Trade-Offs in Distributed Interactive Proofs

Pierluigi Crescenzi
IRIF – CNRS and Université de Paris, France
piluc@irif.fr

Pierre Fraigniaud
IRIF – CNRS and Université de Paris, France
pierref@irif.fr

Ami Paz
IRIF – CNRS and Université de Paris, France
Faculty of Computer Science, University of Vienna, Austria
amipaz@irif.fr

Abstract

The study of interactive proofs in the context of distributed network computing is a novel topic, recently introduced by Kol, Oshman, and Saxena [PODC 2018]. In the spirit of sequential interactive proofs theory, we study the power of distributed interactive proofs. This is achieved via a series of results establishing trade-offs between various parameters impacting the power of interactive proofs, including the number of interactions, the certificate size, the communication complexity, and the form of randomness used. Our results also connect distributed interactive proofs with the established field of distributed verification. In general, our results contribute to providing structure to the landscape of distributed interactive proofs.

2012 ACM Subject Classification Theory of computation → Distributed computing models; Theory of computation → Interactive proof systems; Theory of computation → Distributed algorithms

Keywords and phrases Distributed interactive proofs, Distributed verification

Digital Object Identifier 10.4230/LIPIcs.DISC.2019.13

Related Version A full version of the paper is available at `https://arxiv.org/abs/1908.03363`.

Funding *Pierre Fraigniaud*: Additional support from the INRIA project GANG, and from the ANR project DESCARTES.
Ami Paz: Supported by the Fondation des Sciences Mathématiques de Paris.

Acknowledgements The authors are thankful to Gianlorenzo D'Angelo for fruitful discussions on the topic of this paper, and to Amir Yehudayoff for discussion on his work [27].

1 Introduction

This paper is concerned with distributed network computing, in which n processing nodes occupy the n vertices of a connected simple graph G, and communicate through the edges of G. In this context, *distributed decision* [25] refers to the task in which the nodes have to collectively decide whether the network G satisfies some given graph property, which may refer also to input labels given to the nodes (basic examples of such tasks are whether the network is acyclic or whether the network is properly colored). If the property is satisfied then all nodes must accept, otherwise at least one node must reject. Distributed decision finds immediate applications to distributed fault-tolerant computing, in which the nodes must check whether the current network configuration is in a legal state with respect to some Boolean predicate [22]. (If this is not the case, the rejecting node(s) may raise an alarm or launch a recovery procedure.)

33rd International Symposium on Distributed Computing (DISC 2019).
Editor: Jukka Suomela; Article No. 13; pp. 13:1–13:17
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

While some properties (e.g., whether a given coloring is proper) are locally decidable (LD) by exchanging information between neighbors only, other properties are not (e.g., whether the network is acyclic, or whether a given set of pointers forms a spanning tree of the network). As a remedy, the notion of *proof-labeling scheme* (PLS) was introduced [22], and variants were considered, including *non-deterministic local decision* (NLD) [14], and *locally checkable proofs* (LCP) [18]. All these settings assume the existence of a *prover* assigning *certificates* to the nodes, and a distributed *verifier* in charge of verifying that these certificates form a distributed proof that the network satisfies some given property. For instance, acyclicness can be certified by a prover picking one arbitrary node u, and assigning to each node v a certificate $c(v)$ equal its distance to u. The distributed verifier running at every node v checks that v has one neighbor w satisfying $c(w) = c(v) - 1$, and all its other neighbors w' satisfying $c(w') = c(v) + 1$. If the network contains a cycle, then these equalities will be violated in at least one node.

Note that the prover is not necessarily an abstract entity, as an algorithm constructing some distributed data structure (e.g., a spanning tree) may construct in parallel a proof that this data structure is correct. Interestingly, all (Turing decidable) graph properties can be certified by PLS and LCP with $O(n^2)$-bits certificates [22], and this is tight [18] – for instance, symmetry[1] (Sym) was shown to require $\Omega(n^2)$-bit certificates. However, not only such universal certification requires high space complexity at the nodes for storing large certificates, it also requires high communication complexity between neighbors for verifying the correctness of these certificates. Hence, the concern is minimizing the size of the certificates for specific graph properties, e.g., minimum-weight spanning trees (MST) [21].

Recently, the notion of *randomized* proof-labeling schemes (RPLS) was introduced [15]. RPLS assumes that the distributed verifier is randomized, and the global verdict provided by the nodes about the correctness of the network configuration should hold with probability at least $2/3$. Such randomized distributed verification schemes were proven to be very efficient in terms of communication complexity (with $O(\log n)$-bit messages exchanged between neighbors), but this often holds at the cost of actually *increasing* the size of the certificates provided by the prover.

Another recent direction for reducing the certificate size introduces a *local hierarchy* of complexity classes defined by *alternating quantifiers* (similarly to the polynomial hierarchy [28]) for local decision [11]. Interestingly, many properties requiring $\Omega(n^2)$-bit certificates with a locally checkable proof stand at the bottom levels of this hierarchy, with $O(\log n)$-bit certificates. This is for instance the case of non 3-colorability ($\overline{\mathsf{3Col}}$), which stands at the second level of the hierarchy, and Sym, which stands at the third level of the hierarchy. More generally, all monadic second order graph properties belong to this hierarchy with $O(\log n)$-bit certificates. However, it is not clear how to implement the protocols resulting from this hierarchy.

Even more recently, a very original and innovative approach was adopted [20, 24], bearing similarities with the local hierarchy but perhaps offering more algorithmic flavor. This approach considers *distributed interactive proofs*. Such proofs consist of a constant number of interactions between a centralized prover M (a.k.a. Merlin) and a randomized distributed verifier A (a.k.a. Arthur). For instance, a dAM protocol is a protocol with two interactions: Arthur queries Merlin by sending a random string, and Merlin replies to this query by sending a certificate. Similarly, a dMAM protocol involves three interactions: Merlin provides a

[1] G is symmetric if G has a non trivial automorphism, i.e., a one-to-one mapping from the set of nodes to itself preserving edges, and distinct from the identity map.

certificate to Arthur, then Arthur queries Merlin by sending a random string, and finally Merlin replies to Arthur's query by sending another certificate. This series of interactions is followed by a phase of distributed verification performed between every node and its neighbors, which may be either deterministic or randomized.

Although the interactive model seems weaker than the alternation of quantifiers in the local hierarchy, many properties requiring $\Omega(n^2)$-bits certificates with a locally checkable proof admit an Arthur-Merlin protocol with small certificates, and very few interactions. For instance, this is the case of Sym that admits a dMAM protocol with $O(\log n)$-bit certificates, and a dAM protocol with $O(n \log n)$-bit certificates [20] (on the other hand, any dAM protocol for Sym requires $\Omega(\log \log n)$-bit certificates [20]). It is also known that non symmetry ($\overline{\mathsf{Sym}}$) can be decided by a dAMAM protocol with $O(\log n)$-bit certificates [24]. These results raise several interesting questions, such as:

1. Are there ways to establish trade-offs between space complexity (i.e., the size of the certificates) and communication complexity (i.e., the size of the messages exchanged between nodes)? The dMA protocols [20] as well as the RPLS protocols [15] enable to gain a lot in terms of message complexity, but at the cost of still high space complexity. Would it be possible to compromise between these two complexities? In particular, would it be possible to reduce the certificate size at the cost of increasing the communication complexity?
2. The theory of distributed decision has somehow restricted itself to distributed randomness, in the sense that each node has only access to a private source of random coins. These coins are public to the prover, but remain private to the other nodes. Shared randomness is known to be stronger than private randomness for communication complexity, as witnessed by, e.g., deciding equality [23]. How much shared randomness could help in the context of distributed decision?
3. Last but not least, are there general reduction theorems between Arthur-Merlin classes for trading the number of interactions with the certificate size? In the centralized setting, it is known that $\mathsf{AM}[k] = \mathsf{AM}[2]$ for any $k \geq 2$, but it is not known whether a similar claim holds in the distributed setting [20]. Also, in the centralized setting the Sipser–Lautemann theorem tells us that $\mathsf{MA} \subseteq \Sigma_2 \cap \Pi_2$ and $\mathsf{MA} \subseteq \mathsf{AM} \subseteq \Pi_2$, but it is not known whether the distributed Arthur-Merlin classes stand so low in the local alternating hierarchy too.

Our Results

In this paper, we study the power of distributed interactive proofs. This is achieved via a series of results establishing trade-offs between various parameters impacting the power of interactive proofs, including the number of interactions, the certificate size, the communication complexity, and the form of randomness used. Our results also connect distributed interactive proofs with the established field of distributed verification. We address the above three questions as follows. For the first question, we show how to apply techniques developed in the framework of multi-party communication complexity to get trade-offs between space and communication for the classical triangle detection problem. For the second question, we show that shared randomness helps significantly, enabling to exponentially reduce the communication complexity while preserving the space complexity for important problems such as spanning tree, and a vast class of optimization problems, including, for example, maximum independent set and minimum dominating set. For the third question, we give a general technique for reducing the number of interactions at the cost of increasing the certificate and message size.

More specifically, for the first question, we explore the trade-off of space vs. communication, and establish that for every α there exists a Merlin-Arthur protocol for triangle-freeness, which uses $O(\log n)$ bits of shared randomness, $\widetilde{O}(n/\alpha)$-bit certificates and $\widetilde{O}(\alpha)$-bit messages between nodes (Theorem 4). To our knowledge, this is the first example of a decision task for which one can trade communication for space. In addition, the proof reveals an interesting connection between dMA and communication complexity with a referee. Note that, for $\alpha = \sqrt{n}$, we obtain a distributed Merlin-Arthur protocol for triangle-freeness with message and space complexities $\widetilde{O}(\sqrt{n})$ bits. In contrast, any proof-labeling scheme for triangle-freeness must have certificate size at least $n/e^{O(\sqrt{\log n})}$ bits (Proposition 5). A similar tradeoff can be obtained when using distributed randomness, though with higher space complexity.

Regarding the second question, we explore the significance of having access to shared randomness. We show that, for any minimization problem π in graphs whose admissibility can be decided locally, there exists a Merlin-Arthur protocol for certifying the existence of a solution whose cost is at most k, using $O(\log n)$ bits of shared randomness, with $O(\log n)$-bit certificates and $O(\log\log n)$-bit messages between nodes (see Theorem 6). The same result holds for maximization problems whose admissibility can be decided locally. Note that this class of problems includes, for example, maximum independent set, minimum dominating set, and minimum vertex cover (potentially weighted). This exponentially improves the communication complexity of locally checkable proofs for such problems. The same communication complexity could be obtained using randomized proof-labeling schemes, but at the cost of increasing the certificate size to up to $O(n \log n)$ bits. As another interesting result in the context of exploring the significance of having access to shared randomness, we show that even shared randomness remains limited both in terms of the certificate size and of the amount of communication. We show that every Arthur-Merlin protocol for Sym, and every Arthur-Merlin protocol for $\overline{\mathsf{Sym}}$ must have both certificate size and message size $\Omega(\log\log n)$ bits, even with shared randomness (see Theorem 10). Interestingly, for the class of graphs used in the proof of this latter result, there is a Merlin-Arthur protocol with certificates and messages of constant length. This shows that the inclusion $\mathsf{MA} \subseteq \mathsf{AM}$ which holds in the centralized setting does not hold in the distributed setting.

Finally, we consider general reductions within the Arthur-Merlin hierarchy, and compare the power of this hierarchy to the power of proof-labeling schemes with certificates of linear size. We show that, for every σ and γ, any graph property verifiable with an Arthur-Merlin protocol with 3 or 4 interactions (dMAM or dAMAM) using σ-bit certificates and γ-bit messages can also be verified by an Arthur-Merlin (dAM) protocol using $O(n\sigma^2)$-bit certificates and $O(n\gamma\sigma)$-bit messages (see Theorem 11 and Corollary 12). Although the linear blowup in terms of both certificate size and message complexity may seem huge at a first glance, it fits (up to logarithmic factors) with the different results obtained previously [20, 24] regarding Sym and graph non-isomorphism ($\overline{\mathsf{Iso}}$). Finally, we compare the power of Arthur-Merlin protocols with an arbitrarily large number of interactions with the power of proof-labeling schemes. We show that there exists a graph property admitting a proof-labeling scheme with certificates and messages on $O(n)$ bits, that cannot be solved by an Arthur-Merlin protocol with $o(n)$-bit certificates, for any fixed number $k \geq 0$ of interactions between Arthur and Merlin, even using shared randomness, and even with messages of unbounded size (see Theorem 14). This latter result demonstrates that, in general, one cannot trade the number of interactions between Merlin and Arthur for reducing the certificate size, at least for certificates of linear size.

Most of our results are stated by assuming that nodes have access to shared randomness. However, as all our protocols are local, all our 1-round protocols can be simulated by 2-round protocols using distributed randomness. In general, our results contribute to providing structure to the landscape of distributed interactive proofs.

Related Work

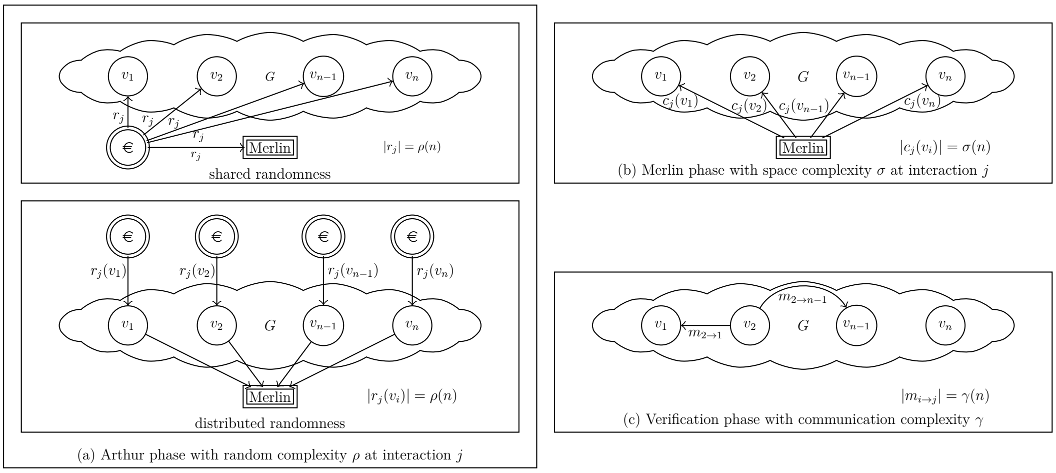
Local decision (LD) and the central notion of locally-checkable labellings were introduced and thoroughly studied in the 90s [25]. Local verification was introduced fifteen years later [22], through the original notion of proof-labeling schemes (PLS). Proof-labeling schemes find important applications to self-stabilization, but are subject to some restrictions (only certificates are exchanged between neighbors). These restrictions were lifted by considering the general notion of locally checkable proofs (LCP) [18]. By definition, we have $\mathsf{LCP} = \Sigma_1\mathsf{LD}$ (the same way $\mathsf{NP} = \Sigma_1\mathsf{P}$). A third notion of distributed verification was introduced, by considering the class NLD [14]. NLD differs from $\Sigma_1\mathsf{LD}$ in the fact that, in NLD, the certificates cannot depend on the identities given to the nodes.

Randomized versions of local decision and local verification have been considered in the literature [15, 13, 14, 20]. A Merlin-Arthur (dMA) protocol is actually a randomized version of locally checkable proof ($\Sigma_1\mathsf{LD}$) that was previously studied [15]: Merlin provides each node with a certificate, and Arthur performs a randomized verification algorithm at each node. The benefit of using dMA over $\Sigma_1\mathsf{LD}$ can be exponential in terms of communication complexity (i.e., of the size of the messages exchanged between neighbors), at the cost of a linear increase in space complexity (i.e., of the size of the certificates provided by Merlin) [15].

Very closely related to the line of work about interactive distributed proofs is the local hierarchy $\mathsf{LH} = \bigcup_{k\geq 0}\left(\Sigma_k\mathsf{LD} \cup \Pi_k\mathsf{LD}\right)$ with certificates and messages of logarithmic size [11], extending the known logLCP class [18]. In particular, it is proved that $\mathsf{Sym} \in \Sigma_3\mathsf{LD}$ [11]. Also, it is easy to show that $\overline{\mathsf{3Col}} \in \Pi_2\mathsf{LD}$. In contrast, placing Sym or $\overline{\mathsf{3Col}}$ in $\Sigma_1\mathsf{LD}$ requires $\Omega(n^2)$-bit certificates [18]. The same hierarchy was later considered, but under the constraint that the certificates must not depend on the identifier assignment to the nodes. With $O(n^2)$-bit certificates, this hierarchy collapses at the second level $\Pi_2\mathsf{LD}$ [5]. (This is in contrast with the hierarchy in which certificates can depend on the node identifiers, which collapses at the first level $\Sigma_1\mathsf{LD}$ with $O(n^2)$-bit certificates.) Nevertheless, apart from the bottom levels, the hierarchies based on alternating quantifiers with $O(\log n)$-bit certificates are essentially the same [5]. See the recent survey [10] for more results on distributed decision.

This work is inspired by very recent achievements in the field of distributed interactive protocols [20, 24]. In addition to the aforementioned results regarding Sym and $\overline{\mathsf{Sym}}$, two versions of $\overline{\mathsf{Iso}}$ (Given two sub-graphs G_1 and G_2 of G, $G_1 \not\sim G_2$, that is, are G_1 and G_2 non-isomorphic?) were considered. For the easiest version, in which the input of each node v is formed by the two sets of neighbors of v in the two graphs G_1 and G_2, a dAMAM protocol with $O(\log n)$-bit certificates for deciding $G_1 \not\sim G_2$ was designed [24]. For the more complicated version, in which $G_1 = G$ (that is, G_1 is the communication network) and the input of each node v is the set of neighbors of v in G_2, a dAMAM protocol with $O(n \log n)$-bit certificates for deciding $G_1 \not\sim G_2$ was proposed [20], and an Arthur-Merlin protocol with a constant number of interactions and $O(\log n)$-bit certificates, for deciding $G_1 \not\sim G_2$, was successively designed [24]. Interestingly, this latter result is obtained via a general connection between efficient centralized computation (under various models) and the ability to design Arthur-Merlin protocols with a constant number of interactions between the prover and the verifier, using logarithmic-size certificates.

We use a variety of techniques and results from the theory of *communication complexity* [23]. Specifically, we use an Arthur-Merlin style protocol for two-party disjointness [1], in a recent variant [2] that allows a trade-off between communication complexity and certificate size. We also use recent lower bounds for the equality and non-equality problems in the same setting [17]. Finally, we use multi-party communication protocol with a referee for the SumZero problem with bounded inputs [26], and with unbounded inputs [19].



(a) Arthur phase with random complexity ρ at interaction j

(b) Merlin phase with space complexity σ at interaction j

(c) Verification phase with communication complexity γ

Figure 1 The three different phases of a distributed Arthur-Merlin protocol (the Arthur phase can make use of either shared or distributed randomness).

2 Model and Definitions

A *network configuration* is a triple (G, id, x) where $G = (V, E)$ is a connected simple graph, $\mathrm{id} : V \to \{1, \dots, n^c\}$ for some constant $c \geq 1$ is the *identity* one-to-one assignment to the nodes, and $x : V \to \{0,1\}^*$ is the *input label* assignment (i.e., the state of the node). A *distributed language* is a collection $\mathcal{L}$ of network configurations. Note that it may be the case that, for some language $\mathcal{L}$, $(G, \mathrm{id}, x) \in \mathcal{L}$ while $(G, \mathrm{id}', x) \notin \mathcal{L}$ for two different identity assignments id and id$'$. This typically occurs for languages where the label of a node refers to the identities of its neighbors, e.g., for encoding spanning trees. (Throughout the paper, we assume that all considered distributed languages are Turing-decidable). *Distributed decision* for $\mathcal{L}$ is the following task: given any network configuration (G, id, x), the nodes of G must collectively decide whether $(G, \mathrm{id}, x) \in \mathcal{L}$. If this is the case, then *all* nodes must accept, otherwise *at least one node* must reject (with certain probabilities, depending on the model).

We consider *interactive protocols* for distributed decision [20]. A distributed interactive protocol $\mathcal{P}$ consists of a constant series of interactions between a *prover* called *Merlin*, and a *verifier* called *Arthur* (see Fig. 1 for a visual representation of such a protocol). The prover Merlin is centralized, and has unlimited computing power. It is aware of the whole network configuration (G, id, x) under consideration, but it cannot be trusted. The verifier Arthur is distributed, and has bounded knowledge, that is, at each node v, Arthur is initially aware solely of $(\mathrm{id}(v), x(v))$, i.e., of its identity and its input label.

In a distributed *Arthur-Merlin* interactive protocol performed on $\mathcal{I} = (G, \mathrm{id}, x)$, whenever Arthur is the one that starts interacting, it picks a random string $r_1(v)$ at each node v of G (this random string might be private to each node, or the nodes may have access to shared randomness). Given the collection r_1 of random strings selected by the nodes, Merlin provides every node v with a *certificate* $c_1(v) = \mathsf{p}(v, \mathcal{I}, r_1)$, where $\mathsf{p} : \{0,1\}^* \to \{0,1\}^*$. At this point, Arthur picks another random string $r_2(v)$ at each node v. Then Merlin replies to each node v by sending a second certificate $c_2(v) = \mathsf{p}(v, \mathcal{I}, r_1, r_2)$, and so on. Whenever Merlin is the one that starts interacting, the process starts with Merlin constructing a binary string $c_0(v) = \mathsf{p}(v, \mathcal{I})$ that it sends to every node v. These interactions proceed for a constant number

$k \geq 0$ of times, and Merlin interacts last, by sending $c_{\lfloor \frac{k}{2} \rfloor}(v)$ to every node v. A sequence of interactions can then be summarized by a *transcript* $\pi(\mathcal{I}, \mathsf{p}, r) = \Big(\pi_v(\mathcal{I}, \mathsf{p}, r)\Big)_{v \in V}$, where $r = \Big(r_1, \ldots, r_{\lfloor \frac{k}{2} \rfloor}\Big)$ with $r_i = (r_i(v))_{v \in V}$ for $i = 1, \ldots, \lfloor \frac{k}{2} \rfloor$, and

$$\pi_v(\mathcal{I}, \mathsf{p}, r) = \Big(c_0(v), r_1(v), c_1(v), \ldots, r_{\lfloor \frac{k}{2} \rfloor}(v), c_{\lfloor \frac{k}{2} \rfloor}(v)\Big)$$

with $c_0(v) = \varnothing$ if Arthur starts the interactions. In other words, an Arthur-Merlin protocol with k interactions results in a transcript with $c_0(v) = \varnothing$ if k is even, and with $c_0(v)$ equal to the first certificate provided by Merlin otherwise. The Arthur-Merlin protocol completes by performing a *deterministic* distributed verification algorithm v executed at each node. Specifically, Algorithm v proceeds as follows at every node v:

1. A message $M_{v,u}$ destined for every neighboring node u of v is forged, and sent to u. This message may depend on the identity $\mathrm{id}(v)$, the input $x(v)$, all random strings generated by Arthur at v, and all certificates received by v from Merlin.
2. Based on all the knowledge accumulated by v (i.e., its identity, its input label, the generated random strings, the certificates received from Merlin, and all the messages received from its neighbors), Algorithm v accepts or rejects at node v.

A distributed Arthur-Merlin protocol $\mathcal{P}$ thus consists of two consecutive stages: (1) interactions between the nodes and the prover p (Arthur-Merlin phases), and (2) communication among neighboring nodes (algorithm v). Note that, for the sake of simplifying the presentation, and unifying the comparison with previous work, we restrict ourselves to verification algorithms v that perform in a single round. Performing more than one round enables to improve the complexity of verification protocols in some cases [12]. However, this does not conceptually change the nature of the protocol. For zero interactions (i.e., $k = 0$), a distributed Arthur-Merlin protocol simply consists in performing a (deterministic) decision algorithm at each node [22]. For one interaction (i.e., $k = 1$), a distributed Arthur-Merlin verification protocol is a (1-round) locally-checkable proof algorithm [18].

▶ **Definition 1.** *The class* $\mathsf{dAM}[k](\sigma, \gamma)$ *is the class of languages* $\mathcal{L}$ *for which there exists a distributed Arthur-Merlin verification protocol with at most* $k \geq 0$ *interactions between Arthur and Merlin, where Merlin provides certificates of at most* $\sigma \geq 0$ *bits to the nodes, and the verification algorithm* v *exchanges messages of at most* $\gamma \geq 0$ *bits between nodes, such that, for every configuration* $\mathcal{I} = (G, \mathrm{id}, x)$*,*

$$\begin{cases} (G, \mathrm{id}, x) \in \mathcal{L} & \Rightarrow & \exists \mathsf{p} : \Pr_r[\mathsf{v}(\pi(\mathcal{I}, \mathsf{p}, r)) \text{ accepts at all nodes}] \geq 2/3; \\ (G, \mathrm{id}, x) \notin \mathcal{L} & \Rightarrow & \forall \mathsf{p} : \Pr_r[\mathsf{v}(\pi(\mathcal{I}, \mathsf{p}, r)) \text{ rejects in at least one node}] \geq 2/3. \end{cases}$$

The definition of distributed *Merlin-Arthur* interactive protocols, and of $\mathsf{dMA}[k](\sigma, \gamma)$ is similar, apart from the fact that, as opposed to Arthur-Merlin protocols in which Merlin always interacts last, Arthur has one more "interaction" during which it picks a random bit-string $r'(v)$ at every node v, which is used to perform a *randomized* verification algorithm v. Therefore, for $k \geq 1$, a Merlin-Arthur protocol with k interactions can be defined as an Arthur-Merlin protocol with $k - 1$ interactions, but where the verification algorithm v is randomized. For zero interactions, a distributed Merlin-Arthur protocol simply consists in performing a (deterministic) decision algorithm at each node [22]. For one interaction, a distributed Merlin-Arthur protocol is a (1-round) randomized decision algorithm as studied previously [13]. For two interactions, a distributed Merlin-Arthur protocol is a (1-round) randomized locally-checkable proof algorithm, also as studied previously [15].

In the following, we may avoid mentioning the parameters σ and γ when they are clear from the context, or when they are respectively identical in the two terms of an equality. For small values of $k \geq 2$, $\mathsf{dAM}[k]$ and $\mathsf{dMA}[k]$ are rewritten as an alternating sequence of As and Ms. For instance, $\mathsf{dAM}[2] = \mathsf{dAM}$, $\mathsf{dMA}[2] = \mathsf{dMA}$, $\mathsf{dAM}[3] = \mathsf{dMAM}$, $\mathsf{dMA}[3] = \mathsf{dAMA}$, and $\mathsf{dAM}[4] = \mathsf{dAMAM}$, and so on. For $k \leq 1$, it follows from the definition that $\mathsf{dAM}[0] = \mathsf{dMA}[0] = \mathsf{LD}$. We also have $\mathsf{dAM}[1] = \Sigma_1\mathsf{LD}$ and $\mathsf{dMA}[1] = \mathsf{BPLD}(2/3, 2/3)$ where the class $\mathsf{BPLD}(p, q)$ is the distributed version of BPP [16], with p being the acceptance probability of the interactive protocol on legal instances, and q being the rejection probability of the protocol on illegal instances [13]. As a last example, dMA is the class of languages that can be decided by a randomized locally checkable proof, as studied previously [15].

As opposed to the sequential setting in which it is known that $\mathsf{AM}[k] = \mathsf{AM}[2]$ for all $k \geq 2$, it is not known whether such "collapse" occurs in the distributed setting. Therefore, we define the Arthur-Merlin *hierarchy* as $\mathsf{dAMH}(\sigma, \gamma) = \bigcup_{k \geq 0} \mathsf{dAM}[k](\sigma, \gamma)$. That is, $\mathcal{L} \in \mathsf{dAMH}(\sigma, \gamma)$ if and only if there exists $k \geq 0$ such that $\mathcal{L} \in \mathsf{dAM}[k](\sigma, \gamma)$.

Boosting the Success Probability

In classical, sequential randomized algorithm, the success probability constant $2/3$ can be easily increased using repetitions. On the other hand, it was shown that this boosting technique is not applicable for randomized distributed decision algorithms in general [13], making the choice of constant significant when considering such settings. The inability of boosting in the distributed setting is due to the fact that, when repeating the algorithm on a "no" instance for several times, different nodes may reject in different repetitions, causing each node to sees very few rejections and decide on acceptance. Somewhat surprisingly, we can show that in the case of distributed Arthur-Merlin protocols (i.e., $\mathsf{dAM}[k]$ classes), parallel repetition is possible, and at a relatively low blowup in communication and certificates. This allows us to boost the success probability, as follows.

▶ **Proposition 2.** *Let $1 > p' > p > 1/2$. If there exists an Arthur-Merlin verification protocol $\mathcal{P}$ with $k \geq 2$ interactions that enables to verify a distributed language $\mathcal{L}$ with σ-bit certificates, γ-bit messages, and success probability p, then there exists an Arthur-Merlin verification protocol $\mathcal{P}'$ with k interactions that enables to verify $\mathcal{L}$ with $\sigma + O(\log n)$-bit certificates, messages on $\gamma + O(\log n)$ bits, and success probability p'.*

Proof. Moving from success probability p to success probability $p' > p$ is achieved in a standard way, by merely repeating $\mathcal{P}$ a constant number of times (depending on p and p'), and adopting the majority of the outcomes. However, this cannot be done in a straightforward manner because, for a configuration $(G, x, \mathrm{id}) \notin \mathcal{L}$, it may be the case that the (at least one) node rejecting (G, x, id) is different at each repetition. Therefore, during the last interaction with the prover, Merlin provides every node with a local encoding of a spanning tree T enabling to count the number of executions of $\mathcal{P}$ resulting in at least one node rejecting. It is known that certificates of $O(\log n)$ bits suffice for certifying such a tree [22]. The root of the tree T accepts or rejects depending on whether the majority of executions of $\mathcal{P}$ accepted or rejected, respectively. ◀

3 Space vs. Communication

In this section, we study the trade-off between space and communication complexity for Merlin-Arthur interactive protocols. Specifically, we consider the classical triangle-freeness problem, and establish a trade-off between space and communication for this problem. Recall that a graph $G = (V, E)$ is triangle-free if, for every three nodes u, v, w in V, either $\{u, v\} \notin E$, $\{u, w\} \notin E$, or $\{v, w\} \notin E$. We denote by Δ_{free} the corresponding distributed language. There is a recent deterministic distributed algorithm for triangle-freeness running in $\widetilde{O}(\sqrt{n})$ rounds in the CONGEST model [7]. A general scheme for designing dMA protocol has also been proposed [15]. This scheme enables to reduce communication complexity to $O(\log n)$, at the cost of increasing the space complexity to $O(n^2)$. When more interactions are allowed, say a constant number k, a recent reduction [24] – from centralized small-space algorithms to our setting – implies a protocol with $O(\log n)$-bit certificates and $O(\log n)$-bit messages for the triangle-freeness problem, i.e., $\Delta_{\mathsf{free}} \in \mathsf{dMA}[k](\log n,\ \log n)$ for some constant $k > 1$.

In order to prove the trade-off between space and communication complexities for triangle-freeness, we first state the following result, which will be used at several places in the paper.

▶ **Lemma 3.** *For any network of maximum degree d, there exists a proof-labeling scheme (and thus a $\Sigma_1\mathsf{LD}$ protocol) with $O(\log n)$-bit certificates providing each node v with the certified value n of the number of nodes, and a color $c(v) \in \{1, \dots, \min\{d^2 + 1, n\}\}$ such that c forms a certified proper distance-2 coloring of the network.*

Proof idea. The certification of the number of nodes can be done by using a rooted spanning tree and by counting nodes in the sub-trees. The certification of a proper distance-2 coloring can be done by assigning colors to nodes, with every node checking that all its neighbors have different colors, all different from its own color. ◀

Since triangle-freeness is a local property, nodes do not need to be represented by identifiers that are different throughout the entire network. Instead, identifiers resulting from a proper distance-2 coloring suffice. Therefore, using Lemma 3, we can assume that nodes are provided with identifiers in $\{1, \dots, n\}$ such that $\mathrm{id}(u) \neq \mathrm{id}(v)$ whenever the distance between u and v is at most 2.

▶ **Theorem 4.** *For every $\alpha = O(n)$, there exists a Merlin-Arthur protocol for triangle-freeness, using $O(\log n)$ bits of shared randomness, with $O(\frac{n}{\alpha} \log n)$-bit certificates and $O(\alpha \log n)$-bit messages between nodes. In short $\Delta_{\mathsf{free}} \in \mathsf{dMA}(\frac{n}{\alpha} \log n,\ \alpha \log n)$.*

Proof. We identify the space $\{1, \dots, n\}$ of IDs with $[n/\alpha] \times [\alpha]$, for some $\alpha = O(n)$ of choice. Each node u thus has a set S_u of pairs of the form (i, t) representing its neighbors. Let q be a prime such that $cn\alpha < q \leq 2cn\alpha$, for a large enough constant $c > 1$, and let $\mathbb{F}_q$ be the field of q elements. Each node u represents S_u as α functions $\psi_{S_u,t} : [n/\alpha] \to \{0, 1\}$, where $\psi_{S_u,t}(i) = 1 \iff (i, t) \in S_u$. Node u then extends these functions to polynomials $\Psi_{S_u,t} : \mathbb{F}_q \to \mathbb{F}_q$ of degree at most $n/\alpha - 1$ that agree with $\psi_{S_u,t}$ on $[n/\alpha]$. To make sure that an edge $\{u, v\}$ is not a part of a triangle, the nodes u and v need to verify that $S_u \cap S_v = \emptyset$, which is equivalent to $\Psi_{S_u,t}(i) \cdot \Psi_{S_v,t}(i) = 0$ for all $i \in [n/\alpha]$ and $t \in [\alpha]$. Node u then defines its *neighbors polynomials* $\Psi_{uv,t} = \Psi_{S_u,t} \cdot \Psi_{S_v,t}$ for every $v \in S_u$, and every $t \in [\alpha]$. Let $\Psi_u = \sum_{t \in [\alpha]} \sum_{v \in S_u} \Psi_{uv,t}$. The degree of each polynomial $\Psi_{uv,t}$ is at most $2(n/\alpha - 1)$, and thus this is also the case for the degree of Ψ_u. Node u is not part of a triangle if and only if $\Psi_{uv,t}(i) = 0$ for every $t \in [\alpha]$, $i \in [n/\alpha]$ and $v \in S_u$. Since $q > n\alpha$, it follows that u is not part of a triangle if and only if $\Psi_u(i) = 0$ for every $i \in [n/\alpha]$. (For each i, we have a sum of $n\alpha$ values, each in $\{0, 1\}$.)

Merlin assigns to node u the certificate Φ_u, which is supposed to be equal to Ψ_u. Since this is a polynomial of degree at most $2(\frac{n}{\alpha} - 1)$, the same number of coefficients are sufficient for representing Φ_u. Therefore, the certificates are of $O(\frac{n}{\alpha} \log q)$ bits, which are actually $O(\frac{n}{\alpha} \log n)$ bits, as $q \leq 2cn\alpha = O(n^2)$.

Each node u first verifies that $\Phi_u(i) = 0$ for every $i \in [n/\alpha]$. Then, it checks that indeed $\Phi_u = \Psi_u$, as follows. The protocol uses the shared randomness to choose a field element $i_0 \in \mathbb{F}_q$ known to all nodes. Each node v broadcasts $\{\Psi_{S_v,t}(i_0) : t \in [\alpha]\}$ to each of its neighbors, using $O(\alpha \log q) \leq O(\alpha \log n)$ bits of communication. Node u then computes

$$\Psi_u(i_0) = \sum_{t\in[\alpha]} \sum_{v\in S_u} \Psi_{uv,t}(i_0) = \sum_{t\in[\alpha]} \sum_{v\in S_u} \Psi_{S_u,t}(i_0) \cdot \Psi_{S_v,t}(i_0)$$

and accepts only if $\Phi_u(i_0) = \Psi_u(i_0)$. The probability that two non-equal polynomials on $\mathbb{F}_q$ of degree at most $2(\frac{n}{\alpha} - 1)$ are equal at a random point i is at most $2(\frac{n}{\alpha} - 1)/q$. Since $q > cn\alpha$, the probability of error can be made arbitrarily small by choosing c large enough. ◀

▶ Remark. Similar trade-offs can still be obtained even if nodes have only access to distributed randomness. For instance, in two rounds, with the same notations as in the proof of Theorem 4, we can have each node u choose its own random $i_u \in \mathbb{F}_q$, and send it to all its neighbors v. To get a 1-round dMA protocol, with $O(\frac{n^2}{\alpha} \log n)$-bit certificates, and $O(\alpha \log n)$-bit communication, Merlin sends to u a specific certificate for each edge incident to u. That is, u gets a polynomial Φ_v for each $v \in S_u$, which equals (allegedly) to $\Psi_{uv}(i) = \sum_{t\in T} \Psi_{uv,t}(i) = \sum_{t\in T} \Psi_{S_u,t}(i) \cdot \Psi_{S_v,t}(i)$ on each $i \in \mathbb{F}_q$. In this case, v chooses i_v at random locally, and sends to u the value i_v in addition to the α evaluations $\Psi_{S_v,t}(i_v)$ for all $t \in [\alpha]$.

A particular application of Theorem 4 is the existence of a Merlin-Arthur protocol with both space and message complexities $\widetilde{O}(\sqrt{n})$. This contrasts with the following lower bound.

▶ **Proposition 5.** *Any proof-labeling scheme for triangle-freeness must have certificate size at least $n/e^{O(\sqrt{\log n})}$ bits.*

Proof idea. The lower bound graph construction for the BROADCAST-CONGESTED-CLIQUE model [9] obviously gives a lower bound to the weaker, BROADCAST-CONGEST model. This lower bound is based on a lower bound for multiparty communication complexity of disjointness [27], which also applies for the non-deterministic case. Finally, as noted in previous work on PLS [18, 6], a lower bound for non-deterministic communication complexity in the BROADCAST-CONGEST model implies a certificate-size lower bound for PLS. ◀

We can then conclude that, as opposed to the dMA protocol of Theorem 4, any PLS for triangle freeness must use almost-linear communication. Put differently, the trivial protocol of sending all the list of neighbors is almost optimal, even if non-determinism is used.

4 Distributed vs. Shared Randomness

In this section we compare the power of distributed interactive protocols using *shared* randomness (the nodes have access to a common source of random coins) with the power of protocols using *distributed* randomness (each node has access to a private source of random coins only) – in both cases, the outcomes of the random trials are public to Merlin.

Certifying Solutions to Optimization Problems

We consider optimization problems on graphs, such as finding a minimum dominating set, or a maximum independent set, and their weighted counterparts. Similar problems where previous studied in the context of non-interactive distributed verification [11]. In such a problem π, an admissible solution is a set S of nodes satisfying a set of constraints depending on π, and the quality of a solution S is measured by its weight $w(S) = \sum_{s\in S} w(s)$ where $w(s)$

is the weight of node s, given as input (where $w(s) = 1$ for every node s when considering only the cardinality of the solution). We assume that all weights are polynomial in the size n of the network. A set S is distributively encoded by a Boolean variable $x(v)$ at each node v, indicating whether the node is in S or not. We consider two distributed languages:

- The language Adm_π is composed of all configurations $(G, (w, x), \mathrm{id})$ such that x encodes an *admissible* solution for π in the weighted graph G (weights are assigned by w).
- The language $\mathsf{OptVal}_{\pi,k}$, for $k \geq 0$, is composed of all configurations (G, w, id) such that there exists an admissible solution for π of weight at most k (respectively, at least k) for the minimization (respectively, maximization) problem π.

This framework can easily be extended to study problems whose solutions are sets of edges.

▶ **Theorem 6.** *For any optimization problem π on graphs such that checking whether a given solution x is admissible can done by exchanging $O(\log\log n)$ bits between neighbors, there exists a Merlin-Arthur protocol for $\mathsf{OptVal}_{\pi,k}$, using $O(\log n)$ bits of shared randomness, with $O(\log n)$-bit certificates and $O(\log\log n)$-bit messages between nodes. In short, $\mathsf{OptVal}_{\pi,k} \in \mathsf{dMA}(\log n,\ \log\log n)$.*

Proof idea. A trivial starting point for protocols solving π is by having Merlin mark the nodes of the solution using the certificates. Computing the weight of the given solutions is a global task, and thus much harder in the distributed setting. To solve it, we aggregate the solution weight over a spanning tree. However, this still requires larger messages than desired; the crucial part in the proof is in using multi-party communication complexity protocol with a referee for the SumZero problem. Using this protocol, we can reduce the messages to their desired size. ◀

For the statement of Theorem 6, we assumed that all weights are polynomial in the size n of the network. If the weights are m-bit long, we can adapt the proof, and show that there exists a Merlin-Arthur protocol for $\mathsf{OptVal}_{\pi,k}$, using $O(\log(m + \log n))$ bits of shared randomness, with $O(\log n)$-bit certificates and $O(\log n)$-bit messages between nodes. In short, $\mathsf{OptVal}_{\pi,k} \in \mathsf{dMA}(\log n,\ \log n)$ even with weights exponential in n.

Certifying Coloring and Lucky Labeling

Similar arguments as the ones used to establish Theorem 6 allow us to verify specific optimization problems, for which checking that a solution is admissible is not easy. We exemplify this with the coloring problem, and its variant the *lucky labeling* problem [4, 8].

Checking that a given graph coloring is proper is a simple task, which can be solved by having each node broadcast its color. Here, we show that verifying a given c-coloring can be done using a dMA protocol with $O(\Delta \log\log c)$-bit certificates and $O(1)$ bits of communication in networks of maximum degree Δ. This stands in contrast to the trivial verification algorithm where the communication is of $O(\log c)$ bits. In the dMA protocol, Merlin provides every node v with the location $p(v, u)$ of a bit where the colors of u and v differ, for each of its neighbors u. A node v then checks with each neighbor u the fact that $p(v, u) = p(u, v)$, and at the same time sends to u the value of the corresponding bit. The Merlin step requires space $O(\Delta \log\log c)$. The Arthur step requires constant communication when shared randomness is available using the equality protocol.

▶ **Lemma 7.** *There is a Merlin-Arthur protocol for verifying a given c-coloring using $O(\log\log c)$ bits of shared randomness, certificates of size $O(\log\log c)$ bits, and constant communication complexity.*

We apply this remark to the so-called *lucky labeling.* Let $\ell : V \to \{1, \dots, c\}$, and, for every node v of G, let $S(v) = \sum_{u \in N(v)} \ell(u)$. The labeling ℓ is *lucky* if, for every two adjacent nodes u and v, we have $S(u) \neq S(v)$. The lucky coloring number of a graph G, denoted by $\eta(G)$, is the least positive integer c such that G has a lucky labeling $\ell : V \to \{1, \dots, c\}$. We refer to [4, 8] for properties of the lucky coloring number. In particular, it is conjectured that $\eta(G) \le \chi(G)$, and it is known that $\eta(G) \le \Delta^2 - \Delta + 1$, even for list lucky labeling.

Verifying a given graph coloring is trivial, even without labels or interaction. To verify an upper bound on the chromatic number χ of a graph, there is a simple PLS giving each node a color. The situation with lucky labeling is much more subtle: it is impossible to verify lucky labeling in a single round (this can be easily seen by considering different labelings on a short path). There is a simple PLS for verifying a given labeling is lucky, or for bounding η from above, which gives each node v the sum $S(v)$, and also labels for the latter case.

This PLS has label size and communication of $O(\log \Delta)$. Applying RPLS [15] gives $\sigma = O(\Delta \log \Delta)$ and $\gamma = O(\log \log \Delta)$, which can be reduced to $\gamma = O(1)$ using shared randomness. Here, we show an MA protocol using shared randomness with $\sigma = O(\Delta \log \log \Delta)$ and $\gamma = O(\log \log \Delta)$, establishing another trade-off between space and communication.

▶ **Theorem 8.** *For every $\lambda = O(n)$, there exists a Merlin-Arthur protocol for $\eta(G) \le \lambda$, using $O(\log \log \Delta)$ bits of shared randomness, with $O(\Delta \log \log \Delta)$-bit certificates and $O(\log \log \Delta)$-bit messages between nodes. In short, lucky labeling is in* $\mathsf{dMA}(\Delta \log \log \Delta,\ \log \log \Delta)$.

Proof. Merlin sends to every node v its label $\ell(v)$ and the alleged sum $S(v)$ of the labels of its neighbors. For the verification, the nodes verify that the sums $S(v)$ constitute a proper coloring, using the protocol from Lemma 7. In addition, they use a multiparty protocol for the SumZero problem [26, 3], in order to verify $S(v) = \sum_{u \sim v} \ell(u)$. ◀

A similar protocol can be used for the problem of verifying that a given labeling is lucky.

A General Reduction Between Distributed and Shared Randomness

Interestingly, assuming distributed randomness does not limit the power of Arthur-Merlin protocols compared to *shared* randomness, up to a small additive factor in the certificate size. The same holds for Merlin-Arthur protocols, but solely up to one additional interaction between Arthur and Merlin. This result is not hard to achieve using a classical spanning-tree verification technique already applied in proof labeling schemes [22]. Yet, it both generalizes and simplifies the previous results on shared vs. distributed randomness [20].

▶ **Theorem 9.** *For any distributed language $\mathcal{L}$, and for any number $k \ge 1$ of interactions, and for any certificate size $\sigma \ge 0$, if $\mathcal{L} \in \mathsf{dAM}[k](\sigma, \gamma)$ (respectively, $\mathcal{L} \in \mathsf{dMA}[k](\sigma, \gamma)$) using $\rho(n)$ shared random bits, then $\mathcal{L} \in \mathsf{dAM}[k](\sigma + \log n + \rho, \gamma + \log n + \rho)$ (respectively, $\mathcal{L} \in \mathsf{dAM}[k+1](\sigma + \log n + \rho, \gamma + \log n + \rho)$) with distributed randomness.*

Proof idea. We simulate the shared randomness protocol by using the randomness of a single node as the shared randomness. Merlin disseminates the randomness to all nodes, and a spanning tree is used to verify that the disseminated random string is the desired one. ◀

Lower bound for shared randomness

For many verification problems, the number of random bits used by Arthur remains limited, typically $\rho(n) = O(\log n)$, which shows that, often, shared randomness does not add much power to Arthur-Merlin protocols. The next result states a lower bound on the certificate and message size in the case of Sym and $\overline{\mathsf{Sym}}$, when using shared randomness.

▶ **Theorem 10.** *Any Arthur-Merlin protocol for (non) symmetry must have certificate and message size* $\Omega(\log\log n)$*. In short,* $\mathsf{Sym}, \overline{\mathsf{Sym}} \notin \mathsf{dAM}(o(\log\log n), \infty) \cup \mathsf{dAM}(\infty, o(\log\log n))$*, even using shared randomness.*

Proof. In [17], a communication complexity variant of Arthur-Merlin protocols has been proposed. In this variant, Arthur consists of two parties, Alice and Bob, and the input is split between them: Alice holds x, Bob holds y, and they wish to decide whether the value of a specified function f with input x and y is equal to 1. At the beginning, Alice and Bob start by tossing some coins, then Merlin publishes a certificate, and finally Alice and Bob separately decide whether to accept (the acceptance/rejection criteria are the same as for the Arthur-Merlin protocols). The communication cost of the protocol is defined as the worst-case length of Merlin's certificates. At the end of the paper, the authors observe that, with respect to this variant of Arthur-Merlin protocols, any such protocol for Eq and for $\overline{\mathsf{Eq}}$ must have communication complexity $\Omega(\log\log n)$. We will now show that the existence of a dAM protocol with one interaction for the Sym (respectively, $\overline{\mathsf{Sym}}$) problem with certificate size $o(\log\log n)$ would imply a two-player Arthur-Merlin protocol for Eq (respectively, $\overline{\mathsf{Eq}}$) with communication complexity $o(\log\log n)$. The theorem thus follows.

Given two binary vectors $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$, recall that $\mathsf{Eq}(x, y) = 1$ if and only if $x_i = y_i$ for every i with $1 \leq i \leq n$ (for the sake of simplicity, we will assume that $x \neq \mathbf{0}$ and that $y \neq \mathbf{0}$, but our construction can be also adapted to the case in which $x = \mathbf{0}$ or $y = \mathbf{0}$). We now define a graph $G_{x,y}$ such that $\mathsf{Sym}(G_{x,y}) = 1$ if and only if $\mathsf{Eq}(x, y) = 1$ (and, hence, $\overline{\mathsf{Sym}}(G_{x,y}) = 1$ if and only if $\overline{\mathsf{Eq}}(x, y) = 1$). The graph includes $6n + 2$ nodes a, b, a_i, b_i, u_i, v_i, y_i, and z_i, for $1 \leq i \leq n$, and the following edges (see Fig. 2):

- (a, b), (a, a_i), for $1 \leq i \leq n$ such that $x_i = 1$, and (b, b_i), for $1 \leq i \leq n$ such that $y_i = 1$;
- (u_i, a_j) and (v_i, b_j), for $1 \leq i \leq j \leq n$;
- (u_i, u_j) and (v_i, v_j), for $1 \leq i < j \leq n$, and (u_i, y_j) and (v_i, z_j), for $1 \leq i, j \leq n$;
- (y_i, y_{i+1}) and (z_i, z_{i+1}), for $1 \leq i < n$, and (y_1, y_n) and (z_1, z_n).

Clearly, if $x = y$, then $\mathsf{Sym}(G_{x,y}) = 1$: indeed, we can simply map each a-node (respectively, u-node and y-node) to the corresponding b-node (respectively, v-node and z-node). On the other hand, because of the degree distribution of its nodes, any non-trivial automorphism of $G_{x,y}$ has to map the u-nodes to the corresponding v-nodes: this in turn implies that, because of their neighborhoods, each a node has to be mapped to the corresponding b-node. Hence, since the mapping is an automorphism, the neighborhood of node a and node b has to be the same: that is, $x = y$.

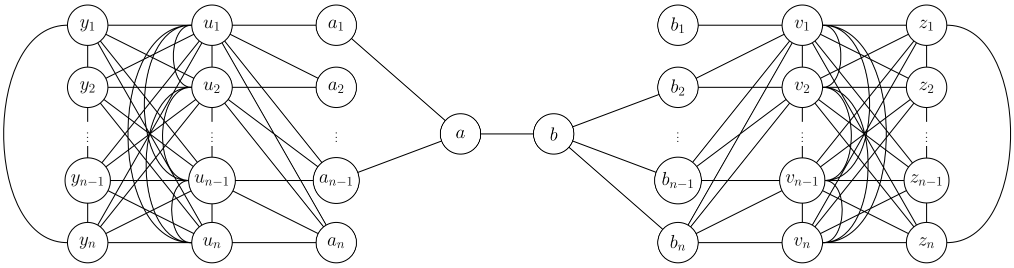


Figure 2 The graph used to reduce Eq (respectively, $\overline{\mathsf{Eq}}$) to Sym (respectively, $\overline{\mathsf{Sym}}$): in this case, $x_2 = x_n = y_1 = 0$.

Let us now suppose that there exists a dAM protocol $\mathcal{P}$ with one interaction for the Sym (respectively, $\overline{\mathsf{Sym}}$) problem which uses certificates of size $o(\log\log n)$. We now show how $\mathcal{P}$ can be used to design an Arthur-Merlin protocol for Eq (respectively, $\overline{\mathsf{Eq}}$) with communication complexity $o(\log\log n)$ (for the sake of brevity, we will show this statement for Sym and Eq: the proof for $\overline{\mathsf{Sym}}$ and $\overline{\mathsf{Eq}}$ is almost identical). Given x (respectively, y), Alice (respectively, Bob) can construct the (a,u,y)-subgraph (respectively, (b,v,z)-subgraph) of $G_{x,y}$: let $G^A_{x,y}$ (respectively, $G^B_{x,y}$) denote such subgraph. After having sent to Merlin the shared random string r, Alice and Bob waits for Merlin's certificate which is supposed to be formed by the two certificates π_a and π_b that nodes a and b would have received during the execution of $\mathcal{P}$ with random string r. By simulating $\mathcal{P}$ for every possible certificate assignment to the nodes of $G^A_{x,y}$ (respectively, $G^B_{x,y}$), Alice (respectively, Bob) can verify whether there exists an assignment that makes all the nodes of its corresponding subgraph accept: if this is the case, Alice (respectively, Bob) accepts. By definition, we have that if $x = y$, then, for any random string r there exist a certificate assignment to $G^A_{x,y}$ (respectively, $G^B_{x,y}$) and a certificate for Alice and Bob which make Alice and Bob accept. On the other hand, if $x \neq y$, then, for at least $2/3$ of all possible random strings, any certificate assignment to $G^A_{x,y}$ (respectively, $G^B_{x,y}$) and any certificate for Alice and Bob makes Alice and Bob reject. Since the size of the certificate for Alice and Bob is twice the size of the certificate size of $\mathcal{P}$, this implies that this protocol is an Arthur-Merlin protocol for Eq with communication complexity $o(\log\log n)$. This contradicts the lower bound observed in [17]: we have thus proved that $\mathsf{Sym}, \overline{\mathsf{Sym}} \notin \mathsf{dAM}(o(\log\log n), \infty)$.

The above proof can be adapted in order to obtain a lower bound on the communication complexity of any dAM for the Sym and $\overline{\mathsf{Sym}}$ problems. Indeed, instead of asking Merlin for the certificates of the nodes a and b, Alice and Bob ask Merlin for the message transmitted on the edge (a,b). They then try to find a certificate assignment that suits this message. Hence, we have also shown that $\mathsf{Sym}, \overline{\mathsf{Sym}} \notin \mathsf{dAM}(\infty, o(\log\log n))$ and the theorem follows. ◀

A result similar to previous theorem was proved in [20] in an ad-hoc manner, but only for the Sym problem and with respect to space complexity. The authors have recently reported to improve the lower bound from $\log\log n$ to $\log n$ [24].

5 Interactions vs. Space and Communication

In this section, we explore the power given to interactive protocols by allowing many interactions between Merlin and Arthur, in terms of both space and communication complexity.

Reducing the number of interactions

The following general result allows us to reduce the number of interactions between Arthur and Merlin, at the cost of increasing the certificate size and the communication cost of the protocol.

▶ **Theorem 11.** *For any two functions σ and γ, $\mathsf{dMAM}(\sigma,\gamma) \subseteq \mathsf{dAM}(n\sigma^2, n\sigma\gamma)$.*

Proof idea. We modify a dMAM protocol, so that Merlin gives all the certificates at the end, instead of at interactions 1 and 3. This results in a very low success probability, so we repeat the new protocol for $n\sigma$ times in parallel, and accept only if all occurrences are accepting. ◀

▶ **Corollary 12.** *For any two functions σ and γ, $\mathsf{dAMAM}(\sigma,\gamma) \subseteq \mathsf{dAM}(n\sigma^2, n\sigma\gamma)$.*

As a direct application of Theorem 11, we have that Sym admits a dAM protocol with one interaction and certificate size $O(n \log^2 n)$. This is a consequence of the theorem and of the existence of a dAM protocol with two interactions [20]. It is worth noting that this consequence of our general reduction result is only a log factor away from the "ad-hoc" result of [20] which establishes the existence of a dAM protocol with one interaction and certificate size $O(n \log n)$. Corollary 12 can be applied to a more recent result[24], which establishes the existence of a dAM protocol with three interactions and certificate size $O(\log n)$ for the non-isomorphism graph problem, in the case in which the nodes can communicate on both graphs. As a consequence of the corollary, we have that this problem admits a dAM protocol with one interaction and certificate size $O(n \log^2 n)$. As far as we know, this is the first dAM protocol with one interaction for this version of the non-isomorphism graph problem An interesting open question is whether such a protocol can exist also for the problem in which nodes can communicate only on one graph, while the other graph is locally given as input to the nodes themselves. For this latter problem, a dAM protocol with one interaction and certificate size $O(n \log n)$ was given [20], as well as an dAM protocol with a constant number of interactions and certificate size $O(\log n)$ [24]. Observe that the two applications of Theorem 11 and of Corollary 12 are obtained at the cost of an increase of the communication complexity by a factor $\widetilde{O}(n)$. We do not know if this linear increase of communication complexity can be avoided in general.

The Arthur-Merlin Hierarchy

We analyze the power of the Arthur-Merlin hierarchy. Recall that, for any $\sigma \geq 0$ and $\gamma \geq 0$, $\mathsf{dAMH}(\sigma, \gamma) = \bigcup_{k \geq 0} \mathsf{dAM}[k](\sigma, \gamma)$. We show that increasing the number of interactions cannot help much for reducing the certificate size to $o(n)$, even for languages defined on a very simple subclass of regular graphs, with 1-bit inputs, and admitting a locally checkable proof with $O(n)$-bit certificates.

▶ **Theorem 13.** *There exists a distributed language $\mathcal{L}$ on cycles, with 1-bit inputs, admitting a locally checkable proof with $O(n)$-bit certificates, and $O(n)$-bit messages, that is outside the Arthur-Merlin hierarchy with $o(n)$-bit certificates, even with messages of unbounded size, and even if the verifier performs an arbitrarily large constant number of rounds, whenever Arthur generates $\rho(n) = o(n)$ random bits at each node for each interaction with Merlin. In short, there exists a distributed language $\mathcal{L}$ on regular graphs satisfying $\mathcal{L} \in \Sigma_1\mathsf{LD}(O(n), O(n)) \setminus \mathsf{dAMH}(o(n), \infty)$.*

Proof idea. The main argument of the proof is that the number of transcripts of Arthur-Merlin protocol with $o(n)$-bit certificates, $o(n)$-bit random strings, and $k = O(1)$ interactions, is smaller than the number of distributed languages on n-node graphs, even with 1-bit input labels, and even on the ring. On the other hand, with $\Theta(n)$-bit certificates, all such languages can be decided by a locally checkable proof on the ring. ◀

We complete this section by showing that, in contrast to the previous theorem, every distributed language on regular graphs with $O(1)$-bit inputs has a locally checkable proof with $O(n)$-bit certificates, and a 2-round verifier.

▶ **Theorem 14.** *Every distributed language on d-regular graphs with $O(1)$-bit input labels belongs to $\Sigma_1\mathsf{LD}(\widetilde{O}(n), O(dn))$, with a verifier performing two rounds.*

Proof idea. To prove such a general claim, one must allow each node to study the structure and inputs of the whole graph. To this end, we apply the probabilistic method and show that there is a "balanced" assignment of information to the nodes, such that each node can use the information in its 1-neighborhood in order to reconstruct the entire graph structure. ◀

Since $\Sigma_1\mathsf{LD} \subseteq \mathsf{dAM} \cap \mathsf{dMA}$, an immediate corollary of this theorem is that every distributed language on d-regular graphs with $O(1)$-bit inputs belongs to $\mathsf{dAM}(\widetilde{O}(n), O(dn))$. Using a known RPLS protocol [15], we can also show that each such language belongs to $\mathsf{dMA}(\widetilde{O}(dn), O(\log n))$.

References

1 Scott Aaronson and Avi Wigderson. Algebrization: A New Barrier in Complexity Theory. *TOCT*, 1(1):2:1–2:54, 2009. doi:10.1145/1490270.1490272.

2 Amir Abboud, Aviad Rubinstein, and R. Ryan Williams. Distributed PCP Theorems for Hardness of Approximation in P. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 25–36, 2017. doi:10.1109/FOCS.2017.12.

3 Daniel Apon, Jonathan Katz, and Alex J. Malozemoff. One-round multi-party communication complexity of distinguishing sums. *Theor. Comput. Sci.*, 501:101–108, 2013. doi:10.1016/j.tcs.2013.07.026.

4 Maria Axenovich, Jochen Harant, Jakub Przybylo, Roman Soták, Margit Voigt, and Jenny Weidelich. A note on adjacent vertex distinguishing colorings of graphs. *Discrete Applied Mathematics*, 205:1–7, 2016. doi:10.1016/j.dam.2015.12.005.

5 Alkida Balliu, Gianlorenzo D'Angelo, Pierre Fraigniaud, and Dennis Olivetti. What can be verified locally? *J. Comput. Syst. Sci.*, 97:106–120, 2018. doi:10.1016/j.jcss.2018.05.004.

6 Keren Censor-Hillel, Ami Paz, and Mor Perry. Approximate Proof-Labeling Schemes. In *Structural Information and Communication Complexity - 24th International Colloquium, SIROCCO*, pages 71–89, 2017. doi:10.1007/978-3-319-72050-0_5.

7 Yi-Jun Chang, Seth Pettie, and Hengjie Zhang. Distributed Triangle Detection via Expander Decomposition. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 821–840, 2019. doi:10.1137/1.9781611975482.51.

8 Sebastian Czerwinski, Jaroslaw Grytczuk, and Wiktor Zelazny. Lucky labelings of graphs. *Inf. Process. Lett.*, 109(18):1078–1081, 2009. doi:10.1016/j.ipl.2009.05.011.

9 Andrew Drucker, Fabian Kuhn, and Rotem Oshman. On the power of the congested clique model. In *ACM Symposium on Principles of Distributed Computing, PODC*, pages 367–376, 2014. doi:10.1145/2611462.2611493.

10 Laurent Feuilloley and Pierre Fraigniaud. Survey of Distributed Decision. *Bulletin of the EATCS*, 119, 2016. URL: http://eatcs.org/beatcs/index.php/beatcs/article/view/411.

11 Laurent Feuilloley, Pierre Fraigniaud, and Juho Hirvonen. A Hierarchy of Local Decision. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP*, pages 118:1–118:15, 2016. doi:10.4230/LIPIcs.ICALP.2016.118.

12 Laurent Feuilloley, Pierre Fraigniaud, Juho Hirvonen, Ami Paz, and Mor Perry. Redundancy in Distributed Proofs. In *32nd International Symposium on Distributed Computing, DISC*, pages 24:1–24:18, 2018. doi:10.4230/LIPIcs.DISC.2018.24.

13 Pierre Fraigniaud, Mika Göös, Amos Korman, Merav Parter, and David Peleg. Randomized distributed decision. *Distributed Computing*, 27(6):419–434, 2014. doi:10.1007/s00446-014-0211-x.

14 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.

15 Pierre Fraigniaud, Boaz Patt-Shamir, and Mor Perry. Randomized proof-labeling schemes. *Distributed Computing*, 32(3):217–234, 2019.

16 John Gill. Computational Complexity of Probabilistic Turing Machines. *SIAM J. Comput.*, 6(4):675–695, 1977. doi:10.1137/0206049.

17 Mika Göös, Toniann Pitassi, and Thomas Watson. Zero-Information Protocols and Unambiguity in Arthur-Merlin Communication. *Algorithmica*, 76(3):684–719, 2016. doi:10.1007/s00453-015-0104-9.

18 Mika Göös and Jukka Suomela. Locally Checkable Proofs in Distributed Computing. *Theory of Computing*, 12(1):1–33, 2016. doi:10.4086/toc.2016.v012a019.

19 Karthik C. S., Bundit Laekhanukit, and Pasin Manurangsi. On the parameterized complexity of approximating dominating set. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pages 1283–1296, 2018. doi:10.1145/3188745.3188896.

20 Gillat Kol, Rotem Oshman, and Raghuvansh R. Saxena. Interactive Distributed Proofs. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing, PODC*, pages 255–264, 2018. URL: https://dl.acm.org/citation.cfm?id=3212771.

21 Amos Korman and Shay Kutten. Distributed verification of minimum spanning trees. *Distributed Computing*, 20(4):253–266, 2007. doi:10.1007/s00446-007-0025-1.

22 Amos Korman, Shay Kutten, and David Peleg. Proof labeling schemes. *Distributed Computing*, 22(4):215–233, 2010. doi:10.1007/s00446-010-0095-3.

23 Eyal Kushilevitz and Noam Nisan. *Communication complexity.* Cambridge University Press, 1997.

24 Moni Naor, Merav Parter, and Eylon Yogev. The Power of Distributed Verifiers in Interactive Proofs. *CoRR*, abs/1812.10917, 2018. arXiv:1812.10917.

25 Moni Naor and Larry J. Stockmeyer. What Can be Computed Locally? *SIAM J. Comput.*, 24(6):1259–1277, 1995. doi:10.1137/S0097539793254571.

26 Noam Nisan. The communication complexity of threshold gates. In *Combinatorics, Paul Erdos is Eighty*, volume 1, pages 301–315, 1993.

27 Anup Rao and Amir Yehudayoff. Simplified Lower Bounds on the Multiparty Communication Complexity of Disjointness. In *30th Conference on Computational Complexity, CCC*, pages 88–101, 2015. doi:10.4230/LIPIcs.CCC.2015.88.

28 Larry J. Stockmeyer. The Polynomial-Time Hierarchy. *Theor. Comput. Sci.*, 3(1):1–22, 1976. doi:10.1016/0304-3975(76)90061-X.