Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 124 (2022)

On Interactive Proofs of Proximity with Proof-Oblivious Queries

Oded Goldreich* Guy N. Rothblum† Tal Skverer‡

September 9, 2022

Abstract

Interactive proofs of proximity (IPPs) offer ultra-fast approximate verification of assertions regarding their input, where ultra-fast means that only a small portion of the input is read and approximate verification is analogous to the notion of approximate decision that underlies property testing. Specifically, in an IPP, the prover can make the verifier accept each input in the property, but cannot fool the verifier into accepting an input that is far from the property (except for with small probability).

The verifier in an IPP system engages in two very different types of activities: interacting with an untrusted prover, and querying its input. The definition allows for arbitrary coordination between these two activities, but keeping them separate is both conceptually interesting and necessary for important applications such as addressing temporal considerations (i.e., at what time is each of the services available) and facilitating the construction of zero-knowledge schemes. In this work we embark on a systematic study of IPPs with proof-oblivious queries, where the queries should not be affected by the interaction with the prover. We assign the query and interaction activities to separate modules, and consider different limitations on their coordination.

The most strict limitation requires these activities to be totally isolated from one another; they just feed their views to a separate deciding module. We show that such systems can be efficiently emulated by standard testers.

Going to the other extreme, we only disallow information to flow from the interacting module to the querying module, but allow free information flow in the other direction. We show that extremely efficient one-round (i.e., two-message) systems of such type can be used to verify properties that are extremely hard to test (without the help of a prover). That is, the complexity of verifying can be polylogarithmic in the complexity of testing. This stands in contrast the MAPs (viewed as 1/2-round systems) in which proof-oblivious queries are as limited as our isolated model.

Our focus is on an intermediate model that allows shared randomness between the querying and interacting modules but no information flow between them. In this case we show that 1-round systems are efficiently emulated by standard testers but 3/2-round systems of extremely low complexity exist for properties that are extremely hard to test. One additional result about this model is that it can efficiently emulate any IPP for any property of low-degree polynomials.

*Weizmann Institute of Science, Rehovot, Israel.

†Apple Machine Learning Research. Part of this work was done while at the Weizmann Institute of Science.

‡Weizmann Institute of Science, Rehovot, Israel.

ISSN 1433-8092

Contents

1 Introduction

This paper initiates a systematic study of a natural type of interactive proofs of proximity, a notion which is a hybrid of interactive proofs and property testing. Specifically, interactive proofs of proximity (IPPs) combine the paradigm of verifying delegated computation with the paradigm of ultra-fast computation that refers to an approximate version of the actual input. Since any proof system is defined in terms of its verification procedure, which in turn presumes a model of computation, we start from the model of computation underlying property testing (see, e.g., the textbook [14]).

1.1 From property testing to interactive proofs of proximity

Loosely speaking, property testing typically refers to sub-linear time probabilistic algorithms for deciding whether a given object has a predetermined property or is far from any object having this property. Such algorithms, called testers, obtain local views of the object by making adequate queries; that is, the object is seen as a function and the testers get oracle access to this function (and thus may be expected to work in time that is sub-linear in the size of the object). In particular, one often seeks testers of extremely low query complexity (e.g., query complexity that is polylogarithmic in the size of the object).

Needless to say, such low level of complexity for testing cannot be achieved for all problems of interest, which leads to the natural question of whether traditional (non-interactive) proofs or interactive proofs can assist us in such cases. That is, viewing low query complexity as our base line, we are considering non-interactive and interactive proof systems in which the verifier makes few queries and reads short proofs (resp., short messages from the prover). Indeed, we are talking about "NP" and "IP" analogs of property testing (viewed as an analogue of "P"). Such analogs, termed *MA-proofs of proximity* (MAPs)[1] and *interactive proofs of proximity* (IPPs), were introduced and studied in [20] and [23], respectively.[2]

The seemingly innocuous analogy between the well-known complexity classes $\mathcal{MA}$ and $\mathcal{IP}$ and the "property testing versions" termed MAP and IPP ignores the question of orchestrating the two activities of the verifier (i.e., querying the oracle and interacting with the prover). The trivial answer is that these two activities are performed by the same entity concurrently with free information flow from one activity to the other. But *is this concurrency and free information flow essential*?

The foregoing question is not merely interesting *per se*. There may be settings in which such concurrency is not possible or that a free information flow is not desirable. For example, it may be that query access to the input is only available at one time, whereas the interaction with the prover is only available afterwards. Alternatively, as outlined at the end of Section 1.2, having queries that are oblivious of the prover's messages seems a first step towards obtaining a natural notion of zero-knowledge IPPs (cf. [4]). Indeed, our focus is on IPPs that employ *proof-oblivious queries*.

We mention that proof-oblivious queries were considered before, both in the context of MAPs (cf. [20, Def. 2.2]) and in the context of IPPs (cf. [23, Fn. 2] and [18, Sec. 5]). In particular, it was

[1] Note that in a randomized setting, as is inherent for algorithms that read small parts of their input, alleged (non-interactive) proofs are verified probabilistically. Hence, such verification is actually analogous to MA rather than to NP.

[2] Actually, the work of [23] predated [20], and both were predated by [9], which presents an extremely general framework that contains both IPPs and PCPPs as a special case.

shown that MAPs with proof-oblivious queries can be efficiently emulated by a standard tester [20, Thm. 4.2]: If the former use proofs of length p and q queries, then the tester has query complexity $O(p \cdot q)$. This should be contrasted with the fact that there are MAPs (with proof-dependent queries) of extremely low complexity for properties that are extremely hard to test (see [20, Thm. 1.1]). We interpret these facts as saying that proof-oblivious queries severely limit the power of MAPs. But *is it so also for IPPs*?

Before answering the foregoing question, we observe that, in the context of IPPs, *the notion of proof-oblivious queries may have several different natural interpretations* (which all coincide in the context of MAPs).[3] In fact, we initiate a systematic study of proof-oblivious queries (*PO-queries*) in the context of IPPs. Jumping ahead, we telegraphically highlight the following results (which will be properly reviewed in Section 1.3):

- Under the most strict interpretation of IPPs with proof-oblivious queries (i.e., the "isolated" model), these systems can be efficiently emulated by a standard tester.
- Under the most liberal interpretation of IPPs with proof-oblivious queries (i.e., the "general" model), there exist such systems of extremely low complexity for properties that are extremely hard to test. This holds even for 1-round systems (in which a single message by the verifier followed by a single message by the prover), in contrast to MAPs (which are 1/2-round IPPs).
- Under an intermediate (and natural) interpretation of IPPs with proof-oblivious queries (i.e., the "pre-coordinated" model), it holds that 1-round systems are efficiently emulated by testers but 3/2-round systems of extremely low complexity exist for properties that are extremely hard to test.

We now turn to a more detailed account of the underlying definitions and results.

1.2 Defining IPPs and types of IPPs with PO-queries

An interactive proof of proximity is a two-party protocol for parties called verifier and prover. The verifier has oracle access to a function $f : [n] \to \Sigma$, and also gets explicit inputs n and $\epsilon > 0$, where ϵ is called the proximity parameter. The prover gets f as explicit input, and its aim is to convince the verifier that f is in some predetermined set Π_n, called the property. In analogy to the definition of interactive proof systems [16], we require that the prover can convince the verifier to accept any f in Π (w.h.p.), but cannot fool the verifier into accepting f that is ϵ-far from Π_n (except for with low probability). Indeed, the main deviation from the definition of a (standard) interactive proof system is that *the soundness requirement is made only for inputs that are ϵ-far from Π_n*, where f is ϵ-far from Π_n if for every $g \in \Pi_n$ it holds that $|\{i \in [n] : f(i) \neq g(i)\}| > \epsilon \cdot n$. The actual definition refers to an optimal prover strategy, denoted P, and focuses on the communication and query complexities of the system.

Definition 1.1 (interactive proofs of proximity systems (IPPs) [23]):[4] *Let $\Pi = \bigcup_{n \in \mathbb{N}} \Pi_n$ such that Π_n is a set of functions over $[n]$. A randomized and interactive oracle machine, denoted V,*

[3] While [18] do not formally define proof-oblivious systems, their claims seem to refer to two different notions. See Section 1.6.

[4] As discussed in Section 1.5, we avoid the good convention of specifying a ("honest") prover strategy for the completeness condition.

constitutes a verifier *for an* interactive proof of proximity for (the property) Π *if the following two conditions hold.*

(*completeness*): *On input n, ϵ and oracle access to any $f \in \Pi_n$, after interacting with an optimal prover P, the verifier accepts with probability at least $2/3$.*

(*soundness*): *On input n, ϵ and oracle access to any $f : [n] \to \Sigma$ that is ϵ-far from Π_n, after interacting with an optimal prover P, the verifier accepts with probability at most $1/3$.*

We say that the system has perfect completeness *if the verifier accepts each $f \in \Pi$ with probability 1. The* query complexity of V *is $q : \mathbb{N} \times [0,1] \to \mathbb{N}$ if, on input n, ϵ and oracle access to any $f : [n] \to \Sigma$, the verifier makes at most $q(n, \epsilon)$ queries to f. The* communication complexity of the system *is $c : \mathbb{N} \times [0,1] \to \mathbb{N}$ if, on input n, ϵ and oracle access to any $f : [n] \to \Sigma$, the parties exchange at most $c(n, \epsilon)$ bits.*

Standard testers can be viewed as a special case in which the communication complexity is zero; that is, effectively there is no prover. MAPs correspond to the special case in which the communication is unidirectional, with the prover sending a single message (of length at most c).

The general notion of proof-oblivious queries merely asserts that the queries made by the verifier are oblivious of the messages received from the prover. In order to clarify what this means as well as discuss alternative notions, we decompose the verifier V into three (randomized) modules: A querying module denoted Q, which is in charge of querying the oracle, a interacting module denoted I, which is in charge of interacting with the prover, and a deciding module denoted D, which takes the final decision based on the information handed to it by the other two modules. Now, the general notion (or model) of proof-oblivious queries postulates that the interaction from the querying module Q to the other modules is unidirectional (from Q to I and D). Actually, it suffices to have Q send a single message to I, which in turn sends a single message to D. (Actually, in this case, we may combine I and D into a single entity.)

In contrast, the most strict interpretation of proof-oblivious queries means that the querying and interacting modules are totally uncoordinated; in particular, they do not communicate with one another nor do they receive any message from the deciding module (since otherwise D may serve as a subliminal communication channel between Q and I). In the corresponding model, called the isolated model, the two main modules (i.e., Q and I) are totally isolated from one another. The only communication between the three modules amount to Q and I sending their view to the deciding module D; specifically, the querying module sends the answers that it has obtained from the oracle along with its internal coin tosses, whereas the interacting module sends the messages that it has obtained from the prover along with its internal coin tosses. Without loss of generality, each module sends a single message to D. See illustration on the l.h.s of Figure 1.

Note that in the isolated model the two messages received by D (from Q and I, respectively) are statistically independent. In particular, Q and I each have their own source of randomness, and these sources are independent of one another. In contrast, in the pre-coordinated model, these two parties are fed by the same source of randomness, although they are free to use disjoint parts of it. See illustration at the center of Figure 1. Hence, although the communication pattern (in the pre-coordinated model) is the same as in the isolated model, the messages that D receives may be correlated, since Q and I are using the same source of randomness. In particular, I may send to the prover messages that are related to the queries that Q made to the oracle, but unlike in the general model (see illustration on the r.h.s of Figure 1) these messages may not depend on the answers provided by the oracle.

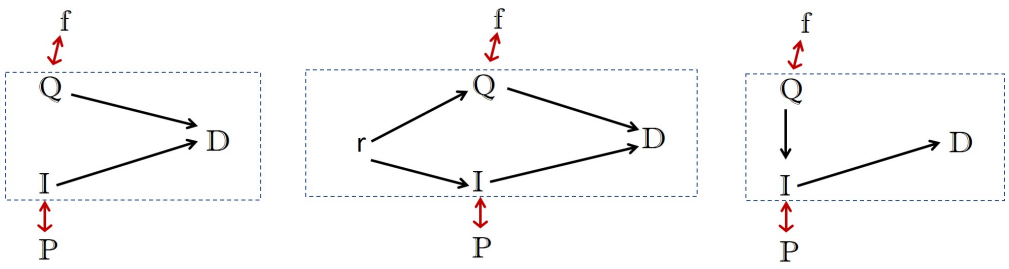

Figure 1: The isolated, pre-coordinated, and general models of PO-queries.

Public-coin IPPs. Recall that public-coin interactive proofs (of proximity) are ones in which each message sent by the verifier is a predetermined subsequence of the randomness of the verifier. Note that any public-coin IPP that uses PO-queries is, by definition, in the pre-coordinated model, because its interacting module is not allowed to use the information it might have received from the querying module in determining its messages to the prover. On the other hand, any IPP of the isolated model can be emulated by a public-coin IPP of the isolated model.

Obtaining zero-knowledge IPPs. Loosely speaking, IPPs that use proof-oblivious queries, even in the general sense, may be easily converted into zero-knowledge IPPs (e.g., as defined in [4]). In case these IPPs (with PO-queries) are of the public-coin type, we just have the (zero-knowledge) prover send commitments to its original messages, and prove at the very end (in zero-knowledge manner) that the original verifier would have accepted the corresponding (de-committed) information. This does not disrupt the new verifier's actions, since its queries are proof-oblivious and its messages to the prover are random. When using general (i.e., private-coin) IPPs (with PO-queries), one may use secure two-party computation to enable the verifier to ("blindly") deliver messages to the prover; these messages are computed based on the verifier's randomness and the previously committed messages of the prover, while the prover provides (in secrecy) the corresponding de-committed values, and the verifier remains ignorant about its own messages. Alternatively, one may use fully-homomorphic encryptions instead of the commitment, and have the verifier send encryptions of its own messages.

1.3 Our main results

As stated upfront, our focus is on interactive proofs of proximity (IPPs) that use proof-oblivious queries (PO-queries). We initiate a systematic study of such systems, obtaining various results that distinguish the three models that were presented in Section 1.2 and relating them to MAPs and to standard testers.

When relating the various models, we consider an emulation of one model in a second model to be efficient if the communication and query complexity in the second model are polynomial in the complexities in the first model. In contrast, separation results assert at least a sub-exponential gap between the corresponding complexities. Our first result asserts that IPPs in the isolated model can be efficiently emulated by standard testers.

Theorem 1.2 (efficient emulation of the isolated model by testers): *Suppose that* Π *is a property*

that can be verified in the isolated model (of IPPs with proof-oblivious queries) *using q queries and c bits of communication. Then,* Π *has a standard tester of query complexity* $O((c+1)\cdot q)$.

Theorem 1.2 significantly extends [20, Thm. 4.2], which establishes an analogous emulation of *MAPs with proof-oblivious queries*. Recall that MAPs are 1/2-round IPPs (i.e., the "interaction" is unidirectional with the prover sending a single message to the verifier), and in this case all three models of proof-oblivious queries coincide. In contrast, in Theorem 1.2 the communication between the prover and the verifier is only restricted by the total amount of bits communicated.

While the isolated model offers limited advantage over a standard tester, we show that the general model (of IPPs with proof-oblivious queries) is significantly stronger. In fact, not only is it stronger than standard testers, it is even not efficiently emulated by general MAPs (i.e., ones that are *not* restricted to proof-oblivious queries).

Theorem 1.3 (the general model of PO-queries cannot be efficiently emulated by MAPs): *There exists a property* Π *that can be verified in the general model* (of IPPs with proof-oblivious queries) *using* $O(1/\epsilon)$ *queries and* $O(\epsilon^{-1}\log n)$ *bits of communication, but any MAP for* Π *that uses a proof of length p must use at least* $\Omega(n^{1/2}/(p+1))$ *queries. Furthermore, the IPP with proof-oblivious queries uses a single round of communication.*

Theorem 1.3 is proved for the property consisting of all permutations over $[n]$, while using the lower bound established in [18, Lem. 4.3]. For the upper bound we use a proof system different than the (1-round) IPP presented in [18, Sec. 4.1], since the latter uses proof-dependent queries (i.e., it does not satisfy the condition of proof-oblivious queries).

The results regarding the model of pre-coordinated (PO-queries) IPPs bridge the two extremes captured by Theorems 1.2 and 1.3. On the one hand, *1-round* IPPs in the pre-coordinated model can be efficiently emulated by standard testers. On the other hand, 3/2-*round* IPPs in the pre-coordinated model are significantly stronger than standard testers. Recall that in 1-round IPP the communication consists of two messages (i.e., the first message is sent from the verifier to the prover who responds with a single message), whereas in 3/2-round IPP the communication consists of three messages (i.e., the prover sends a single message, which is followed by a communication round as in a 1-round IPP).

Theorem 1.4 (the pre-coordinated model of PO-queries – 1 round vs 1.5 rounds): *The following dichotomy holds regarding IPPs* (with proof-oblivious queries) *in the pre-coordinated model.*

1. *Any property that can be verified in the pre-coordinated model, using a 1-round IPP with q queries and c bits of communication, has a standard tester of query complexity* $O((c+1)\cdot q)$.
2. *There exists a property* Π *that can be verified in the pre-coordinated model using a* 3/2-*round IPP with* $O(1/\epsilon)$ *queries and* $O(\epsilon^{-1}\log n)$ *bits of communication, but any tester for* Π *uses at least* $\Omega(n^{1/2})$ *queries.*

Indeed, Part 1 asserts an efficient emulation at a complexity level that matches the bound in Theorem 1.2 (which refers to the isolated model), whereas Part 2 provides a separation analogous to Theorem 1.3 (which refers to the general model). Note that Part 1 implies a separation between 1-round IPPs in the general model (of proof-oblivious queries) and 1-round IPPs in the pre-coordinated model. On the other hand, the separation in Part 2 is only with respect to testers (rather than MAPs as in Theorem 1.3). In order to prove the hardness of the emulation of the pre-coordinated model by general MAPs, we use more rounds of the pre-coordinated model.

	1/2-round (MAP)	1-round	3/2-round	$O(1)$-round	$\omega(1)$-round
isolated	YES [20, Thm. 4.2]	$\longleftarrow$			YES (Thm. 1.2)
public-coin		$\longleftarrow$		YES (Thm. 1.8)	NO (Thm. 1.5)
pre-coordinated	$\downarrow$	YES (Thm. 1.4(1))	NO (Thm. 1.4(2))	$\longrightarrow$	
general POQ		NO (Thm. 1.3)	$\longrightarrow$		

Table 1: Can standrad testers efficiently emulate IPPs that use PO-queries?

Theorem 1.5 (the pre-coordinated model of PO-queries cannot be efficiently emulated by MAPs): *There exists a property* Π *that can be verified in the pre-coordinated model* (of IPPs with proof-oblivious queries) *using* $\mathrm{poly}(\epsilon^{-1}\log n)$ *queries and* $\mathrm{poly}(\log n)$ *bits of communication, but any MAP for* Π *that uses a proof of length* p *must use at least* $\Omega(n^{0.999}/(p+1))$ *queries. Furthermore, the IPP with proof-oblivious queries uses* $O(\log n)$ *rounds of public-coin communication.*

Theorem 1.5 is proved by observing that the proof of [20, Thm. 3.28] uses an IPP that can be implemented in the pre-coordinated model. Specifically, we observe that the celebrated sum-check protocol of [22] can be implemented in the pre-coordinated model (see Section 4.3). The general picture that emerges from the results reviewed so far (and the following Theorem 1.8) is summarized in Table 1.

A focus on the pre-coordinated model. We find the pre-coordinated model especially appealing, since it allows the two main modules to operate independently of one another (but based on the same randomness, which is essential in light of Theorem 1.2). Theorems 1.4 and 1.5 frame our study of the pre-coordinated model, which is aimed at a finer understanding of what these proof systems can achieve. We loosely state two positive results and one negative result. The first result refers to a natural class of (general) MAPs.

Theorem 1.6 (3/2-rounds of IPPs in the pre-coordinated model can efficiently emulate a natural class of MAPs): *Suppose that* Π *is a property of functions from* $[n]$ *to* $[m]$ *that can be verified by a MAP that uses a proof of length* ℓ *and makes* q *non-adaptive and uniformly distributed queries. Then,* Π *can be verified in the pre-coordinated model by a* 3/2*-round IPP that uses* $\widetilde{q} = \widetilde{O}(q)$ *queries and total communication* $O(\widetilde{q}\cdot\ell + \widetilde{q}^2\cdot\log(nm))$.

We stress that the hypothesis only requires that each query of the MAP verifier is uniformly distributed; their joint distribution, which may also depend on the given proof-string, is arbitrary (beyond this requirement).[5] We comment that the 3/2-round (pre-coordinated model) IPP of Theorem 1.4 (Part 2) is essentially obtained as a special case of Theorem 1.6. On the other hand, the result we actually obtain (see Theorem 4.7) is much more general than Theorem 1.6. The next result refers to a much wider class of IPPs, but restricts the class of properties.

Theorem 1.7 (IPPs in the pre-coordinated model can efficiently emulate general IPPs for any property of low-degree polynomials): *Let* Π *be a property* (equiv., a subset) *of* m*-variate polynomials of total degree* d *over a finite field* $\mathcal{F}$*, and suppose that* Π *can be verified by an* r*-round public-coin*

[5] Indeed, we are interested in the case that the original MAP uses proof-dependent queries. In this case, the dependence on the proof-string is manifested in the dependent between the queries.

IPP of query complexity q and communication complexity c, and that $d \leq |F|/O(1+\log_{|\mathcal{F}|} q)$. Then, Π can be verified in the pre-coordinated model by a $(r+q)$-round IPP that uses $O(q+d)$ queries and total communication $O(c) + q \cdot (d+m) \cdot O(\log(q+|\mathcal{F}|))$. Furthermore, if the original IPP uses non-adaptive queries, then the emulation can be done in $r+1$ rounds.

Recall that general r-round IPPs can be efficiently emulated by $O(r)$-round public-coin IPPs (see [23], following [17]); hence the result stated in Theorem 1.7 extends to the case that Π can be verified by any IPP, but the resulting round and communication complexities are increased in a suitable manner (so to account for the public-coin emulation). On the other hand, the verifier in the resulting IPP (i.e., in the pre-coordinated model) is not public-coin, and this is inherent because any property (including ones that have an efficient MAP but are hard to test) can be embedded in a property of polynomials whereas the following Theorem 1.8 limits the power of public-coin system.[6] Indeed, it follows that *IPPs of the pre-coordinated model cannot be efficiently emulated by public-coin IPPs of the pre-coordinated model* (cf. Corollary 1.9).

Theorem 1.8 (efficient emulation by testers of public-coin $O(1)$-round IPPs in the pre-coordinated model): *Let Π be a property that can be verified by an r-round public-coin IPP in the pre-coordinated model using q queries and c bits of communication. Then, Π has a standard tester of query complexity $(\mathrm{poly}(r) \cdot c)^r \cdot q$.*

Recall that any public-coin IPP that uses PO-queries is, by definition, in the coordinated model. The public-coin restriction in Theorem 1.8 is inherent in light of Theorem 1.4 (Part 2). Indeed, combining Theorems 1.8 and 1.4, we get.

Corollary 1.9 (private-coin 3/2-round IPPs in the pre-coordinated model cannot be efficiently emulated by $O(1)$-round public-coin IPPs in the pre-coordinated model): *There exists a property Π that can be verified in the pre-coordinated model using a 3/2-round IPP with $O(1/\epsilon)$ queries and $O(\epsilon^{-1} \log n)$ bits of communication, but any r-round public-coin IPP in the pre-coordinated model for Π of query complexity q uses at least $\Omega((n/q^2)^{1/2r})$ bits of communication.*

Indeed, Corollary 1.9 provides yet another natural example of the gap between public-coin and general interactive proof systems.[7]

Turning back to Theorem 1.8, we mention that it extends (from public-coin IPPs of the pre-coordinated model) to IPPs of the pre-coordinated model that use *proof-oblivious messages* (i.e., the messages sent by the verifier are oblivious of the prover's messages): See Theorem 4.10. On the other hand, the emulation provided by Theorem 1.8 is relatively tight, since the proof system of [20, Lem. 3.29] has an r-round version that yields (via this emulation) a tester with almost optimal query complexity (see Appendix A.3).

[6] For example, starting from [20, Thm. 1.1], we have a property of functions $f : [n] \to \{0,1\}$ that has a MAP that uses a proof of length $O(\log n)$ and makes $O(1/\epsilon)$ queries, but cannot be tested with $n^{0.999}$ queries. Associating $[n]$ with H^m such that $|H| = \log_2 n$ and considering the low-degree extensions of these functions over a field $\mathcal{F}$ of size $O(m \cdot |H|)$, we obtain a property of degree $m \cdot |H|$ polynomials that has a MAP that uses a proof of length $O(\log n)$ and $O(m \cdot |H|/\epsilon) = O(\epsilon^{-1} \log^2 n)$ non-adaptive queries, and is at least as hard to test as the original property. Note that the domain of the new functions has size $s \stackrel{\text{def}}{=} |\mathcal{F}^m| = o(n^2)$; hence, hardness of testing holds for query complexity $s^{0.49}$.

[7] The best-known gaps were demonstrated in the context of zero-knowledge (cf. [13, Thm. 4.5.11] vs [13, Sec. 4.9.1]) and relatively efficient proving (cf. [24]). Closer to our context are the gaps shown for IPPs in the distribution testing setting [7, Sec. 6] and in the sample-based setting [11, Sec. 4.2].

1.4 Our techniques

Theorems 1.2, 1.4(1), 1.6, 1.7, and 1.8 are proved by emulating one model on another. In particular, we indicate the limitation of various types of IPPs that use PO-queries by showing that they can be efficiently emulated by standard testers. In contrast, we illustrate the power of IPPs that use PO-queries by showing that they can efficiently emulate general IPPs of certain types. We now review these two different types of emulations.

1.4.1 Emulating IPPs that use PO-queries by standard testers

We indicate the limitation of some types of IPPs (of the isolated and coordinated models) by showing that they can be efficiently emulated by standard testers. Such emulations are used in the proof of Theorem 1.2, Part 1 of Theorem 1.4, and Theorem 1.8. These emulations are best illustrated in the simplest contents of Theorem 1.2, which refers to the isolated model (of IPPs with PO-queries).

The starting point is the representation of all possible executions of a standard interactive proof system by a "game-tree" in which internal vertices represents execution prefixes and their children represent possible moves of the relevant party (cf. [3, Sec. 4] and [12, Apdx C.1]). The value of a leaf in the tree is the probability that the verifier accepts conditioned on the corresponding sequence of messages sent during the execution, where the value is either 0 or 1 in the special case that the sequence of messages fully determines the verifier's randomness (e.g., when the system is of the public-coin type). The value of an internal vertex associated with the verifier equals the expected value of its children, when the expectation is according to the (conditional) probability space that determines the next message of the verifier.[8] The value of an internal vertex associated with the prover equals the maximal value among the values assigned to its children. Hence, the value of the root of the game-tree represents the probability that the verifier accepts, which in turn represents the expected value of the leaf reached in a random execution (with an optimal prover).

We stress that the foregoing description refers to standard interactive proof systems (not to IPPs). Turning to the isolated model, we observe that the value of a leaf in the tree may be defined as the probability that the (decision module of the) verifier accepts in a random execution of the querying module, when (the deciding module is) also fed with the corresponding output of the interacting module (i.e., the corresponding sequence of messages sent during the execution of the interacting module). The key observation is this value is a function of the identity of the specific leaf (corresponding to a specific interaction transcript) and *the outcome of a random process that does not depend on the identity of this leaf*. In other words, the same random process (representing the querying module) is used in all leaves. Hence, all that we need is (constant additive error) approximations of the values of all leaves, and all these approximations can be obtained based on the same repeated invocations of the random process. Thus, it suffices to invoke the random process for a number of times that is logarithmic (rather than linear) in the number of leaves. Using these approximated values of all leaves, we can compute the approximate value of the root of the tree, and Theorem 1.2 follows (i.e., the tester invokes the querying module $O(c)$ times, where each invocation makes q queries, where c denotes the communication complexity of the IPPs and q its query complexity).

[8] Note that this selection from a conditional probability space is not necessarily how the real (possibly private-coin) verifier acts, but this is how we view the effects of its actions in the analysis.

Turning to the pre-coordinated model, which is the focus of Theorems 1.4 and 1.8, we note that it is no longer the case that the random execution of the querying module is the same in all leaves (corresponding to all interaction transcripts). Indeed, the PO-queries condition implies that these executions do not depend on the prover messages, but they are conditioned by the choices made by the interacting module, since the querying and interacting modules use the same source of randomness. In particular, different conditional spaces may lead to different sequences of queries, and so our goal is to show that it suffices to use much fewer conditional spaces than the number of leaves. In the case of one-round IPPs (of the pre-coordinated model) this is achieved by observing that the conditional spaces are oblivious of the last prover message, and that it suffices to sample a constant number of verifier messages. This leads to establishing Part 1 of Theorem 1.4. In the case of public-coin IPPs, the verifier messages are also oblivious of the prover messages, and so the relevant parameter is the product of the number of verifier messages used in each round. Proving that it suffices to sample $\text{poly}(r) \cdot c$ verifier messages for each of the r rounds, allows to establish Theorem 1.8. In other words, in this case, we prune the game-tree, leaving only $\text{poly}(r) \cdot c$ children in each vertex that corresponds to a verifier move.

1.4.2 Emulating some types of IPPs by IPPs that use PO-queries

We illustrate the power of IPPs in the coordinated model by showing that they can efficiently emulate general IPPs of certain types. Such emulations are used in the proofs of Theorems 1.6 and 1.7.

Recall that Theorem 1.6 refers to any MAP that uses a proof of length ℓ and makes q non-adaptive and uniformly distributed queries to a function $f : [n] \to [m]$, where these queries may depend on the proof given to the verifier (i.e., the individual queries may be related in a way that depends on the given proof). The key idea is to query f at a uniformly distributed point $u \in [n]$, and place u as the i^{th} query of a random execution of the MAP, where $i \in [q]$ is selected uniformly at random, then ask the prover to provide the corresponding execution of the MAP, and accept if and only if the provided transcript is accepting and matches $f(u)$. That is, our 3/2-round IPP makes the random query u, and enters an interaction with the prover, who is supposed to send the MAP-proof as its first message. Denoting this message by π, our verifier selects $i \in [q]$ uniformly at random, and selects a *random r such that on randomness r, upon given the MAP-proof π, the i^{th} query of MAP-verifier equals u*. Our verifier sends r (or the corresponding query sequence) to the prover, who is supposed to respond with the corresponding answers. Letting $\overline{a} = (a_1, ..., a_q)$ denote the actual prover response, the deciding module accepts the value $f(u)$ provided by the querying module and the transcript $(\pi, r, \overline{a})$ provided by the interacting module if and only if $a_i = f(u)$ and the MAP-verifier would have accepted the proof π, when using randomness r and getting the oracle answer-sequence $(a_1, ..., a_q)$. The error is reduced to a constant by repeating the foregoing system for $O(q)$ times, in parallel.

Turning to Theorem 1.7, recall that we are given an r-round public-coin IPP of query complexity q and communication complexity c for some property of low degree polynomials. Here, we make a random query per each query of the original prover, and ask our prover for (the univariate polynomial that describes) the values of the tested polynomial on the line that connects our random query and the real query. Specifically, the interacting module first emulates the original public-coin IPP, while relying on the fact that the verifier's messages are independent of the verifier's queries and the answers to these queries, and later it tries to obtain the answers to these queries as determined by the corresponding univariate ("line") polynomials. (Actually, we use low-degree

curves rather than lines, and, in addition, we also check that the tested function is a low-degree polynomial.)[9] If our prover provides the wrong univariate polynomial (for a line), then the deciding module catches it with high probability (since we know the value of a random point on this line), and otherwise it uses the value (of the corresponding query) as determined by this univariate polynomial. We comment that the foregoing idea was applied quite extensively in the study of PCPs, starting with [10, 2], but it seems that its first appearance in the context of IPPs with proof-oblivious queries is due to [18, Lem. 5.4].

1.5 Some comments about our conventions

This section contains brief comments about some of our conventions. We first note that although many results are stated in terms of properties of functions over $[n]$, one should view n as a generic (or varying) parameter rather than as fixed; formally, the results should be restated as in Definition 1.1. Also, whenever we write 0.999 (or 0.99), we actually mean any constant smaller than 1.

On the computational complexity of the strategies. In this work we focus on the communication and query complexities of IPPs, while ignoring their computational complexities (both for the verifier and for the prescribed prover). This makes the negative results, which establish the limited power of certain IPPs (vis-a-vis testers), stronger. We mention that our positive results, which establish the power of certain IPPs, are actually obtained using relatively low computational complexity. In particular, with the exception of Theorem 1.6, the verifier strategy we present runs in time that is almost-linear in its query and communication complexities. In the exceptional case, the computational complexity of the verifier depends on the complexity of "reversed sampling" (i.e., sampling a random-pad that generates a given query), which is low in natural cases. In general, the computational complexity of IPPs and property testers is a secondary consideration, which is left for follow-up studies.

Using optimal prover strategies. The foregoing discussion justifies not specifying a prescribed proof-strategy for the completeness conditions, but rather referring to an optimal prover strategy both in the completeness and soundness condition. As far as the soundness condition is concerned, nothing is lost by this convention, but not specifying a prover (i.e., a honest prover) strategy for the completeness condition hinders the desire to have strategies that posses additional features such as relative efficiency or zero-knowledge. The gain in the convention adopted in the current work is that it slightly simplifies the definitions and facilitates some of the proofs; specifically, the accepting probability of a verifier V on any input can be expressed as a function $p_V : \{0,1\}^* \to [0,1]$ that depends solely on V.

Alphabet (or range of the functions). The properties that we consider are sets of functions over $[n]$. Typically, these functions are either Boolean or range over $[n]$, but we also use functions of the form $f : \mathcal{F}^m \to \mathcal{F}$ for some finite field $\mathcal{F}$. In all cases, the emulations preserve the function, so the type of queries is the same in both models. Lastly, we mention that one can always represent function of the form $f : [n] \to \Sigma$ by Boolean functions over $[n']$ such that $n' = n \cdot O(\log |\Sigma|)$. In this

[9] We cannot rely on the fact that this test is conducted by the original IPPs, since we have to verify that the tested function is close to a low-degree polynomial in order to establish the soundness of the emulation.

case one encodes elements of Σ by codewords (taken from a code with constant relative distance); this is done in order to preserve relative distances up to a constant factor.

Communication rounds. Our notion of a communication round refers to the exchange of a pair of messages; hence, for $r \in \mathbb{N}$, an r-round IPP refers to the case that each of the two parties sends r messages, and means that the first message is sent by the verifier. In contrast, an $(r-0.5)$-round protocol starts with the prover sending a message, and the verifier sending only $r-1$ messages. In both cases, the last message is by the prover, who sends a total of r messages.

Non-adaptive queries. When saying that a general IPP uses non-adaptive queries we mean that the queries are determined based on the verifier's randomness and on the message it has received from the prover, but do not depend (directly) on the answers to prior queries. (An indirect dependence may arise if the verifier leaks information on its queries and the prover's messages depend on this information and on the value of the tested function at these queries.) Needless to say, non-dapative queries in IPPs that use proof-oblivious queries, depend only on the randomness of the querying module.

1.6 Related works

As mentioned upfront, proof-oblivious queries were considered before, both in the context of MAPs (cf. [20, Def. 2.2]) and in the context of IPPs (cf. [23, Fn. 2] and [18, Sec. 5]). However, the focus of these works was elsewhere. Specifically, the focus of [20] was on introducing the MAP model (of non-interactive proofs of proximity), which is viewed as an NP (or MA) version of property testing, and demonstrating its power (see, e.g., [20, Thm. 1.1]). Within this context, showing that MAPs with proof-oblivious queries can be efficiently emulated by standard testers [20, Thm. 1.5] was viewed as an indication that proof-dependent queries are essential to the power of MAPs.

Likewise, the focus of [23] was on defining IPPs in their full generality and on studying their power. They mention that in some applications it may be helpful if the verifier's queries can be made in advance of interacting with the prover, and note that all protocols in their work have this "oblivious-queries" feature, but they do not formally define or further study this feature. The focus of [18] is on separating MAPs (viewed as 1/2-round IPPs) from *one-round* IPPs (see [18, Thm. 1.1]). Within this context, showing that one-round IPPs that use "proof oblivious" queries can be efficiently emulated by standard tester [18, Thm. 1.2] is viewed as indication that proof-dependent queries are essential for that separation. As noted in Footnote 3, the notion of "proof oblivious" queries is not formally defined in [18], but it seems that the proof of [18, Thm. 1.2], which is presented in [18, Sec. 5.1], refers to the isolated model. We mention that a comment made (in passing) in [18, Sec. 1.1.1], which asserts that the sum-check protocol uses "proof oblivious" queries, seems to refer to the pre-coordinated model. The same holds for the contents of [18, Sec. 5.2.2].[10]

The notion of sample-based IPPs (SIPPs), introduced and studied in [11], considers IPPs in which the verifier can only obtain the value of the input at a sample of uniformly and independent distributed locations. These SIPPs can be viewed as a special case of our general model (of IPPs with PO-queries), whereas public-coin SIPPs can be viewed as a special case of our pre-coordination

[10] Hence, the chasm between public and private coins claimed in [18, Sec. 5.2] seems unsubstantiated, since the limits of the public coin version seem to be proved only for the isolated model, whereas the protocol for the private coin version is in the pre-coordinated model.

model. We mention that [11, Thm. 3.1] asserts that a natural class of standard non-adaptive testers can be efficiently emulated by SIPPs [11, Thm. 3.1]. The transformation underlying the proof of [11, Thm. 3.1] is similar to our proof of Theorem 1.6, although the results and the context are different.

We briefly mention one other line of work that has considered limits to the coordination of the verifier's activities in an interactive proof. We refer to the work of [6], which is in the streaming model of interactive proofs [5, 8]. In this model, they make the distinction between protocols in which the verifier's messages to the prover may depend on the input, and protocols in which this is not allowed. The latter is a restriction on the coordination between the verifier's interaction and input-examination activities, but it is not of the type we study in this work. We remark that our pre-coordinated and isolated models do guarantee that the verifier's interaction with the prover does not depend on the input (whereas the general PO-query model does not provide this guarantee).

1.7 Organization

In Section 2 we provide a more formal description of the three models (of IPPs with PO-queries) studied in this paper. The actual contents of of this paper is presented in Section 3–5: Section 3 deals with the isolated model and contains the proof of Theorem 1.2, which provides a good warm-up towards the proofs of Part 1 of Theorem 1.4 and Theorem 1.8. The latter proofs are presented in Section 4, which deals with the pre-coordinated model, and contains also the proofs of Theorems 1.6 and 1.7. The short proof of Theorem 1.3 appears in Section 5, which deals with the general model of PO-queries.

2 Models and Notation

In order to define the various models of *proof-oblivious queries* (PO-queries) interactive proofs of proximity (IPP), we decompose the verifier into three modules, called *querying*, *interacting*, and *deciding*, and denoted Q, I, and D, respectively. The querying module (i.e., Q) is the only part that queries the input function, and the interacting module (i.e., I) is the only part that interacts with the prover. The final decision is made by the deciding module (i.e., D), which is fed with the outputs of the two other modules. The three models differ by the restrictions imposed on the coordination between them, where in all models the querying module gets no information from the other modules.

Recall that in standard interactive proofs, one denotes the output of the verifier by $\langle P, V\rangle(x)$, where x is a common input. In extensions that allow private inputs (see, e.g., [13, Def. 4.2.10]), one uses the notation $\langle P(y), V(z)\rangle(x)$, where z and y are corresponding private inputs.[11] Here we have no real common input, so the output of the *interacting module* I, which may include its entire view, is denoted $\langle P(y), I(z)\rangle$. By default (unless stated otherwise), P will denote the optimal prover strategy, so there is no need to quantify over all strategies. Note that the prover has free access to the tested function, denoted f.

The most restricted model – "isolated" modules. Here the querying and interacting modules are isolated, and feed their respective outputs (which equals their entire views of the execution)

[11] These extensions are for formulating additional features such as relatively-efficient proving and auxiliary-input zero-knowledge.

to the deciding module. Each of these modules has its own randomness; actually, the deciding module may be deterministic (since it may use randomness provided to it by one of the other modules). In this case, we write the random variable that represents the decision of the verifier as

$$D(Q^f(R_Q), \langle P(f), I(R_I)\rangle), \tag{1}$$

where R_Q and R_I are independent random variables representing the randomness of each module. We refer to systems captured by Eq. (1) as belonging to the isolated model.

The intermediate model – "pre-coordinated" modules. Here the main two modules are pre-coordinated by their shared randomness. In this case, we write the random variable representing the decision as

$$D(Q^f(R), \langle P(f), I(R)\rangle), \tag{2}$$

where R is a random variable representing the shared randomness of both module. Again, without loss of generality, the deciding module may be deterministic. We refer to systems captured by Eq. (2) as belonging to the pre-coordinated model.

The least restricted model – general PO-queries. Here, the interacting module gets the outcome of the querying module. Hence, we can write the random variable representing the decision as $D(\langle P(f), I(Q^f(R), R)\rangle)$. Actually, we may omit R from the input to I, since the output of the querying module may include R. Hence, the random variable representing the decision of the verifier is written as

$$D(\langle P(f), I(Q^f(R))\rangle). \tag{3}$$

Indeed, in this case, we can combine the interacting and deciding module, or rather integrate the deciding module in the interacting module (and write the verifier's decision as $\langle P(f), I(Q^f(R))\rangle$). We refer to systems captured by Eq. (3) as belonging to the general (PO-queries) model. An alternative definitional approach to the general PO-queries model is presented in Appendix A.1.

3 Emulating the isolated model by standard testers

In this section we prove **Theorem 1.2**, which asserts that *any property that can be verified in the isolated model using q queries and c bits of communication, can be tested using $O((c+1)\cdot q)$ queries*. We mention that this result is quite tight, since emulation cost $O((c+1)\cdot q)^{0.999}$ would contradict [20, Thm. 3.12].[12]

Proof: The key observation is that, using relatively few queries, we can approximate *the probability that D accepts a fixed function f on a fixed interaction transcript*, where the probability space is merely over the randomness of the querying module (and does not depend on the interaction transcript). Specifically, for a fixed transcript τ and function f, we define

$$p^f_\tau \stackrel{\text{def}}{=} \Pr_\omega[D(Q^f(\omega), \tau) = 1]. \tag{4}$$

[12] Specifically, the proof of [20, Thm. 3.12] presents, for any $\alpha \in (0,1)$, a property Π that cannot be tested with n^α queries such that, for any $p > 0$, there exists a MAP of Π that uses a proof of length p and makes $\text{poly}(1/\epsilon)\cdot n^{\alpha+o(1)}/p$ *PO-queries*.

Note that this value can be approximated up to an ϵ additive term with error probability δ by invoking Q^f for $O(\epsilon^{-2}\log(1/\delta))$ times. We stress that these invocations of Q^f are oblivious of the value of τ, and so the same invocations can be used towards approximating several p^f_τ's (while increasing the error bound by a corresponding factor).

Likewise, we can define the corresponding value also for partial transcripts: When the next message is by the verifier, and $C_{\tau'}$ denotes the set of random choices that are consistent with the partial transcript τ', we have

$$p^f_{\tau'} = \mathrm{E}_{\omega\in C_{\tau'}}\left[p^f_{\tau',\alpha(\omega,\tau')}\right] \tag{5}$$

where $\alpha(\omega,\tau')$ is the next message sent by the verifier (on partial transcript τ') when using randomness ω. When the next message is by the prover, we have

$$p^f_{\tau'} = \max_\beta\left\{p^f_{\tau',\beta}\right\} \tag{6}$$

where β represents the prover's message. Indeed, these values (i.e., the $p^f_{\tau'}$'s) represent the (conditional) probability that the verifier accepts f, given the partial transcript τ'. In particular, the value that corresponds to the empty transcript (i.e., p^f_λ) represents the probability that the verifier accepts f, when interacting with an optimal prover strategy.

Another important observation is that the values that correspond to all partial transcripts can be computed based on the values that correspond to all the full transcripts. This is done using Eq. (5) and Eq. (6), just as it is done in the case of standard interactive proof systems; indeed, we may view the process Q^f as a last message by the verifier in such an interaction, and average over it. (Indeed, the computational complexity of scanning $C_{\tau'}$ (in the case of a verifier message) and all possible prover messages (in the case of a prover message) is irrelevant here.)

The crucial point is that we can approximate all p^f_τ, where τ is a full transcript, up to a constant additive deviation, using $O((\log 2^c)\cdot q) = O((c+1)\cdot q)$ queries to f. The point is that the same $O(c+1)$ invocations of Q^f can be used to approximate all 2^c values p^f_τ's. Based on these values, we can approximate all $p^f_{\tau'}$ for all partial τ', including the empty one (i.e., λ). Now, as noted above, p^f_λ is the probability that the verifier (Q,I,D) accepts f; that is,

$$p^f_\lambda = \mathrm{E}_{\tau\leftarrow\langle P(f),I\rangle)}\left[\Pr_\omega[D(Q^f(\omega),\tau)=1]\right] = \Pr_{\omega,\omega'}[D(Q^f(\omega),\langle P(f),I(\omega')\rangle))=1],$$

where the last equality relies on the definition of the isolated model. Hence, $p^f_\lambda \geq 2/3$ if f has the property, whereas $p^f_\lambda \leq 1/3$ if f is far from the property, and approximating p^f_λ up to an additive deviation of 0.1 allows for distinguishing these two cases. We stress that the analysis, which is rooted at Eq. (4), does not presume that the queries are non-adaptive. ■

Remark: Note that *c can be upper bounded by $\widetilde{O}(\ell)$, where ℓ is the total length of the prover's messages*. To prove the latter claim, we note that an averaging step taken at a node of the max-averaging game-tree (corresponding to Eq. (5)) can be approximated by sampling $\mathrm{poly}(\ell)$ of its children, such that with probability at least $1-\exp(-\Omega(\ell^2))$ the approximation is within an additive term of $0.01/\ell$. Hence, the number of leaves in the pruned game-tree is $\mathrm{poly}(\ell)^\ell$, and the claim follows by applying the union bound. Actually, *we can use $c = \ell + O(r\cdot\log\ell)$, where r denotes the number of rounds*, since in this case the number of leaves in the pruned game-tree is $2^\ell\cdot\mathrm{poly}(\ell)^r$.

4 On the power of the pre-coordinated model

Building on the proof of Theorem 1.2, which was presented in Section 3, we first prove Part 1 of Theorem 1.4, which asserts that standard testers can efficiently emulate *one-round* IPPs of the pre-coordinated model. This emulation is presented in Section 4.1, and we shall revisit its ideas in Section 4.4, when proving Theorem 1.8 (which refers to public-coin systems). But before doing this revisiting, we present some positive results (i.e., results that illustrate the power of the pre-coordinated model). Specifically, in Section 4.2 we prove Theorem 1.6 (and its extension (i.e., Theorem 4.7)), and in Section 4.3 we prove Theorem 1.7. (Part 2 of Theorem 1.4 is proved in Section 4.2, and serves as warm-up towards the proofs of Theorems 1.6 and 4.7.)

4.1 Emulating the one-round version by standard testers

The following result implies Part 1 of Theorem 1.4, which referred to the total communication complexity of the IPP.

Theorem 4.1 (one-round IPPs of the pre-coordinated model are efficiently emulated by standard testers): *Suppose that* Π *is a property that can be verified in the pre-coordinated model by a one-round system that uses* q *queries and a proof-message of length at most* ℓ*. Then,* Π *has a standard tester of query complexity* $O((\ell+1)\cdot q)$*.*

Proof: As stated upfront, the proof builds on the proof of Theorem 1.2, which was presented in Section 3. Here we consider the one-round version of the pre-coordinated model, in which the verifier sends a single message answered by a single message of the prover. In this case, we let $\mathtt{msg}(\omega)$ denote the verifier's message (sent by I), when using randomness ω, and $P(f,\alpha)$ denote the response of the (optimal) prover upon receiving the message α and having input $f:[n]\to[m]$. Hence, $\langle P(f),I(\omega)\rangle$ can be written as $(\omega,P(f,\mathtt{msg}(\omega))$. Recalling that ω is part of $Q^f(\omega)$, we omit the appearance of ω from $\langle P(f),I(\omega)\rangle=(\omega,P(f,\mathtt{msg}(\omega))$. Consequently, in this case, we have

$$(Q^f(\omega),\langle P(f),I(\omega)\rangle)\equiv(Q^f(\omega),P(f,\mathtt{msg}(\omega))). \tag{7}$$

The key observation, which is captured by Eq. (7), is that the verifier's message does not depend on f. Hence, conditioning on $\mathtt{msg}(\omega)=\alpha$ is analogous to considering a uniform probability space over $\mathtt{msg}^{-1}(\alpha)$, where the point is that this probability space does not depend on f (i.e., $\mathtt{msg}^{-1}(\alpha)$ does not depend on $P(f,\alpha)$).

Analogously to the proof of Theorem 1.2, another key observation is that, for every fixed value of the verifier-message α (and every f), we can approximate

$$p^f_\alpha \stackrel{\text{def}}{=} \max_\beta\left\{\Pr_\omega\left[D(Q^f(\omega),\beta)=1\,\middle|\,\mathtt{msg}(\omega)=\alpha\right]\right\} \tag{8}$$

by invoking Q^f on random choices in $\{\omega:\mathtt{msg}(\omega)=\alpha\}$. Specifically, $O(\epsilon^{-2}\log(1/\delta))$ invocations suffice for approximating each $p^f_{\alpha,\beta}$ up to an ϵ additive term with error probability δ, where

$$p^f_{\alpha,\beta}\stackrel{\text{def}}{=}\Pr_\omega[D(Q^f(\omega),\beta)=1|\mathtt{msg}(\omega)=\alpha].$$

As in Section 3, setting $\delta=\Omega(2^{-\ell})$ allows us to approximate (for each α) all $p^f_{\alpha,\beta}$'s (up to a constant term, with high probability) by using $O(\ell+1)$ invocations of Q^f. The crucial point is

that, *for a fixed α, all $p^f_{\alpha,\beta}$'s can be approximating by using the same invocations of Q^f*, because these invocations refer to sampling uniformly the same set (i.e., $\{\omega : \mathtt{msg}(\omega) = \alpha\}$). (Indeed, the computational complexity of sampling $\mathtt{msg}^{-1}(\alpha)$ is irrelevant here.) Hence, for each α, making $O((\ell+1)\cdot q)$ queries to f allows to approximate the value of p^f_α up to an additive deviation of 0.01 with probability at least 0.99.

Lastly, note that

$$\Pr_\omega\left[D(Q^f(\omega), P(f, \mathtt{msg}(\omega))) = 1\right] = \mathrm{E}_\omega\left[p^f_{\mathtt{msg}(\omega)}\right] \tag{9}$$

and so it suffices to approximate the r.h.s of Eq. (9) up to a constant deviation, which can be done by sampling a constant number of ω's. Specifically, for $t = O(1)$, selecting $\omega_1, ..., \omega_t \in [n]$ uniformly at random, with probability at least 0.9 it holds that

$$\mathrm{E}_\omega\left[p^f_{\mathtt{msg}(\omega)}\right] = \frac{1}{t}\cdot\sum_{i\in[t]} p^f_{\mathtt{msg}(\omega_i)} \pm 0.1,$$

which establishes the foregoing claim. Hence, making $O((\ell+1)\cdot q)$ queries to f allows to approximate the l.h.s of Eq. (9) up to an additive deviation of 0.11 with probability at least 0.89. Recalling that the l.h.s of Eq. (9) equals the probability that the verifier (of the pre-coordinated model) represented by (Q, I, D) accepts f, it follows that Π can be tested within query complexity $O((\ell+1)\cdot q)$. (Note that, like in the proof of Theorem 1.2, we do not need to assume that Q is non-adaptive.) ■

Digest and reflection. We stress that it was necessary and sufficient to approximate $p^f_{\mathtt{msg}(\omega)}$ for a constant number of ω's, and that for each ω the value of $p^f_{\mathtt{msg}(\omega)}$ can be approximated by taking $O(\ell+1)$ samples from the same probability space (i.e., the uniform distribution over $\{s : \mathtt{msg}(s) = \mathtt{msg}(\omega)\}$). In contrast, trying to extend the argument to 3/2-round interactions would have required us to approximate $p^f_{\beta',\mathtt{msg}(\omega,\beta')}$, where $\mathtt{msg}(\omega,\beta')$ is the verifier's message (which depends on the prover's first message β'), for all possible β''s. The problem is that each of these β''s requires sampling from a potentially different probability space (i.e., the set $\{s : \mathtt{msg}(s,\beta') = \mathtt{msg}(\omega,\beta')\}$), whereas the different samples may lead to different queries of Q^f. As we shall see in Theorem 4.5, this difficulty is inherent. We observe that these difficulties are avoided if one aims at emulating 3/2-round systems (of the pre-coordinated model) by MAPs (rather than by testers).

Theorem 4.2 (3/2-round IPPs of the pre-coordinated model are efficiently emulated by MAPs): *Suppose that Π is a property that can be verified in the pre-coordinated model by a 3/2-round system that uses q queries and prover messages of total length ℓ. Then, Π can be verified by a MAP of query complexity $O((\ell+1)\cdot q)$ and proof-length ℓ.*

(In general, for any $r \geq 0$, every $(r+1)$-round IPPs of the pre-coordinated model can be efficiently emulated by an r-round (general) IPP, but this is interesting only for $r \in \{0, 0.5\}$ because for $r \geq 1$ any $(r+1)$-round IPP can be efficiently emulate by an r-round IPP.)

Proof Sketch: Building on the proof of Theorem 4.1, we observe that it suffices to approximate $p^f_{\beta'}$ for a single β' that is sent by the prover (as its first message). That is, the proof in the MAP system consists of the first message sent by the prover in the 3/2-round system, and the verifier just approximates $p^f_{\beta'}$ exactly as the tester approximated p^f_λ in the proof of Theorem 4.1. ■

Separation of 1-round systems. Considering the set of permutations over $[n]$, denoted PERM, and using Theorem 4.1, we demonstrate a separation between 1-round systems in the pre-coordinated model and 1-round system with general PO-queries.

Corollary 4.3 (1-round IPP with general PO-queries vs 1-round IPP in the pre-coordinated model):

- PERM *can be verified by a one-round IPP with proof-oblivious queries, using* $O(\epsilon^{-1} \log n)$ *communication and* $O(1/\epsilon)$ *queries.*
- *If* PERM *can be verified in the pre-coordinated model by a one-round system of query complexity* q *and a prover message of length* ℓ*, then* $(\ell + 1) \cdot q = \Omega(\sqrt{n})$*.*

Proof: As detailed in the proof of Theorem 5.1, the first part follows by using a verifier that selects uniformly $i \in [n]$, queries the function at i, sends the image to the prover, and accepts if and only if the prover returns i. The second item follows by combining Theorem 4.1 with the (known) lower bound on the query complexity of testing PERM (see [18, Lem. 4.3], which is also used in the proof of Theorem 5.1). ■

MAPs vs 1-round systems of the pre-coordinated type. Recall that MAPs are known to be stronger than standard testers [20, 15]. Combining this fact with Theorem 4.1, we infer that MAPs cannot be efficiently emulated by one-round systems in the pre-coordinated model.

Corollary 4.4 (MAP vs 1-round pre-coordinated model): *There exists a property* Π *of* n*-bit strings that satisfies the following two conditions.*

- Π *can be verified by a MAP using a proof of length* $O(\log n)$ *and* $\mathrm{poly}(1/\epsilon)$ *queries.*
- *If* Π *can be verified in the pre-coordinated model by a one-round system of query complexity* q *and a prover message of length* ℓ*, then* $(\ell + 1) \cdot q = \Omega(n^{0.999})$*.*

We comment that the MAP used in the first item uses non-adaptive queries, but it is not the case that each of these queries is uniformly distributed in $[n]$ (and so Theorem 1.6 does not apply to it).

4.2 Emulating some MAPs by 3/2-rounds of the pre-coordinated model

In contrast to Theorem 4.1, we show that there is a gap between the power of 3/2-round systems and standard testers. This implies that 3/2 rounds of the pre-coordinated model are significantly stronger than one round.

Theorem 4.5 (the pre-coordinated model cannot be efficiently emulated by testers): *There exists a property of* n*-bit strings that can be verified by a* 3/2*-round IPP in the pre-coordinated model using* $O(\epsilon^{-1} \log n)$ *communication and* $O(1/\epsilon)$ *queries, but testing it requires* $\Omega(\sqrt{n})$ *queries.*

This establishes Part 2 of Theorem 1.4. Theorem 4.5 is proved by transforming a known MAP for the property into a 3/2-round system in the pre-coordinated model; this transformation illustrates the idea that underlies our emulation of a natural class of MAPs by 3/2-rounds of the pre-coordinated model (see proof of Theorem 4.6).

Proof: Consider the following MAP for the set $S = \{uuvv : uv \in \{0,1\}^{n/2}\}$, which is a property that requires $\Omega(\sqrt{n})$ queries for testing.[13] The prover sends the length of u, and the verifier checks (by using adequate queries) that "matching" locations (selected at random) do hold the same value, where $i < j$ are matching locations if either $i \in [|u|]$ and $j = |u| + i$ or $i \in [2|u| + 1, 2|u| + |v|]$ and $j = |v| + i$. The verifier in our proof system (for the pre-coordinated model) proceeds as follows: *It first queries the input at a random location, denoted $i \in [n]$, and the initiates a 3/2-round interaction with the prover. When the prover sends $|u|$, the verifier asks it to provide the value of the input string at location i', where i' is selected at random to be either i or the location matched to i* (e.g., if $i \in [|u|]$, then i' is selected uniformly in $\{i, |u| + i\}$). *When the prover answers, the verifier accepts if and only if the prover's answer equals the answer obtained from the oracle.*

Note that this procedure can be implemented in the pre-coordinated model using 3 messages, because the pre-coordinated randomness may include i as well as a bit that is used to select either i or its matched location. The soundness error is upper-bounded by observing that if the tested string is ϵ-far from S, then the prover fails with probability greater than ϵ, because the hypothesis implies that more than $2\epsilon n$ locations are matched to a location that holds a different value (where the matching is according to the first message sent by the prover). Hence, this 3/2-round system has soundness error less than $1 - \epsilon$, while using one query and a logarithmic amount of communication.

The soundness error can be reduced by parallel repetition, just as in the case of standard interactive proof systems (where the crucial observation is that the value of an execution in the pre-coordinated model can be represented by a max-average game-tree (see Appendix A.2)). Note that the resulting 3/2-round system is in the pre-coordinated model and that it uses $O(1/\epsilon)$ queries and $O(\epsilon^{-1} \log n)$ bits of communication. ■

Generalizing the emulation that underlies the proof of Theorem 4.5. We now apply the idea underlying the proof of Theorem 4.5 to emulate a natural class of MAPs by 3/2-round IPPs of the pre-coordinated model. We refer to MAPs that make non-adaptive queries, where non-adaptivity refers to the indepedence of queries on the answers to prior queries. Furthermore, we assume that for every (alleged) proof, each of the queries made by the verifier is uniformly distributed, although there may be dependencies between the queries. For simplicity of exposition, we shall also assume that these MAPs have perfect completeness (i.e., never err on inputs in the property). Note that the MAP that was used in the proof of Theorem 4.5 is of this type.

Theorem 4.6 (emulating MAPs that use non-adaptive and uniformly distributed queries): *Suppose that a property of functions from $[n]$ to Σ can be verified by a MAP of perfect completeness that uses a proof of length ℓ and q non-adaptive queries that are each uniformly distributed in $[n]$. Then, this property can be verified in the pre-coordinated model by a 3/2-round system that uses $O(q)$ queries and total communication $O(q \cdot \ell + q^2 \cdot (\log_2 n + \log_2 |\Sigma|))$.*

We stress that the hypothesis only requires that each query of the MAP verifier is uniformly distributed; their joint distribution, which may also depend on the given proof-string, is arbitrary (beyond this requirement). We mention that the resulting 3/2-round IPP uses uniformly and independently distributed queries (i.e., it is actually sample-based (cf. [11])).

[13] The lower bound was originally proved in [1] for the context-free set $S = \{uu^\top vv^\top : uv \in \{0,1\}^{n/2}\}$, but the same proof applies to the current set. Essentially, in order to distinguish a random string in S from a totally random n-bit long string, one must query the tested string $uuvv$ at two locations that are at distance $|u|$ or $|v|$ apart.

Proof: Let V be such a MAP verifier. Denote the (non-adaptive) queries made by V after receiving the alleged proof $\pi \in \{0,1\}^\ell$ by $Q_1(r,\pi),....,Q_q(r,\pi)$, where $r \in \{0,1\}^m$ is V's randomness, and denote V's final verdict (when given oracle access to $f:[n] \to \Sigma$) by $V(r,\pi,f(Q_1(r,\pi)),....,f(Q_q(r,\pi)))$. Now, the (three-message) PO-queries system proceed as follow.

Construction 4.6.1 (the basic system): *Using randomness $\omega = (i,j,\omega') \in [n] \times [q] \times \{0,1\}^{m'}$, for some adequate m', the system proceed as follow.*

- *The querying module obtains the value of $f(i)$, by querying f at i, and hands $(i,f(i))$ to the deciding module.*
- *The interacting module starts by receiving a message π from the prover; the honest/designated prover will send the proof sent in the MAP. Next, using its coins ω, it determines a random $r \in \{0,1\}^m$ such that $Q_j(r,\pi)=i$, where the selection of r in the set $\{s \in \{0,1\}^m : Q_j(s,\pi)=i\}$ is determined by ω'. The verifier sends $\overline{q} \stackrel{\text{def}}{=} (Q_1(r,\pi),....,Q_q(r,\pi))$ to the prover, and denotes the prover's response by $\overline{a} \stackrel{\text{def}}{=} (a_1,...,a_q)$.* (Needless to say, the honest/designated prover sets $a_k = f(Q_k)$ for every $k \in [q]$.)

  (Hence, the triplet of messages exchanged is $(\pi,\overline{q},\overline{a})$, and $(\pi,\overline{a})$ is sent to the deciding module along with j and r.)
- *The deciding module, receiving (i,v) from the querying module and $(j,r,\pi,a_1,...,a_q)$ from the interacting module, accepts if and only if both $a_j = v$ and $V(r,\pi,a_1,....,a_q)=1$.*

The total (verifier–prover) communication of this system is $\ell + q \cdot \lceil \log_2 n \rceil + q \cdot \lceil \log_2 |\Sigma| \rceil$.

Clearly, the completeness error of Construction 4.6.1 is upper-bounded by the completeness error of the MAP, which we assumed to be zero. We shall show next that the soundness error is upper-bounded by $1-(2/3q)$; hence, we derive a standard IPP (with error $1/3$) by repeating the foregoing system $O(q)$ times, in parallel, and accepting if and only if all invocations accept.

Claim 4.6.2 (soundness claim): *If f is ϵ-far from the property, then Construction 4.6.1 rejects with probability at least $2/3q$.*

Proof: Recall that for every f that is ϵ-far from the property, and for every π, it holds that

$$\Pr_r[V(r,\pi,f(Q_1(r,\pi)),....,f(Q_q(r,\pi)))=1] \leq 1/3.$$

The key observation is that, for every π and $j \in [q]$, the following two processes generate identical distributions:

1. Selecting $i \in [n]$ uniformly at random and outputting $(Q_1(r,\pi),...,Q_q(r,\pi))$ such that r is uniformly distributed in $\{s \in \{0,1\}^m : Q_j(s,\pi)=i\}$.
2. Selecting $r \in \{0,1\}^m$ uniformly at random, and outputting $(Q_1(r,\pi),...,Q_q(r,\pi))$.

Hence, for any π and any strategy P' of answering the verifier's message (in the IPP), it holds that

$$\Pr_{i,j,r}\left[\begin{array}{ll} & V(r,\pi,P'(Q_1(r,\pi),....,Q_q(r,\pi)))=1 \\ \& & P'(Q_1(r,\pi),....,Q_q(r,\pi))) = (f(Q_1(r,\pi)),...,f(Q_q(r,\pi))) \end{array} \;\middle|\; Q_j(r,\pi)=i\right] \leq 1/3$$

which implies

$$\Pr_{i,j,r}\left[\begin{array}{ll} & V(r,\pi,P'(Q_1(r,\pi),....,Q_q(r,\pi)))=0 \\ \vee & P'(Q_1(r,\pi),....,Q_q(r,\pi)))_j \neq f(Q_j(r,\pi)) \end{array}\;\middle|\; Q_j(r,\pi)=i\right] \geq 2/3q$$

because $(p_1,...,p_q) \neq (f_1,...,f_q)$ implies $\Pr_j[p_j \neq f_j] \geq 1/q$. The claim follows. ■

Conclusion. Using error reduction (and accepting iff each of the $O(q)$ parallel executions accepts), the theorem follows.

(Note that the perfect completeness of the MAP was only used in order to employ a consensus vote for acceptance (i.e., we accepted iff all invocations of Construction 4.6.1 accept).) ■

A straightforward extension. The argument extends to the case that for every $j \in [q]$ the distribution of $Q_j(\cdot,\pi)$ is oblivious of π. In this case, the querying module selects its query according to the distribution $Q_j(\cdot,\pi_0)$, where π_0 is an arbitrary fixed ℓ-bit string. That is, the joint randomness has the form $(s,j,\omega') \in \{0,1\}^m \times [q] \times \{0,1\}^{m'}$, and the query i is set to $Q_j(s,\pi_0)$. Actually, note that, without loss of generality, the Q_j's are identical (i.e., consider permuting the queries at random); hence, the real point in this extension is that the distribution of $Q_j(\cdot,\pi)$ need not be uniform, but rather needs to be oblivious of π.

Generalizing Theorem 4.6. Making the foregoing assumption (i.e., $Q_j(\cdot,\pi) \equiv Q_1(\cdot,\pi)$ for every j and π), we consider the case that all query distributions $Q_1(\cdot,\pi)$ are *dominated by a single distribution*. Specifically, we say that a distribution X i s γ-dominated by a distribution Y if for every z it holds that $\Pr[X=z] \leq \Pr[Y=z]/\gamma$. (Indeed, this notion is meaningful only for $\gamma < 1$, whereas every distribution is 1-dominated by itself only.)

Theorem 4.7 (emulating MAPs, a more general case): *Suppose that a property of functions from $[n]$ to Σ can be verified by a MAP that uses a proof of length ℓ and q non-adaptive queries such that all query distributions are γ-dominated by a single distribution; that is, there exists a distribution Y such that for every $j \in [q]$ and $\pi \in \{0,1\}^\ell$ the distribution $Q_j(\cdot,\pi)$ is γ-dominated by Y. Then, this property can be verified in the pre-coordinated model by a 3/2-round system that uses $O(\widetilde{q}/\gamma)$ queries and total communication $O(\widetilde{q} \cdot \ell + \widetilde{q}^2 \cdot (\log_2 n + \log_2 |\Sigma|))$, where $\widetilde{q} = O(q \log q)$.*

Recall that we may assume, without loss of generality, that for every j and π it holds that $Q_j(\cdot,\pi) \equiv Q_1(\cdot,\pi)$.

Proof: By employing error reduction to the MAP, we may assume that it uses $\widetilde{q}$ queries and that the error probabilities (both in the completeness and soundness condition) are upper-bounded by $1/3\widetilde{q}$. Specifically, we invoke the query-based verification of the MAP-proof $O(\log q)$ times. (Note that the proof length (i.e., ℓ) remains intact.) Furthermore, by randomly permuting the queries, we may assume that, for every $\pi \in \{0,1\}^\ell$, it holds that $Q_j(\cdot,\pi)$ and $Q_1(\cdot,\pi)$ are identically distributed (and all are γ-dominated by Y). Letting $X_\pi \stackrel{\text{def}}{=} Q_1(\cdot,\pi)$ be a random variable that represents the distribution of each query, we consider the following system, which generalizes Construction 4.6.1.

Construction 4.7.1 (the generalized basic system): *Let $t = \lceil 1/\gamma \rceil$. Abusing notation, we assume that the following y_i's are selected according to Y, which is not necessarily uniform over $[n]$. In this case, using randomness $\omega = (y_1,...,y_t,\omega_1,....,\omega_t,j,\omega') \in [n]^t \times (\{0,1\}^{m'})^t \times [\widetilde{q}] \times \{0,1\}^{m'}$, for some adequate m', the system proceeds as follow.*

- *The querying module obtains the value of $f(y_i)$, for every $i \in [t]$, by querying f at y_i, and hands all $(y_i, f(y_i))$'s to the deciding module.*

- *The interacting module proceeds as follows.*

  - *Upon receiving a message π from the prover, the verifier uses the ω_i's to sub-sample the y_i's such that each y_i is included in the sub-sample with probability*

    $$\gamma \cdot \frac{\Pr[X_\pi = y_i]}{\Pr[Y = y_i]} \in [0,1],$$

    where the upper bound is guaranteed by γ-domination. If the sub-sample contains no elements, the interacting module halts while sending a special symbol to the deciding module (which will cause this module to accept).

  - *Otherwise, the interacting module selects the first y_i in the sub-sample, denoted y, and continues as in Construction 4.6.1. That is, using its coins ω, it determines a random $r \in \{0,1\}^m$ such that $Q_j(r,\pi) = y$, where the selection of r in the set $\{s \in \{0,1\}^m : Q_j(s,\pi) = y\}$ is determined by ω'. Next, it sends $\overline{q} \stackrel{\text{def}}{=} (Q_1(r,\pi), ..., Q_{\widetilde{q}}(r,\pi))$ to the prover, and denotes the prover's respond by $\overline{a} \stackrel{\text{def}}{=} (a_1, ..., a_{\widetilde{q}})$.*
    Note that each value y is selected as a sample of Y and included in the sub-sample with probability

    $$\Pr[Y = y] \cdot \gamma \cdot \frac{\Pr[X_\pi = y]}{\Pr[Y = y]} = \gamma \cdot \Pr[X_\pi = y]$$

    and that the sub-sample is empty with probability $(1-\gamma)^t \leq e^{-1}$. Furthermore, when the sub-sample is non-empty, the selected r is uniformly distributed.

  (Hence, the triplet of messages exchanged is $(\pi, \overline{q}, \overline{a})$, and $(\pi, \overline{a})$ is sent to the deciding module along with j, i and r. Recall that $y_i = y = Q_j(r,\pi)$; that is, i is the index of the first query that was included in the sub-sample.)

- *The deciding module, upon receiving $(y_1, v_1), ..., (y_t, v_t)$ from the querying module and $(i, j, r, \pi, \overline{a})$ from the interacting module, where $\overline{a} = (a_1, ..., a_{\widetilde{q}})$, accepts if and only if both $a_j = v_i$ and $V(r, \pi, a_1, ..., a_{\widetilde{q}}) = 1$.* (In case the deciding module received a special symbol from the interacting module, it just accepts.)

Construction 4.6.1 is obtained as a special case when $\gamma = 1$ (and in this case $t = 1$) and Y is the uniform distribution on $[n]$. The query complexity of Construction 4.7.1 is t, and the communication complexity is $\ell + \widetilde{q} \cdot \lceil \log_2 n \rceil + \widetilde{q} \cdot \lceil \log_2 |\Sigma| \rceil$. The completeness error of Construction 4.7.1 is upper-bounded by the completeness error of the MAP, which is at most $1/3\widetilde{q}$. As shown next, the soundness error of Construction 4.7.1 is smaller than $1 - (0.4/\widetilde{q})$; that is, it rejects ϵ-far functions with probability at least $0.4/\widetilde{q}$.

Claim 4.7.2 (soundness claim): *If f is ϵ-far from the property, then Construction 4.7.1 rejects with probability at least $0.4/\widetilde{q}$.*

Proof sketch: Note that, with probability at least $1 - e^{-1} > 0.6$, Construction 4.7.1 "emulates" the execution of MAP on a random r. Hence, if the prover provides answers according to f, then the

execution rejects with probability at least 2/3. On the other hand, whenever the prover provides an answer that does not fit the value of f, it is detected with probability at least $1/\widetilde{q}$. Hence, f is rejected with probability at least $0.6 \cdot \frac{2}{3} \cdot \frac{1}{\widetilde{q}} = 0.4/\widetilde{q}$. ■

Conclusion. Using the fact that the completeness error is smaller than $1/3\widetilde{q}$ and employing error reduction, the theorem follows. Specifically, we execute Construction 4.7.1 for $O(\widetilde{q})$ times, in parallel, and accept if and only if at least $1 - 0.35/\widetilde{q}$ fraction of these executions accept. ■

MAPs vs 3/2-round systems of the pre-coordinated type. Theorems 4.2 and 4.7 sandwich the power of 3/2-round IPPs of the pre-coordinated model in terms of MAPs. On the one hand, these 3/2-round systems are efficiently emulated by MAPs (Theorem 4.2), whereas on the other hand they can efficiently emulate a natural subclass of MAPs (Theorem 4.7). We conjecture that not all MAPs can be efficiently emulated by 3/2-round IPPs of the pre-coordinated model.

4.3 On the power of the multi-round version of the pre-coordinated model

One simple but extremely useful observation regarding multi-round IPPs in the pre-coordinated model is that they can perform the celebrated sum-check protocol of [22]. Recall that this protocol is used for verifying claims of the form $\sum_{x \in H^m} f(x) = v$, where $f : \mathcal{F}^m \to \mathcal{F}$ describes a low-degree polynomial, $H \subset \mathcal{F}$, and $v \in \mathcal{F}$ is a claimed value. The protocol proceeds in m rounds such that in the i^{th} round the verifier sends a uniformly distributed $\omega_i \in \mathcal{F}$, and the final query to f is at the point $\omega_1 \cdots \omega_m \in \mathcal{F}^m$. Indeed, all checks performed during the execution, can be performed at the end (by the deciding module). Given these fact, Theorem 1.5 follows by observing that the proof of [20, Thm. 3.28] refers to this specific property (called Tensor/Sub-cube Sum). For details see Appendix A.3.

We now move to proving Theorem 1.7, which asserts that any public-coin IPP for any property of low-degree polynomials can be efficiently emulated in the pre-coordinated model. Specifically, for a finite field $\mathcal{F}$, number of variables $m \in \mathbb{N}$ and (total) degree bound $d \in \mathbb{N}$, we consider any subset of the set of all m-variate polynomials of total degree at most d over $\mathcal{F}$.

Theorem 4.8 (the main claim of Theorem 1.7): *Let $\Pi_{\mathcal{F},m,d}$ be a property* (i.e., a subset of the set) *of m-variate polynomials of total degree at most d over $\mathcal{F}$. Suppose that $\Pi_{\mathcal{F},m,d}$ can be verified by a r-round public-coin IPP system with query complexity q and communication complexity c, and that $d \leq |F|/O(1+\log_{|\mathcal{F}|} q)$. Then, $\Pi_{\mathcal{F},m,d}$ can be verified in the pre-coordinated model with $O(q)+O(d)$ queries and $O(c) + q \cdot (d+m) \cdot O(\log(q+|\mathcal{F}|))$ bits of communication. Furthermore, this IPP uses $r+q$ rounds.*

Note that the tested functions have size $n = |\mathcal{F}^m|$.

Proof: The key idea is to verify the value of a low-degree polynomial, $f : \mathcal{F}^m \to \mathcal{F}$, at an arbitrary point of interest, $\overline{x} \in \mathcal{F}^m$, by querying f at a random point, $\overline{u} \in \mathcal{F}^m$, and asking the prover to provide the univariate polynomial p that describes the value of f on a line (or low degree curve) that connects the points $\overline{x}$ and $\overline{u}$. The verifier can keep a cheating prover at bay by comparing the value of p at (the point that corresponds to) $\overline{u}$ with the value $f(\overline{u})$, and use the value of p at (the point that corresponds to) $\overline{x}$. This idea was applied quite extensively in the study of PCPs, starting with [10, 2], but it seems that its first appearance in the context of IPPs with proof-oblivious queries is due to [18, Lem. 5.4].

Turning to our specific context, we apply the foregoing idea to the queries that the original verifier wishes to make (during its interaction with the prover). We stress that this can be done only after the original interaction was emulated, because the prover should remain ignorant as to these queries, whereas the aforementioned line reveals a lot of information about a desired query $\overline{x}$. In fact, such a line reveals $\log_2 |\mathcal{F}|^{m-1}$ bits of information about $\overline{x}$ and ditto regarding $\overline{u}$ (but the $\log_2 |\mathcal{F}|$ bits of uncertainty regarding $\overline{u}$ suffice for the verification process). However, since the original IPP is of the public-coin type, *we can emulate the original interaction before asking the prover to provide the values of the original queries*, and establish the correctness of these values later, as outlined above (i.e., using the fact that the verifier has obtained the values of the polynomial at some random points).

The foregoing description suffices for the case that the tested function f is a polynomial of total degree at most d, but this is not necessarily the case in general. Nevertheless, by employing a low-degree test, we can guaranteed that f is close to a degree d polynomial f'. (Indeed, our verifier needs to check that f is a low-degree polynomial; it cannot rely on the fact that the original IPP may perform such a test anyhow, since the soundness of the emulation relies on f being close to a low degree polynomial.)

Assuming that f is close to a degree d polynomial f' implies that, *in expectation*, the fraction of disagreement between f and f' on a random line that passes through $\overline{x}$ equals the global fraction of disagreement between f and f'. But, for the soundness analysis, we need the fraction of disagreement (on such a line) to be small with probability at least $1 - o(1/q)$. While points on a random line that pass through $\overline{x}$ are not random enough to support this claim, points on a random curve of degree $t = O(\lceil \log_{|\mathcal{F}|} q \rceil)$ that passes through $\overline{x}$ are random enough. Hence, we use such curves rather than lines. On input $f : \mathcal{F}^m \to \mathcal{F}$, our (pre-coordinated model) IPP proceeds as follows.

- The *querying module* queries the function f at q random points, denoted $\overline{u}_1, ..., \overline{u}_q \in \mathcal{F}^m$. (These points will be used by the interacting module.)

  In addition, it makes $O(d)$ queries that correspond to $O(1)$ random lines in $\mathcal{F}^m$; these queries will be used by a low-degree test (which will be evaluated by the decision module). This low-degree test is invoked with proximity parameter set to some small constant (e.g., 0.01).

- The *interacting module*, denoted I, proceeds in two stages. In the first stage, I emulates an execution of the original verifier with the prover, *while relying on the fact that the verifier's messages in this interaction are oblivious of the oracle's answers* (per the public-coin hypothesis).

  (We stress that the original verifier may make queries during its interaction with the prover, and these queries may depend on the prover's messages, but the original verifier's messages are independent of the oracle's answers (as well as of the prover's messages).)

  In the second stage, I emulates the original verifier's access to the oracle f. Specifically, letting $\overline{x}_i$ denote the i^{th} query of the original verifier, I sends the canonical description of a random degree t curve $C_i : \mathcal{F} \to \mathcal{F}^m$ that connects $\overline{x}_i$ and $\overline{u}_i$, to the prover, who is supposed to answer with a degree $t \cdot d$ univariate polynomial that describes the value of f on the curve C_i. Denoting the prover's response by P_i, the answer to the query $\overline{x}_i$ is defined to equal the value of P_i at d_i such that $C_i(d_i) = \overline{x}_i$.

  We stress that the query $\overline{x}_i$ is determined by the original verifier's randomness, the transcript of the interaction with the prover, and the answers provided to prior queries. Due to the

latter dependence, the emulation of the q queries takes place in q rounds.

- The *deciding module* accepts if and only if the following three conditions hold:

  1. The values of f provided for the low-degree test (coupled with the description of the $O(1/\epsilon)$ corresponding lines) lead this test to accept.
  2. The original verifier would have accepted given the interaction transcript generated in the first stage and the answers to the queries defined by the polynomials sent by the prover in the second stage. (Recall that the answer to the i^{th} query (i.e., $\overline{x}_i$) is defined to equal the value of the i^{th} polynomial (i.e., P_i) at the adequate point (i.e., d_i such that $C_i(d_i) = \overline{x}_i$).
  3. For every $i \in [q]$, the value of the i^{th} polynomial at the adequate point agrees with the value of f at $\overline{u}_i$; that is, $P_i(r_i) = f(\overline{u}_i)$, where r_i is the location of $\overline{u}_i$ on C_i (i.e., $C_i(r_i) = \overline{u}_i$).

We stress that it is not important to hide d_i from the prover, but it is crucial to hide r_i from the prover. (Indeed, we could have defined C_i to be a random degree t curve such that $C_i(0) = \overline{x}_i$ and $C_i(r_i) = \overline{u}_i$ for a random $r_i \in \mathcal{F}$.)

Evidently, the foregoing system is of the pre-coordinated type, and its completeness follows from the completeness of the original IPP. As for its query and communication complexities, they are determined by the two main activities: The querying module makes one query per each original query and invokes a low-degree test that uses $O(d)$ queries. In the first stage the interacting module merely emulates the original interaction, and in the second stage it sends a (degree t) curve per each original query and receives a univariate polynomial (of degree $t \cdot d$ in return). Hence, the total communication is $c + q \cdot ((t+1) \cdot \log_2 |\mathcal{F}^m| + t \cdot d \cdot \log_2 |\mathcal{F}|)$. If $t > 1$, then using $t \cdot \log_2 |\mathcal{F}| = O(\log q)$, we get a communication bound of $c + q \cdot (m+d) \cdot O(\log q)$, and otherwise the bound is $c + q \cdot (m+d) \cdot O(\log \|F|)$.

Turning to the soundness condition, we may assume that f is 0.01-close to a polynomial f' of total degree d, because otherwise the low-degree test rejects with high probability. Using the fact that the points on a random degree t curve that passes through a fixed point are $(t-1)$-wise independent, it follows that with probability at least $1 - O(t/|\mathcal{F}|)^{(t-1)/2} = 1 - o(1/q)$ the restriction of f to such a random (degree t) curve is 0.02-close to the restriction of f' to that curve. Thus, with probability $1 - o(1)$ it holds that, for each $i \in [q]$, if the prover provide a univariate polynomial that does not agree with f' on the curve C_i, then the verifier rejects with probability at least $1 - \frac{d + 0.02 \cdot |\mathcal{F}|}{|\mathcal{F}|}$. On the other hand, by the soundness of the original IPP, for at least two-thirds of the possible choices of the original verifier, if the prover provides the correct univariate polynomial, for each $i \in [q]$, then our verifier rejects. Hence, our verifier rejects with probability at least

$$\frac{2}{3} \cdot (1 - o(1)) \cdot \left(1 - \frac{d + 0.02 \cdot |\mathcal{F}|}{|\mathcal{F}|}\right)$$

which implies that the soundness error is upper-bounded by $\frac{1}{3} + o(1) + \frac{d + 0.02 \cdot |\mathcal{F}|}{|\mathcal{F}|} < 0.34 + 0.1 + 0.02 = 0.46$ (rather than by 1/3). Employing error reduction, using $O(1)$ parallel repetitions, we are done. ■

Reducing the number of rounds for non-adaptive queries: Establishing the furthermore claim of Theorem 1.7. If V makes non-adaptive queries (i.e., all its queries are determined solely by its public-coin interaction with the prover along with its own randomness), then we may reduce the additive overhead in the number of rounds that the transformation introduces from q to 1. In this case, the second stage of the interacting module may consist of a single round, in which the values of all q queries are provided (by the prover) and verified via the $\overline{u}_i$'s, in parallel. Specifically, the verifier determines all $\overline{x}_i$'s, sends canonical descriptions of the q random curves $C_1, ..., C_q : \mathcal{F} \to \mathcal{F}^m$ that connect the $\overline{x}_i$'s and the $\overline{u}_i$'s, to the prover, who is supposed to answer with q (degree $t \cdot d$) univariate polynomials that describe the value of f on these curves.

Reducing the number of queries. If we assume that V makes non-adaptive queries and that $q \cdot d \leq |\mathcal{F}|/O(\log_{|\mathcal{F}|} q)$ (rather than $d \leq |\mathcal{F}|/O(\log_{|\mathcal{F}|} q)$), then we can modify the emulation to only query f at a single point $\overline{u} \in \mathcal{F}^m$ chosen uniformly at random. This is done by replacing the q random (degree t) curves that go through the q different points, by a single curve of degree $q + t$ that goes through these q points. Specifically, assuming that $\mathcal{F}$ is of prime cardinality, we may use a random (degree $q+t$) curve $C : \mathcal{F} \to \mathcal{F}^m$ such that $C(i) = \overline{x}_i$ for every $i \in [q]$ and $C(j) = \overline{u}$ for a random $j \in \mathcal{F}$. The prover will respond with a single univariate polynomial of degree $(q+t) \cdot d$, and this response will be checked in a corresponding manner. Hence, the query complexity is reduced (from $O(q + d)$) to $O(d + 1)$. Note that the communication complexity does not increase; it may actually decrease to $O(c) + q \cdot (d + m) \cdot O(\log |\mathcal{F}|)$.

Digest, abstraction, and generalization. Polynomials of low-degree are the archetypical example of codewords of a locally testable and (strongly) locally correctable code, and this is the actual role they play in Theorem 4.8. Specifically, our proof uses the fact that codewords are *locally correctable* in a strong sense based on relatively few non-adaptive queries (to the possibly corrupted codeword) such that each of the queries is uniformly distributed in the set of locations (in the codewords). The strong sense, obtained by using low-degree curves, is that the recovered value is correct with very high probability (i.e., probability $1 - o(1/q)$, where q is as in the theorem), and, furthermore, that the queried locations induce a code of constant relative distance. Nevertheless, using the standard notion of local correction (cf., e.g., [19, Def. 3.1]), which only requires that the recovered value is correct with probability at least $2/3$, along with error reduction, would have sufficed (alas with quantitatively inferior bounds). The key observation is that the value of the corrected codeword at an arbitrary location $x \in [n]$ (i.e., the value $\overline{w}_x$, where $\overline{w}$ is the codeword closest to w)[14] can be verified in the pre-coordinated model as follows.

- The querying module queries the input $w \in \Sigma^n$ at a uniformly selected location $u \in [n]$, obtaining the value w_u.
- Letting $q_{\texttt{corr}}$ denote the query complexity of the local corrector, the interactive module selects uniformly an index $j \in [q_{\texttt{corr}}]$, and a random r such that on input $x \in [n]$ and randomness r, the j^{th} query of the local corrector equals u. The interactive module sends r (or, equiv., the sequence of $q_{\texttt{corr}}$ queries) to the prover.

[14] Recall that this is required only when w is close enough to the code; otherwise (i.e., when case w is not close enough to the code), the local tester will reject. Indeed, the local tester will be invoked with a proximity parameter that is upper-bounded by the correction distance of the local corrector.

(Here we rely on the hypothesis that the local corrector makes non-adaptive queries that are each uniformly distributed in $[n]$.)

- The deciding module accepts if and only if the j^{th} answer provided by the prover equals the value w_u. In this case the corrected value of w_x is obtained by applying the local corrector to the sequence of $q_{\mathtt{corr}}$ answers provided by the prover.

Note that if the prover does not provide correct answers (i.e., answers that yield a wrong value for $\overline{w}_x$), then the verifier rejects with probability at least $1/q_{\mathtt{corr}}$. Hence, invoking this process for $O(q_{\mathtt{corr}} \log q)$ times (and ruling by majority) yields an error probability of $o(1/q)$. Using the corresponding local tester (for codewords), and plugging this into the proof of Theorem 4.8, allows us to obtain a corresponding result for any property of the corresponding code (i.e., any subset of codewords).

Actually, it is more appealing to state the result in terms of *holographic interactive proofs*, a notion introduced in [21]. Loosely speaking, these are ordinary interactive proofs for promise problems that consist of codewords of a predetermined code $C \subset \Sigma^n$. That is, fixing the code C and some set $C' \subseteq C$, the promise problem consists of distinguishing inputs in C' from inputs in $C \setminus C'$, whereas nothing is required with respect to inputs in $\Sigma^n \setminus C$. Alternatively (as in [21, Def. 8]), one may consider the set $S \subseteq \Sigma^k$ of strings encoded by the code C (viewed as a mapping from Σ^k to Σ^n), and require that encodings of strings in S are always accepted, whereas encoding of strings in $\Sigma^k \setminus S$ are rejected (with probability at least 2/3).

Theorem 4.9 (generalization of Theorem 4.8, loosely stated): *Let $C \subset \Sigma^n$ be a code that is locally testable using $q_{\mathtt{test}}$ queries and locally correctable using $q_{\mathtt{corr}}$ non-adaptive queries that are are each uniformly distributed in $[n]$. For any $C' \subseteq C$, suppose that C' can be verified by a r-round public-coin* holographic interactive proof system *with query complexity q and communication complexity c. Then, C' can be verified in the pre-coordinated model with $\widetilde{O}(q) \cdot q_{\mathtt{corr}} + O(q_{\mathtt{test}})$ queries and $O(c) + \widetilde{O}(q) \cdot {q_{\mathtt{corr}}}^2 \cdot \log(n + |\Sigma|)$ bits of communication. Furthermore, this IPP uses $r + q$ rounds.*

The complexity bounds can be improved to $\widetilde{O}(q) + O(q_{\mathtt{test}})$ and $O(c) + \widetilde{O}(q) \cdot q_{\mathtt{corr}} \cdot \log(n + |\Sigma|)$ respectively, if the local corrector is "robust" in the (natural) sense that, when querying an input w that is sufficiently close to C, then it recovers the value correctly (with high probability) even when the sequence of answers is adversarially tampered (at a sufficiently constant rate).[15] In this case, if prover provide answers that yield a wrong value for $\overline{w}_x$, then the verifier rejects with probability $\Omega(1)$ rather than with probability at least $1/q_{\mathtt{corr}}$.

4.4 Emulating the public-coin version of the pre-coordinated model

Recall that we couldn't extend the efficient emulation of the one-round pre-coordinated model (by a standard tester) even to 3/2-round interaction (see digest in Section 4.1). This is because the verifier's message may depend arbitrarily on the prover's first message, denoted β'. Consequently, for each possible β', when we approximate $p^f_{\beta',\mathtt{msg}(\omega,\beta')}$ by sampling, we may need to sample from a different set of random strings that corresponds to the verifier message $\mathtt{msg}(\omega, \beta')$ (i.e, the set $\{s : \mathtt{msg}(s, \beta') = \mathtt{msg}(\omega, \beta')\}$), whereas different samples may lead to different queries (by Q).

[15] Note that this extra condition is required. For example, consider a local corrector that makes the same query twice and outputs a bluntly wrong value if the two answers are different (which will never happen without adversarial tampering).

In contrast, as we shall show below, making the assumption that each verifier's message does not depend on any of the previous *prover* messages, allows to extend the emulation to any number of rounds. The key notion is of proof-oblivious messages, which are defined as verifier's messages that are independent of the prover's (prior) messages. Indeed, public-coin IPPs are an important special case of using proof-oblivious messages, but the latter notion extends beyond public-coin interactions (e.g., any one-round IPP constitutes an IPP that uses proof-oblivious messages).

Theorem 4.10 (emulating multi-round IPPs of the pre-coordinated model that use proof-oblivious messages): *Let Π be a property that can be verified by an r-round IPP in the pre-coordinated model that uses proof-oblivious messages, makes q queries, and receives prover messages of total length ℓ. Then, Π has a standard tester of query complexity $(\mathrm{poly}(r) \cdot \ell)^r \cdot q$.*

Indeed, Theorem 1.8 follows as a special case. We mention that the emulation provided by Theorem 4.10 is relatively tight, since the proof system of [20, Lem. 3.29] has an r-round version that yields (via this emulation) a tester with almost optimal query complexity (see Appendix A.3). Specifically, for any $r \in [o((\log n)/\log\log n)]$, there exists a property Π of functions from $[n]$ to $[n^{1/(r+1)}]$ that cannot be tested within query complexity $n/O(\log n)^{r+1}$ but can be verified by an r-round public-coin IPP in the pre-coordinated model using $O(1/\epsilon) \cdot n^{1/(r+1)}$ queries and prover messages of total length $\widetilde{O}(n^{1/(r+1)})$.

Proof: Recalling that the verifier's messages in this model depend only on its randomness, denoted ω, we let $\mathtt{msg}_i(\omega)$ denote the i^{th} message sent by the verifier when using randomness ω. Without loss of generality (up to a factor of r), we assume that the prover's messages are of predetermined length, and let ℓ_i denote the length of the i^{th} prover's message. Indeed, $\ell = \sum_{i\in[r]} \ell_i$ equals the total length of the prover's messages.

We denote a generic message of the verifier (resp., prover) in the i^{th} round by α_i (resp., β_i). As in Section 3, we denote by $p^f_{\alpha_1,\beta_1,...,\alpha_i,\beta_i}$ the probability that the verifier accepts f, conditioned on the partial i-round transcript being $\alpha_1,\beta_1,\ldots,\alpha_i,\beta_i$. Our tester will estimate p^f_λ (up to a small constant deviation, with high probability). Towards this end, it will also be useful to consider the corresponding notation for $(i-0.5)$-round partial transcripts; that is, $p^f_{\alpha_1,\beta_1,...,\alpha_i}$ denotes the probability that the verifier accepts f, conditioned on the partial $(i-0.5)$-round transcript being $\alpha_1,\beta_1,\ldots,\alpha_i$. In particular, the following result will be used with $i=1$, while being proved by reverse induction (starting at $i=r$).

Lemma 4.10.1 (main lemma): *For each $i \in [r]$, let $\epsilon_i = \frac{r-i+1}{10r}$ and $\delta_i = O(r^2\ell')^{-i} \cdot 2^{-\sum_{j\in[i-1]}\ell_j}$, where $\ell' = \ell + r\cdot\log_2\ell$. Then, for every f, every $i\in[r]$, and every $\tau=(\alpha_1,\beta_1,\ldots,\alpha_i)$, one can estimate $p^f_{\alpha_1,\beta_1,...,\alpha_i}$ up to an additive deviation of ϵ_i with error probability at most δ_i by making*

$$q_i \stackrel{\text{def}}{=} O(r^2\cdot\ell')^{r-i+1}\cdot q$$

queries to f. Furthermore, the joint distribution of these q_i queries depends only on the α_j's ($j\in[i]$), and is independent of the β_j's ($j\in[i-1]$).

In particular, for every α_1, we can approximate $p^f_{\alpha_1}$ up to an additive deviation of 0.1 with probability at least 0.99 by making $O(r^2\ell')^r\cdot q$ queries. As in Section 4.1, we are done by observing that it suffices to approximate $p^f_{\alpha_1}$ for a constant number of random α_1's (as drawn according to $\mathtt{msg}_1$).

Proof: As stated above, the lemma is proved by (reverse) induction on i. For the base case ($i = r$), we fix a $(r - 0.5)$-round transcript $\tau_r = (\alpha_1, \beta_1, \ldots, \beta_{r-1}, \alpha_r)$ and observe that

$$p^f_{\tau_r} = \max_{\beta_r} \left\{ p^f_{\tau_r, \beta_r} \right\}$$

Now, as for the one-round case (treated in Section 4.1), for this fixed τ_r, we can approximate all $p^f_{\tau_r,\beta_r}$'s (i.e., for all β_r's) using the same invocations of Q^f, where Q represents the querying module of the verifier (whose final decision is based on the transcript of the interaction and on the output of the querying module).[16] This is because these approximations can be obtained by sampling from the same set $\{s : \overline{\mathtt{msgs}}_r(s) = (\alpha_1, \alpha_2, \ldots, \alpha_r)\}$, where $\overline{\mathtt{msgs}}_i(\omega) = (\mathtt{msg}_i(\omega), \mathtt{msg}_2(\omega), ..., \mathtt{msg}_i(\omega))$. Specifically, by making $O(\epsilon^{-2} \cdot \log(1/\delta))$ invocations of Q^f, we get an estimate for each individual $p^f_{\tau_r,\beta_r}$ that has additive deviation at most ϵ with probability at least $1 - \delta$. Setting $\epsilon = \epsilon_r$ and $\delta = 2^{-\ell_r} \cdot \delta_r$, and taking a union bound over all $\beta_r \in \{0,1\}^{\ell_r}$, we obtain an estimate of $p^f_{\tau_r}$ up to additive deviation ϵ_r with error probability at most δ_r using

$$\begin{aligned} O(\epsilon_r^{-2} \cdot \log(2^{\ell_r}/\delta_r)) \cdot q &= O(r^2 \cdot \log((r^2\ell')^r \cdot 2^\ell)) \cdot q \\ &= O(r^2 \cdot (\ell + r \cdot \log(r^2\ell')) \cdot q \\ &= O(r^2 \cdot (\ell + r \cdot \log \ell)) \cdot q \\ &= O(r^2 \cdot \ell') \cdot q \;=\; q_r \end{aligned}$$

queries to f. As is evident from the foregoing description, the joint distribution of these q_r queries is independent of the β_j's (since the randomness fed to Q is sampled from a space that only depends on the α_j's).

Turning to the (reversed) induction step, we assume that the claim holds for some $i > 1$, and show that it holds for $i - 1$. Let $\tau_{i-1} = (\alpha_1, \ldots, \beta_{i-1}, \alpha_{i-1})$ be some a fixed $((i-1) - 0.5)$-round transcript, and note that

$$p^f_{\tau_{i-1}} = \max_{\beta_{i-1}} \left\{ p^f_{\tau_{i-1},\beta_{i-1}} \right\} \tag{10}$$

$$= \max_{\beta_{i-1}} \left\{ \mathrm{E}_\omega \left[p^f_{\tau_{i-1},\beta_{i-1},\mathtt{msg}_i(\omega)} \middle| \overline{\mathtt{msgs}}_{i-1}(\omega) = (\alpha_1, \ldots, \alpha_{i-1}) \right] \right\} \tag{11}$$

where we use the fact that $\mathtt{msg}_i(\omega)$ is only conditioned by the value of $\overline{\mathtt{msgs}}_{i-1}(\omega)$.[17] Now, we take a sample of $t = O(\epsilon^{-2} \log(1/\delta))$ independently and uniformly distributed elements of $\{s : \overline{\mathtt{msgs}}_{i-1}(s) = (\alpha_1, \ldots, \alpha_{i-1})\}$, denote the resulting t-multiset by S, and observe that, for every β_{i-1}, with probability at least $1 - \delta$, it holds that

$$\frac{1}{t} \cdot \sum_{s \in S} p^f_{\tau_{i-1},\beta_{i-1},\mathtt{msg}_i(s)} \;=\; \mathrm{E}_\omega \left[p^f_{\tau_{i-1},\beta_{i-1},\mathtt{msg}_i(\omega)} \middle| \overline{\mathtt{msgs}}_{i-1}(\omega) = (\alpha_1, \ldots, \alpha_{i-1}) \right] \pm \epsilon. \tag{12}$$

For every $s \in S$ and every $\beta_{i-1} \in \{0,1\}^{\ell_{i-1}}$, we invoke the induction hypothesis regarding $p^f_{\tau_{i-1},\beta_{i-1},\mathtt{msg}_i(s)}$. Taking a union bound over these $t \cdot 2^{\ell_{i-1}}$ invocations (of the hypothesis), we

[16] Recall that, in the pre-coordinated model, the verifier is represented by the triple (Q, I, D), and its decision regarding f is represented by $D(Q^f(\omega), \langle P(f), I(\omega)\rangle)$, where the i^{th} message sent by $I(\omega)$ is $\mathtt{msg}_i(\omega)$.

[17] Indeed, in the special case of a public-key verifier, this conditioning is trivial (i.e., $\mathtt{msg}_i(\omega)$ is independent of the previous messages).

infer that with error probability at lesat $1 - t \cdot 2^{\ell_{i-1}} \cdot \delta_i$, all of the relevant $p^f_{\tau_{i-1},\beta_{i-1},\mathtt{msg}i(s)}$'s were estimated up to an additive deviation of ϵ_i. Combining Eq. (10)&(11) and Eq. (12), it follows that, with probability at least $1 - 2^{\ell_{i-1}} \cdot t \cdot \delta_i + \delta$, we estimate $p^f_{\tau_{i-1}}$ up to additive deviation of at most $\epsilon_i + \epsilon$. The main claim will follow by using an adequate setting of ϵ and δ, but before doing so we show that the query complexity is $t \cdot q_i$.

The latter claim relies on the observation that, *for every β_{i-1} and each sample point $s \in S$, the value of $p^f_{\tau_{i-1},\beta_{i-1},\mathtt{msg}_i(s)}$ is estimated using a distribution of queries that depends only on $\alpha_1, ..., \alpha_{i-1}$ and $\mathtt{msg}_i(s)$*. This observation relies on the induction hypothesis (for i), and implies the same fact for the value of $p^f_{\tau_{i-1}}$. Hence, the query complexity at the $i - 1^{\text{st}}$ level is $t \cdot q_i$, which equals q_{i-1}, provided that $t = O(r^2 \cdot \ell')$.

Turning to the quantitative analysis, we observe that using $\epsilon = 1/10r$ and $\delta = 2^{-O(\ell')}$, it follows that $\epsilon_i + \epsilon = \frac{r-i+1}{10r} + \frac{1}{10r} = \epsilon_{i-1}$ and $t = O(r^2 \cdot \ell')$, whereas the error probability (at the $i - 1^{\text{st}}$ level) is upper-bounded by

$$\begin{aligned} 2^{\ell_{i-1}} \cdot t \cdot \delta_i + \delta &= 2^{\ell_{i-1}} \cdot O(r^2\ell') \cdot O(r^2\ell')^{-i} \cdot 2^{-\sum_{j\in[i-1]} \ell_j} + 2^{-O(\ell')} \\ &= O(r^2\ell')^{-(i-1)} \cdot 2^{-\sum_{j\in[i-2]} \ell_j} \end{aligned}$$

which equals δ_{i-1}. The claim follow. ■

Conclusion. As stated upfront, Lemma 4.10.1 implies that, for every α_1, we can approximate $p^f_{\alpha_1}$ up to constant deviation of 0.1 with error probability at most 0.01 by making

$$q' \stackrel{\text{def}}{=} O(r^2 \cdot \ell')^r \cdot q \;=\; O(r^2 \cdot (\ell + r \cdot \log \ell))^r \cdot q \;=\; O(r^3 \cdot \ell)^r \cdot q$$

queries. Furthermore, we can estimate p^f_λ, which represents the probability that the verifier accepts f, up to an additive deviation of 0.01 with probability 0.99, by the average of a constant number of $p^f_{\alpha_1}$'s (for random α_1's). Combining these two facts, it follows that p^f_λ can be estimated up to an additive deviation of 0.11 with probability at least 0.98 using $O(q')$ queries to f. This allows for distinguishing between the case of $p^f_\lambda \geq 2/3$ and the case of $p^f_\lambda \leq 1/3$. ■

5 On the power of the general model

In this section we prove Theorem 1.3 asserting that the general model of PO-queries cannot be efficiently emulated by MAPs. Specifically, we show that a natural property that is extremely hard to verify by (general) MAPs can be verified by 1-round IPP that uses proof-oblivious queries. The property, denoted PERM, consists of the set of all permutations over $[n]$.

Theorem 5.1 (PERM is hard for MAPs but easy for the general model of PO-queries): PERM *can be verified in the general model* (of IPPs with proof-oblivious queries) *using $O(1/\epsilon)$ queries and $O(\epsilon^{-1} \log n)$ bits of communication, but any MAP for Π that uses a proof of length p must use at least $\Omega(n^{1/2}/(p+1))$ queries. Furthermore, the IPP with proof-oblivious queries uses a single round of communication.*

Proof: The hardness of PERM for MAPs was established in [18, Lem. 4.3] and contrasted with a 1-round IPP presented in [18, Sec. 4.1], but this IPP uses proof-dependent queries (i.e., it does

not satisfy the PO-queries condition).[18] Instead, we present the following simple IPP: The verifier queries the tested function $f : [n] \to [n]$ at a random point $\omega \in [n]$, sends $f(\omega)$ to the prover, and accepts if and only if the prover answers with ω.

Clearly, the verifier always accept when f is a permutation. On the other hand, if f is ϵ-far from being a permutation, then there are more than ϵn points $i \in [n]$ such that $f^{-1}(f(i))$ is not a singleton. If the verifier happens to select such a point i, then the prover's answer leads the verifier to accept with probability at most $1/|f^{-1}(f(i))| \leq 1/2$. Repeating the process $O(1/\epsilon)$ times, in parallel, yields the desired IPP. ■

[18] Their IPP selects a random point $\omega \in [n]$, sends ω to the prover, and queries f on the prover's answer, denoted v. That is, the prover is suppose to answer with $v \in f^{-1}(\omega)$, and the verifier accepts iff $f(v) = \omega$. We stress that the verifier queries f at a point provided by the prover.

Acknowledgements

We are grateful to Ron Rothblum for helpful discussions.

O.G.'s research was partially supported by the Israel Science Foundation (grant No. 1041/18) and by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 819702). G.R.'s research has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 819702), from the Israel Science Foundation (grant number 5219/17), and from the Simons Foundation Collaboration on the Theory of Algorithmic Fairness.

References

[1] N. Alon, M. Krivelevich, I. Newman, and M. Szegedy. Regular Languages are Testable with a Constant Number of Queries. *SIAM Journal on Computing*, Vol. 30 (6), pages 1842–1862, 2001.

[2] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof Verification and Intractability of Approximation Problems. *Journal of the ACM*, Vol. 45, pages 501–555, 1998. Preliminary version in *33rd FOCS*, 1992.

[3] M. Bellare, O. Goldreich, and S. Goldwasser. Randomness in Interactive Proofs. *Comput. Complex.*, Vol. 3, pages 319–354, 1993.

[4] I. Berman, R. Rothblum, and V. Vaikuntanathan. Zero-Knowledge Proofs of Proximity. In the *9th ITCS*, pages 19:1-19:20, 2018.

[5] A. Chakrabarti, G. Cormode, A. McGregor, and J. Thaler. Annotations in Data Streams. *ACM Trans. Algorithms*, Vol. 11 (1), pages 7:1–7:30, 2014.

[6] A. Chakrabarti, G. Cormode, A. McGregor, J. Thaler, and S. Venkatasubramanian. Verifiable Stream Computation and Arthur-Merlin Communication. SIAM Journal on Computing, Vol. 48 (4), pages 1265–1299, 2019.

[7] A. Chiesa and T. Gur. Proofs of Proximity for Distribution Testing. In the *9th ITCS*, pages 53:1–53:14, 2018. See full version in *ECCC*, TR17-155, 2017.

[8] G. Cormode, J. Thaler, and K. Yi. Verifying Computations with Streaming Interactive Proofs. *Proc. VLDB Endow.*, Vol. 5 (1), pages 25–36, 2011.

[9] F. Ergun, R. Kumar, and R. Rubinfeld. Fast Approximate PCPs. In *31st STOC*, pages 41–50, 1999.

[10] U. Feige, D. Lapidot, and A. Shamir. Multiple Non-Interactive Zero-Knowledge Proofs Under General Assumptions. *SIAM Journal on Computing*, Vol. 29 (1), pages 1–28, 1999. Preliminary version in *31st FOCS*, 1990.

[11] G. Goldberg and G.N. Rothblum. Sample-Based Proofs of Proximity. In the *13th ITCS*, pages 77:1–77:19, 2022. See full version in *ECCC*, TR21-146, 2021.

[12] O. Goldreich. *Modern Cryptography, Probabilistic Proofs and Pseudorandomness*. Algorithms and Combinatorics series (Vol. 17), *Springer*, 1998.

[13] O. Goldreich. *Foundation of Cryptography – Basic Tools*. Cambridge University Press, 2001.

[14] O. Goldreich. *Introduction to Property Testing*. Cambridge University Press, 2017.

[15] O. Goldreich, T. Gur, and I. Komargodski. Strong Locally Testable Codes with Relaxed Local Decoders. *ACM Trans. Comput. Theory*, Vol. 11 (3), pages 17:1–17:38, 2019.

[16] S. Goldwasser, S. Micali, and C. Rackoff. The Knowledge Complexity of Interactive Proof Systems. *SIAM Journal on Computing*, Vol. 18, pages 186–208, 1989. Preliminary version in *17th STOC*, 1985.

[17] S. Goldwasser and M. Sipser. Private Coins versus Public Coins in Interactive Proof Systems. *Advances in Computing Research: a research annual*, Vol. 5 (Randomness and Computation, S. Micali, ed.), pages 73–90, 1989. Extended abstract in *18th STOC*, 1986.

[18] T. Gur, Y. Liu, and R. Rothblum. An Exponential Separation Between MA and AM Proofs of Proximity. *Comput. Complex.*, Vol. 30 (2), Art. 12, 2021.

[19] T. Gur, and G. Ramnarayan, and R. Rothblum. Relaxed Locally Correctable Codes. In the *9th ITCS*, pages 27:1–27:11, 2018. See *ECCC*, TR17-143, 2017.

[20] T. Gur and R. Rothblum. Non-interactive proofs of proximity. *Comput. Complex.*, Vol. 27 (1), pages 99–207, 2018. Preliminary version in *ECCC*, TR13-078, 2013.

[21] T. Gur and R. Rothblum. A Hierarchy Theorem for Interactive Proofs of Proximity. In the *8th ITCS*, pages 39:1–39:43, 2017.

[22] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic Methods for Interactive Proof Systems. *Journal of the ACM*, Vol. 39, No. 4, pages 859–868, 1992. Preliminary version in *31st FOCS*, 1990.

[23] G. Rothblum, S. Vadhan, and A. Wigderson. Interactive Proofs of Proximity: Delegating Computation in Sublinear Time. In *45th ACM Symposium on the Theory of Computing*, pages 793–802, 2013.

[24] S. Vadhan. On Transformation of Interactive Proofs that Preserve the Prover's Complexity. In *32nd ACM Symposium on the Theory of Computing*, pages 200–207, 2000.

Appendices

A.1 An alternative definition of the general PO-queries model

Our definitional approach is based on decomposing the verifier into three modules, called *querying*, *interacting* and *deciding*, and postulates models based on the interaction between these modules. In particular, the general model of proof-oblivious queries only postulates that the querying module gets no information from the other modules.

Inspired by [20, Def. 2.2], which refers to MAPs only, one can take a different approach towards defining the general PO-queries model, an approach that does not rely on the decomposition of the verifier into modules. To present this alternative definition, we first define the *query set* of an IPP verifier.

Definition A.1 (the verifier's query set): *Let V be an IPP verifier. For an integer $n \in \mathbb{N}$, proximity parameter $\epsilon > 0$, input $f : [n] \to [m]$, randomness ω, and arbitrary prover strategy P, we define the corresponding* query set*, denoted $\mathcal{Q}^f_{V,P}(n, \epsilon, \omega)$, as the set of locations* (in $[n]$) *queried by V when using randomness ω in its interaction with the prover P and while querying the input f.*

We stress that $\mathcal{Q}^f_{V,P}(n, \epsilon, \omega)$ is well-defined also in case V makes adaptive queries to f; this is the reason that f appears in the notation.

Since n is implicit in f and ϵ remains invariant in our discussion, we omit both of them from the notation; that is, we replace $\mathcal{Q}^f_{V,P}(n, \epsilon, \omega)$ by $\mathcal{Q}^f_{V,P}(\omega)$. Also, without loss of generality, we assume that P is a deterministic strategy, and so for a fixed ω, the query set of V is fully determined. We now present the alternative definition,

Definition A.2 (alternative definition of the general PO-queries model): *Let V be an IPP verifier. We say that V makes* proof-oblivious queries *if for every $n \in \mathbb{N}$, $\epsilon > 0$, and f, and every* fixed *ω, it holds that $\mathcal{Q}^f_{V,P_1}(\omega) = \mathcal{Q}^f_{V,P_2}(\omega)$ for any pair of prover strategies P_1 and P_2.*

We show that this definition is, in fact, equivalent to the definition of the general PO-queries model based on the decomposition into module (as captured in Eq. (3)).

Proposition A.3 (equivalence of the two definitions): *For every IPP verifier V, the following two conditions are equivalent.*

1. *V satisfies Definition A.2.*
2. *V can be decomposed into three modules Q, I and D that satisfy the conditions of the general proof-oblivious queries model; that is, for every prover strategy P and for every ω, it holds that $D(\langle P(f), I(Q^f(\omega))\rangle) = \langle P(f), V^f(\omega)\rangle)$.*

Proof: Starting from Condition 2, let Q, I and D denote the modules that V is composed of. Fixing n, ϵ, f, and a random string ω, consider any two prover strategies P_1 and P_2. Since Q is the only module of V that queries f, and since Q does not interact with the prover (nor obtains any information about the prover's answers from any other module), it follows that Q's queries are independent of P_i, and Condition 1 follows.

Turning to the opposite direction and assuming that Condition 1 holds, we derive three modules (Q, I, D) that satisfy the requirements of the general PO-queries model and perfectly emulate V.

Querying module (Q)**:** Let P_0 be an arbitrary prover strategy (e.g., the prover that always responds with a string of zeros). Then, on input n, ϵ, ω and oracle access to f, the module Q emulates an execution of $\langle P_0, V^f(n, \epsilon; \omega)\rangle$, while using its oracle access to f to answer queries of V.

Note that Q does not communicate with a prover, but rather emulates P_0 by itself.

Interacting module (I)**:** Recall that on input n, ϵ, the module I receives the output of the querying module, which consists of the randomness ω and the sequence of answers provided by f to queries of Q^f. Let us denote the latter sequence by $\overline{a}$. The module I interacts with the actual prover, denoted P, by invoking a copy of V on input n, ϵ and randomness ω, relaying messages between this copy and P, and providing V with answers to its queries (according to the record in $\overline{a}$).

The point is that, by Condition 1, when using randomness ω, the queries of V during the interaction with P are identical to its queries during an interaction with P_0, whereas these are exactly the queries made by Q on randomness ω. Hence, the answers are recorded in $\overline{a}$, and I uses these answers when V issues these queries.

Deciding module (D)**:** Recall that on input n, ϵ, the module D receives the output of the interacting module, which consists of the randomness ω, the sequence of answers (denoted $\overline{a}$) provided to the queries made to f, and the sequence of messages (denoted $\overline{\beta}$) sent by the prover. The module D invokes a copy of V, on input n, ϵ and randomness ω, and emulates its interaction with the prover (by using $\overline{\beta}$) as well as the oracle's answers (according to the record in $\overline{a}$). Once V outputs its decision, D outputs it as is.

Again, D's emulation of the execution of V is perfect, and so its output is exactly the one that V would have given on the same randomness.

Hence, for every ω, it holds that $D(\langle P(f), I(Q^f(\omega))\rangle) = \langle P(f), V^f(\omega))\rangle)$, which establishes Condition 2. ■

A.2 On parallel repetition of IPPs

We prove that, in the pre-coordinated model, parallel repetition reduces error as expected. Essentially, this is the case since the execution of systems in this model can be represented by max-average game-trees, just as in the case of standard interactive proof systems (cf. [3, Sec. 4] and [12, Apdx C.1]). Actually, this holds also for general IPPs, but the proof is more confusing in that context. We stress that, in all cases, we consider general interactive proof systems that may not be of the public-coin type and may not have perfect completeness.

Game-trees of standard interactive proof systems. Fixing an input to such an interactive proof system, we consider the tree of all possible executions, and associate a value to each vertex in such a tree. The children of an internal vertex correspond to possible messages that the relevant party may send at the corresponding step. The verifier selects one of these children at random, according to some distribution that is not necessarily uniform, and the value of the corresponding vertex equals the expected value of its children. (Note that this selection from a conditional probability space is not necessarily how the real (possibly private-coin) verifier acts, but this is how

we view the effect of its actions in the analysis.) An optimal prover selects a child of maximal value, and this is the value of the parent vertex. The value of a leaf in the tree is the probability that the verifier accepts conditioned on the corresponding sequence of messages sent during the execution, where the value is either 0 or 1 in the special case that the sequence of messages fully determines the verifier's randomness (e.g., if the verifier sends its entire randomness as its last message). Note that the value of the root of the game-tree represents the probability that the verifier accepts, which in turn represents the expected value of the leaf reached in a random execution (with an optimal prover).

Looking at t parallel executions of such a system, we consider an analogous game-tree, but define the value of a leaf in this tree as the average of the values of the corresponding t leaves in the game-tree of the original system. That is, this value is the average of the acceptance probability in the t executions, not the probability of accepting in the t-way parallel execution, which we did not define (i.e., we did not say when the parallel execution accepts). We define the value of an internal vertex in the tree of t executions analogously to the way it was defined for the game-tree of a single execution. The reader can easily verify that *the value of each vertex in the tree that describes the t-way parallel execution equals the average of the t corresponding vertices in game-tree of a single execution.*[19] Furthermore, going over the argument, it follows that the value of a leaf reached in a random execution of the t-way parallel system (with an optimal prover) is the average of t independent random variables such that each random variable represent the value of a random execution of the original system.

[19] This can be proven by induction from the leaves of the tree. Denoting the value of a vertex v in the game-tree of a single execution by $\mathtt{val}(v)$, and the value of a vertex $\overline{v}$ in the tree that describes a t-way parallel execution by $\overline{\mathtt{val}}(\overline{v})$, we prove that $\overline{\mathtt{val}}(v_1,...,v_t) = \sum_{i\in[t]} \mathtt{val}(v_i)/t$, by assuming that equality holds in each child of $(v_1,...,v_t)$. In the case of a verifier step, letting $R(v)$ be the distribution on vertices representing possible messages used by the verifier at vertex v (of the original game-tree), we use the fact that

$$\begin{aligned}
\overline{\mathtt{val}}(v_1,...,v_t) &= \mathrm{E}_{(c_1,...,c_t)\leftarrow(R(v_1),,,.,R(v_t))}\left[\overline{\mathtt{val}}(c_1,....,c_t)\right] \\
&= \mathrm{E}_{(c_1,...,c_t)\leftarrow(R(v_1),,,.,R(v_t))}\left[\frac{1}{t}\cdot\sum_{i\in[t]}\mathtt{val}(c_i)\right] \\
&= \frac{1}{t}\cdot\sum_{i\in[t]}\mathrm{E}_{c_i\leftarrow R(v_i)}[\mathtt{val}(c_i)] \\
&= \frac{1}{t}\cdot\sum_{i\in[t]}\mathtt{val}(v_i).
\end{aligned}$$

Likewise, in the case of a prover step, letting $C(v)$ denote the set of vertices representing possible messages used by the prover at vertex v (of the original game-tree), we use the fact that

$$\begin{aligned}
\overline{\mathtt{val}}(v_1,...,v_t) &= \max_{(c_1,...,c_t)\in C(v_1)\times\cdots\times C(v_t)}\left[\overline{\mathtt{val}}(c_1,....,c_t)\right] \\
&= \max_{(c_1,...,c_t)\in C(v_1)\times\cdots\times C(v_t)}\left[\frac{1}{t}\cdot\sum_{i\in[t]}\mathtt{val}(c_i)\right] \\
&= \frac{1}{t}\cdot\sum_{i\in[t]}\max_{c_i\in C(v_i)}[\mathtt{val}(c_i)] \\
&= \frac{1}{t}\cdot\sum_{i\in[t]}\mathtt{val}(v_i).
\end{aligned}$$

Claim A.4 (main claim): *Let X be a random variable that represents the value of random leaf in the tree corresponding to t parallel executions of the original system, when the prover strategy aims at maximizing this value. Then, $X = \frac{1}{t} \cdot \sum_{i \in [t]} Z_i$ such that the Z_i's are independent random varaibles each representing the value of random leaf in the tree corresponding to an execution of the original system, when the prover strategy aims at maximizing this value.*

This can be seen by fixing the first i executions, and looking at the $i+1^{\text{st}}$ execution. The key point is that maximizing the average of unrelated values calls for maximizing each of the values.

Game-trees of systems in the pre-coordinated model. As stated upfront, we merely observe that the same considerations can be applied to the game-tree that corresponds to an IPP system in the pre-coordinated model. Specifically, we view the querying process as taking place after the entire interaction is completed. Hence, fixing an input f, the value of a *leaf* in this game-tree is defined as the probability that the verifier accepts conditioned on the corresponding sequence of messages sent during the execution, where this probability is determined by the answers (of f) to the (random) queries made by the verifier. (That is, again, we look at the verifier's action as if it draws its randomness from a conditional probability space, although it actually behaves differently.) Hence, we established the following fact.

Proposition A.5 (parallel repetition of the pre-coordinated model): *Let V be an IPP in the pre-coordinated model. When running t executions of $\langle P(f), V^f \rangle$ in parallel, the number of accepting executions is the sum of t independent 0-1 random variables that are each 1 with probability that equals the probability of accepting in a single execution of $\langle P(f), V^f \rangle$.*

Hence, error reduction by independent (parallel) executions is available just as in the case of ordinary randomized algorithms. Specifically, if we need to distinguish the case of rejecting with probability at most $1/3q$ from the case of rejection with probability at least $0.4/q$, then $O(q)$ repetitions suffice.

Game-trees of systems in any interactive proof of proximity. Although it may be confusing, the same considerations can be applied to the game-tree that corresponds to a general IPP system. In this case, for a fixed f, the conditional distributions from which the verifier draws its messages depend on f. The fact that the real verifier may not know f is irrelevant to the analysis, which is looking at the execution from a "know all" view point. Furthermore, the same holds with respect to the queries made by the verifier. Hence, we actually have.

Theorem A.6 (parallel repetition of general IPP systems): *Let V be an arbitrary IPP. When running t executions of $\langle P(f), V^f \rangle$ in parallel, the number of accepting executions is the sum of t independent 0-1 random variables that are each 1 with probability that equals the probability of accepting in a single execution of $\langle P(f), V^f \rangle$.*

A.3 The IPP for tensor/sub-cube sum, generalized and revised

We review the celebrated sum-check protocol of [22], viewing it as applied to an input function $f : \mathcal{F}^m \to \mathcal{F}$, and observing that in this case this protocol is implementable by a public-coin IPP *in the pre-coordinated model.*

Recall that the sum-check protocol is used for verifying claims of the form $\sum_{x\in H^m} f(x) = v$, where $f : \mathcal{F}^m \to \mathcal{F}$ describes (or is claimed to describe) a low-degree polynomial, $H \subset \mathcal{F}$, and $v \in \mathcal{F}$ is a claimed value. Using the standard setting of $m = \frac{|\mathcal{F}|}{\Theta(\log |\mathcal{F}|)}$ and $n = |\mathcal{F}|^m$, this yields a system of poly$(\log n)$ complexity for a property (called Tensor/Sub-cube Sum) that cannot be tested with $n^{0.999}$ queries (cf. [20, Thm. 3.28]). Observing that this IPP can be implemented in the pre-coordinated model establishes Theorem 1.5. A more general trade-off, obtained by setting $m = r$, yields the relative tightness of the emulation (of public-coin r-round IPPs of the pre-coordinated model) captured by Theorem 4.10. The tightness is further narrowed by using an auxiliary observation.

The actual property: Tensor/Sub-cube sum. For a finite field $\mathcal{F}$, a subset $H \subset \mathcal{F}$ and an integer $m \in \mathbb{N}$ satisfying $m \cdot 2(|H| - 1) < \frac{|\mathcal{F}|}{10}$, we consider the Subcube-Sum property that consists of the set of m-variate polynomials f over $\mathcal{F}$ of individual degree at most $d \stackrel{\text{def}}{=} 2 \cdot (|H| - 1)$ that sum to 0 on the sub-cube H^m:

$$\sum_{x_1,...,x_m\in H^m} f(x_1, \ldots, x_m) = 0$$

By [18, Cor. 3.16], the query complexity of testing Subcube-sum is $\Omega(|H|^m/\log |\mathcal{F}|)$. On the other hand, the sum-check protocol takes m rounds, where in each round the parties reduce the number of variables in the sum (by one unit), by communicating $d+1 < 2|H|$ field elements. Details follow.

A generic round in the sum-check protocol. The parties enter the i^{th} round with a claim of the form

$$\sum_{x_i,...,x_m\in H^m} f(\omega_1, ..., \omega_{i-1}, x_i, \ldots, x_m) = v_{i-1} \tag{13}$$

where $\omega_1, ..., \omega_{i-1}$ and v_{i-1} were determined in prior rounds (and $v_0 = 0$). In the i^{th} round, the prover is supposed to send the univariate (degree d) polynomial

$$p_{\omega_1,...,\omega_{i-1}}(z) \stackrel{\text{def}}{=} \sum_{x_{i+1},...,x_m\in H^m} f(\omega_1, ..., \omega_{i-1}, z, x_{i+1}, \ldots, x_m)$$

and the verifier replies with a uniformly chosen $\omega_i \in \mathcal{F}$. Assuming that the prover responded with $\widetilde{p}_i$, the verifier checks that $\sum_{x\in H} \widetilde{p}_i(x) = v_{i-1}$. Both parties then set $v_i \leftarrow \widetilde{p}_i(\omega_i)$. After the last iteration, the verifier queries f at the point $\omega = (\omega_1, ..., \omega_m)$, and accepts if and only if $f(\omega) = v_m$.

The foregoing description suffices for the case that f is guaranteed to be a polynomial of individual degree d. Otherwise, this property has to be tested first; the verifier can do this by itself (by using $O(1/\epsilon) \cdot md$ queries). The sum-check analysis is then applied to the corresponding polynomial f' that is ϵ-close to f.

Observe that the resulting m-round IPP is of the public-coin type and that it can be implemented in the pre-coordinated model, because all checks can be relegated to the decision module. The query complexity of this IPP is $1 + O(dm/\epsilon)$, and the total communication is $m \cdot (d+1) \cdot \log_2 |\mathcal{F}| = O(m|H| \log |\mathcal{F}|)$. The fact that the overall query complexity is larger than $m|H|$ suggests to avoid the last round and instead let the verifier check $\sum_{x\in H} f(\omega', x) = v_{m-1}$, where $\omega' = (\omega_1, ..., \omega_{m-1})$.

Actually, we query f on the entire axis-parallel line $(\omega_1, ..., \omega_{m-1}, \cdot)$, and also reject if the values do not fit a univariate polynomial of degree d. (In the analysis, we use that fact that if f is ϵ-close to an m-variate polynomial of individual degree d, then, with probability at least 0.9, it holds that f disagrees with f' on at most $10\epsilon \cdot |\mathcal{F}|$ points on a random axis-parallel line.)

On the relative tightness of Theorem 4.10. We shall demonstrate the relative tightness of Theorem 4.10 by using the foregoing version of the IPP for Subcube Sum. For any number of rounds $r \leq 0.5 \log_2 n / \log_2 \log_2 n$, we set $m = r + 1$ and $n = |\mathcal{F}|^m$, which implies $|\mathcal{F}| = n^{1/m} \geq (\log_2 n)^2$. We choose H as big as possible; that is, $|H| = \Theta(|\mathcal{F}|/m)$. Applying Theorem 4.10 to the foregoing protocol yields a tester of query complexity

$$\begin{aligned} O(c^r \cdot q) &= O(m|H| \log |\mathcal{F}|)^{m-1} \cdot (O(m|H|/\epsilon) + |\mathcal{F}|) \\ &< O(1/\epsilon) \cdot \frac{O(|\mathcal{F}| \log |\mathcal{F}|)^m}{\log |\mathcal{F}|} \\ &= O(1/\epsilon) \cdot \frac{O((1/m) \cdot \log n)^m}{\log |\mathcal{F}|} \cdot n, \end{aligned}$$

whereas the aforementioned lower bound (of [18, Cor. 3.16]) on query complexity of testing is

$$\begin{aligned} \frac{\Omega(|H|^m)}{\log |\mathcal{F}|} &= \frac{\Omega(|\mathcal{F}|/m)^m}{\log |\mathcal{F}|} \\ &= \frac{\Omega(1/m)^m}{\log |\mathcal{F}|} \cdot n. \end{aligned}$$

Hence, the emulation is tight up to a factor of $O(1/\epsilon) \cdot O(\log n)^{r+1}$.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il