Distributed Computing
12 - Complexities, Volume and Exercises

Mikaël Rabie
Université de Paris, IRIF
Creating Complexities
Complexities on Graphs with the LOCAL Model

Figures from https://jukkasuomela.fi/landscape-of-locality/
For any $k > 0$, there exists a problem $P_k$ that is solvable in time $\Theta(n^{1/k})$. 

Chang, Pettie (2019)
Comb Tree: A path $V_2$ of degree 3 nodes. To each of those nodes there is a path appended. For each node $v \in V_2$ either:

- Its appended path is 2-colored ($v \in D_2$)
- It is 2-colored with its $V_2 \setminus D_2$ neighbors

Question: Find a $O(\sqrt{n})$ algorithm.
Comb Tree: A path $V_2$ of degree 3 nodes. To each of those nodes there is a path appended. For each node $v \in V_2$ either:

- Its appended path is 2-colored ($v \in D_2$)
- It is 2-colored with its $V_2 \setminus D_2$ neighbors

Question: Find a $O(\sqrt{n})$ algorithm.
Question: On which graphs the problem needs $\Omega(\sqrt{n})$?
Question: On which graphs the problem needs $\Omega(\sqrt{n})$?
Question: On which graphs the problem needs $\Omega(\sqrt{n})$?

Question: How to generalize this problem with the same results on any graphs?
Question: On which graphs the problem needs $\Omega(\sqrt{n})$?

Question: How to generalize this problem with the same results on any graphs?

- $G_1$: input graph
- $V_1 = \{ v \in V(G_1) : \text{degree}(v) \leq 2 \}$
- $G_2 = G_1 \setminus V_1$
- $V_2 = \{ v \in V(G_2) : \text{degree}(v) \leq 2 \}$
- $V_1$ nodes accept if, in regard to its neighbors in $V_1$:
  - They are 2-colored with colors 1 and 2
  - They all are in color 3
- $D_2 = \{ v \in V_2 : \text{its } V_1 \text{ neighbors are 2-colored} \}$
  - $D_2$ nodes accept if they are in color 4
- $V_2 \setminus D_2$ nodes accept if, in regard to its neighbors in $V_2 \setminus D_2$:
  - They are 2-colored with colors 1 and 2
  - They have 2 of those neighbors and are in color 3
Generalization to $n^{1/k}$

- $G_1$: input graph
- $V_i = \{v \in V(G_i): \text{degree}(v) \leq 2\}$
- $G_i = G_{i-1} \setminus V_{i-1}$
- $V_{k+1} = V(G_{k+1})$
- A vertex in $V(G)$ is **exempted** if
  - it has a lower level exempted neighbor
  - it has a lower level 2-colored neighbor
  - it is in $V_{k+1}$
- $D_i \subseteq V_i$ are the exempted nodes of level $i$
- $V_i \setminus D_i$ nodes ($i \leq k$) accept if, in regard to its neighbors in $V_i \setminus D_i$:
  - They are 2-colored with colors 1 and 2
  - They have 2 of those neighbors and are in color 3
The Centralized Local Model
SLOCAL Model

- Each node is activated one after another, to compute its own output
- A node has access to the outputs already computed to produce its own
- Complexity: maximal radius needed among nodes
- **Greedy** problems can be solved in radius $O(1)$
SLOCAL Model

- Each node is activated one after another, to compute its own output
- A node has access to the outputs already computed to produce its own
- Complexity: maximal radius needed among nodes
- **Greedy** problems can be solved in radius $O(1)$
SLOCAL Model

- Each node is activated one after another, to compute its own output
- A node has access to the outputs already computed to produce its own
- Complexity: maximal radius needed among nodes
- **Greedy** problems can be solved in radius $O(1)$
SLOCAL Model

- Each node is activated one after another, to compute its own output
- A node has access to the outputs already computed to produce its own
- Complexity: maximal radius needed among nodes
- **Greedy** problems can be solved in radius $O(1)$
SLOCAL Model

- Each node is activated one after another, to compute its own output
- A node has access to the outputs already computed to produce its own
- Complexity: maximal radius needed among nodes
- **Greedy** problems can be solved in radius $O(1)$
SLOCAL Model

- Each node is activated one after another, to compute its own output
- A node has access to the outputs already computed to produce its own
- Complexity: maximal radius needed among nodes
- Greedy problems can be solved in radius $O(1)$
SLOCAL Model

- Each node is activated one after another, to compute its own output
- A node has access to the outputs already computed to produce its own
- Complexity: maximal radius needed among nodes
- Greedy problems can be solved in radius $O(1)$
SLOCAL Model

- Each node is activated one after another, to compute its own output
- A node has access to the outputs already computed to produce its own
- Complexity: maximal radius needed among nodes
- **Greedy** problems can be solved in radius $O(1)$
SLOCAL Model

- Each node is activated one after another, to compute its own output
- A node has access to the outputs already computed to produce its own
- Complexity: maximal radius needed among nodes
- **Greedy** problems can be solved in radius $O(1)$
Each node is activated one after another, to compute its own output
- A node has access to the outputs already computed to produce its own
- Complexity: maximal radius needed among nodes
- Greedy problems can be solved in radius $O(1)$
SLOCAL Model

- Each node is activated one after another, to compute its own output
- A node has access to the outputs already computed to produce its own
- Complexity: maximal radius needed among nodes
- **Greedy** problems can be solved in radius $O(1)$
CentLOCAL Model

- In parallel, each node $v$:
  - Knows its own $Id_v$ and degree $d_{Id_v}$
  - At each step, they send a request $(Id_u, k)$, with $k \leq d_{Id_u}$
  - They get $(Id_w, d_{Id_w}, k')$ such that $(u, v) \in E$ are connected by port $k$ from $u$ and $k'$ from $w$
- Complexity: maximal number of requests from a node
CentLOCAL Model

- In parallel, each node $v$:
  - Knows its own $Id_v$ and degree $d_{Id_v}$
  - At each step, they send a request $(Id_u, k)$, with $k \leq d_{Id_u}$
  - They get $(Id_w, d_{Id_w}, k')$ such that $(u, v) \in E$ are connected by port $k$ from $u$ and $k'$ from $w$

- Complexity: maximal number of requests from a node

Request: $(14, 2)$
CentLOCAL Model

- In parallel, each node $v$:
  - Knows its own $Id_v$ and degree $d_{Id_v}$
  - At each step, they send a request ($Id_u$, $k$), with $k \leq d_{Id_u}$
  - They get ($Id_w$, $d_{Id_w}$, $k'$) such that $(u, v) \in E$ are connected by port $k$ from $u$ and $k'$ from $w$
- Complexity: maximal number of requests from a node

Request: (14,2)
- In parallel, each node $v$: 
  - Knows its own $Id_v$ and degree $d_{Id_v}$
  - At each step, they send a request $(Id_u, k)$, with $k \leq d_{Id_u}$
  - They get $(Id_w, d_{Id_w}, k')$ such that $(u,v) \in E$ are connected by port $k$ from $u$ and $k'$ from $w$
- Complexity: maximal number of requests from a node

Request: $(14, 2) \Rightarrow (2, 4, 1)$
CentLOCAL Model

- In parallel, each node \( v \):
  - Knows its own \( Id_v \) and degree \( d_{Id_v} \)
  - At each step, they send a request \((Id_u, k)\) with \( k \leq d_{Id_u} \)
  - They get \((Id_w, d_{Id_w}, k')\) such that \((u, v) \in E\) are connected by port \( k \) from \( u \) and \( k' \) from \( w \)
- Complexity: maximal number of requests from a node

Request: \((14,3)\)
CentLOCAL Model

- In parallel, each node $v$:
  - Knows its own $Id_v$ and degree $d_{Id_v}$
  - At each step, they send a request $(Id_u, k)$, with $k \leq d_{Id_u}$
  - They get $(Id_w, d_{Id_w}, k')$ such that $(u, v) \in E$ are connected by port $k$ from $u$ and $k'$ from $w$
- Complexity: maximal number of requests from a node

Request: $(14, 3) \Rightarrow (8, 2, 2)$
CentLOCAL Model

- In parallel, each node $v$:
  - Knows its own $Id_v$ and degree $d_{Id_v}$
  - At each step, they send a request $(Id_u, k)$, with $k \leq d_{Id_u}$
  - They get $(Id_w, d_{Id_w}, k')$ such that $(u, v) \in E$ are connected by port $k$ from $u$ and $k'$ from $w$
- Complexity: maximal number of requests from a node

Request: (2,3)
CentLOCAL Model

- In parallel, each node $v$:
  - Knows its own $Id_v$ and degree $d_{Id_v}$
  - At each step, they send a request $(Id_u, k)$, with $k \leq d_{Id_u}$
  - They get $(Id_w, d_{Id_w}, k')$ such that $(u, v) \in E$ are connected by port $k$ from $u$ and $k'$ from $w$
- Complexity: maximal number of requests from a node

Request: $(2, 3) \Rightarrow (10, 4, 3)$
Greedy Problems

Problem $A$ can be solved in time $\Theta(f(n))$ in the LOCAL model

$\Rightarrow$ $A$ can be solved in time $\Omega(f(n))$ and $O(\Delta f(n))$ in the CentLOCAL model.
Problem $A$ can be solved in time $\Theta(f(n))$ in the LOCAL model.

$\Rightarrow$ $A$ can be solved in time $\Omega(f(n))$ and $O\left(\Delta^{f(n)}\right)$ in the CentLOCAL model.
Problem $A$ can be solved in time $\Theta(f(n))$ in the LOCAL model

$\Rightarrow A$ can be solved in time $\Omega(f(n))$ and $O\left(\Delta^f(n)\right)$ in the CentLOCAL model

**Even et. al (2018)**

There is a CentLOCAL algorithm in time $O(\Delta \times \log^* n + \Delta^3)$ for $\leq \Delta^2$-coloring a graph.

There is a CentLOCAL algorithm in time $O(\Delta \times \log^* n + \Delta^3)$ for orienting a graph where the longer oriented path is of length $\leq \Delta^2$.

Any greedy problem can be solved in time $O(f(\Delta) \times \log^* n)$. 
In the CentLOCAL model, if $n$ is not given in advance and identifiers do not require to be polynomial in $n$, there is no problem whose time complexity is in $\omega(\log^* n) \cap o(n)$.
Rosenbaum and Suomela (2020)
In the CentLOCAL model, if \( n \) is not given in advance and identifiers do not require to be polynomial in \( n \), there is no problem whose time complexity is in \( \omega(\log^* n) \cap o(n) \).

- Take \( N \) such that \( T(N) \ll N \)
- Do a distance \( N \)-coloring
- Simulate the algorithm with the new identifiers
Maximal Matching Lower Bound
Maximal Matching Algorithm

- Computer network with port numbering
- Bipartite, 2-colored graph
- \( \Delta \)-regular (here \( \Delta = 3 \))

Output: maximal matching
Maximal Matching Algorithm

Very simple algorithm

unmatched white nodes:
send proposal to port 1
Maximal Matching Algorithm

Very simple algorithm

unmatched white nodes:
send proposal to port 1

black nodes:
accept the first proposal you get, reject everything else
(break ties with port numbers)
**Very simple algorithm**

**unmatched white nodes:**
send *proposal* to port 1

**black nodes:**
*accept* the first proposal you get, *reject* everything else
(break ties with port numbers)
Maximal Matching Algorithm

Very simple algorithm

unmatched white nodes:
send *proposal* to port 2
Maximal Matching Algorithm

Very simple algorithm

unmatched white nodes: send proposal to port 2

black nodes: accept the first proposal you get, reject everything else (break ties with port numbers)
Maximal Matching Algorithm

**Very simple algorithm**

**unmatched white nodes:**
send *proposal* to port 2

**black nodes:**
*accept* the first proposal you get, *reject* everything else
(break ties with port numbers)
Maximal Matching Algorithm

Very simple algorithm

unmatched white nodes: send proposal to port 3
Maximal Matching Algorithm

Very simple algorithm

unmatched white nodes:
send proposal to port 3

black nodes:
accept the first proposal you get, reject everything else
(break ties with port numbers)
Very simple algorithm

unmatched white nodes: send proposal to port 3

black nodes: accept the first proposal you get, reject everything else (break ties with port numbers)
Maximal Matching Algorithm

Very simple algorithm

Finds a maximal matching in $O(\Delta)$ communication rounds

Note: running time does not depend on $n$
Maximal Matching in $\Delta$-regular graphs

- Maximal Matching, $\Sigma = \{M, O, P\}$
- $B = (MO^{\Delta-1} | P^\Delta)$, $W = (M[PO]^{\Delta-1} | O^\Delta)$
Maximal Matching in $\Delta$-regular graphs

- Maximal Matching, $\Sigma = \{M, O, P\}$
- $B = (MO^{\Delta-1} \mid P^\Delta)$, $W = (M[PO]^{\Delta-1} \mid O^\Delta)$

**Question**: Do one-round reduction on the $W$-algorithm.
Parametrized Matching

Parametrized problem $\Pi_{\Delta}(x, y)$

- Addition of joker output $X$.
- $B_{\Delta}(x, y) = \left( MO^{d-1} | P^d \right) O^y X^x$
- $W_{\Delta}(x, y) = \left( [MX][POX]^{d-1} | [OX]^d \right) [POX]^y [MPOX]^x$

Questions:

1. Prove that: Maximal Matching in $T$ rounds, $\Rightarrow \Pi_{\Delta}(0, 0)$ in $T$ rounds.
2. (At home) Prove that: $\Pi_{\Delta}(0, 0)$ on $W$ in $T$ rounds $\Rightarrow \Pi_{\Delta}(1, 0)$ on $B$ in $T - 1$ rounds.

We have: $\Pi_{\Delta}(x, y)$ in $T$ rounds $\Rightarrow \Pi_{\Delta}(x + 1, y + x)$ in $T - 1$ rounds.

3. What lower bound on the complexity can you deduce?
Weak $k$-matching

**Weak $k$-matching**: For each node $v$,

- $v$ is matched to $l \in [1, k]$ nodes
- $v$ is unmatched and all the neighbors of $v$ are matched

**Questions**: Suppose that we can solve Maximal Matching in $o(\Delta)$ rounds

1. Prove that we solve weak $\Delta^{1/2}$-Matching in $o(\Delta^{1/2})$ rounds.
2. Deduce that we solve $\Pi_{\Delta}(\Delta^{1/2} - 1, 0)$ in $o(\Delta^{1/2})$ rounds.
3. Deduce that we solve $\Pi_{\Delta}(\Delta^{1/2} + o(\sqrt{\Delta}), o(\Delta))$ in 0 rounds.
Weak $\sqrt{\Delta}$-matching
**Weak \( k \)-matching**

**Weak \( k \)-matching** : For each node \( v \),

- \( v \) is matched to \( l \in [1, k] \) nodes
- \( v \) is unmatched and all the neighbors of \( v \) are matched

**Questions** : Suppose that we can solve Maximal Matching in \( o(\Delta) \) rounds

1. Prove that we solve weak "\( \Delta^{1/2} \)-Matching" in \( o(\Delta^{1/2}) \) rounds.
2. Deduce that we solve \( \Pi_\Delta(\Delta^{1/2} - 1, 0) \) in \( o(\Delta^{1/2}) \) rounds.
3. Deduce that we solve \( \Pi_\Delta(\Delta^{1/2} + o(\sqrt{\Delta}), o(\Delta)) \) in 0 rounds.

**Reminder** :

- \( B_\Delta(x, y) = \left( \MO^{d-1} \right| \P^d \right) \O^y \X^x \)
- \( W_\Delta(x, y) = \left( [\MX][\POX]^{d-1} \right| [\OX]^d \right) \POX^y \MPOX^x \)

\[ d = \Delta - x - y \]
Weak \( k \)-matching

**Weak \( k \)-matching**: For each node \( v \),

- \( v \) is matched to \( l \in [1, k] \) nodes
- \( v \) is unmatched and all the neighbors of \( v \) are matched

**Questions**: Suppose that we can solve Maximal Matching in \( o(\Delta) \) rounds

1. Prove that we solve weak "\( \Delta^{1/2} \)-Matching" in \( o(\Delta^{1/2}) \) rounds.
2. Deduce that we solve \( \Pi_\Delta(\Delta^{1/2} - 1, 0) \) in \( o(\Delta^{1/2}) \) rounds.
3. Deduce that we solve \( \Pi_\Delta(\Delta^{1/2} + o(\sqrt{\Delta}), o(\Delta)) \) in 0 rounds.

**Balliu et. al (2019)**

There is no deterministic distributed algorithm that solves Maximal Matching in \( o(\Delta) \) rounds in the LOCAL model.
