Efficient Low Diameter Clustering

with strong diameter in the CONGEST model

Christian Micheletti

Università degli Studi di Padova

About this Presentation

Università degli Studi di Padova

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

Intuition

- 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

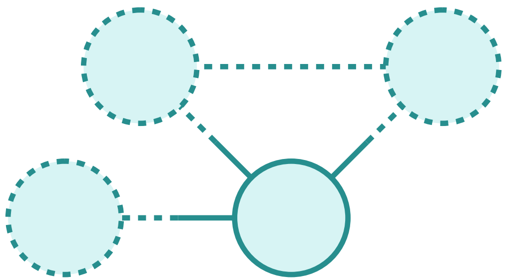
Distribution ⇒ Multiple processors

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

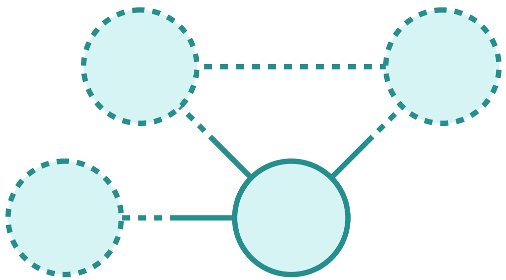
💡 Each node gives a partial solution

- 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 ≤ 1 arc between two nodes



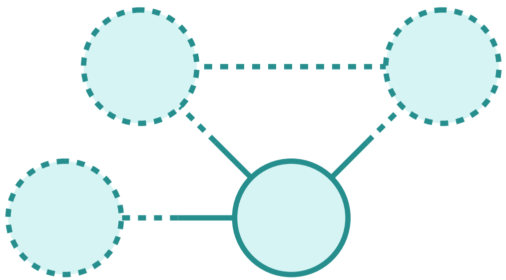
- In the **PN-Network** a node only knows its ***neighbours***
  - And how to "contact" them
- There are no **self loops**
- Connection is two-way
- There is ≤ 1 arc between two nodes



🚨 A node can't see the whole topology

A First Simple Model

- 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 ≤ 1 arc between two nodes



🚨 A node can't see the whole topology

🚨 All nodes appear identical

We add **unique identifiers** to the nodes

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

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

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.[1] 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] Without loss of generality.

W.l.o.g.[2] 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

[2] 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 id(v) = 0 *"waves hello"* to neighbours
  - ...sending them a message

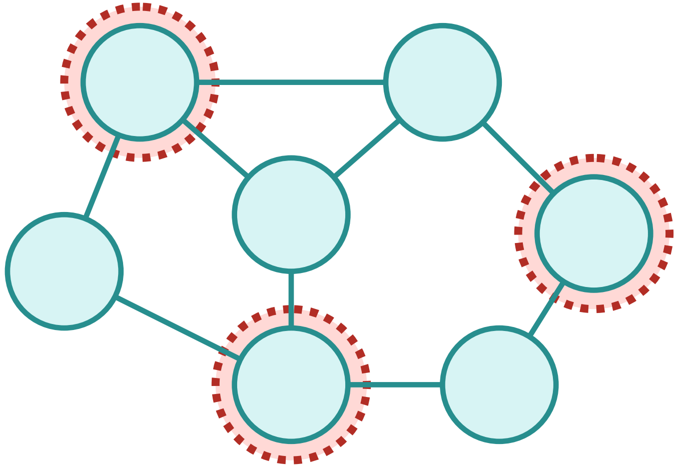
Wave: A First Example

- The node with 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**

Wave: A First Example

- The node with 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 $O(diam(G))$

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



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

A Second Example (Naive-MIS)

💡 Let's leverage id(v) to select the next MIS node

- At round #i, node v : 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

A Second Example (Naive-MIS)

💡 Let's leverage id(v) to select the next MIS node

- At round #i, node v : 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**

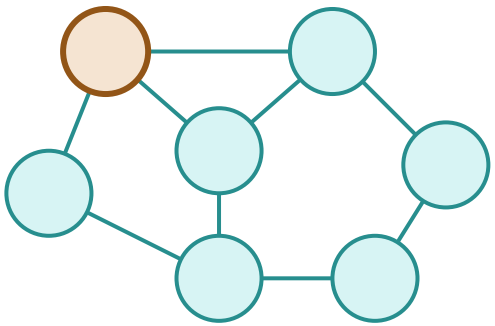
💡 Running a centralized algorithm on a single node would take O(1) rounds

- 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

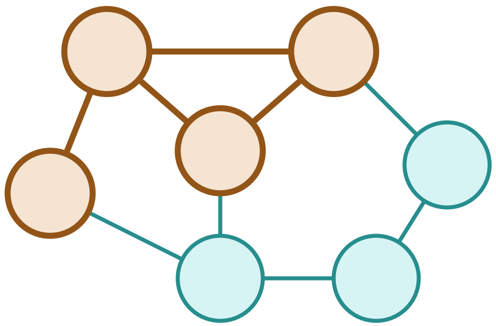
Gathering All

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

$ball(0, v)$



$ball(1, v)$



- 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...

🚨 $\Rightarrow$ We can't use Gather-All in the CONGEST model

Solving problems in CONGEST

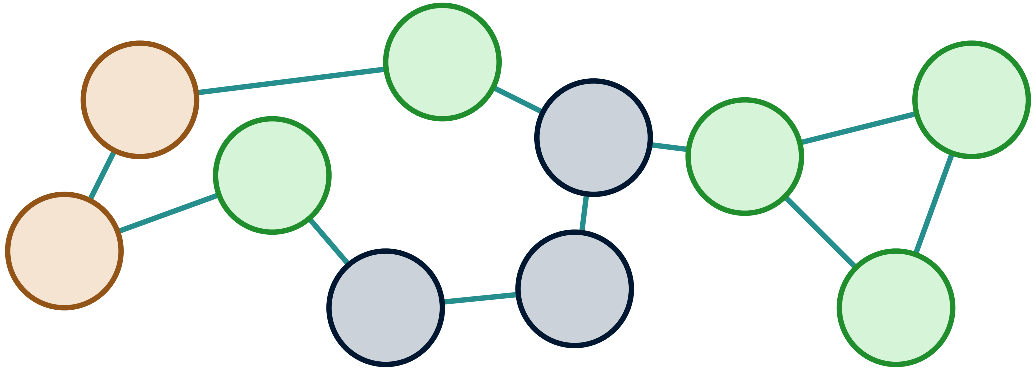
- Censor-Hillel et al. [1] provided an algorithm that solves MIS in $O\left(diam(G)\log^2 n\right)$ in CONGEST

🚨 The diameter can be very large

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

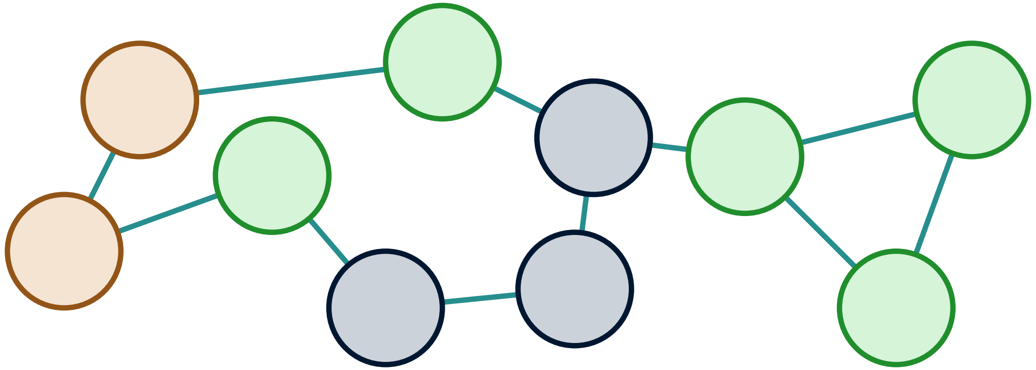
Network Decomposition

- 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



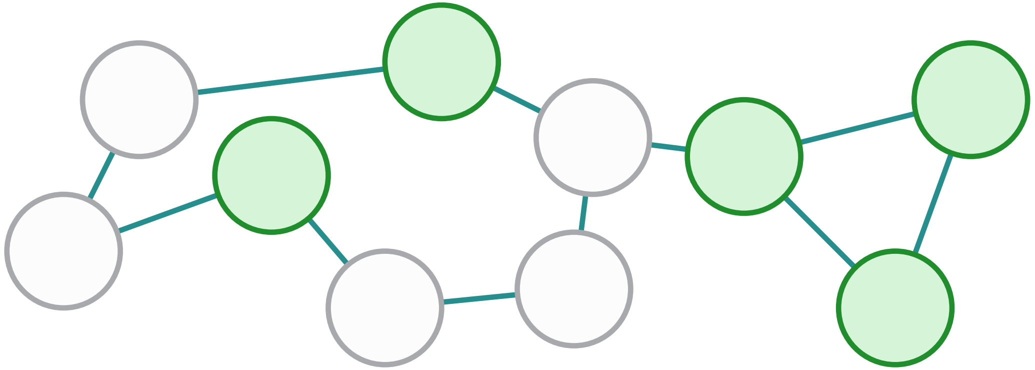
Solving MIS in a color gives a partial solution

- 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"***



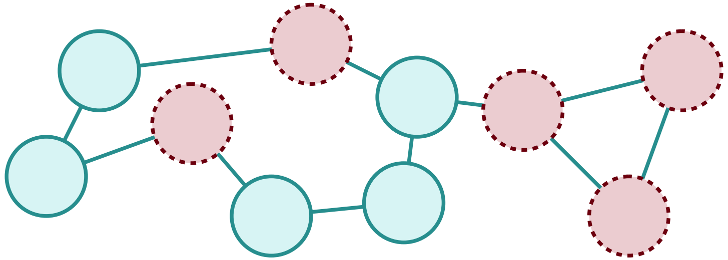
A ***low diameter clustering*** $\mathscr{C} \subseteq 2^V$ for a graph G with diameter d is such:

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



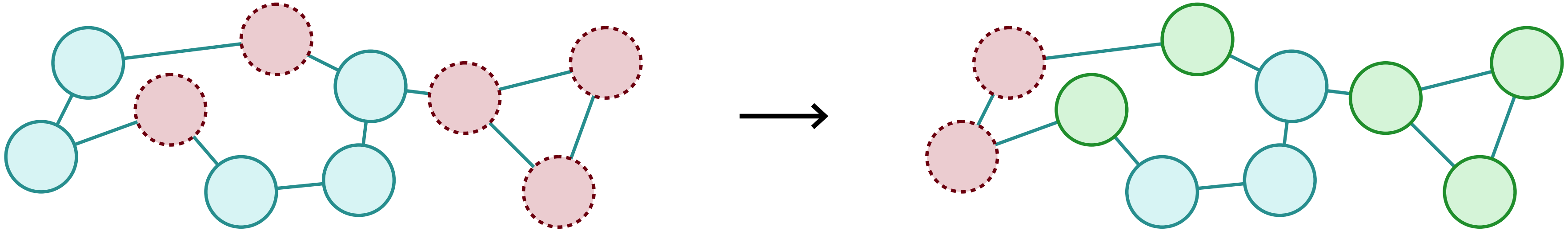
Main iteration:

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



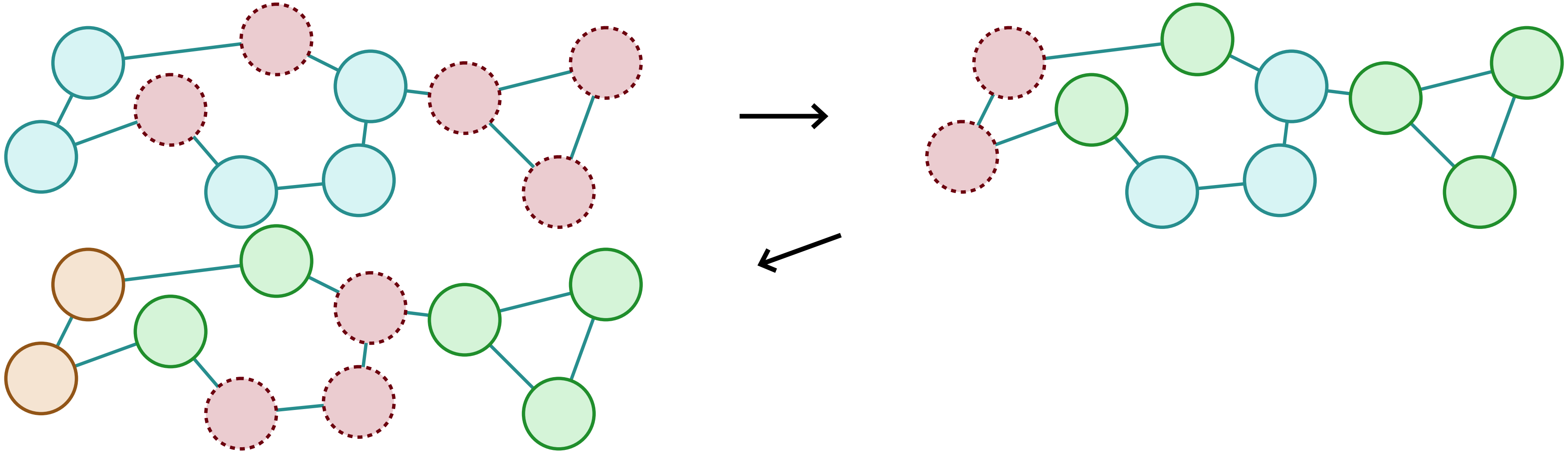
Main iteration:

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



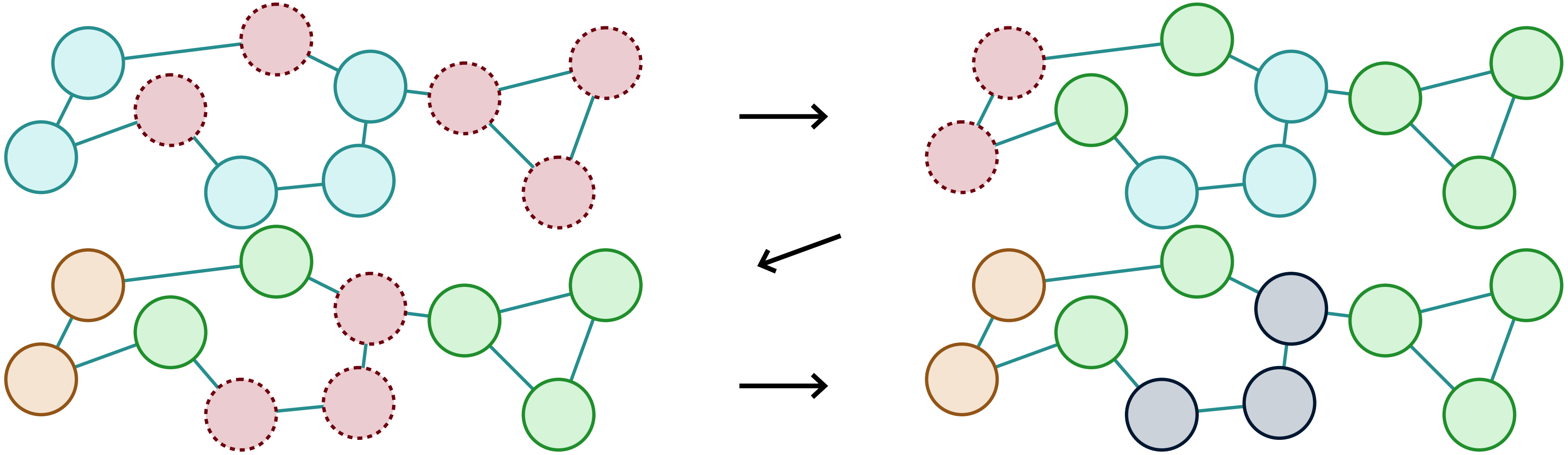
Main iteration:

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:

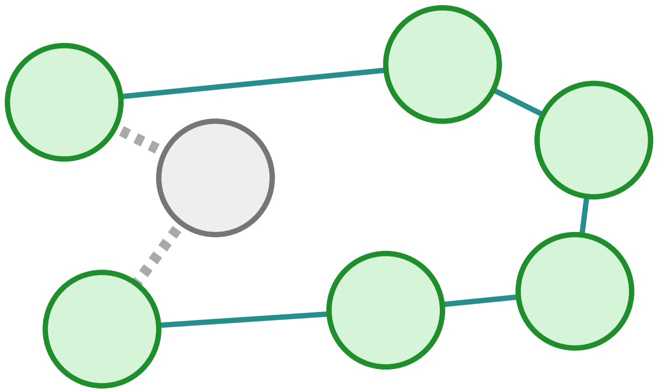
1. Find a low diameter clustering
2. Assign a free color to its nodes
3. Repeat to discarded nodes until there are no more left



- Our previous definition of diamter 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 $d = O(\log^3 n)$
  - Directly strong diameter

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

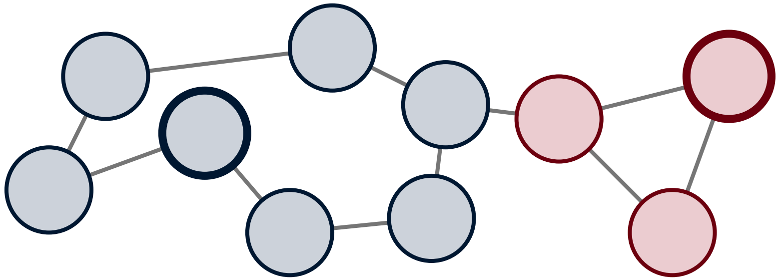
Objectives:

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 nodes separating c.c.s "too far away"

Objectives:

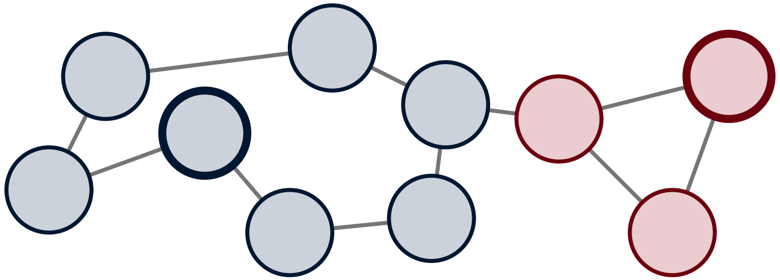
1. Creating connected components with ***"low"*** diameters
   - Keep track of the "center" of the c.c.
     - A.k.a. ***Terminal***
   - We will merge c.c.s
     - Only one terminal is going to be the new center
   - Remove nodes separating c.c.s "too far away"
2. Cluster **at least half** of the nodes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

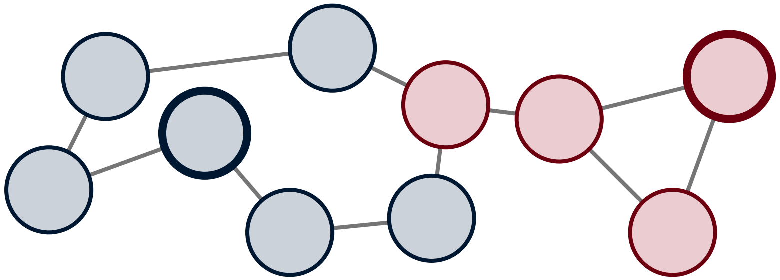
Visual Hint



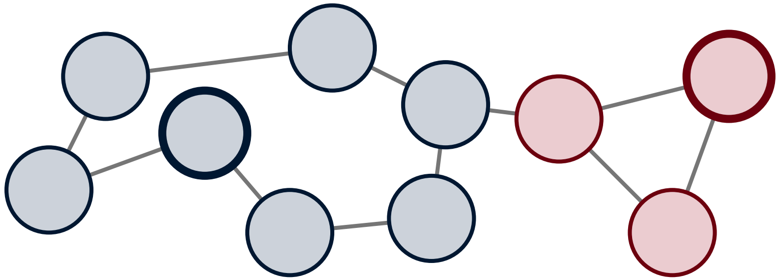
Visual Hint

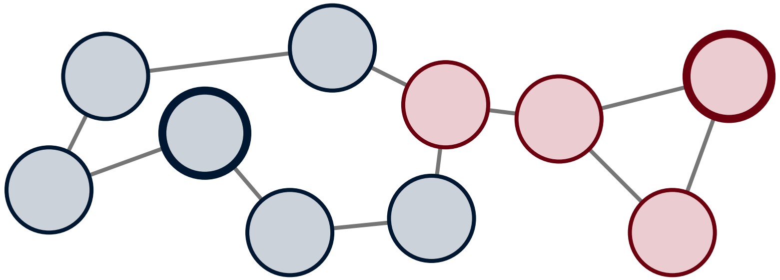
Università degli Studi di Padova

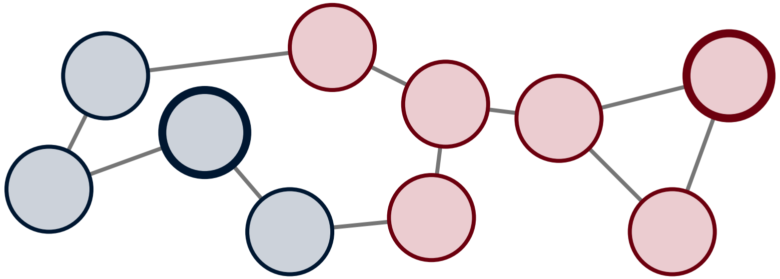




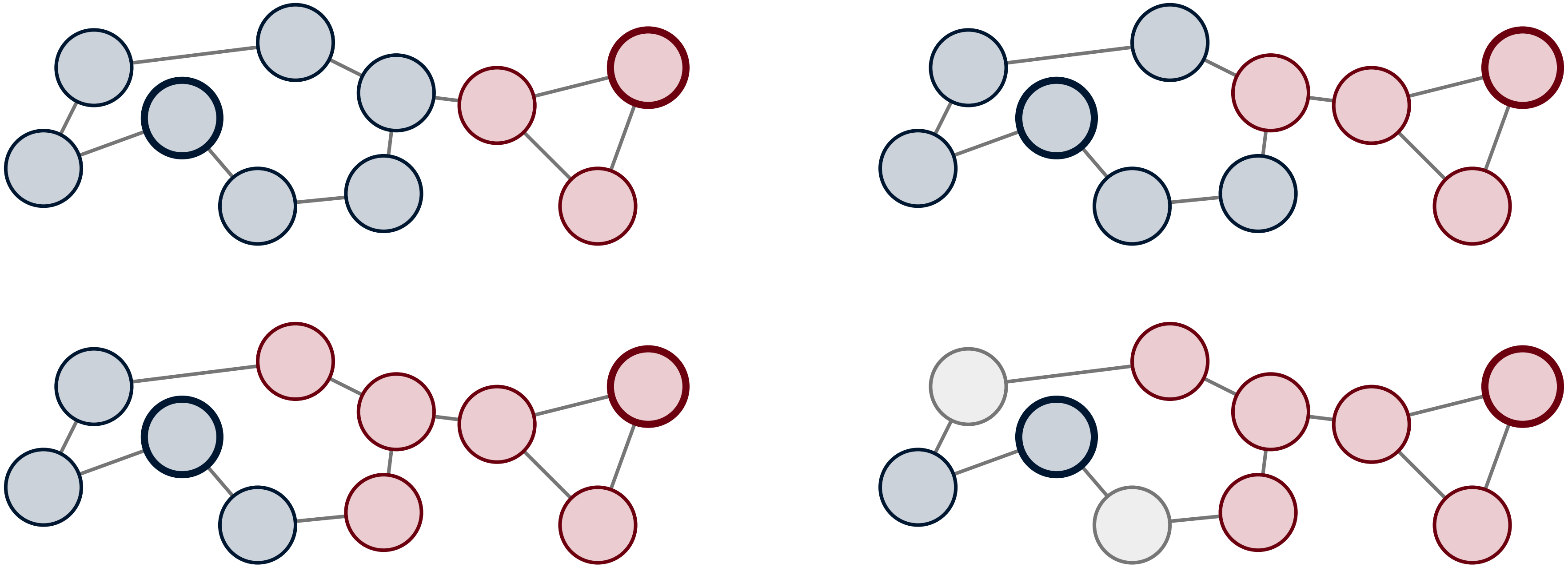
Visual Hint







Visual Hint



Phases

- There are $b = \log\left(\max_{v \in V} \mathrm{id}(v)\right) = O(\log n)$ **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. $dist_G(Q_i, v) \leq R_i$ for all $v \in V_i$
   - **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 $\mathrm{id}(q_1)[0..i] = \mathrm{id}(q_2)[0..i]$
   - For $i = 0$ it's trivially true
   - For $i = b$ there is ≤ 1 terminal in each c.c.

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

Phase Invariants $\forall i \in [0..b]$

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 different 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 $F_0..F_{2b^2}$

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 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, the 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

Observations

🎶 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
  $\Rightarrow$ At most $\frac{|V|}{2b}$ nodes are lost in each phase
  $\Rightarrow$ After the b phases at most $\frac{|V|}{2}$ nodes are removed

Observations

🎶 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
  $\Rightarrow$ At most $\frac{|V|}{2b}$ nodes are lost in each phase
  $\Rightarrow$ After the b phases at most $\frac{|V|}{2}$ nodes are removed
- `red` trees grow at most of $(i+1)\cdot O\left(\log^2 n\right)$

High Level Pseudocode

$V_0 \leftarrow V$

$Q_0 \leftarrow V$

for $i \in 0..b-1$ **do**

  INIT F_0

  for $j \in 0..2b^2-1$ **do**

    BUILD $V_j^{propose}$ $O(diam(T_v))$ $2b^2 = O(\log^2 n)$ $b = O(\log n)$

    $F_{j+1} \leftarrow$ STEP

  $V_{i+1} \leftarrow V(F_{2b^2})$

  $Q_{i+1} \leftarrow roots(F_{2b^2})$

Complexity

- Recall invariant (1.)
  - $\forall v \in V : dist_G\left(Q_i, v\right) = O\left(\log^3 n\right)$, for all $i \in 0..b$
  - Hence, $diam\left(T_v\right) = O\left(\log^3 n\right)$, for all $v \in V$
- Complexity is $\#steps \cdot \#phases \cdot O\left(diam\left(T_v\right)\right)$
  $= O(\log n) \cdot O\left(\log^2 n\right) \cdot O\left(\log^3 n\right)$

$\Rightarrow$ The algorithm runs in $O\left(\log^6 n\right)$ communication steps

UNIVERSITÀ DEGLI STUDI DI PADOVA

Recap

- 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 $(O(\log n) \cdot \ldots)$
  - In parallel in the clusters $(\ldots \cdot O(\log^3 n \cdot \log^2 n))$

- We've seen how to build a **low diameter clustering**
  - In $O\left(\log^6 n\right)$ 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 $O\left(\log^7 n\right)$
- We solve MIS [1] for each color $(O(\log n) \cdot \ldots)$
  - In parallel in the clusters $\left(\ldots \cdot O\left(\log^3 n \cdot \log^2 n\right)\right)$

$\Rightarrow$ We end up solving MIS in $O\left(\log^7 n\right)$ 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