New Classes of Distributed Time Complexity

Dennis Olivetti

Aalto University, Finland

Based on

- **New Classes of Distributed Time Complexity** - Alkida Balliu, Juho Hirvonen, Janne H. Korhonen, Tuomo Lempiäinen, Dennis Olivetti, and Jukka Suomela [STOC'18]
- **Almost Global Problems in the LOCAL Model** - Alkida Balliu, Sebastian Brandt, Dennis Olivetti, and Jukka Suomela [Submitted]

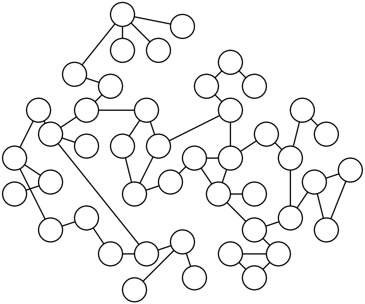
Slides based on "New Classes of Distributed Time Complexity", Janne H. Korhonen

Outline

- LOCAL Model
- Locally Checkable Labellings
- Results
- Proof idea

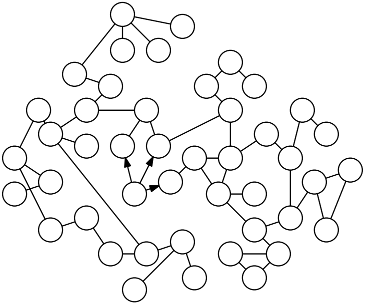
LOCAL Model

- Distributed
- Unlimited bandwidth
- Unlimited computational power
- Nodes have IDs
- In this talk:
  - deterministic algorithms



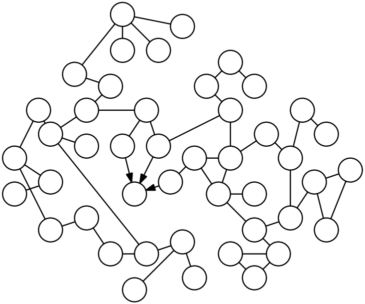
LOCAL Model

- Distributed
- Unlimited bandwidth
- Unlimited computational power
- Nodes have IDs
- In this talk:
  - deterministic algorithms



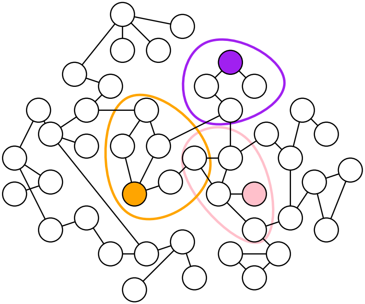
LOCAL Model

- Distributed
- Unlimited bandwidth
- Unlimited computational power
- Nodes have IDs
- In this talk:
  - deterministic algorithms



LOCAL Model

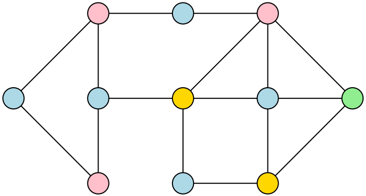
- Distributed
- Unlimited bandwidth
- Unlimited computational power
- Nodes have IDs
- In this talk:
  - deterministic algorithms



Locally Checkable Labellings

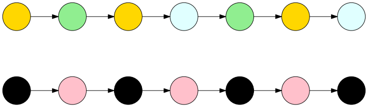
LCL Problems:

- Introduced by Naor and Stockmeyer in 1995
- Constant-size input labels
- Constant-size output labels
- The maximum degree is constant
- Validity of the output is locally checkable



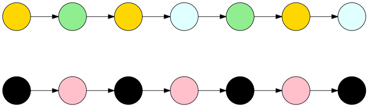
LCL on Cycles

- There are only three possible time complexities:
  - $\Theta(1)$: trivial problems
  - $\Theta(log^* n)$: local problems (symmetry breaking)
  - $\Theta(n)$: global problems



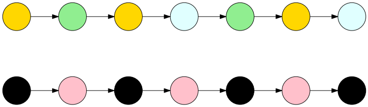
LCL on Cycles

- There are only three possible time complexities:
  - $\boldsymbol{\Theta(1)}$: trivial problems
  - $\boldsymbol{\Theta(log^* n)}$: local problems (symmetry breaking)
  - $\boldsymbol{\Theta(n)}$: global problems
- Automatic speedups:
  - Any $o(\log^* n)$-rounds algorithm can be converted to a $O(1)$-rounds algorithm [Naor and Stockmeyer, 1995]
  - Any $o(n)$-rounds algorithm can be converted to a $O(\log^* n)$-rounds algorithm [Chang, Kopelowitz and Pettie, 2016]



LCL on Cycles

- There are only three possible time complexities:
  - $\Theta(1)$: trivial problems
  - $\Theta(log^* n)$: local problems (symmetry breaking)
  - $\Theta(n)$: global problems
- Automatic speedups:
  - Any $o(\log^* n)$-rounds algorithm can be converted to a $O(1)$-rounds algorithm [Naor and Stockmeyer, 1995]
  - Any $o(n)$-rounds algorithm can be converted to a $O(\log^* n)$-rounds algorithm [Chang, Kopelowitz and Pettie, 2016]
- On cycles with no input, given an LCL description, we can *decide* its time complexity. [Naor and Stockmeyer, 1995] [Brandt et al, 2017]



LCL on Cycles

LCL on Trees

[Chang and Pettie, 2017]:

- Any $n^{o(1)}$-rounds algorithm can be converted to a $O(\log n)$-rounds algorithm
- There are problems of complexity $\Theta(n^{1/k})$

LCL on Trees

LCL on Trees (Our Results)

LCL on General Graphs

- There are problems with complexity $\Theta(\log n)$ [Brandt et al, 2016] [Chang, Kopelowitz and Pettie, 2016] [Ghaffari and Su, 2017]

LCL on General Graphs

- There are problems with complexity $\Theta(\log n)$ [Brandt et al, 2016] [Chang, Kopelowitz and Pettie, 2016] [Ghaffari and Su, 2017]
- Any $o(\log \log^* n)$-rounds algorithm can be converted to a $O(1)$-rounds algorithm using the same techniques of [Naor and Stockmeyer, 1995]

LCL on General Graphs

- There are problems with complexity $\Theta(\log n)$ [Brandt et al, 2016] [Chang, Kopelowitz and Pettie, 2016] [Ghaffari and Su, 2017]
- Any $o(\log \log^* n)$-rounds algorithm can be converted to a $O(1)$-rounds algorithm using the same techniques of [Naor and Stockmeyer, 1995]
- Any $o(\log n)$-rounds algorithm can be converted to a $O(\log^* n)$-rounds algorithm [Chang, Kopelowitz and Pettie, 2016]

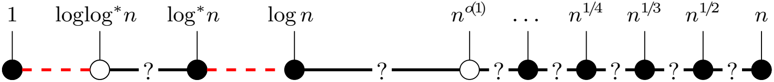
LCL on General Graphs

- There are problems with complexity $\Theta(\log n)$ [Brandt et al, 2016] [Chang, Kopelowitz and Pettie, 2016] [Ghaffari and Su, 2017]
- Any $o(\log \log^* n)$-rounds algorithm can be converted to a $O(1)$-rounds algorithm using the same techniques of [Naor and Stockmeyer, 1995]
- Any $o(\log n)$-rounds algorithm can be converted to a $O(\log^* n)$-rounds algorithm [Chang, Kopelowitz and Pettie, 2016]
- Many problems require $\Omega(\log n)$ and $O(\text{poly} \log n)$

LCL on General Graphs

- There are problems with complexity $\Theta(\log n)$ [Brandt et al, 2016] [Chang, Kopelowitz and Pettie, 2016] [Ghaffari and Su, 2017]
- Any $o(\log \log^* n)$-rounds algorithm can be converted to a $O(1)$-rounds algorithm using the same techniques of [Naor and Stockmeyer, 1995]
- Any $o(\log n)$-rounds algorithm can be converted to a $O(\log^* n)$-rounds algorithm [Chang, Kopelowitz and Pettie, 2016]
- Many problems require $\Omega(\log n)$ and $O(\text{poly} \log n)$
- Different scenario with randomized algorithms

LCL on General Graphs

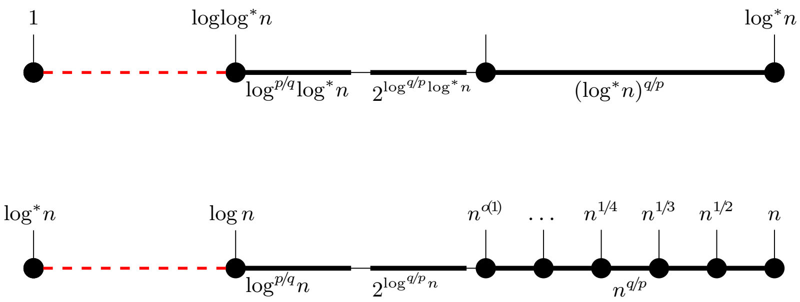


LCL on General Graphs?

LCL on General Graphs (Our Results)

LCL on General Graphs (Our Results)

LCL on General Graphs (Our Results)



General Idea

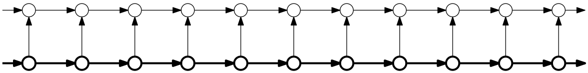
- We start from an LCL problem Π on cycles:
  - Π_l has complexity $T(n) = \Theta(\log^* n)$
    - 3 colouring
  - Π_g has complexity $T(n) = \Theta(n)$
    - a variant of 2 colouring
- We build a speed-up construction:
  - in ℓ rounds a node "sees" at distance $f(\ell) = \ell g(\ell)$
  - we obtain an easier version of Π
  - new complexity:
    - $\Theta(f^{-1}(T(n)))$

Example

- We start from a cycle

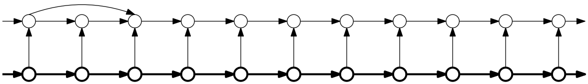
Example

- We add shortcuts on top of the cycle, $g(\ell) = 2^\ell$



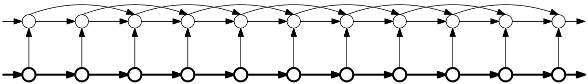
Example

- We add shortcuts on top of the cycle, $g(\ell) = 2^{\ell}$



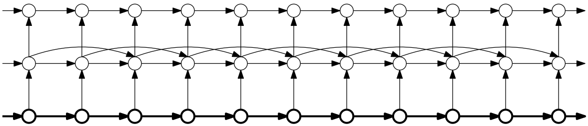
Example

- We add shortcuts on top of the cycle, $g(\ell) = 2^{\ell}$



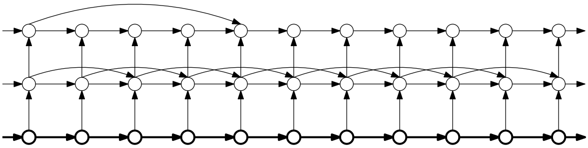
Example

- We add shortcuts on top of the cycle, $g(\ell) = 2^\ell$



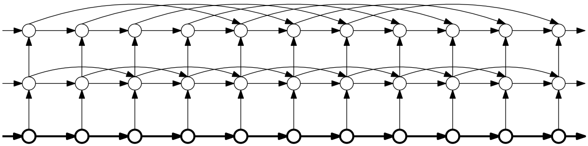
Example

- We add shortcuts on top of the cycle, $g(\ell) = 2^\ell$



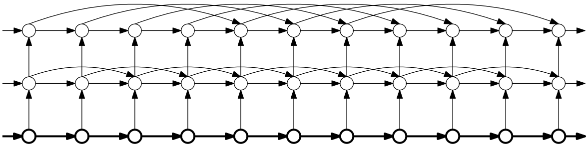
Example

- We add shortcuts on top of the cycle, $g(\ell) = 2^{\ell}$



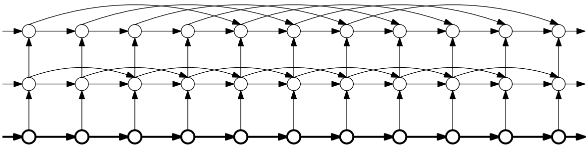
Example

- Π can be solved in $\Theta(f^{-1}(T(n)))$ rounds using the shortcuts



Example

- Problem: this is not a valid LCL



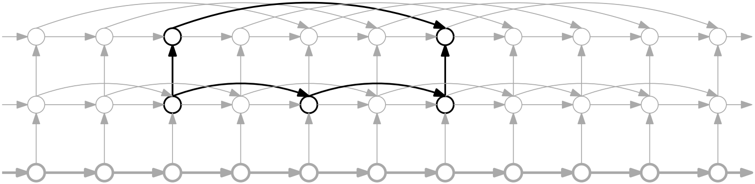
Valid LCL

- An LCL problem must be defined on any graph, not just on some "relevant" instances
- What if the shortcuts are missing?
- What if a cycle is not present at all?

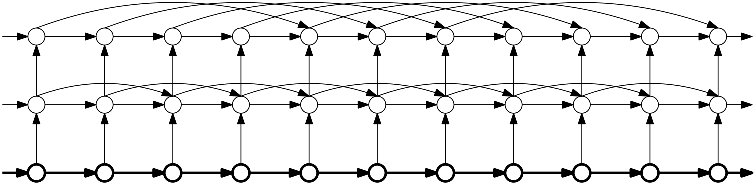
Fixing the details

- Input:
  - a graph
  - a proof that the graph is a relevant instance
    - it must be locally checkable
- Output:
  - Solve Π, or
  - Prove that there is an error in the input proof, or in the graph structure
    - it must be locally checkable

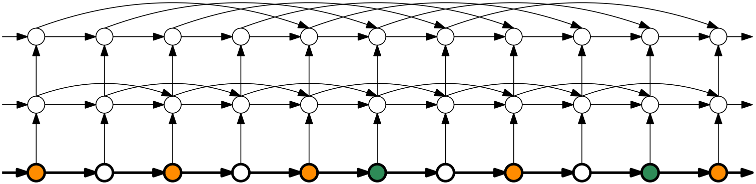
Local checkability of the input



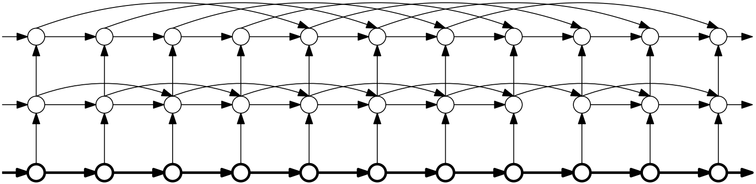
Correct instance



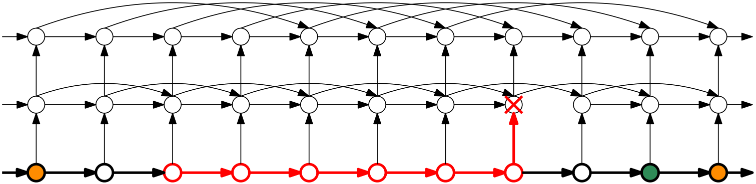
Correct instance



Incorrect instance

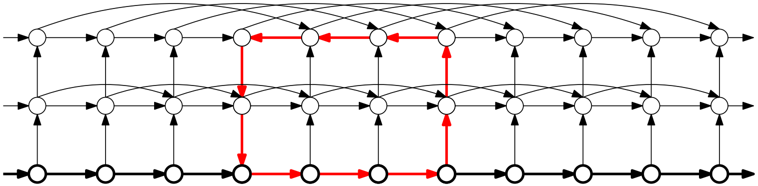


Incorrect instance

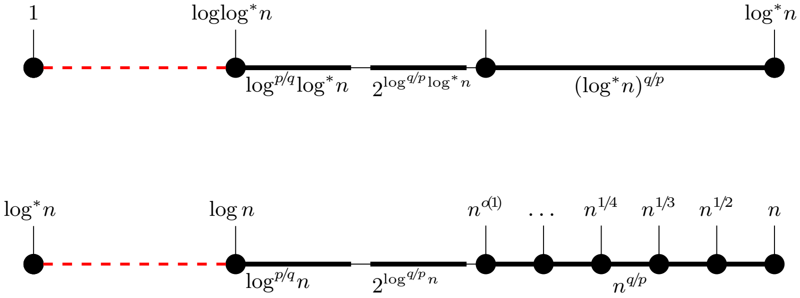


Hardness balance

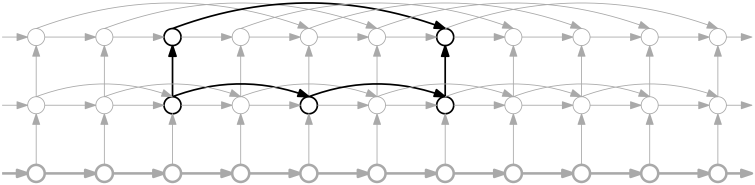
- On incorrect instances, it should be easy to prove that there is an error
- On correct instances, it should be impossible, or hard, to prove that there is an error



Using different $g(\ell)$



Using different $g(\ell)$



Which shortcut constructions can be locally checked?

Link Machine Programs

- Constant number of registers
- Reset
  - $r_1 \leftarrow 1$
- Addition
  - $r_1 \leftarrow r_2 + r_3$
- If statements with equality comparison
  - if $r_1 = r_2$
  - if $r_1 \neq r_2$
- $g(\ell)$ = value of the maximum register after ℓ executions of the program

Link Machine Programs: Examples

- $g(\ell) = 2^\ell$
  - $r_1 \leftarrow r_1 + r_1$

Link Machine Programs: Examples

- $g(\ell) = 2^\ell$
  - $r_1 \leftarrow r_1 + r_1$
- $g(\ell) = \Theta(\ell^2)$
  - $r_1 \leftarrow r_1 + 1$
  - $r_2 \leftarrow r_2 + r_1$

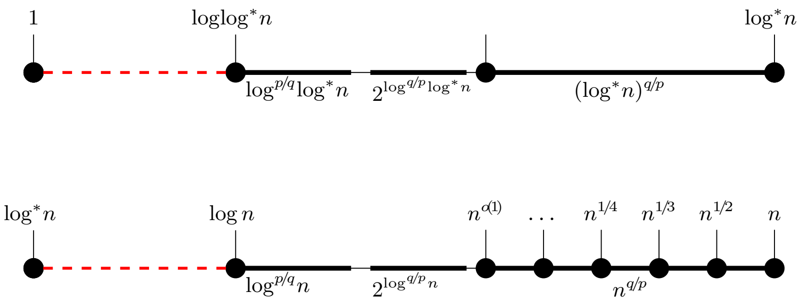
Link Machine Programs: Building Blocks

Program P	Input	Output	Growth
COUNT	–	$y = \ell$	ℓ
ROOT'_k	–	$y = \Theta(\ell^{1/k})$	$\Theta(\ell^{1/k})$
ROOT_k	x	$y = \Theta(x^{1/k})$	$\Theta(x)$
POW_k	x	$y = \Theta(x^k)$	$\Theta(x^k)$
EXP	x	$y = 2^{\Theta(x)}$	$2^{\Theta(x)}$
LOG	x	$y = \Theta(\log x)$	$\Theta(x)$

Link Machine Programs: $g(\ell)$

Program P		Growth
$\mathrm{POW}_p \circ \mathrm{ROOT}'_q$		$\Theta(\ell^{p/q})$
$\mathrm{EXP} \circ \mathrm{POW}_q \circ \mathrm{ROOT}'_p$	$(p \geq q)$	$2^{\Theta(\ell^{q/p})}$
$\mathrm{EXP} \circ \mathrm{POW}_q \circ \mathrm{ROOT}_p \circ \mathrm{LOG} \circ \mathrm{COUNT}$	$(p \geq q)$	$2^{\Theta(\log^{q/p} \ell)}$

Results



1
$\log\log^* n$
$\log^* n$
$\log^{p/q}\log^* n$
$2^{\log^{q/p}\log^* n}$
$(\log^* n)^{q/p}$
$\log^* n$
$\log n$
$n^{o(1)}$
$\ldots$
$n^{1/4}$
$n^{1/3}$
$n^{1/2}$
n
$\log^{p/q} n$
$2^{\log^{q/p} n}$
$n^{q/p}$

Conclusions and Open Problems

- Are there other gaps on trees?
- What happens between $\Omega(\log \log^* n)$ and $O(\log^* n)$ on trees?
- What about polynomial complexities with sub-diameter time/sub-linear volume?
- What are meaningful subclasses of LCL problems where there are gaps again?

Thank you!