Decidability of distributed complexity of locally checkable problems on paths

Alkida Balliu, Sebastian Brandt, Yi-Jun Chang, Dennis Olivetti, Mikaël Rabie, Jukka Suomela

ANR DESCARTES/ESTATE

Tuesday, April 2
LCL Problems on Paths

......

1  28  37  52  8  32  46  47  73  5  3  ......
LCL Problems on Paths
LCL Problems on Paths

1 Coloring.

Maximal Independent Set.

Independent Set.
LCL Problems on Paths
LCL Problems on Paths

\[ \begin{array}{ccccccccccc}
1 & 28 & 37 & 52 & 8 & 32 & 46 & 47 & 73 & 5 & 3 \\
\end{array} \]
LCL Problems on Paths

......

1  28  37  52  8  32  46  47  73  5  3 ......
LCL Problems on Paths

1 Coloring.

Maximal Independent Set.

Independent Set.
LCL Problems on Paths
LCL Problems on Paths

……
……

1 28 37 52 8 32 46 47 73 5 3

1 2 3 2 3 1 2 1 3 2 1
LCL Problems on Paths

3 Coloring.
3 Coloring.

2 Coloring.
LCL Problems on Paths

- 3 Coloring.
- 2 Coloring.
- Maximal Independent Set.
LCL Problems on Paths

- 3 Coloring.
- 2 Coloring.
- Maximal Independent Set.
- Independent Set.
3 Coloring a Path

Worst case communication complexity: $\Theta(n)$. 

Mikaël RABIE
Decidability of LCL Problems on Paths  Tuesday, April 2 3 / 19
3 Coloring a Path

Worst case communication complexity: $\Theta(n)$. 

Mikaël RABIE

Decidability of LCL Problems on Paths

Tuesday, April 2
3 Coloring a Path

Worst case communication complexity: $\Theta(n)$. 

Mikaël RABIE

Decidability of LCL Problems on Paths

Tuesday, April 2
3 Coloring a Path

Worst case communication complexity: $\Theta(n)$. 

Mikaël RABIE
Decidability of LCL Problems on Paths
3 Coloring a Path

Worst case communication complexity: $\Theta(n)$. 

Mikaël RABIE

Decidability of LCL Problems on Paths
Tuesday, April 2
3 Coloring a Path

Worst case communication complexity: $\Theta(n)$. 

Mikaël RABIE

Decidability of LCL Problems on Paths
3 Coloring a Path

Worst case communication complexity: $\Theta(n)$. 

Mikaël Rabie
3 Coloring a Path

Worst case communication complexity: $\Theta(n)$. 
3 Coloring a Path

Worst case communication complexity: $\Theta(n)$.

Mikaël Rabie

Decidability of LCL Problems on Paths

Tuesday, April 2
3 Coloring a Path

Worst case communication complexity: $\Theta(n)$. 

Mikaël RABIE
Decidability of LCL Problems on Paths
Tuesday, April 2

3 / 19
Worst case communication complexity: $\Theta(n)$. 
3-coloring in $O(\log^* n)$ Communications

Cole, Vishkin (1986)

There exists a LCL algorithm to 3-color a path in $O(\log^* n)$ communications.
From $n$ colors to $\log n$ colors

\[
\begin{align*}
&42 \\
&\cdots \\
&102 \\
&\cdots \\
&36 \\
&\cdots
\end{align*}
\]
The $\log^* n$ Complexity

3-coloring a Path

From $n$ colors to $\log n$ colors

After $\log^* n$ iterations, $O(1)$ bits.

Mikaël RABIE
Decidability of LCL Problems on Paths  
Tuesday, April 2  
5 / 19
From $n$ colors to $\log n$ colors
From $n$ colors to $\log n$ colors

$\log^* n$ Complexity

3-coloring a Path

Mikaël RABIE
Decidability of LCL Problems on Paths
Tuesday, April 2
From $n$ colors to $\log n$ colors

After $\log^* n$ iterations, $O(1)$ bits.
From $n$ colors to $\log n$ colors

After $\log^* n$ iterations, $O(1)$ bits.

Mikaël RABIE
Decidability of LCL Problems on Paths
Tuesday, April 25 / 19
From $n$ colors to $\log n$ colors

After $\log^* n$ iterations, $O(1)$ bits.
From \( n \) colors to \( \log n \) colors

After \( \log^* n \) iterations, \( O(1) \) bits.
From $n$ colors to $\log n$ colors

$n$ colors $\Rightarrow \log n$ bits $\Rightarrow 2 \log n$ new colors $\Rightarrow \log \log n + 1$ bits
From $n$ colors to $\log n$ colors

$n$ colors $\Rightarrow \log n$ bits $\Rightarrow 2 \log n$ new colors $\Rightarrow \log \log n + 1$ bits

After $\log^* n$ iterations, $O(1)$ bits.
Coloration Lower Bound

Linial (1992)

An algorithm which colors the $n$-cycle with three colors requires time at least $\frac{1}{2}(\log^* n - 3)$. The same bound holds also for randomized algorithms.
Speed up Algorithm

$A$ : algorithm that $k$-colors nodes in $T$ rounds.
Speed up Algorithm

\( A : \) algorithm that \( k \)-colors nodes in \( T \) rounds.
Speed up Algorithm

$A$ : algorithm that $k$-colors nodes in $T$ rounds.

$c \in [1, k]$
Speed up Algorithm

$A$ : algorithm that $k$-colors nodes in $T$ rounds.

$c \in [1, k]$
Speed up Algorithm

$A$ : algorithm that $k$-colors nodes in $T$ rounds.

$c \in [1, k]$
Speed up Algorithm

\(A\) : algorithm that \(k\)-colors nodes in \(T\) rounds.

\[
\forall \text{id} \leq n \quad T - 1 \quad S_L \in 2^k \quad T - 1
\]
Speed up Algorithm

$A$ : algorithm that $k$-colors nodes in $T$ rounds.
**Speed up Algorithm**

$A$ : algorithm that $k$-colors nodes in $T$ rounds.

$c \in [1, k]$
Speed up Algorithm

A : algorithm that $k$-colors nodes in $T$ rounds.

$S_L \# S_R \in 2^k \times 2^k$
**Speed up Algorithm**

\( A \): algorithm that \( k \)-colors nodes in \( T \) rounds.

\[
T \quad c \in [1, k] \quad T
\]

\[
T - 1 \quad S_L \# S_R \in 2^k \times 2^k \quad T - 1
\]

\[
S_L \cap S_R = \emptyset
\]
A : algorithm that $k$-colors nodes in $T$ rounds.
**Speed up Algorithm**

A: algorithm that $k$-colors nodes in $T$ rounds.
Speed up Algorithm

$A$: algorithm that $k$-colors nodes in $T$ rounds.

$S_L \cap S_R = \emptyset$ & $S'_L \cap S'_R = \emptyset$
**Speed up Algorithm**

*A*: algorithm that \( k \)-colors nodes in \( T \) rounds.

\[
S_L \cap S_R = \emptyset \quad \text{and} \quad S'_L \cap S'_R = \emptyset
\]

\[
S'_L \cap S_R \neq \emptyset
\]
**Speed up Algorithm**

\( A \) : algorithm that \( k \)-colors nodes in \( T \) rounds.

\[ S_L \cap S_R = \emptyset \; \& \; S'_L \cap S'_R = \emptyset \]

\[ S'_L \cap S_R \neq \emptyset \]

\[ S_L \# S_R \neq S'_L \# S'_R \]
Speed up Algorithm

\( A \) : algorithm that \( k \)-colors nodes in \( T \) rounds.
\( A_1 \) : algorithm that \( 4^k \)-colors edges in \( T - 1 \) rounds.
Speed up Algorithm

$A_1$ : algorithm that $4^k$-colors edges in $T - 1$ rounds.

$c \in [1, 4^k]$
Speed up Algorithm

$A$ : algorithm that $k$-colors nodes in $T$ rounds.

$A_1$ : algorithm that $4^k$-colors edges in $T − 1$ rounds.

\[ \forall id \leq n \]

\[ c \in [1, 4^k] \]

\[ T − 1 \]

\[ T − 1 \]
Speed up Algorithm

$A$ : algorithm that $k$-colors nodes in $T$ rounds.

$A_1$ : algorithm that $4^k$-colors edges in $T - 1$ rounds.
Speed up Algorithm

$A$: algorithm that $k$-colors nodes in $T$ rounds.
$A_1$: algorithm that $4^k$-colors edges in $T - 1$ rounds.
Speed up Algorithm

$A$ : algorithm that $k$-colors nodes in $T$ rounds.
$A_1$ : algorithm that $4^k$-colors edges in $T - 1$ rounds.
Speed up Algorithm

$A$: algorithm that $k$-colors nodes in $T$ rounds.
$A_1$: algorithm that $4^k$-colors edges in $T - 1$ rounds.
$A_2$: algorithm that $4^{4^k}$-colors nodes in $T - 1$ rounds.

$c \in [1, 4^k]$
Speed up Algorithm

$A$ : algorithm that $k$-colors nodes in $T$ rounds.
$A_1$ : algorithm that $4^k$-colors edges in $T - 1$ rounds.
$A_2$ : algorithm that $4^{4^k}$-colors nodes in $T - 1$ rounds.

... 

$A_{\log^* n}$ : algorithm that $n$-colors nodes in $T - \frac{\log^* n}{2}$ rounds.
Speed up Algorithm

$A$ : algorithm that $k$-colors nodes in $T$ rounds.
$A_1$ : algorithm that $4^k$-colors edges in $T - 1$ rounds.
$A_2$ : algorithm that $4^{4^k}$-colors nodes in $T - 1$ rounds.

... $A_{\log^* n}$ : algorithm that $n$-colors nodes in $T - \frac{\log^* n}{2}$ rounds.

$\Rightarrow$ any 3-coloring algorithm needs at least $\frac{\log^* n}{2}$ rounds.
Independent Set at some Distance

- From 3–coloring to Maximal Independent Set.
Independent Set at some Distance

- From 3–coloring to Maximal Independent Set. Nodes at distance 2 or 3.
Independent Set at some Distance

- From 3-coloring to Maximal Independent Set.
  Nodes at distance 2 or 3.
- Reiterate on the Independent Node.
  Nodes at distance between 4 and 9.

Time complexity: $O(s \log^* n)$. 

Mikaël Rabie
Decidability of LCL Problems on Paths
Tuesday, April 2
Independent Set at some Distance

- From 3–coloring to Maximal Independent Set. Nodes at distance 2 or 3.
  
  ... 

- Reiterate $k$ times. Nodes at distance between $2^k$ and $3^k$. 

Time complexity: $O(s \log^* n)$. 

Mikaël RABIE
Independent Set at some Distance

- From 3-coloring to Maximal Independent Set.
  Nodes at distance 2 or 3.

- Reiterate on the Independent Node.
  Nodes at distance between 4 and 9.
  ... 

- Reiterate $k$ times.
  Nodes at distance between $2^k$ and $3^k$.

- Cut the segments in sizes between $s$ and $2s$ for some $s \leq 2^k$. 

Time complexity: $O(s \log^* n)$. 

Mikaël Rabie
Independent Set at some Distance

- From 3-coloring to Maximal Independent Set.
  Nodes at distance 2 or 3.

- Reiterate on the Independent Node.
  Nodes at distance between 4 and 9.
  ...

- Reiterate $k$ times.
  Nodes at distance between $2^k$ and $3^k$.

- Cut the segments in sizes between $s$ and $2s$ for some $s \leq 2^k$.

Time complexity: $O(s \log^* n)$. 
Transition Automata

Node : Sequence of outputs.
Edge : Connecting to an admissible next output.
Transition Automata

Node: Sequence of outputs.
Edge: Connecting to an admissible next output.
Transition Automata

Node: Sequence of outputs.
Edge: Connecting to an admissible next output.

![Diagram of Transition Automata]
Transition Automata

Node : Sequence of outputs.
Edge : Connecting to an admissible next output.
Transition Automata

Node: Sequence of outputs.
Edge: Connecting to an admissible next output.
Decidability on Paths without Inputs

Complexity Separation on Paths

Naor, Sotckmeyer (1995)

If the input graph is an unlabeled path or cycle, the time complexity is decidable.
The different time complexities are $O(1)$, $\Theta(\log^* n)$ and $\Omega(n)$. 
Naor, Sotckmeyer (1995)

If the input graph is an unlabeled path or cycle, the time complexity is decidable.
The different time complexities are $O(1)$, $\Theta(\log^* n)$ and $\Omega(n)$. 
Problem on Paths with Inputs

...... 1 28 37 52 8 32 46 47 73 5 3 ......
Problem on Paths with Inputs

1 28 37 52 8 32 46 47 73 5 3

......
3-color the red nodes.

Carry the color through the blue nodes.
3-color the red nodes.

Carry the color through the blue nodes.
For any LCL problem on cycle graphs, its complexity is either $\Omega(n)$ or $O(\log^* n)$. Moreover, there is an algorithm that decides whether the problem has complexity $\Omega(n)$ or $O(\log^* n)$ on cycle graphs; for the case the complexity is $O(\log^* n)$, the algorithm outputs a description of an $O(\log^* n)$-round deterministic LOCAL algorithm that solves it.

For any LCL problem on cycle graphs, its complexity is either $\Omega(\log^* n)$ or $O(1)$. Moreover, there is an algorithm that decides whether the problem has complexity $\Omega(\log^* n)$ or $O(1)$ on cycle graphs; for the case the complexity is $O(1)$, the algorithm outputs a description of an $O(1)$-round deterministic LOCAL algorithm that solves it.
It is PSPACE-hard to distinguish whether a given LCL problem with input labels can be solved in $O(1)$ time or needs $\Omega(n)$ time on globally oriented path graphs.
Decidability on Paths with Inputs

Turing Machine Encoding

Decision is PSPACE hard

Decidability of LCL Problems on Paths
Error Detection

\[\begin{array}{cccccccccccccccc}
\text{Input} & q_0 & q_0 & q_0 & q_0 & q_0 & q_1 & q_1 & q_1 & q_1 & q_1 & q_1 & q_1 & q_1 & q_1 & q_1 & q_1 \\
\text{Output} & a & T & F & F & F & F & T & F & F & F & F & F & F & T & F & F & F & F \\
\end{array}\]

Decision is PSPACE hard
It is PSPACE-hard to distinguish whether a given LCL problem without input labels can be solved in $O(1)$ time or needs $\Omega(n)$ time on trees with degree $\Delta = 3$. 

Balliu, Brandt, Chang, Olivetti, Rabie, Suomela (2018)
Encoding a Number in a Tree

Encoding $6 = 0110_2$
Encoding a Number in a Tree

Encoding $6 = 0110_2$
Encoding a Number in a Tree

Encoding $6 = 0110_2$
Encoding a Number in a Tree

Encoding $6 = 0110_2$

![Tree diagram showing the binary encoding of 6]

0

left

1

right

0

left

1

right

1

left

0

right
Encoding a Number in a Tree

Encoding $6=0110_2$
Encoding a Number in a Tree

Encoding $6=0110_2$
Encoding a Number in a Tree

Encoding $6=0110_2$
Encoding a Number in a Tree

Encoding $6=0110_2$
Encoding the Input of the Path

...... a1 a2 a3 a4 a5 a6 ......
Encoding the Input of the Path