Complexity Landscape for Local Certification

Nicolas Bousquet
CNRS, INSA Lyon, UCBL, LIRIS, UMR5205, F-69622 Villeurbanne, France

Laurent Feuilloley
CNRS, INSA Lyon, UCBL, LIRIS, UMR5205, F-69622 Villeurbanne, France

Sébastien Zeitoun
CNRS, INSA Lyon, UCBL, LIRIS, UMR5205, F-69622 Villeurbanne, France

Abstract

An impressive recent line of work has charted the complexity landscape of distributed graph algorithms. For many settings, it has been determined which time complexities exist, and which do not (in the sense that no local problem could have an optimal algorithm with that complexity). In this paper, we initiate the study of the landscape for *space complexity* of distributed graph algorithms. More precisely, we focus on the local certification setting, where a prover assigns certificates to nodes to certify a property, and where the space complexity is measured by the size of the certificates.

Already for anonymous paths and cycles, we unveil a surprising landscape:

- There is a gap between complexity $O(1)$ and $\Theta(\log\log n)$ in paths. This is the first gap established in local certification.
- There exists a property that has complexity $\Theta(\log\log n)$ in paths, a regime that was not known to exist for a natural property.
- There is a gap between complexity $O(1)$ and $\Theta(\log n)$ in cycles, hence a gap that is exponentially larger than for paths.

We then generalize our result for paths to the class of trees. Namely, we show that there is a gap between complexity $O(1)$ and $\Theta(\log\log d)$ in trees, where d is the diameter. We finally describe some settings where there are no gaps at all.

To prove our results we develop a new toolkit, based on various results of automata theory and arithmetic, which is of independent interest.

2012 ACM Subject Classification Theory of computation → Distributed algorithms

Keywords and phrases Local certification, proof-labeling schemes, locally checkable proofs, space complexity, distributed graph algorithms, complexity gap

Digital Object Identifier 10.4230/LIPIcs.DISC.2025.18

Related Version *Full Version*: `https://arxiv.org/abs/2505.20915` [14]

Funding This work is supported by the ANR grant ENEDISC (ANR-24-CE48-7768).

Acknowledgements The authors would like to thank Thomas Colcombet for answering our questions about Chrobak normal form, and the reviewers for useful comments.

1 Introduction

1.1 Approach

Time complexity landscapes for distributed graph algorithms

For distributed graph algorithms, the most classic measure of complexity is *time*, measured by the number of rounds before output. One of the most fruitful research programs on this topic has been the one of charting the complexity landscape [42]. That is, instead of considering specific problems and asking for their time complexity, a long series of papers

have answered the following question: given a complexity function, is there a problem with that complexity? In this paper, we aim at starting the analogue research program for the *space complexity* of distributed graph algorithms.

Let us mention some elements about the time complexity landscape, in order to draw analogies with our setting later. For the sake of simplicity, we only discuss deterministic complexity of locally checkable labelings (LCLs) in bounded degree graphs in the LOCAL model. Before 2016, there were a few of classic complexities: $O(1)$ for very simple tasks, $\Theta(\log^* n)$ for tasks that could be reduced to coloring, and $O(n)$ for global problems (throughout the paper, n refers to the number of nodes in the graph). Then, several seminal papers established that some problems have complexity $\Theta(\log n)$ [15, 21], and that there is no problem whose complexity strictly lies between $\Theta(\log^* n)$ and $\Theta(\log n)$ [21]. This gap sharply contrasts with classic computational theory where the time hierarchy theorem prevents the existence of such gaps [38]. It was also proved that in some intervals infinitely many complexities exist [6, 7, 19, 22], but not *any* complexity function had a corresponding problem. A key strategy in this research area is to understand the landscape for specific classes of networks such as paths [3], grids [16], trees [4, 5, 6], minor-closed graphs [20], etc. In this paper, we will follow a similar approach: characterizing which space complexities are possible depending on the structure of the network.

New direction: A landscape approach to space

The *space* used by distributed algorithms is much less studied than the time, especially if we restrict to distributed graph algorithms. Some models where space is a well-studied measure are population protocols [2], massively parallel computing (MPC) [39] and models similar to Stone Age [26], but they are not relevant to our approach, either because they differ too much from distributed graph computing (having one-to-one communication or a centralized component) or because they fix the space complexity to constant. As far as we know, the only major line of research considering space complexity for distributed graph algorithms is self-stabilization, and more specifically the state model, where the nodes update their states by reading the states of their neighbors [1, 25]. In that model, the algorithm has to cope with transient faults, and consequently it is common to design algorithms (at least implicitly) with two components : one that builds a solution and one that checks the solution, and can reset the system if needed. *Local certification* was introduced to study specifically the space needed for the checking phase, by abstracting the computation of the solution into an oracle, and this is our focus today.

Informally, a local certification of a property is an assignment of labels to the nodes of the graph, such that the nodes can collectively check the correctness of a given property by inspecting the labels in their neighborhoods. This notion is known under different names, depending on the specific model considered: *proof-labeling scheme* [40], locally checkable proofs [36], non-deterministic local decision [33], etc. We will formally define our model in Section 2, and refer to [27] for an introduction to the notion. The (space) complexity of a certification is the maximum size of the label given to a node, and it is known to be basically equal to the space complexity of self-stabilizing algorithms.[1] Finally, since local certification is about decision problems, it is now standard to focus on checking graph properties (instead of solutions to graph problems), and to refer to the set of correct graphs as a *language*.

[1] There are some fine prints to this statement, that we will discuss later, in Section 1.4.

In local certification, the situation is similar to the pre-2016 situation for the time complexity as described above. That is, there are a few well-established complexity regimes – $O(1)$, $\Theta(\log n)$, $\Theta(n)$ and $\Theta(n^2)$ – but no solid explanation as of why these are so common. But unlike for distributed time complexity (for LCLs on trees), we can prove that for a wide range of complexities, there exists a problem with that complexity.

▶ **Theorem 1.** *For general graphs with identifiers, for any non-decreasing function $f(n)$ in $\Omega(\log n)$ and $O(n^2)$, there exists a property that can be certified with $O(f(n))$ bits, but not in $o(f(n))$ bits.*

The proof (deferred to Appendix A) follows a standard construction of [36]. The graphs satisfying the property are the ones made of two copies of a graph H, linked by a path. The graph H is chosen so that it can be encoded on $f(n)$ bits, and a certification consists in giving this encoding to all nodes, so that every node knows H. It is known that this encoding can be checked locally when identifiers are given, hence we get the $O(f(n))$ upper bound. A counting argument allows to prove a matching lower bound [36].

At first sight, Theorem 1 gives a trivial answer to our question about the complexity landscape of local certification. But it relies crucially on several assumptions. First, in the upper bound it is necessary to have unique identifiers to avoid being fooled by symmetries at the checking phase. Also, because of the identifiers and because the value of n needs to be certified, $\Omega(\log n)$ bits are needed. Finally, to make the counting argument of the lower bound work, one cannot restrict too much the graphs in which to apply this theorem. This rises the three following questions that we tackle in this paper.

▶ **Question 2.** *What happens for complexities in $o(\log n)$?*

In other words, is the $\Omega(\log n)$ a limitation of the proof technique, or could there be a gap between constant and $\Theta(\log n)$, as the current state of our knowledge suggests?

▶ **Question 3.** *What about anonymous networks?*

This is especially relevant in conjunction with the previous question, since a unique identifier cannot be encoded in a certificate of $o(\log n)$ bits. Note that even beyond that regime, the impact of the identifiers on local decision is a well-studied topic [31, 30, 32, 29].

▶ **Question 4.** *What about structured graphs, like paths, cycles and trees?*

In the spirit of the work made on the distributed time complexity landscape, we would like to understand local certification on restricted graph classes, in which counting arguments do not apply, or only partially.

1.2 Main results

We start with a simple setting: anonymous paths without input labels. In this setting, a path is characterized by its length, and a language is defined by a set of authorized lengths. Naturally, we assume that the nodes *do not* have the knowledge of n (otherwise the problem is trivial). Given $O(\log n)$ bits, we can recognize any language. This is because the prover can certify the exact length of the path, by giving to every node its distance to a chosen endpoint as a certificate. The nodes can then check locally the correctness of this counter, and the last node can check whether the length belongs to the authorized sizes. On the other hand, with a constant number of bits, we can encode properties like "the path has length $k \mod m$", via counters modulo m (where m is a constant). To build a language whose

complexity would be strictly between these regimes, it is tempting to consider properties of the form: "the path has length k modulo $f(n)$" for arbitrary function f, and for which a counter would use $\log f(n)$ bits. Unfortunately, the nodes are unable to check that the $f(n)$ is correct, since they do not have access to n. This obstacle does not seem easy to bypass and, it is reasonable to conjecture that there is a gap between $O(1)$ and $\Theta(\log n)$. We do establish the existence of a gap, but it only goes up to $\Theta(\log \log n)$.

▶ **Theorem 5.** *Let $c > 1$ and $N \in \mathbb{N}$. Let $\mathcal{P}$ be a property on paths that can be certified with certificates of size $s(n) := \left\lfloor \frac{\log \log n}{c} \right\rfloor$ for all $n \geqslant N$. Then, $\mathcal{P}$ can also be certified with constant-size certificates.*

This is *the first gap established in local certification.* At first, this $\log \log n$ looks like an artifact of the proof but, surprisingly, it is not! We next prove that there exists a language for which the optimal certificate size is $\Theta(\log \log n)$. (Given the previous theorem, it is sufficient to prove that this language can be certified with $O(\log \log n)$ bits but not with $O(1)$ bits.)

▶ **Theorem 6.** *There exist properties on paths that can be certified with certificates of size $O(\log \log n)$, but not with certificates of size $O(1)$.*

It is *the first time that this regime appears in the area of local certification* (here under the promise that the graph is a path). The language we use is the set of paths whose length is not a product of consecutive primes; we will come back to it.

Now, moving on to cycles, there is a second surprise. We expect to see the same landscape in paths and cycles, but the intermediate regime actually disappears in cycles, and there is a gap between $O(1)$ and $\Theta(\log n)$.

▶ **Theorem 7.** *Let $c > 12$ and $N \in \mathbb{N}$. Let $\mathcal{P}$ be a property on cycles that can be certified with certificates of size $s(n) := \left\lfloor \frac{\log n}{c} \right\rfloor$ for every integer $n \geqslant N$. Then, $\mathcal{P}$ can also be certified with constant-size certificates.*

Finally, we study the case of trees, where the picture is more complex, as it depends on the exact setting considered. In some settings, we can prove that there is no gap (any reasonable function corresponds to the optimal certificate size for a language), and in some other cases, the gap from paths persists. These settings depend on three parameters: (1) whether we consider certificate size as a function of the number of nodes n or of the diameter d, (2) whether the nodes can only see what is at distance 1 or at larger distance (in other words the verification radius is 1 or larger), and (3) whether the maximum degree is bounded or not. We establish a classification of these different settings. In particular, we prove that *the gap for paths is generalized to arbitrary trees*, when parameterized by the diameter, and the verification radius is 1.

▶ **Theorem 8.** *Let $c > 2$ and $D \in \mathbb{N}$. Let $\mathcal{P}$ be a property in trees (of unbounded degree) that can be certified using $s(d) := \left\lfloor \frac{\log \log d}{c} \right\rfloor$ bits for all $d \geqslant D$ (where d is the diameter). Then, $\mathcal{P}$ can also be certified with constant-size certificates.*

We then show that two assumptions in Theorem 8 are optimal. Namely, we prove in Theorem 9 that we can not hope for a gap in n instead of d in trees (even in caterpillars, that is, paths with leafs attached), and we establish in Theorem 10 that it is essential that the vertices are able to see only at distance only 1, because there is no more gap if the verification radius r is at least 2 (again, even in caterpillars).

▶ **Theorem 9.** *Let $f : \mathbb{N} \to \mathbb{R}$ be a non-decreasing function such that $\lim_{n\to+\infty} f(n) = +\infty$ and for all integers $1 \leqslant s \leqslant t$ we have $f(t) - f(s) \leqslant \log t - \log s$. Then, there exists a property on caterpillars (of unbounded degree) that can be certified with certificates of size $O(f(n))$ and not with certificates of size $o(f(n))$.*

▶ **Theorem 10.** *Let $r > 1$. Let $f : \mathbb{N} \to \mathbb{N}$ be a non-decreasing function such that $\lim_{n\to+\infty} f(n) = +\infty$ and for all integers $1 \leqslant s \leqslant t$ we have $f(t) - f(s) \leqslant \log t - \log s$. Then, if the vertices can see at distance r, there exists a property on caterpillars (of unbounded degree) that can be certified with certificates of size $O(f(d))$ and not with certificates of size $o(f(d))$, where d is the diameter.*

Note that the condition on $f(t) - f(s)$ only ask for a function which is sublogarithmic and whose growth is sublogarithmic, to avoid arbitrarily long plateaus followed by big jumps. This condition is satisfied by all the usual functions ($\sqrt{\log n}$, $\log\log n$, $\log^* n$, etc.).

All the results mentioned so far are in the anonymous setting. We also explore the impact of identifiers and of the knowledge of n on this landscape, but we delay this discussion for now.

1.3 Main techniques

Automata and arithmetic perspectives

To give an overview of our techniques, let us start by describing two perspectives on constant size certification in anonymous paths. As said earlier, a basic building block is to certify with a counter that the length is equal to $i \mod k$ for some constant i and k. One can see this behavior as a run in an automaton with k states inducing a cycle whose initial state being $0 \mod k$ and the final state being $i \mod k$. More generally, any constant size certification can be turned into a finite state automaton. There are various ways to do it, but basically, it consists in having states describing (pairs of) certificates, and transitions that connect states that would be accepted by the local verifier. The existence of an accepting run in the automaton is equivalent to the existence of a certificate assignment making the local verifier accept (see Section 3.1 for a more detailed explanation). (Also see [28], and also [23] for a similar perspective in the context of local problems.) Another point of view is the one of arithmetic. Intuitively, a set of lengths of anonymous paths that are accepted by a constant size certification corresponds to a combination of congruence relations (in other words, such a set is equal to the union of arithmetic progressions up to some point). This point of view allows, for example, to derive that the set of lengths of anonymous paths having a given constant size certification is eventually periodic.

Now, moving on to non-constant certifications, we need to introduce a non-uniform automata model. Indeed, since larger paths imply that we can use larger certificates, it also means larger automata. For each certificate size i, we will have an automaton $\mathcal{A}_i$, and we will require that an incorrect instance is rejected by all these automata, while a correct instance is accepted by at least one (small enough) $\mathcal{A}_i$. If a property has non-constant complexity, then it means that we will need arbitrarily large automata. If there exists a certification of size $s(n)$, then for any instance of length n, the automaton $\mathcal{A}_{s(n)}$ will have to accept. Now from the arithmetic perspective, there is an equivalence between having non-constant complexity and the fact that the set of correct lengths is not eventually periodic.

Establishing gaps in paths and trees

To establish a gap between constant size and $\Theta(\log\log n)$ size in paths, the reasoning is the following.[2] Consider a language S that does not have a constant size certification. For an arbitrary i, we focus on the paths of S that are recognized by $\mathcal{A}_i$ but not by $\mathcal{A}_1, ..., \mathcal{A}_{i-1}$ (assume this is non-empty, which is the case for infinitely many i). This set is itself recognized by an automata: the intersection of $\mathcal{A}_i$ with the complement of the union of $\mathcal{A}_1, ..., \mathcal{A}_{i-1}$. Studying the state complexity of this new automaton (which we do not discuss here) leads us to the fact that it must accept a path of length at most 2^{2^i}. Finally, this upper bound on the minimum length for which some certificate size is needed translates into a lower bound constraint for the optimal complexity, and the double exponential translates into the double logarithm of the theorem.

This proof can be generalized without much modification to labeled paths (that is, paths with inputs) and to larger verification radius (see the full version [14]). This is pretty straightforward, given the previous proof: we use classic automata instead of unary automata, and a slightly different transformation from certification to automata. On the contrary, the proof that gives the right constants for Theorem 5 utilizes Chrobak normal form for unary automata (see full version), and does not generalize easily to the labeled case.

The proof of the analogous gap in trees (Theorem 8), is based on the same insights, but is much more involved. Basically, we adapt our automata to walk in the tree, reading the pending subtrees as labels. Two challenges are that a given tree can be read in many different ways, and that the alphabet is infinite. We argue that the complexity will be captured by the walks that follow a maximum path in the tree, and that the intersection, complement and union that are used in the proof are not harmed by the infinite alphabet.

The $\log\log n$ language

We now turn our attention to Theorem 6, which establishes the existence of a language with optimal certification size $\Theta(\log\log n)$ in anonymous unlabeled paths. Let us start by giving some intuition about how we came to this result. Intuitively, having optimal certificates of size $\Theta(\log\log n)$ means that certificates of size k allow to differentiate between correct and incorrect instances of size order 2^{2^k}. When we implement simple counters modulo some constant using k bits, the modulo is of order 2^k, hence we can distinguish between correct and incorrect instances of size 2^k but not more, in the sense that q and $q+2^k$ will be classified in the same category. Hence we need to do an "exponential jump" in terms of distinguishing capability. We now make two observations. First, given a set of pairs $(r_i, m_i)_{i\leq\ell}$, the fact that the path length is *not equal* to $r_i \mod m_i$ for some $i \leq \ell$ can be certified with certificates of size $O(\max_i \log(m_i))$. Indeed, since it is sufficient to prove that one of the modulo equation is not satisfied, we can simply give explicitly m_i in all the certificates, in addition to the counter modulo m_i. Second, using the Chinese remainder theorem, for any two numbers a and b, there must exist an integer m exponentially smaller than $\max(a, b)$ such that $a \not\equiv b \mod m$. This makes the existence of a $\Theta(\log\log n)$ language believable, but making it into a real construction is another kettle of fish. We end up considering the language of the path whose length *is not* the product of consecutive primes. The scheme consists in certifying that either there is a gap or a repetition in the sequence of primes, via adequate counters, or there is an inconsistency between the largest prime used and what the length of the path is equal to, modulo a well-chosen (small) integer.

[2] Actually this proof gives worse constants than the ones of Theorem 5, but it is the one that generalizes to other settings.

The case of cycles

Theorem 7 states that in cycles there is a gap between constant and logarithmic certification size. This is in sharp contrast with the case of paths, which is surprising given the similarity of the two structures. At a very intuitive level, the difference stems from the fact that the only congruence that we can check in cycles are of the form $0 \mod m$, and not arbitrary $i \mod m$. Indeed, in a path, it is easy to have one endpoint checking counter value zero and the other checking counter value i, while in cycles there are no endpoints. The natural way to simulate the endpoints is to designate a specific vertex v in the cycle to check that the counter starts at 0 from one side and reaches i from the other side, but this is not robust. Indeed there could be more than one designated vertices and still all nodes could accept if each segment is $i \mod m$. This breaks the congruence, except if $i = 0$. This restriction in the congruences prevents us from using the arithmetic tools of the proof of Theorem 6.

Let us now sketch the proof of the gap between certificates of size $O(1)$ and $\Theta(\log n)$ in cycles. We again use an automata-like point of view, but we need to adapt it to work without starting and ending nodes. We consider a sequence of directed certificate graphs[3] $(G_i)_i$. Here, a cycle C with certificates of size i $(c_1, \dots, c_n)$ is accepted by G_i if and only if $(c_1, c_2), (c_2, c_3), \cdots, (c_n, c_1)$ is a closed walk in G_i. For the proof, we consider the first length n_k that is accepted by the k-th graph G_k (corresponding to certificates of size k), but not by smaller ones. Now, if the certificate size is in $o(\log n)$, this walk is much longer than the size of G_k, and therefore it has a lot of cycles. We define a notion of elementary cycle of the graph, and decompose this walk as a linear combination of elementary cycles. If d is the greatest common divisor of these elementary cycles, then $n_k = d \times q$, for some q. We consider a set of numbers of the form $p \times d$, such that (1) p is prime, (2) $p \times d < n_k$ and (3) $p \times d$ is still much larger than the size of G_k. By a generalization of Bézout's identity, (3) implies that all these numbers are accepted by G_k hence they belong to the language. But since they are smaller than n_k, by minimality, they must be recognized by smaller graphs too. We argue by pigeon-hole principle that there must exist one graph $G_{k'}$, $k' < k$ that has two walks of lengths $p_1 d$ and $p_2 d$ that intersect. Then by concatenating portions of the two walks, we can prove that $G_{k'}$ actually accepts all the cycles of size $ap_1 d + bp_2 d$. Finally, using again a generalization of Bézout's identity we can prove that the cycle of length n_k is also recognized by $G_{k'}$, which contradicts the minimality of k.

"No gap" results

We also establish that for several settings there is no gap in the complexity, that is, for any well-behaved function f, there exists a property that has certificates of size $O(f(n))$ but not $o(f(n))$.

Let us sketch the technique of the upper bound for the setting of Theorem 10: restricting to caterpillars, using certificates whose size is a function of d, with verification radius 2. Given a function f, we define a sequence of integers (b_k), such that the k-th term is roughly $f^{-1}(\log k)$. The correct instances are of the following form: a path of length b_k, for some k, such that the i-th node has i leaves, and except for the first one, which has b_k leaves instead of 1. A compact way to certify these instances is to give k to every node. The nodes in the middle check the growth of the number of attached leaves (thanks to their radius 2 verification) and the endpoints check that they have b_k leaves. The diameter is b_k, and the number of bits used is $\log k$ which is basically $f(d)$.

[3] We consider "graphs" and not "automata" here since we have no initial or accepting states, a sequence of certificate only gives us a walk in the graph of pairs of certificates.

1.4 Discussions and open problems

Before we move on to the technical parts, let us discuss open problems and future directions.

Full understanding of paths. For paths, we do not know what happens between $\Theta(\log\log n)$ and $\Theta(\log n)$. By sparsifying the set of primes considered in the $\log\log n$ language (Theorem 6), we can get languages for which the natural upper bound can be positioned in between these two regimes, but Theorem 5 does not provide a matching lower bound anymore, hence we cannot prove that there is no gap.

▶ **Open problem 11.** *Is there a gap between* $\Theta(\log\log n)$ *and* $\Theta(\log n)$ *in paths?*

To solve this question, it would good to get a better understanding of the $\Theta(\log\log n)$ regime, or even a characterization. (For now, we just have one example of a property in this regime.)

General graphs. For general graphs, we prove in the full version that if the radius is larger than 1, then there is no gap. For radius 1, it is unclear whether we should expect the same gap as for trees or not. Our automata-related tools seem too weak to tackle this case (the generalization from trees to graphs in automata theory is notoriously intricate).

▶ **Open problem 12.** *Is there a gap between* $O(1)$ *and* $\Theta(\log\log d)$ *for general graphs? For bounded degree graphs?*

The role of identifiers and of the knowledge of n. Our main results are for the anonymous setting, where the nodes do not have the knowledge of n. In the full version, we explore several settings with identifiers or (approximate) knowledge of n. We can for example prove that a very sharp estimate of n allows to break the $\log\log n$ barrier of Theorem 5, while arbitrarily large identifiers do not help. It is still very unclear how these different assumptions affect the complexity landscape.

Extensions to self-stabilizing algorithms: back space complexity in algorithms. We described in the introduction our wish to chart the space landscape of distributed graph algorithms, and as a first step we focused on local certification. A natural next step is to transfer our results to self-stabilizing algorithms. As mentioned earlier, the two are tightly connected, since the space used for silent self-stabilization is basically captured by the space needed to certify the solution correctness [10]. Actually, [10] implements a transformation from a local certification to a self-stabilizing algorithm, that does require an additive $O(\log n)$ for the memory of the algorithm in comparison to the local certification size. This is usually harmless, but in the setting of this paper, which focuses on sublogarithmic regime, the result is unusable. The restricted topology might allow to shave the additional logarithmic term.

▶ **Open problem 13.** *Are the optimal certificate size and the optimal memory of a self-stabilizing algorithm asymptotically equal in restricted topologies (paths, cycles, trees), even in the sublogarithmic regime?*

When it comes to transferring the sub-log-logarithmic gap for trees to (silent) self-stabilizing algorithms, it would actually be enough to understand whether labelings accepted by tree automata (which are equivalent to monadic second-order on trees) can be built in constant space in a self-stabilizing manner. Incidentally, understanding when one can make this transfer while keeping polynomial time is also a very intriguing question (see [9] for a discussion).

Different landscapes. Theorem 1 establishes that for any super-logarithmic complexity f there exists a property that requires exactly that complexity. But the construction is very unnatural, since the nodes need to know the function f, and the instances are extremely specific. Hence the following question.

▶ **Open problem 14.** *Are there super-logarithmic gaps in the complexity of natural properties, for a reasonable definition of "natural"? What about monadic second-order (MSO) properties?*

It is known that there are properties for which the optimal certification size is $\Theta(n)$, for example having diameter 3 [18, 12] and also $\Theta(n^2)$ [36]. As far as we know the only works about what happens in between are [11] and [41], that have established that for forbidden subgraphs, there are many polynomial complexities, when using verification radius 2.

We mention MSO properties in the open problem because they have received a considerable amount of attention in recent years in local certification [8, 24, 28, 35, 34], and capture many classic properties and problems through logic.

Finally, another research direction is to chart landscapes for other parameters. For example, [13] explored the certification complexity as a function of the maximum degree.

1.5 Organization of the conference version

For this conference version, we focus on the results on paths and cycles, which allows to give most of the insights within the page limit. Some of the proofs of the lemmas and claims are deferred to the appendix. The proof of our more involved results as well as discussions of the knowledge on n and the identifier assumptions only appear in the full version [14].

2 Model and definitions

2.1 Graphs

All the graphs we consider are finite, simple, loopless, connected, and undirected. For completeness, let us recall several basic graph definitions. Let G be a graph. We denote its set of vertices (resp. edges) by $V(G)$ (resp. by $E(G)$), or simply by V (resp. by E) if G is clear from the context. Let $u, v \in V(G)$. The *distance* between u and v, denoted by $d(u, v)$, is the smallest number of edges in a path from u to v. The *diameter* of G is the largest distance between any two vertices. If G is a path, its *length* is its number of vertices (or equivalently, its diameter plus one). We say that G is a *caterpillar* if, when removing all the degree-1 vertices in G, the resulting graph is a path, called the *central path* of G (which is induced by the set of all vertices of degree at least two in G).

2.2 Local certification

Let $G = (V, E)$ be a graph. We will sometimes assume that the vertices of G are equipped with *unique identifiers* and/or with *inputs*. An *identifier assignment* for G is an *injective* mapping from V to some set I (the set of identifiers) and an *input function* is a mapping from V to some set L (the set of labels). If G is equipped with identifiers, we say that we are in the *locally checkable proof model*, else we say that we are in the *anonymous model*. If the vertices of G have inputs, we say that G is *labeled*. Finally, let C be a non-empty set. A *certificate assignment* of G with certificates in C is a mapping $c : V \to C$.

Let $r \geqslant 1$ and c be a certificate assignment for G. Let $u \in V$. The *view of u at distance r* consists in:

- all the vertices at distance at most r from u, and all the edges having at least one endpoint at distance at most $r-1$,
- the restriction of c to these vertices,
- the restriction of the identifier assignment (if any) and of the input function (if any) to these vertices.

A *verification algorithm (at distance r)* is a function taking as input the view at distance r of a vertex, and outputting a decision, *accept* or *reject*. In all this paper, if r is not mentioned in a statement of a result, it is by default equal to 1. When we will consider settings where $r \geqslant 2$, it will always be explicitly written.

Let $\mathcal{C}$ be a class of (possibly labeled) graphs and $\mathcal{P}$ be a property on graphs in $\mathcal{C}$. Note that if we consider labeled graphs, the fact that a graph $G \in \mathcal{C}$ satisfies the property $\mathcal{P}$ does not depend only on the structure of G: it depends also on its input function. In other words, it is possible that a graph $G \in \mathcal{C}$ satisfies $\mathcal{P}$ and that another graph $G' \in \mathcal{C}$ with the same structure as G but with a different input function does not satisfy $\mathcal{P}$. Let $s : \mathbb{N} \to \mathbb{N}$. We say that there exists a *certification scheme for $\mathcal{P}$* with certificates of size s if there exists a verification algorithm such that the two following conditions are satisfied:

- (Completeness) For every n-vertex graph $G \in \mathcal{C}$ that satisfies $\mathcal{P}$, and for every identifier assignment of G (if we are in the locally checkable proof model), there exists a certificate assignment in $\{0, \dots, 2^{s(n)} - 1\}$ such that the verification of every vertex accepts (we say that the graph is globally accepted).
- (Soundness) For every graph $G \in \mathcal{C}$ that does not satisfy $\mathcal{P}$, for every identifier assignment of G (if we are in the locally checkable proof model), for every $k \in \mathbb{N}$ and every certificate assignment in $\{0, \dots, 2^k - 1\}$, at least one vertex rejects.

Let us emphasize that, if G does not satisfy $\mathcal{P}$, then for any assignment of certificates *of any size*, at least one vertex rejects. Let us also point out the fact that in a certification scheme for a property $\mathcal{P}$ in some class $\mathcal{C}$ (in this paper, we will for instance consider the cases where $\mathcal{C}$ is the class of paths, of cycles, of trees...), the vertices have the promise that the graph belongs to $\mathcal{C}$. In other words, the certification scheme depends on the property $\mathcal{P}$ *and* on the class $\mathcal{C}$, and we are not concerned by the output of the verification procedure of the vertices in graphs that do not belong to $\mathcal{C}$.

3 Gap between $O(1)$ and $\Theta(\log \log n)$ in paths

The goal of this section is to prove the following result:

▶ **Theorem 15.** *Let $c > 2$ and $N \in \mathbb{N}$. Let $\mathcal{P}$ be a property on paths that can be certified with certificates of size $s(n) := \left\lfloor \frac{\log \log n}{c} \right\rfloor$ for all $n \geqslant N$. Then, $\mathcal{P}$ can also be certified with constant-size certificates.*

Note that the constant c is larger here than in Theorem 5. We prove the stronger version in the full version.

3.1 Preliminary: automata point of view

For every property $\mathcal{P}$ on paths, we can associate a subsets S of integers such that a path is accepted if and only if its length is in S. The property $\mathcal{P}$ is equivalent to S and in the rest of the proof, we will completely forget the property $\mathcal{P}$ and only focus on S. For every $k \in \mathbb{N}$, let us denote by C_k the set of certificates of size k, and by S_k the set of lengths of the paths that are accepted with certificates in C_k. We have: $S = \bigcup_{k \in \mathbb{N}} S_k$.

The set S_k is a regular language that is accepted with the following nondeterministic finite automaton $\mathcal{A}_k$ over a unary alphabet. The set of states is the set of pairs of certificates of size k plus two additional states, i and f, that is: $C_k^2 \cup \{i, f\}$. There is a single initial state which is i and a single final state which is f. The transitions are the following:

- for every $c_1, c_2 \in C_k$, we put a transition between states i and (c_1, c_2) if a vertex of degree 1 that has certificate c_1 and has a neighbor with certificate c_2 accepts;
- for every $c_1, c_2, c_3 \in C_k$, we put a transition between states (c_1, c_2) and (c_2, c_3) if a vertex of degree 2 that has certificate c_2 and has two neighbors with certificates c_1 and c_3 accepts;
- for every $c_1, c_2 \in C_k$, we put a transition between states (c_1, c_2) and f if a vertex of degree 1 that has certificate c_2 and has a neighbor with certificate c_1 accepts;
- if there exists $c \in C_k$ such that an isolated vertex with certificate c accepts, we put a transition from i to f.

Let us give an example to make things more concrete. Assume that we want to certify that the length of a path is divisible by 3. There is an easy way to do it by using three certificates 0, 1, and 2. The prover fixes an endpoint u and for every vertex v, the certificate it gives to v is $d(u, v) \mod 3$. Then, every vertex v checks that one of the two following conditions is satisfied:

- v has degree 1, has certificate 0 or 2, and its neighbor has certificate 1, or
- v has degree 2, and the set of certificates of v and its two neighbors is $\{0, 1, 2\}$.

The automaton corresponding to these certificates is represented on Figure 1.

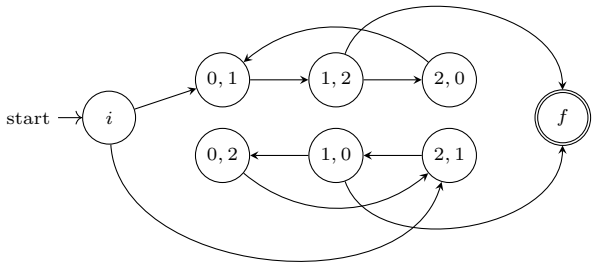

Figure 1 The automaton corresponding to the certificates used to certify that the length of a path is divisible by 3. The states corresponding to the tuples $(0, 0)$, $(1, 1)$ and $(2, 2)$ are not represented because they have no incoming nor outgoing transitions. The final state is the state f.

▶ **Lemma 16.** *For every $t \in \mathbb{N}$, a path on t vertices is accepted with certificates of size k if and only if there exists an accepting run of length t (i.e., going through t transitions) in $\mathcal{A}_k$.*

The intuition behind this lemma has been given earlier and the formal proof is in the full version.

3.2 Proof via state complexity

▶ **Lemma 17.** *Let Σ be a (possibly infinite) alphabet, and let $L_{\mathcal{A}}, L_{\mathcal{B}} \subseteq \Sigma^*$ be two languages over Σ recognized by nondeterministic finite automata $\mathcal{A}$ and $\mathcal{B}$, having $n_{\mathcal{A}}$ and $n_{\mathcal{B}}$ states respectively. Then:*

- $L_{\mathcal{A}} \cup L_{\mathcal{B}}$ *can be recognized by an automaton having* $n_{\mathcal{A}} + n_{\mathcal{B}}$ *states*
- $L_{\mathcal{A}} \cap L_{\mathcal{B}}$ *can be recognized by an automaton having* $n_{\mathcal{A}} n_{\mathcal{B}}$ *states*
- $\overline{L_{\mathcal{A}}}$ *can be recognized by an automaton having* $2^{n_{\mathcal{A}}}$ *states*

These statements are folklore, see the full version for explanations.

Proof of Theorem 15. Using the notations introduced previously, the automaton $\mathcal{A}_k$ has $M_k := 2^{2k} + 2$ states and recognizes S_k. Let $\mathcal{P}$ be a property that can not be recognized with constant-size certificates. Then, the set $X \subseteq \mathbb{N}$ containing all the integers $k \in \mathbb{N}$ such that $S_k \not\subseteq \bigcup_{i \leqslant k-1} S_i$ is infinite (this set X contains all the integers k such that there exists a path that can be accepted with certificates of size k but not with smaller size). For $k \in X$, let n_k be the smallest integer in $S_k \setminus \bigcup_{i \leqslant k-1} S_i$. Let $k \in X$ be such that $n_k \geqslant N$ (such an integer $k \in X$ exists because X is infinite and for all distinct $k, k' \in X$ we have $n_k \neq n_{k'}$).

Since $n_k \in S_{s(n_k)}$ and $n_k \notin S_i$ for $i < k$, we have $s(n_k) \geqslant k$. By Lemma 17, $\bigcup_{i \leqslant k-1} S_i$ can be recognized by an automaton that has $\sum_{i=1}^{k-1} M_i \leqslant 2^{2k}$ states. Thus, by Lemma 17, $\overline{\bigcup_{i \leqslant k-1} S_i}$ can be recognized by an automaton that has at most $2^{2^{2k}}$ states. Again by Lemma 17, $S_k \setminus \bigcup_{i \leqslant k-1} S_i$ can be recognized by an automaton having at most $M_k \cdot 2^{2^{2k}}$ states. Since n_k is the smallest integer in $S_k \setminus \bigcup_{i \leqslant k-1} S_i$, it is at most equal to the number of states of this automaton, so it follows that $n_k \leqslant M_k \cdot 2^{2^{2k}} \leqslant 2^{2^{2k+1}}$. Finally we get $s(n_k) \geqslant k \geqslant \frac{\log \log n_k - 1}{2}$, and the result follows. ◀

4 A property with optimal size $\Theta(\log \log n)$ in unlabeled paths

The goal of this Section is to prove the following theorem.

▶ **Theorem 6.** *There exist properties on paths that can be certified with certificates of size* $O(\log \log n)$*, but not with certificates of size* $O(1)$*.*

Before proving Theorem 6, let us show the following result:

▶ **Lemma 18.** *Let* $m, t \in \mathbb{N}$ *and* $m \geqslant 2$*. Certifying that the length a path on* n *vertices satisfies* $n \equiv t \mod m$ *can be done with certificates of size* $O(\log m)$*.*

The proof of this statement can be found in the full version of the paper.

▶ Remark 19. For every $m, t \in \mathbb{N}$ with $m \geqslant 2$, we can also certify that the length n of a path satisfies $n \not\equiv t \mod m$ with certificates of size $O(\log m)$, with the same proof (just by replacing $\mathsf{Distance}[u] = t$ by $\mathsf{Distance}[u] \neq t$ at the end). In particular, with certificates of size $O(\log m)$, we can certify that m divides n, or that m does not divide n.

Finally, let us introduce some notations and give some useful properties. For every $k \geqslant 1$, let us denote by p_k the k-th prime number (i.e. $p_1 = 2, p_2 = 3, p_3 = 5 \ldots$), and let a_k be the product of the k first prime numbers: $a_k := \prod_{i=1}^{k} p_i$. Let $S \subseteq \mathbb{N}$ be the set $\{a_k \mid k \geqslant 1\}$.

▶ **Lemma 20.** *[37] We have* $p_k = \Theta(k \log k)$ *and* $a_k = 2^{k \log k (1 + o(1))}$*.*

▶ **Lemma 21.** *There exists* $c > 0$ *such that, for every even integer* $n \geqslant 2$*, there exists* $k \leqslant c \log n$ *such that* p_k *divides* n *and* p_{k+1} *does not divide* n*.*

Proof. Let n be an even integer and let k be the smallest integer such that p_k divides n and p_{k+1} does not divide n (which exists because n is divisible by $p_1 = 2$). Then, n is divisible by a_k, so $a_k \leqslant n$. Using Claim 20, we get $2^{k \log k (1 + o(1))} \leqslant n$, so $k \log k (1 + o(1)) \leqslant \log n$, and the result follows. ◀

▶ **Lemma 22.** *Let $1 \leqslant s < t$. There exists $m \leqslant \lceil \log t \rceil$ such that $s \not\equiv t \mod m$.*

Proof. By contradiction, assume that for all $m \leqslant \lceil \log t \rceil$, we have $s \equiv t \mod m$. Then, by the Chinese remainder theorem, we get $s \equiv t \mod p$, where p is the least common multiple of $1, 2, \ldots, \lceil \log t \rceil$. It is well-known that this least common multiple is at least t, so we have $1 \leqslant s < t \leqslant p$. Together with $s \equiv t \mod p$, this implies $s = t$, which contradicts the assumption $s < t$. ◀

▶ **Proposition 23.** *Let $n \geqslant 1$, and c be the constant of Lemma 21. Then, $n \notin S$ if and only if at least one of the three following conditions is satisfied:*

1. *n is odd, or*
2. *there exists $1 \leqslant \ell < k \leqslant c \log n$ such that p_ℓ does not divide n and p_k divides n, or*
3. *there exists $1 \leqslant k \leqslant c \log n$ and $1 \leqslant m \leqslant \lceil \log n \rceil$ such that p_k divides n, p_{k+1} does not divide n and $n \not\equiv a_k \mod m$.*

Proof. First, assume that one of the three conditions is satisfied. If condition 1. holds, n is odd, so $n \notin S$ because S contains only even integers. If condition 2. holds, then $n \notin S$ because all the integers in S which are divisible by p_k are also divisible by p_ℓ for all $\ell \leqslant k$. If condition 3. holds, then $n \notin S$, because the only integer in S which is divisible by p_k and not by p_{k+1} is a_k.

Conversely, assume that $n \notin S$, and let us show that at least one of the three conditions is satisfied. Assume that conditions 1. and 2. are not satisfied, and let us show that condition 3. holds. Since condition 1. is not satisfied, n is even, so by Lemma 21, there exists $k \leqslant c \log n$ such that p_k divides n and p_{k+1} does not divide n. Since condition 2. is not satisfied, n is divisible by a_k, so $a_k < n$ (this inequality is strict because, by assumption, $n \notin S$). Finally, by Lemma 22, there exists $m \leqslant \lceil \log n \rceil$ such that $n \not\equiv a_k \mod m$, so condition 3. is satisfied. ◀

Note that the third item is used to avoid accepting numbers with some prime used twice. It is tempting to check this condition directly instead of using our indirect check. But in general this would use integers that are too large, for example in the case where $n = q^2$ with q a prime number.

We are now able to prove Theorem 6.

Proof of Theorem 6. Recall that we assume that the input graph is a path P. Let $\mathcal{P}$ be the property of being a path whose length is *not* in S. First, observe that $\mathcal{P}$ cannot be certified with constant-size certificates. Indeed, properties on paths that can be certified with constant size-certificates are paths whose length is in a set that is eventually periodic (the most simple proof for it uses Chrobak normal form, see the full version), and the set $\overline{S}$ is not. Now, let us show that $\mathcal{P}$ can be certified with certificates of size $O(\log \log n)$.

Let $n \geqslant 1$. If $n \notin S$, the prover certifies that at least one of the three conditions of Proposition 23 is satisfied. More precisely:

- if n is odd, the prover certifies it. By Lemma 18, this needs $O(1)$ bits.
- if there exists $1 \leqslant \ell < k \leqslant c \log n$ such that p_k divides n and p_ℓ does not divide n, the prover writes k and ℓ in the certificate of each vertex, and certifies that p_k divides n and that p_ℓ does not divide n. Since $k \leqslant c \log n$ and $p_k = \Theta(k \log k)$, by Lemma 18 and Remark 19, this needs $O(\log \log n)$ bits.
- if there exists $1 \leqslant k \leqslant c \log n$ and $1 \leqslant m \leqslant \lceil \log n \rceil$ such that p_k divides n, p_{k+1} does not divide n and $n \not\equiv a_k \mod m$, the prover writes k and m in the certificate, certifies that p_k divides n, p_{k+1} does not divides n and that $n \not\equiv a_k \mod m$. By Lemma 18 and Remark 19, this needs $O(\log \log n)$ bits.

The verification of the vertices just consists in checking that the condition given by the prover is indeed satisfied, with the verification procedure of Lemma 18. Note that the vertices do not need to check the bounds on k and m for conditions 2. and 3., because if condition 2. or 3. is satisfied with larger k or m, it also implies that $n \notin S$ (and then $P \in \mathcal{P}$ and P can be also accepted with certificates of size $O(\log \log n)$). These bounds on k and m are only useful to get an upper bound on the size of the certificates. ◀

5 Gap between $O(1)$ and $\Theta(\log n)$ in cycles

In this section we prove the following theorem.

▶ **Theorem 7.** *Let $c > 12$ and $N \in \mathbb{N}$. Let $\mathcal{P}$ be a property on cycles that can be certified with certificates of size $s(n) := \left\lfloor \frac{\log n}{c} \right\rfloor$ for every integer $n \geqslant N$. Then, $\mathcal{P}$ can also be certified with constant-size certificates.*

5.1 Preliminaries on number theory and walks in graphs

Let us give some results from number theory on which we will rely. First, let us recall the prime number theorem:

▶ **Theorem 24** (Prime Number Theorem)**.** *For $n \in \mathbb{N}$, let $\pi(n)$ be the number of prime numbers in $\{1, \ldots, n\}$. Then:*

$$\pi(n) \sim \frac{n}{\ln(n)}$$

From Theorem 24, we can deduce the immediate following corollary:

▶ **Corollary 25.** *Let $c > 12$. For every $n \in \mathbb{N}$, let $\pi_c(n)$ be the number of prime numbers p such that $2^{4n+1} < p \leqslant 2^{n(c/2-2)}$. Then, there exists $n_0 \in \mathbb{N}$ such that for all $n \geqslant n_0$, $\pi_c(n) > n2^{2n}$.*

Proof. For every $n \in \mathbb{N}$, we have $\pi_c(n) = \pi(2^{n(c/2-2)}) - \pi(2^{4n+1})$. Since $c > 12$, by Theorem 24, we have $\pi(2^{4n+1}) = o(\pi(2^{n(c/2-2)}))$. Thus, $\pi_c(n) \sim \pi(2^{n(c/2-2)})$. By applying again the prime number theorem, we get $\pi_c(n) \sim 2^{n(c/2-2)}/(n(c/2-2))$. Thus, $n2^{2n} = o(\pi_c(n))$, and the result follows. ◀

Finally, let us give the following generalization of Bézout's identity that will be useful in the proof of Theorem 7:

▶ **Lemma 26** ([17])**.** *Let $\ell_1, \ldots, \ell_t$ be positive integers. Let $m := \max(\ell_1, \ldots, \ell_t)$ and $d := \gcd(\ell_1, \ldots, \ell_t)$. Then, for every integer $n \geqslant m^2$ which is divisible by d, there exists non-negative integers $a_1, \ldots a_t$ such that $\sum_{i=1}^{t} a_i \ell_i = n$.*

We now move on to some definitions about walks in graphs. A *closed walk* in $\mathcal{G}_k$ is a directed path that begins and ends in the same vertex (it is allowed to pass through the same vertex or the same edge multiple times). The *length* of a closed walk is its number of edges. A closed walk that does not pass through the same vertex twice (except for the starting and ending vertices which are the same) is called an *elementary cycle*. If $\mathcal{C}$ and $\mathcal{C}'$ are two directed closed walks, we say that $\mathcal{C}'$ is a closed subwalk of $\mathcal{C}$ if a subsequence of vertices in $\mathcal{C}$ is equal to $\mathcal{C}'$. See Figure 2 for an example. Note that the length of an elementary closed walk in $\mathcal{G}_k$ is at most equal to the number of vertices of $\mathcal{G}_k$ which is 2^{2k}.

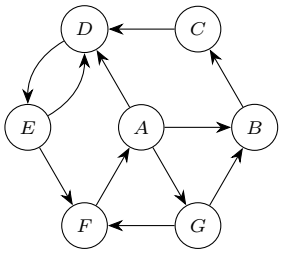

Figure 2 In this directed graph, $ADEDEFA$ is a closed walk of length 6, which is not an elementary cycle. The closed walks $FAGF$ and $CDEFABC$ are elementary cycles and have length 3 and 6 respectively. Moreover, $FAGF$ is a subwalk of $DEFAGFAD$.

5.2 Proof of Theorem 7

All this subsection is devoted to the proof of Theorem 7. The proofs of the claims can be found in Appendix B.

Let $c > 12$ and $N \in \mathbb{N}$. Let S be the set of lengths of cycles in $\mathcal{P}$. Assume by contradiction that there exists a property $\mathcal{P}$ such that, for every integer $n \geqslant N$ satisfying $\mathcal{P}$, the cycle of length n is accepted with certificates of size $s(n)$, where $s(n) \leqslant \left\lfloor \frac{\log n}{c} \right\rfloor$, and that a constant number of certificates are not sufficient to certify $\mathcal{P}$.

For every $k \geqslant 1$, let C_k be the set of certificates of size k, and let S_k be the subset of S corresponding to the cycles accepted with certificates in C_k. Note that $|C_k| = 2^k$, and that we have $S = \bigcup_{k\in\mathbb{N}} S_k$. Let $\mathcal{G}_k = (V_k, E_k)$ be the directed graph where $V_k := C_k^2$ and for all $a, b, c \in C_k$, there is an edge in E_k between (a, b) and (b, c) if and only if a degree-2 vertex with certificate b has a neighbor with certificate a and another neighbor with certificate c accepts (and there are no other edges in E_k, that is there is no edge between (a, b) and (c, d) if $b \neq c$). The directed graph $\mathcal{G}_k$ has 2^{2k} vertices.

▷ Claim 27. For every integer $n \geqslant 3$, $n \in S_k$ if and only there exists a closed walk of length n in $\mathcal{G}_k$.

By Corollary 25, there exists $k_0 \in \mathbb{N}$ such that for all $k \geqslant k_0$, $\pi_c(k) > k2^{2k}$.

Since S is not accepted with a constant number of certificates, the set $X \subseteq \mathbb{N}$ of integers $k \in \mathbb{N}$ such that $S_k \not\subseteq \bigcup_{1\leqslant i<k} S_i$ is infinite. For $k \in X$, let n_k be the smallest integer $S_k \setminus \bigcup_{1\leqslant i<k} S_i$. Finally, let us fix an integer integer $k \in X$, such that $k \geqslant k_0$ and $n_k \geqslant N$ (such an integer $k \in X$ exists because X is infinite and for all distinct $k, k' \in X$ we have $n_k \neq n_{k'}$).

▷ Claim 28. We have $n_k \geqslant 2^{ck}$.

Since $n_k \in S_k$, by Claim 27, there is a closed walk of length n_k in $\mathcal{G}_k$. Let us consider the strongly connected component $\mathcal{G}'_k$ of $\mathcal{G}_k$ containing this closed walk. Let t be the number of elementary cycles in $\mathcal{G}'_k$ that we denote by $\mathcal{C}_1, \ldots, \mathcal{C}_t$, and let $\ell_1, \ldots, \ell_t$ be their lengths. Let $d = \gcd(\ell_1, \ldots, \ell_t)$. We have $d \leqslant 2^{2k}$, because we have $\ell_i \leqslant 2^{2k}$ for every $i \in \{1, \ldots, t\}$ (since $\mathcal{G}_k$ has size 2^{2k}).

▷ Claim 29. Let $\mathcal{C}$ be a closed walk in $\mathcal{G}'_k$, and let ℓ be its length. There exists $b_1, \ldots, b_t \in \mathbb{N}$ such that $\ell = \sum_{i=1}^{t} b_i\ell_i$. Thus, d divides ℓ. In particular, d divides n_k.

▷ Claim 30. Let $m \in \mathbb{N}$ be such that d divides m, and $m \geqslant 2^{4k+1}$. Then, there exists a closed walk in $\mathcal{G}'_k$ of length m. Thus, $m \in S_k$.

Let us now combine these arguments to prove Theorem 7. Before giving the technical details, let us explain the intuition. Let us denote by d the $d := gcd$ of all the lengths of cycles in S_k. Since $\mathcal{G}_k$ has size 2^{2k}, Lemma 30 ensures that all the cycles of length $r \cdot d$ are in $\mathcal{P}$ when r is large enough (but small compared to n_k). Thus there exist many prime numbers p such that pd are in $\mathcal{P}$ and $pd \leq n_k$. By definition of n_k, at least two of them can be certified with the same set of bits and we can obtain a contradiction. Let us now formalize the argument.

Recall that k is an integer such that $k \geqslant k_0$ and $n_k \geqslant N$. Let p be a prime number such that $2^{4k+1} < p \leqslant 2^{k(c/2-2)}$. By Claim 30, we have $pd \in S_k$. Moreover, since $p \leqslant 2^{k(c/2-2)}$ and $d \leqslant 2^{2k}$, we have $pd \leqslant 2^{kc/2}$, that is, $pd \leq \sqrt{n_k}$ using Claim 28. Since n_k is the smallest integer in $S_k \setminus \bigcup_{1 \leqslant i < k} S_i$, we have $pd \in \bigcup_{1 \leqslant i < k} S_i$. For every $i \in \{1, \dots, k-1\}$, let X_i be the set of prime numbers $p \in \{2^{4k+1}+1, \dots, 2^{k(c/2-2)}\}$ such that $pd \in S_i$. Since there are $\pi_c(k)$ prime numbers in $\{2^{4k+1}+1, \dots, 2^{k(c/2-2)}\}$ and we have $\pi_c(k) > k2^{2k}$, by the pigeonhole principle there exists $i < k$ such that $|X_i| > 2^{2k}$. Let us fix this index i.

For every $p \in X_i$, since $pd \in S_i$, there exists a closed walk $\mathcal{C}^{(p)}$ of length pd in $\mathcal{G}_i$. Since $|X_i| > 2^{2k}$, and since $\mathcal{G}_i$ has 2^{2i} vertices and $i < k$, again by the pigeonhole principle there exist $p, q \in X_i$ such that $\mathcal{C}^{(p)}$ and $\mathcal{C}^{(q)}$ have a vertex in common. Since $\mathcal{C}^{(p)}$ has length pd, $\mathcal{C}^{(q)}$ has length qd, and these two cycles have a vertex in common, for every $a, b \in \mathbb{N}$, there exists a closed walk of length $apd + bqd$ in $\mathcal{G}_i$ (obtained by starting from a vertex $u \in \mathcal{C}^{(p)} \cap \mathcal{C}^{(q)}$, taking a times $\mathcal{C}^{(p)}$ and then b times $\mathcal{C}^{(q)}$). Thus, for every $a, b \in \mathbb{N}$, $apd + bqd \in S_i$.

Finally, since $\gcd(pd, qd) = d$ (because p and q are two distinct prime numbers), since $pd, qd \leqslant \sqrt{n_k}$, and since n_k is divisible by d by Claim 29, we can apply Lemma 26 which states that there exists $a, b \in \mathbb{N}$ such that $apd + bpd = n_k$. So $n_k \in S_i$, which a contradiction, because by assumption $n_k \in S_k \setminus \bigcup_{1 \leqslant i < k} S_i$. This concludes the proof of Theorem 7.

6 A property with optimal size $\Theta(\log n)$ in cycles

Let us finish this short version of the paper with a property that can be certified with $O(\log n)$ bits but not with constant-size certificates, to show that the gap stated in Theorem 7 is optimal. The proof is in Appendix C

▶ **Proposition 31.** *Certifying that the length of a cycle is not a power of* 2 *can be done with certificates of size* $O(\log n)$ *but not with certificates of size* $O(1)$.

References

1 Karine Altisen, Stéphane Devismes, Swan Dubois, and Franck Petit. *Introduction to Distributed Self-Stabilizing Algorithms*. Morgan & Claypool Publishers, 2019. doi:10.2200/S00908ED1V01Y201903DCT015.

2 James Aspnes and Eric Ruppert. An introduction to population protocols. *Bull. EATCS*, 93:98–117, 2007.

3 Alkida Balliu, Sebastian Brandt, Yi-Jun Chang, Dennis Olivetti, Mikaël Rabie, and Jukka Suomela. The distributed complexity of locally checkable problems on paths is decidable. In Peter Robinson and Faith Ellen, editors, *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, PODC 2019*, pages 262–271. ACM, 2019. doi:10.1145/3293611.3331606.

4 Alkida Balliu, Sebastian Brandt, Yi-Jun Chang, Dennis Olivetti, Jan Studený, and Jukka Suomela. Efficient classification of locally checkable problems in regular trees. In Christian

Scheideler, editor, *36th International Symposium on Distributed Computing, DISC 2022*, volume 246 of *LIPIcs*, pages 8:1–8:19, 2022. doi:10.4230/LIPICS.DISC.2022.8.

5 Alkida Balliu, Sebastian Brandt, Yi-Jun Chang, Dennis Olivetti, Jan Studený, Jukka Suomela, and Aleksandr Tereshchenko. Locally checkable problems in rooted trees. *Distributed Comput.*, 36(3):277–311, 2023. doi:10.1007/S00446-022-00435-9.

6 Alkida Balliu, Sebastian Brandt, Dennis Olivetti, and Jukka Suomela. Almost global problems in the LOCAL model. *Distributed Comput.*, 34(4):259–281, 2021. doi:10.1007/S00446-020-00375-2.

7 Alkida Balliu, Juho Hirvonen, Janne H. Korhonen, Tuomo Lempiäinen, Dennis Olivetti, and Jukka Suomela. New classes of distributed time complexity. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018*, pages 1307–1318. ACM, 2018. doi:10.1145/3188745.3188860.

8 Dan Alden Baterisna and Yi-Jun Chang. Optimal local certification on graphs of bounded pathwidth. *CoRR*, abs/2502.00676, 2025. doi:10.48550/arXiv.2502.00676.

9 Lélia Blin, Swan Dubois, and Laurent Feuilloley. Silent MST approximation for tiny memory. In Stéphane Devismes and Neeraj Mittal, editors, *Stabilization, Safety, and Security of Distributed Systems - 22nd International Symposium, SSS 2020*, volume 12514, pages 118–132. Springer, 2020. doi:10.1007/978-3-030-64348-5_10.

10 Lélia Blin, Pierre Fraigniaud, and Boaz Patt-Shamir. On proof-labeling schemes versus silent self-stabilizing algorithms. In *Stabilization, Safety, and Security of Distributed Systems - 16th International Symposium, SSS 2014*, volume 8756, pages 18–32, 2014. doi:10.1007/978-3-319-11764-5_2.

11 Nicolas Bousquet, Linda Cook, Laurent Feuilloley, Théo Pierron, and Sébastien Zeitoun. Local certification of forbidden subgraphs. *CoRR*, abs/2402.12148, 2024. doi:10.48550/arXiv.2402.12148.

12 Nicolas Bousquet, Louis Esperet, Laurent Feuilloley, and Sébastien Zeitoun. Renaming in distributed certification. *CoRR*, abs/2409.15404, 2024. doi:10.48550/arXiv.2409.15404.

13 Nicolas Bousquet, Laurent Feuilloley, and Sébastien Zeitoun. Local certification of local properties: Tight bounds, trade-offs and new parameters. In *41st International Symposium on Theoretical Aspects of Computer Science, STACS 2024*, volume 289 of *LIPIcs*, pages 21:1–21:18, 2024. doi:10.4230/LIPICS.STACS.2024.21.

14 Nicolas Bousquet, Laurent Feuilloley, and Sébastien Zeitoun. Complexity landscape for local certification. *CoRR*, abs/2505.20915, 2025. doi:10.48550/arXiv.2505.20915.

15 Sebastian Brandt, Orr Fischer, Juho Hirvonen, Barbara Keller, Tuomo Lempiäinen, Joel Rybicki, Jukka Suomela, and Jara Uitto. A lower bound for the distributed lovász local lemma. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016*, pages 479–488. ACM, 2016. doi:10.1145/2897518.2897570.

16 Sebastian Brandt, Juho Hirvonen, Janne H. Korhonen, Tuomo Lempiäinen, Patric R. J. Östergård, Christopher Purcell, Joel Rybicki, Jukka Suomela, and Przemyslaw Uznanski. LCL problems on grids. In Elad Michael Schiller and Alexander A. Schwarzmann, editors, *Proceedings of the ACM Symposium on Principles of Distributed Computing, PODC 2017*, pages 101–110. ACM, 2017. doi:10.1145/3087801.3087833.

17 Alfred Brauer. On a problem of partitions. *American Journal of Mathematics*, 64(1):299–312, 1942.

18 Keren Censor-Hillel, Ami Paz, and Mor Perry. Approximate proof-labeling schemes. *Theor. Comput. Sci.*, 811:112–124, 2020. doi:10.1016/J.TCS.2018.08.020.

19 Yi-Jun Chang. The complexity landscape of distributed locally checkable problems on trees. In Hagit Attiya, editor, *34th International Symposium on Distributed Computing, DISC 2020*, volume 179 of *LIPIcs*, pages 18:1–18:17, 2020. doi:10.4230/LIPICS.DISC.2020.18.

20 Yi-Jun Chang. The distributed complexity of locally checkable labeling problems beyond paths and trees. In *15th Innovations in Theoretical Computer Science Conference, ITCS 2024,*

January 30 to February 2, 2024, Berkeley, CA, USA, volume 287 of *LIPIcs*, pages 26:1–26:25, 2024. doi:10.4230/LIPICS.ITCS.2024.26.

21 Yi-Jun Chang, Tsvi Kopelowitz, and Seth Pettie. An exponential separation between randomized and deterministic complexity in the LOCAL model. *SIAM J. Comput.*, 48(1):122–143, 2019. doi:10.1137/17M1117537.

22 Yi-Jun Chang and Seth Pettie. A time hierarchy theorem for the LOCAL model. *SIAM J. Comput.*, 48(1):33–69, 2019. doi:10.1137/17M1157957.

23 Yi-Jun Chang, Jan Studený, and Jukka Suomela. Distributed graph problems through an automata-theoretic lens. *Theor. Comput. Sci.*, 951:113710, 2023. doi:10.1016/J.TCS.2023.113710.

24 Linda Cook, Eun Jung Kim, and Tomás Masarík. A tight meta-theorem for LOCAL certification of mso_2 properties within bounded treewidth graphs. *CoRR*, abs/2503.19671, 2025. doi:10.48550/arXiv.2503.19671.

25 Shlomi Dolev. *Self-Stabilization*. MIT Press, 2000. URL: http://www.cs.bgu.ac.il/%7Edolev/book/book.html.

26 Yuval Emek and Roger Wattenhofer. Stone age distributed computing. In Panagiota Fatourou and Gadi Taubenfeld, editors, *ACM Symposium on Principles of Distributed Computing, PODC '13, Montreal, QC, Canada, July 22-24, 2013*, pages 137–146. ACM, 2013. doi:10.1145/2484239.2484244.

27 Laurent Feuilloley. Introduction to local certification. *Discret. Math. Theor. Comput. Sci.*, 23(3), 2021. doi:10.46298/DMTCS.6280.

28 Laurent Feuilloley, Nicolas Bousquet, and Théo Pierron. What can be certified compactly? compact local certification of MSO properties in tree-like graphs. In *PODC '22: ACM Symposium on Principles of Distributed Computing*, pages 131–140. ACM, 2022. doi:10.1145/3519270.3538416.

29 Laurent Feuilloley and Pierre Fraigniaud. Survey of distributed decision. *Bull. EATCS*, 119, 2016. URL: http://eatcs.org/beatcs/index.php/beatcs/article/view/411.

30 Pierre Fraigniaud, Mika Göös, Amos Korman, and Jukka Suomela. What can be decided locally without identifiers? In *ACM Symposium on Principles of Distributed Computing*, pages 157–165. ACM, 2013. doi:10.1145/2484239.2484264.

31 Pierre Fraigniaud, Magnús M. Halldórsson, and Amos Korman. On the impact of identifiers on local decision. In *Principles of Distributed Systems, 16th International Conference, OPODIS 2012*, volume 7702, pages 224–238. Springer, 2012. doi:10.1007/978-3-642-35476-2_16.

32 Pierre Fraigniaud, Juho Hirvonen, and Jukka Suomela. Node labels in local decision. *Theor. Comput. Sci.*, 751:61–73, 2018. doi:10.1016/J.TCS.2017.01.011.

33 Pierre Fraigniaud, Amos Korman, and David Peleg. Towards a complexity theory for local distributed computing. *J. ACM*, 60(5):35:1–35:26, 2013. doi:10.1145/2499228.

34 Pierre Fraigniaud, Frédéric Mazoit, Pedro Montealegre, Ivan Rapaport, and Ioan Todinca. Distributed certification for classes of dense graphs. In Rotem Oshman, editor, *37th International Symposium on Distributed Computing, DISC 2023*, volume 281 of *LIPIcs*, pages 20:1–20:17, 2023. doi:10.4230/LIPICS.DISC.2023.20.

35 Pierre Fraigniaud, Pedro Montealegre, Ivan Rapaport, and Ioan Todinca. A meta-theorem for distributed certification. *Algorithmica*, 86(2):585–612, 2024. doi:10.1007/S00453-023-01185-1.

36 Mika Göös and Jukka Suomela. Locally checkable proofs in distributed computing. *Theory Comput.*, 12(1):1–33, 2016. doi:10.4086/TOC.2016.V012A019.

37 Godfrey Harold Hardy and Edward Maitland Wright. *An introduction to the theory of numbers*. Oxford university press, 1979.

38 Juris Hartmanis and Richard E Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 117:285–306, 1965.

39 Sungjin Im, Ravi Kumar, Silvio Lattanzi, Benjamin Moseley, and Sergei Vassilvitskii. Massively parallel computation: Algorithms and applications. *Found. Trends Optim.*, 5(4):340–417, 2023. doi:10.1561/2400000025.

40 Amos Korman, Shay Kutten, and David Peleg. Proof labeling schemes. *Distributed Comput.*, 22(4):215–233, 2010. doi:10.1007/S00446-010-0095-3.

41 Masayuki Miyamoto. Distributed complexity of p_k-freeness: Decision and certification. *CoRR*, abs/2410.20353, 2024. doi:10.48550/arXiv.2410.20353.

42 Jukka Suomela. Landscape of locality (invited talk). In Susanne Albers, editor, *17th Scandinavian Symposium and Workshops on Algorithm Theory, SWAT*, volume 162 of *LIPIcs*, pages 2:1–2:1. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPICS.SWAT.2020.2.

A No gap above $\log n$ in general graphs with identifiers

We establish that there is no gap in the local certification complexity in general graphs with identifiers in the $\Omega(\log n)$ regime. The proof follows the same route as the proof of the $\Theta(n^2)$ bound for non-trivial automorphism in [36] .

▶ **Theorem 1.** *For general graphs with identifiers, for any non-decreasing function $f(n)$ in $\Omega(\log n)$ and $O(n^2)$, there exists a property that can be certified with $O(f(n))$ bits, but not in $o(f(n))$ bits.*

Since the proof is a rather direct adaptation of the one of [36] and is not central in this paper, we use a more sketchy style than for our other proofs.

Proof. Fix some function f in $\Omega(\log n)$ and $O(n^2)$ consider the following language. A graph is in the language if it is made of two copies of a graph H on $\sqrt{f(n)}$ nodes,where an arbitrary node is linked to its copy by a path of length $n-2\sqrt{f(n)}$. A certification for this language is the following. On a correct instance, every node is given the following pieces of information.

1. Its part of the certification of the size n of the graph, via a spanning tree (see [27]).
2. The adjacency matrix of H.
3. The identifier assignment restricted to the copies of H.
4. Its parts of two spanning trees, pointing to the two nodes that belong both to a copy of H and to the path.

Every node checks the following.

1. Item 1 and 4 above are consistent (again see [27]).
2. If its identifier appears in the list of Item 3, it checks that its neighborhood in the graph is consistent with the neighborhood in H as described by the certificates; except if it is the root of a spanning tree of Item 4, in which case it should have exactly one additional neighbor.
3. If its identifier does not appear in the certificates it should have degree 2.

The correctness is follows from [36], where the same scheme is used except that the central path has constant length (and therefore the identifiers of the nodes on the path can be given to all nodes without overhead).

The spanning tree certifying the number of nodes can be encoded in $O(\log n)$ bits, the adjacency matrices use $O\left(\left(\sqrt{f(n)}\right)^2\right) = O(f(n))$ bits, and the identifier assignment uses $2\sqrt{f(n)}\log n$ bits. Hence in total $O(f(n))$ bits in the regime of the theorem.

Again the lower bound is very similar to the one of [36]. Basically, if we were to use $o(f(n))$ bits, by pigeon-hole principle, there would be two different correct instances of the same size n, for which the same certificates would be used on one edge of the path. Then

we could consider the graph where we take the right part from one instance and the left part from the other (with their accepting certificates), gluing on the edge with identical certificates. This new graph would be accepted, but it is not in our language since the graphs at the end of the path are different. A contradiction. ◀

B Missing proofs for the gap between $O(1)$ and $\Theta(\log n)$ in cycles

In this appendix section, we restate and prove the claim of Section 5 for which the proof was missing.

▷ Claim 27. For every integer $n \geqslant 3$, $n \in S_k$ if and only there exists a closed walk of length n in $\mathcal{G}_k$.

Proof. If $n \in S_k$, there exists an assignment of certificates $c_1, \ldots, c_n$ to the vertices of a n-vertex cycle such that every vertex accepts. For every $i \in \{1, \ldots, n\}$, since the vertex with certificate c_i accepts and has two neighbors with certificates c_{i-1} and c_{i+1} (where $i-1$ and $i+1$ are taken modulo n), by definition there is an edge in $\mathcal{G}_k$ from (c_{i-1}, c_i) to (c_i, c_{i+1}). So this gives an closed walk of length n in $\mathcal{G}_k$. Conversely, if there exists a closed walk $(c_1, c_2), (c_2, c_3), \ldots, (c_n, c_1)$ in $\mathcal{G}_k$, by definition all the vertices of the cycle of length n with certificates $c_1, \ldots, c_n$ accept, so $n \in S_k$. ◁

▷ Claim 28. We have $n_k \geqslant 2^{ck}$.

Proof. By definition of $s(n_k)$, we have $n_k \in S_{s(n_k)}$. Since $n_k \in S_k \setminus \bigcup_{1 \leqslant i < k} S_i$, it follows that $s(n_k) \geqslant k$. Moreover, by assumption, $s(n_k) \leqslant \left\lfloor \frac{\log n_k}{c} \right\rfloor$. So $n_k \geqslant 2^{ck}$. ◁

▷ Claim 29. Let $\mathcal{C}$ be a closed walk in $\mathcal{G}'_k$, and let ℓ be its length. There exists $b_1, \ldots, b_t \in \mathbb{N}$ such that $\ell = \sum_{i=1}^{t} b_i \ell_i$. Thus, d divides ℓ. In particular, d divides n_k.

Proof. Let us prove this result by induction on the length of $\mathcal{C}$. The base case includes all elementary cycles: if $\mathcal{C}$ is an elementary cycle, there exists $j \in \{1, \ldots, t\}$ such that $\ell = \ell_j$, so the result is trivially true. Assume now that $\mathcal{C}$ is not an elementary cycle, and consider the shortest closed subwalk $\mathcal{C}'$ of $\mathcal{C}$. Then, $\mathcal{C}'$ is an elementary cycle (otherwise it would not be the shortest subwalk of $\mathcal{C}$), so $\mathcal{C}' \in \{\mathcal{C}_1, \ldots, \mathcal{C}_t\}$ and its length is equal to ℓ_j for some $j \in \{1, \ldots, t\}$. Let us denote by $\mathcal{C} \setminus \mathcal{C}'$ the closed walk obtained by removing from $\mathcal{C}$ the steps of $\mathcal{C}'$. The length of $\mathcal{C} \setminus \mathcal{C}'$ is $\ell - \ell_j$. Finally, apply the induction hypothesis to the closed walk $\mathcal{C} \setminus \mathcal{C}'$, to obtain integers $b_1, \ldots, b_t \in \mathbb{N}$ such that $\ell - \ell_j = \sum_{i=1}^{t} b_i \ell_i$. The result follows. ◁

▷ Claim 30. Let $m \in \mathbb{N}$ be such that d divides m, and $m \geqslant 2^{4k+1}$. Then, there exists a closed walk in $\mathcal{G}'_k$ of length m. Thus, $m \in S_k$.

Proof. First, we construct greedily a closed walk $\mathcal{C}_0$ in $\mathcal{G}'_k$ that passes through all the vertices of $\mathcal{G}'_k$ (it exists, because $\mathcal{G}'_k$ is strongly connected). For every $u, v \in V(\mathcal{G}'_k)$, the shortest directed path from u to v has length at most the number of vertices of $\mathcal{G}_k$ which is 2^{2k}. Thus, there exists a closed walk $\mathcal{C}_0$ of length $\ell_0 \leqslant (2^{2k})^2 = 2^{4k}$ that passes through all the vertices. By Claim 29, d divides ℓ_0, so d divides $m - \ell_0$. Furthermore, $m - \ell_0 \geqslant 2^{4k}$. Since we have $\max_{1 \leqslant i \leqslant t} \ell_i \leqslant 2^{2k}$, we can apply Lemma 26 to get the existence of integers $a_1, \ldots, a_t \in \mathbb{N}$ such that $m - \ell_0 = \sum_{i=1}^{t} a_i \ell_i$. Finally, to construct a closed walk in $\mathcal{G}'_k$ of length $m = \ell_0 + \sum_{i=1}^{t} a_i \ell_i$, we attach a_i times the elementary cycle $\mathcal{C}_i$ to the closed walk $\mathcal{C}_0$ for every $i \in \{1, \ldots, t\}$ (this is possible, because $\mathcal{C}_0$ passes through all the vertices). By Claim 27, we have $m \in S_k$. ◁

C Proof of a property with optimal size $\Theta(\log n)$ in cycles

Let us restate the result and prove it.

▶ **Proposition 32.** *Certifying that the length of a cycle is not a power of 2 can be done with certificates of size $O(\log n)$ but not with certificates of size $O(1)$.*

Proof. Let C be a cycle of length n, let $u \in V(C)$ and let P be the path obtained from C by deleting one edge adjacent to u. To certify that $n \notin \{2^k, k \in \mathbb{N}\}$, the prover writes in the certificate of every vertex $v \in V(C)$ the tuple (d, i) where $d \geqslant 3$ is an odd integer that divides n, and i is the distance from u to v in P modulo d. Every vertex checks that, if its certificate is (d, i) then its two neighbors have certificates $(d, i-1 \mod d)$ and $(d, i+1 \mod d)$, and that d is indeed odd. Such a certificate has size $O(\log n)$. This scheme is correct: indeed, if all the vertices accept, the length of the cycle should be divisible by d (and conversely, with the certificates described above, all the vertices will accept if the cycle has length divisible by d).

Now, assume by contradiction that certifying that the length n of a cycle is not a power of 2 can be done with certificates of constant size k (or equivalently, that 2^k distinct certificates are sufficient). Let p be an odd prime number such that $p > 2^k$. Let us consider an assignment of certificates to the vertices of a cycle C of length p such that all the vertices accept. Let us number the vertices of C in clockwise order starting from an arbitrary vertex, and for every $i \in \{0, \dots, n-1\}$, let u_i be the i-th vertex in this numbering, and c_i be its certificate. By the pigeonhole principle, there exists $0 \leqslant i < j \leqslant n-1$ such that $(c_i, c_{i+1}) = (c_j, c_{j+1})$ (where $j+1$ is taken modulo n). If $j = i+1$, then a vertex with certificate c_i accepts with two neighbors having certificate c_i. Thus, in this case, any cycle is accepted (by giving the certificate c_i to all the vertices) which is a contradiction. Else, let $\ell_1 := j - i$ and $\ell_2 := p - j + i$. We have $\ell_1, \ell_2 \in \{1, \dots, p-1\}$ and $\ell_1 + \ell_2 = p$. Since p is prime, we get $\gcd(\ell_1, \ell_2) = 1$. Moreover, for any $a_1, a_2 \in \mathbb{N}$, a cycle of size $a_1\ell_1 + a_2\ell_2$ is accepted. Indeed, by cutting such a cycle in a_1 portions of length ℓ_1 and a_2 portions of length ℓ_2, giving the certificates $c_i, \dots, c_{j-1}$ to the vertices in portions of size ℓ_1 and $c_j, \dots, c_{n-1}, c_0, \dots, c_{i-1}$ to the vertices in portions of length ℓ_2, all the vertices accept because they have the same view as a vertex which accepts in C. Using Lemma 26, all the the cycles of length $m \geqslant p^2$ are accepted, which is a contradiction because the cycles whose length is a power of 2 greater than p^2 should not be accepted. ◀