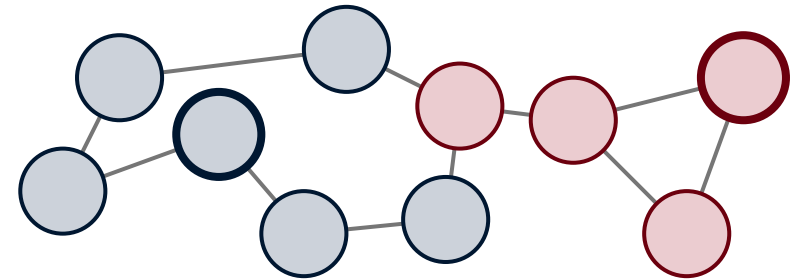
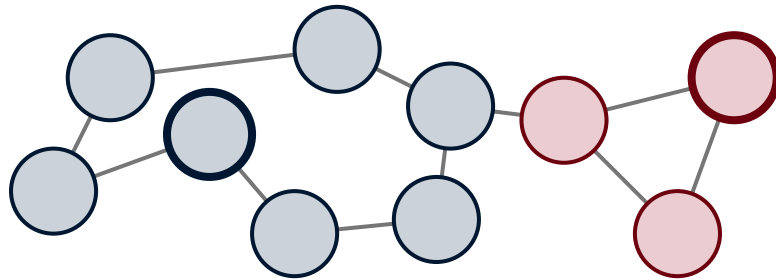
1. Creating connected components with “**low**” diameters
  - Keep track of the “center” of the c.c.
    - A.k.a. **Terminal**
  - We will merge c.c.s
    - Only one terminal is going to be the new center
  - Remove c.c.s with nodes “too far away”
2. Cluster **at least half** of the nodes
  - Required for “**few**” colors **network decompositions**











### **Phases**

- There are  $b = \log(\max_{v \in V} \text{id}(v)) = O(\log n)$  **phases**
- "One phase for each bit in index"
  - Phase  $i \in [0, b - 1]$  computes **terminals set**  $Q_i$

### **Notation:**

- $Q_i$  is the terminals set built *before* phase  $i$
- $Q_b$  is the terminals set built *after* phase  $b - 1$
- $V_i$  is the set of **living nodes** at the beginning of phase  $i$
- $V' = V_b$  is the set of **living nodes** after the last phase



1.  $Q_i$  is  $R_i$ -ruling, i.e.  $\text{dist}_G(Q_i, v) \leq R_i$  for all  $v \in V$ 
  - **We set**  $R_i = i * O(\log^2 n)$ 
    - $Q_0$  is 0-ruling, trivially true with  $Q_0 = V$ 
      - ***“All nodes are terminals at the beginning”***
    - $Q_b$  is  $O(\log^3 n)$ -ruling

*“Each node has polylog distance from  $Q_b$ ”*  
 $\Rightarrow$  Each c.c. has at least one terminal



2. Let  $q_1, q_2 \in Q_i$  s.t. they are in the same c.c in  $G[V_i]$ .  
Then  $\text{id}(q_1)[0..i] = \text{id}(q_2)[0..i]$
- For  $i = 0$  it's trivially true
  - For  $i = b$  there is  $\leq 1$  terminal in each c.c.

Along with invariant (1.), it means that each  
c.c. has polylog diameter!



3.  $|V_i| \geq \left(1 - \frac{i}{2b}\right) |V|$

- $V_0 \geq V$

- $V' \geq \frac{1}{2} |V|$

- ***“The algorithm clusters at least half of the nodes”***

**Objective:** In a c.c. remove from  $Q_i$  all terminals with same  $\text{id}(v)$  prefix except one

- Keep c.c. *“small”*
- Divide it if not possible by removing nodes

**Outline:**

- $2b^2$  *steps*, each computing a forest
- Resulting into a sequence of forests  $F_0 \dots F_{2b^2}$



### Inductive definition:

- $F_0$  is a BFS forest with roots set  $Q_i$
- Let  $T$  be any tree in  $F_j$  and  $r$  its root
  - If  $\text{id}(r)[i] = 0$  the whole tree is **red**, otherwise **blue**
    - **red** vertexes stay **red**
    - Some **blue** nodes stay **blue**
    - Some others *propose* **red** trees to join
      - If accepted, they become **red**
      - Otherwise, they are **deleted**

### ***Proposal:***

$v \in V_j^{propose} \Leftrightarrow v$  is **blue**

$\wedge v$  is the only one in  $path(v, root(v))$   
that neighbours a **red** node

- Define  $T_v$  the (**blue**) subtree rooted at  $v$

$v$  is the only node in  $T_v$  that is also in  $V_j^{propose}$

### *Proposal:*

- Each node in  $V_j^{propose}$  proposes to a **red** neighbour
- Each **red** tree decides to grow or not
  - If it grows, it accepts all proposing subtrees
    - **blue** nodes become **red**
  - If not, all proposing subtrees are **deleted**
- **Criteria:** it decides to grow if gains at least  $\frac{|V(T)|}{2b}$  nodes



If a **red** tree doesn't decide to grow, it will  
neighbour **red** nodes only

- This means it will be able to delete nodes only once in the whole phase
  - ⇒ At most  $\frac{|V|}{2b}$  nodes are lost in each phase
  - ⇒ After the  $b$  phases at most  $\frac{|V|}{2}$  nodes are removed

```
1:  $V_0 \leftarrow V$ 
2:  $Q_0 \leftarrow V$ 
3: for  $i \in 0..b - 1$  do
4:   INIT  $F_0$ 
5:   for  $j \in 0..2b^2 - 1$  do
6:     BUILD  $V_j^{propose}$ 
7:      $F_{j+1} \leftarrow \text{STEP}$ 
8:    $V_{i+1} \leftarrow V(F_{2b^2})$ 
9:    $Q_{i+1} \leftarrow \text{roots}(F_{2b^2})$ 
```

$\left. \begin{array}{l} \text{lines 6-7} \end{array} \right\} O(\text{diam}(T_v))$        $\left. \begin{array}{l} \text{lines 5-7} \end{array} \right\} 2b^2 = O(\log^2 n)$        $\left. \begin{array}{l} \text{lines 3-9} \end{array} \right\} b = O(\log n)$

- Recall invariant (1.)
    - $\forall v \in V : \text{dist}_G(Q_i, v) = O(\log^3 n)$ , for all  $i \in 0..b$
    - Hence,  $\text{diam}(T_v) = O(\log^3 n)$ , for all  $v \in V$
  - Complexity is  $\#steps \cdot \#phases \cdot O(\text{diam}(T_v))$   
 $= O(\log n) \cdot O(\log^2 n) \cdot O(\log^3 n)$
- $\Rightarrow$  The algorithm runs in  $O(\log^6 n)$  communication steps



- We've seen how to build a **low diameter clustering**
  - In  $O(\log^6 n)$  communication steps
  - It clusters at least  $\frac{n}{2}$  nodes



- We've seen how to build a **low diameter clustering**
  - In  $O(\log^6 n)$  communication steps
  - It clusters at least  $\frac{n}{2}$  nodes
- We can apply that until all nodes have a color
  - $O(\log n)$  steps and therefore  $O(\log n)$  colors
  - **Network decomposition** in  $O(\log^7 n)$



- We've seen how to build a **low diameter clustering**
  - In  $O(\log^6 n)$  communication steps
  - It clusters at least  $\frac{n}{2}$  nodes
- We can apply that until all nodes have a color
  - $O(\log n)$  steps and therefore  $O(\log n)$  colors
  - **Network decomposition** in  $O(\log^7 n)$
- We solve MIS [1] for each color ( $O(\log n) \cdot \dots$ )
  - In parallel in the clusters ( $\dots \cdot O(\log^3 n \cdot \log^2 n)$ )

- We've seen how to build a **low diameter clustering**
    - In  $O(\log^6 n)$  communication steps
    - It clusters at least  $\frac{n}{2}$  nodes
  - We can apply that until all nodes have a color
    - $O(\log n)$  steps and therefore  $O(\log n)$  colors
    - **Network decomposition** in  $O(\log^7 n)$
  - We solve MIS [1] for each color ( $O(\log n) \cdot \dots$ )
    - In parallel in the clusters ( $\dots \cdot O(\log^3 n \cdot \log^2 n)$ )
- ⇒ We end up solving MIS in  $O(\log^7 n)$  rounds

# Bibliography

- [1] K. Censor-Hillel, M. Parter, and G. Schwartzman, "Derandomizing Local Distributed Algorithms under Bandwidth Restrictions." [Online]. Available: <https://arxiv.org/abs/1608.01689>
- [2] V. Rozhoň, B. Haeupler, and C. Grunau, "A Simple Deterministic Distributed Low-Diameter Clustering." [Online]. Available: <https://arxiv.org/abs/2210.11784>

- [3] V. Rozhoň and M. Ghaffari, "Polylogarithmic-Time Deterministic Network Decomposition and Distributed Derandomization." [Online]. Available: <https://arxiv.org/abs/1907.10937>
- [4] V. Rozhoň, M. Elkin, C. Grunau, and B. Haeupler, "Deterministic Low-Diameter Decompositions for Weighted Graphs and Distributed and Parallel Applications." [Online]. Available: <https://arxiv.org/abs/2204.08254>