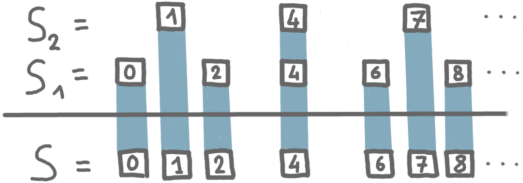
Complexity landscape for local certification

Nicolas Bousquet
Laurent Feuilloley
Sébastien Zeitoun
CNRS · University of Lyon

Low CO2 Research Paper

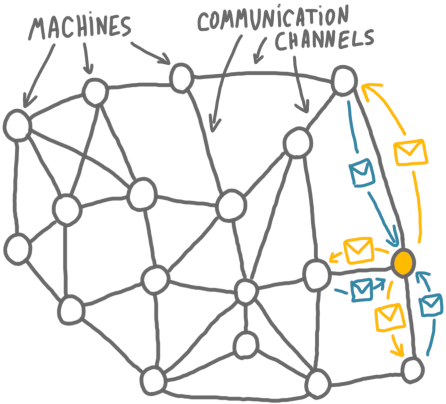


DISC 2025 · Berlin

Time complexity in the LOCAL model

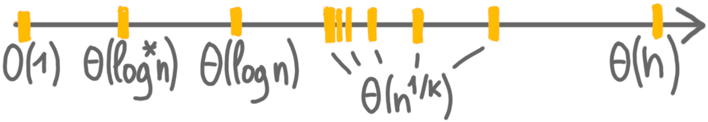
In the LOCAL model:

- The nodes send and receive messages in synchronous rounds.
- The (time) complexity is the number of rounds before termination.



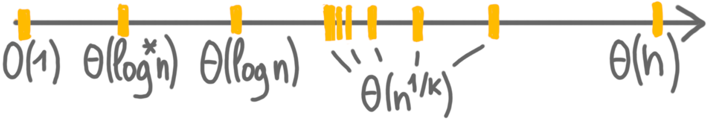
Complexity landscapes in the LOCAL model

Theorem: In the LOCAL model, in bounded-degree trees, the complexity of solving a "local problem" (LCL) can only be: $O(1)$, $\Theta(\log^* n)$, $\Theta(\log n)$, $\Theta(n^{1/k})$, or $\Theta(n)$.



Complexity landscapes in the LOCAL model

Theorem: In the LOCAL model, in bounded-degree trees, the complexity of solving a "local problem" (LCL) can only be: $O(1)$, $\Theta(\log^* n)$, $\Theta(\log n)$, $\Theta(n^{1/k})$, or $\Theta(n)$.



USUAL LANDSCAPE:

Can we get a landscape theorem for space?

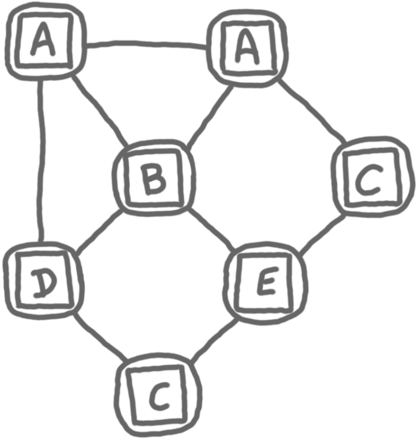
Can we get a landscape theorem for space?

A model: the (simplified) state model

- Every node has a "state".
- At every round: a node reads its states and the states of its neighbors, and update its state.

Space complexity: The maximum size of a state.

A theorem*: In this model, the space complexity of computing a minimum spanning tree is $\Theta(\log^2 n)$.



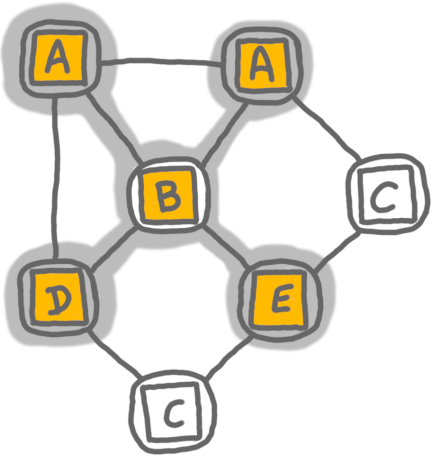
Can we get a landscape theorem for space?

A model: the (simplified) state model

- Every node has a “state”.
- At every round: a node reads its states and the states of its neighbors, and update its state.

Space complexity: The maximum size of a state.

A theorem*: In this model, the space complexity of computing a minimum spanning tree is $\Theta(\log^2 n)$.



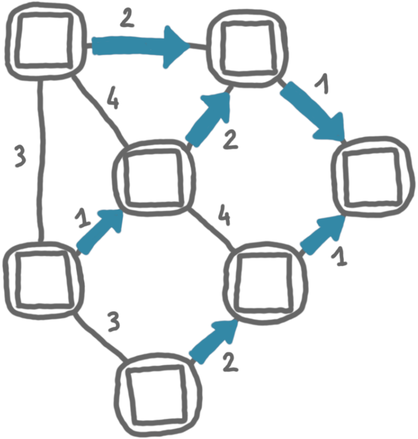
Can we get a landscape theorem for space?

A model: the (simplified) state model

- Every node has a "state".
- At every round: a node reads its states and the states of its neighbors, and update its state.

Space complexity: The maximum size of a state.

A theorem*: In this model, the space complexity of computing a minimum spanning tree is $\Theta(\log^2 n)$.



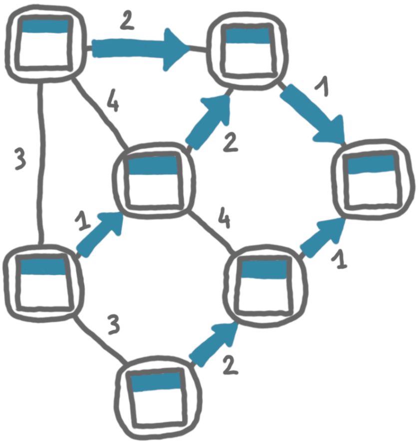
Can we get a landscape theorem for space?

A model: the (simplified) state model

- Every node has a "state".
- At every round: a node reads its states and the states of its neighbors, and update its state.

Space complexity: The maximum size of a state.

A theorem*: In this model, the space complexity of computing a minimum spanning tree is $\Theta(\log^2 n)$.



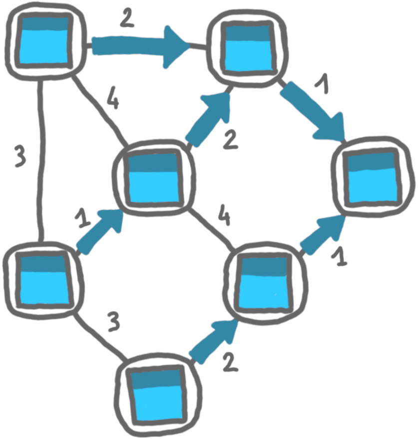
Can we get a landscape theorem for space?

A model: the (simplified) state model

- Every node has a "state".
- At every round: a node reads its states and the states of its neighbors, and update its state.

Space complexity: The maximum size of a state.

A theorem*: In this model, the space complexity of computing a minimum spanning tree is $\Theta(\log^2 n)$.



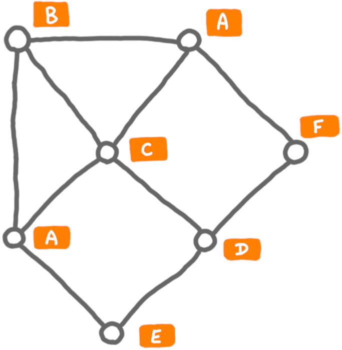
Local certification

Theorem*: The space complexity in the state model is captured by the complexity of local certification.

Focus: Checking that the network satisfies a property $\mathcal{P}$.

Local certification protocol:

- A prover assigns labels, called "certificates", to the nodes.
- Each node can see its certificates and the ones of its neighbors.
- Each node outputs *accept* or *reject*.



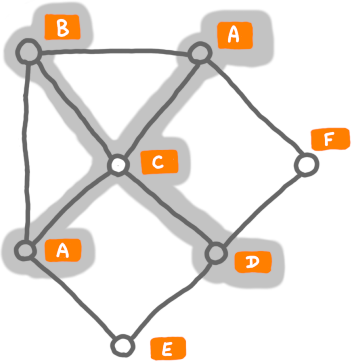
Local certification

Theorem*: The space complexity in the state model is captured by the complexity of local certification.

Focus: Checking that the network satisfies a property $\mathcal{P}$.

Local certification protocol:

- A prover assigns labels, called "certificates", to the nodes.
- Each node can see its certificates and the ones of its neighbors.
- Each node outputs *accept* or *reject*.



Local certification

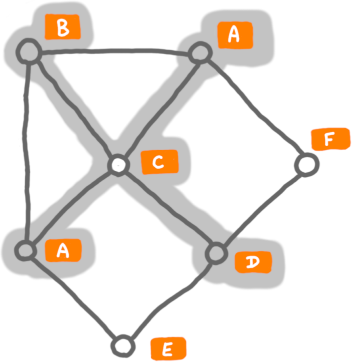
Theorem*: The space complexity in the state model is captured by the complexity of local certification.

Focus: Checking that the network satisfies a property $\mathcal{P}$.

Correctness: We have a local certification for a property $\mathcal{P}$ if for all instances:

- If $\mathcal{P}$ is satisfied, there exists a certificate assignment that makes all nodes accept.
- If $\mathcal{P}$ is *not* satisfied, for all certificates assignment, at least one node rejects.

Complexity: Number of bits per certificate.



Looking for gaps in local certification

Setting: Anonymous unlabeled directed paths.

Property to check: The path length (number of nodes) belongs to some set $S \subseteq \mathbb{N}$.

Observation: When S is the set of even numbers, the optimal complexity is $\Theta(1)$.

Observation: Complexity $O(\log n)$ is sufficient for any S.

Question: Is there a gap between $O(1)$ and $\Theta(\log n)$?

Looking for gaps in local certification

Setting: Anonymous unlabeled directed paths.

Property to check: The path length (number of nodes) belongs to some set $S \subseteq \mathbb{N}$.

Observation: When S is the set of even numbers, the optimal complexity is $\Theta(1)$.

Observation: Complexity $O(\log n)$ is sufficient for any S.

Question: Is there a gap between $O(1)$ and $\Theta(\log n)$?

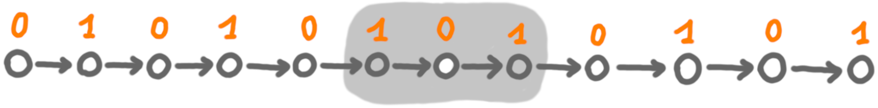
Looking for gaps in local certification

Setting: Anonymous unlabeled directed paths.

Property to check: The path length (number of nodes) belongs to some set $S \subseteq \mathbb{N}$.

Observation: When S is the set of even numbers, the optimal complexity is $\Theta(1)$.

Observation: Complexity $O(\log n)$ is sufficient for any S.



Question: Is there a gap between $O(1)$ and $\Theta(\log n)$?

Looking for gaps in local certification

Setting: Anonymous unlabeled directed paths.

Property to check: The path length (number of nodes) belongs to some set $S \subseteq \mathbb{N}$.

Observation: When S is the set of even numbers, the optimal complexity is $\Theta(1)$.

Observation: Complexity $O(\log n)$ is sufficient for any S.

Question: Is there a gap between $O(1)$ and $\Theta(\log n)$?

Looking for gaps in local certification

Setting: Anonymous unlabeled directed paths.

Property to check: The path length (number of nodes) belongs to some set $S \subseteq \mathbb{N}$.

Observation: When S is the set of even numbers, the optimal complexity is $\Theta(1)$.

Observation: Complexity $O(\log n)$ is sufficient for any S.

Question: Is there a gap between $O(1)$ and $\Theta(\log n)$?

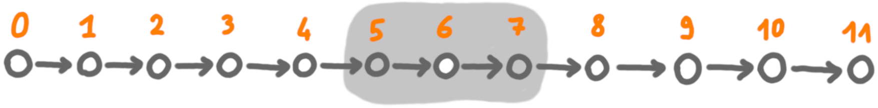
Looking for gaps in local certification

Setting: Anonymous unlabeled directed paths.

Property to check: The path length (number of nodes) belongs to some set $S \subseteq \mathbb{N}$.

Observation: When S is the set of even numbers, the optimal complexity is $\Theta(1)$.

Observation: Complexity $O(\log n)$ is sufficient for any S.



Question: Is there a gap between $O(1)$ and $\Theta(\log n)$?

Looking for gaps in local certification

Setting: Anonymous unlabeled directed paths.

Property to check: The path length (number of nodes) belongs to some set $S \subseteq \mathbb{N}$.

Observation: When S is the set of even numbers, the optimal complexity is $\Theta(1)$.

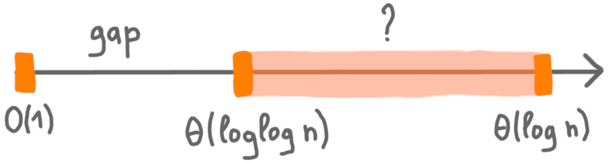
Observation: Complexity $O(\log n)$ is sufficient for any S.

Question: Is there a gap between $O(1)$ and $\Theta(\log n)$?

Our results

Theorem 1: There is a gap in the complexity between $O(1)$ and $\Theta(\log \log n)$.

Theorem 2: There exists a property of complexity $\Theta(\log \log n)$.



Theorem 3: *In cycles*, there is a gap between $O(1)$ and $\Theta(\log n)$.

More results: Generalization from paths to labeled trees, no gap in more general settings, the role of identifiers, etc.

Rest of this talk: Proof sketch for Theorem 1.

Warm-up for the gap proof

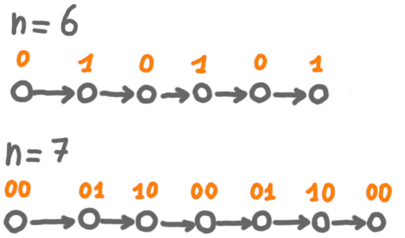
Set-up:

- Directed, anonymous, unlabeled paths.
- The nodes do not know the number of nodes n.
- The prover must use the same number of bits for all nodes.

Example: $S = \{x : x \equiv 0\ [2] \text{ or } x \equiv 1\ [3]\}$

Prover (on correct instances)

- If $n \equiv 0\ [2]$: counter mod 2, on 1 bit.
- If $n \equiv 1\ [3]$: counter mod 3, on 2 bits.



Warm-up for the gap proof

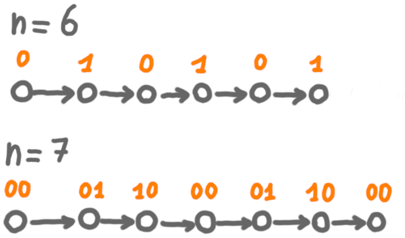
Set-up:

- Directed, anonymous, unlabeled paths.
- The nodes do not know the number of nodes n.
- The prover must use the same number of bits for all nodes.

Example: $S = \{x : x \equiv 0\ [2] \text{ or } x \equiv 1\ [3]\}$

Node verification:

- If 1 bit:
  consistent mod 2, first $= 0$, last $= 1$.
- If 2 bits:
  consistent mod 3, first $= 00$, last $= 00$.
- Otherwise reject.



Slices and automata

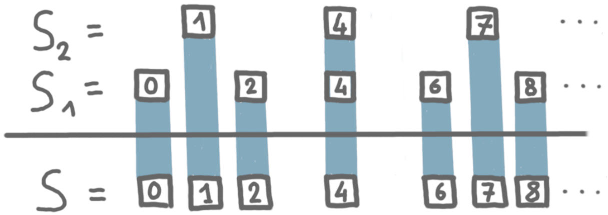
Notation: $S_i = \{$lengths that can be accepted with certificates of size $i.\}$

$$S_2 = \boxed{1} \quad \boxed{4} \quad \boxed{7} \quad \cdots$$

$$S_1 = \boxed{0} \quad \boxed{2} \quad \boxed{4} \quad \boxed{6} \quad \boxed{8} \quad \cdots$$

Slices and automata

Notation: $S_i = \{\text{lengths that can be accepted with certificates of size } i.\}$

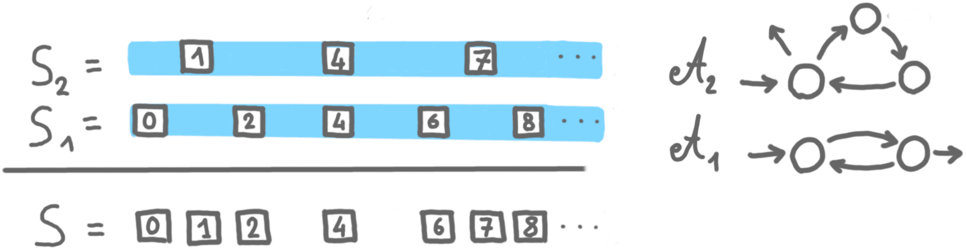


Lemma: For a correct protocol: $S = \cup_i S_i$.

Lemma: Each S_i is recognized by a finite automaton $\mathcal{A}_i$ on $\leq 2^i$ states.
(When we see a set of integers as a language of unary words.)

Slices and automata

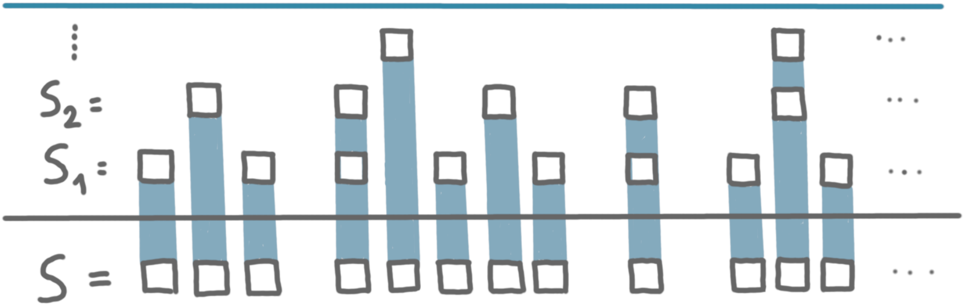
Notation: $S_i = \{$lengths that can be accepted with certificates of size $i.\}$



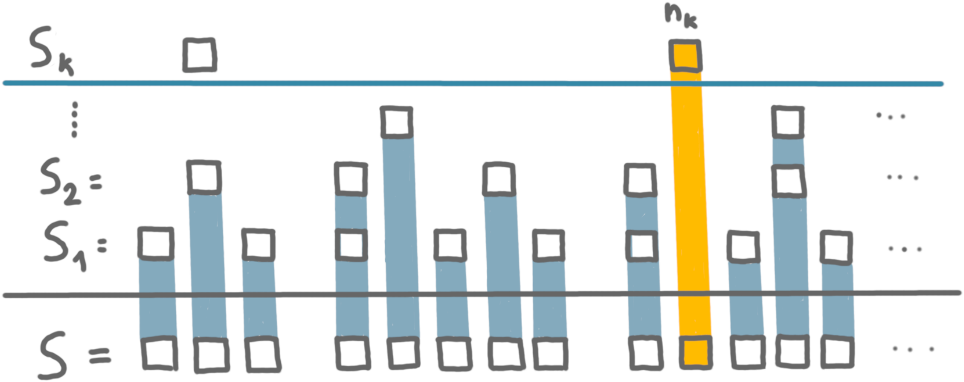
Lemma: For a correct protocol: $S = \cup_i S_i$.

Lemma: Each S_i is recognized by a finite automaton $\mathcal{A}_i$ on $\leq 2^i$ states. (When we see a set of integers as a language of unary words.)

Constant vs. non-constant complexity



Constant vs. non-constant complexity



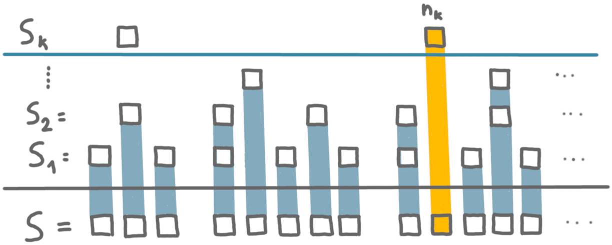
Reformulating the theorem

Notation: For every k, let n_k be the first position for which S_k is needed:

$$n_k = \min\{n : n \in S_k \text{ and } n \notin S_i, i < k\}$$

Theorem: There is a gap in the complexity between $O(1)$ and $\Theta(\log\log n)$.

Reformulation: Non-constant complexity $\Rightarrow$ for infinitely many k, $n_k \leq 2^{2^k}$.



Proof via state complexity

Reformulation: Non-constant complexity $\Rightarrow$ for infinitely many k, $n_k \leq 2^{2^k}$.

$$\begin{aligned} n_k &= \min\{n : n \in S_k \text{ and } n \notin S_i, i < k\} \\ &= \min\{|w| : w \in \mathcal{L}(\mathcal{A}^*)\}, \text{ where } \mathcal{L}(\mathcal{A}^*) = \mathcal{L}(\mathcal{A}_k) \cap \overline{\cup_{i<k}\mathcal{L}(\mathcal{A}_i)} \end{aligned}$$

Lemma: $\mathcal{A}^*$ has at most 2^{2^k} states.

Proof:

- For all i, $\mathcal{A}_i$ has $\leq 2^i$ states.
- Intersection, complement, union operations on finite automata induce an at most exponential blow-up.

Conclusion: n_k is the smallest word accepted by $\mathcal{A}^* \Rightarrow n_k \leq 2^{2^k} \Rightarrow$ Theorem 1.

The two other key results

Theorem 2: There exists a property of complexity $\Theta(\log \log n)$.

Property $\mathcal{P}$ we use: The length is *not* the product of consecutive primes.

$2, 3, 4, 5, 6 = 2 \times 3, 7, 8, 9, \ldots, 29, 30 = 2 \times 3 \times 5, 31, \ldots$

Proof: Play with modular arithmetic for ~~some~~ a long time.

Theorem 3: *In cycles*, there is a gap between $O(1)$ and $\Theta(\log n)$.

Key difference: In cycles, we can only play with a combination of equation of the form $0 \mod m$, which give less expressive power.

Perspectives

Other results:

- The gap generalizes to labeled paths and trees (with diameter instead of n).
- The result are robust to some weak identifier models and vague knowledge of n.
- But there is no gap in the following settings:
  - Above $\log n$ in general identified graphs
  - Below $\log n$ if we allow verification radius 2.
  - With identifiers in $[1, n]$.

Open questions:

- Fill the... gaps in the results.
- Prove gaps in general graphs with identifiers, but only for "natural properties".
- Transfer to real algorithms in the state model.