Hamming Weight Proofs of Proximity with One-Sided Error

Gal Arnon
gal.arnon@weizmann.ac.il
Weizmann Institute

Shany Ben-David
shany.ben-david@biu.ac.il
Bar-Ilan University

Eylon Yogev
eylon.yogev@biu.ac.il
Bar-Ilan University

May 28, 2024

Abstract

We provide a wide systematic study of proximity proofs with one-sided error for the Hamming weight problem Ham_α (the language of bit vectors with Hamming weight at least α), surpassing previously known results for this problem. We demonstrate the usefulness of the one-sided error property in applications: no malicious party can frame an honest prover as cheating by presenting verifier randomness that leads to a rejection.

We show proofs of proximity for Ham_α with one-sided error and sublinear proof length in three models (MA, PCP, IOP), where stronger models allow for smaller query complexity. For n-bit input vectors, highlighting input query complexity, our MA has $O(\log n)$ query complexity, the PCP makes $O(\log\log n)$ queries, and the IOP makes a single input query. The prover in all of our applications runs in expected quasi-linear time. Additionally, we show that any perfectly complete IP of proximity for Ham_α with input query complexity $n^{1-\epsilon}$ has proof length $\Omega(\log n)$.

Furthermore, we study PCPs of proximity where the verifier is restricted to making a single input query (SIQ). We show that any SIQ-PCP for Ham_α must have a linear proof length, and complement this by presenting a SIQ-PCP with proof length $n+o(n)$.

As an application, we provide new methods that transform PCPs (and IOPs) for arbitrary languages with nonzero completeness error into PCPs (and IOPs) that exhibit perfect completeness. These transformations achieve parameters previously unattained.

Keywords: Hamming weight problem; interactive proofs of proximity; interactive oracle proofs

Contents

1 Introduction

A motivating example. On April 14, 2022, businessman Elon Musk made an unsolicited and non-binding offer to purchase the social media company "Twitter, Inc." for $43 billion and take it private, which the board reluctantly accepted. In July, Musk announced his intention to terminate the agreement in the wake of reports that, despite the board's assurance, 5% of Twitter's daily active users were *spambot* accounts. In order to collect data, Musk posted a Twitter poll asking followers about the amount of spambots. In response, Twitter pursued legal action against Musk, which eventually led to the completion of the acquisition on October 27, 2022.

The acquisition was messy, involved extensive litigation, dropped the share price, affected many individuals, and was expensive and time-consuming. The process could have been more straightforward had the parties had the tools to build mutual trust. Specifically, they lacked a method for Twitter to efficiently *prove* to Musk, beyond a reasonable doubt, that the number of spambots is indeed lower than 5%. Musk could have hired experts to examine whether a handful of *specific* users are spambots, but exploring all of the ~ 350 million users is impractical. The appropriate tool to remedy the situation is a *proof of proximity*.

Proofs of proximity. Proofs of proximity are probabilistic proofs with a sublinear time verifier. Since the verifier runs in sublinear time, it cannot even read the entire input. Following work on sublinear time algorithms and property testing [RS96; GGR98], the verifier is given *query access* to the input: the input $\mathbb{x}$ is treated as an oracle and, on query i, the verifier receives $\mathbb{x}[i]$. The goal is to construct probabilistic proofs with sublinear query complexity while minimizing parameters such as verifier running time and communication complexity. Proofs of proximity were first introduced by Ergun, Kumar, and Rubinfeld [EKR04] and further studied by Rothblum, Vadhan, and Wigderson [RVW13] and Gur and Rothblum [GR18], motivated by applications to delegation of computation. Since then, there has been considerable research on proofs of proximity across various models.

The Hamming weight problem. This work focuses on probabilistic proofs of proximity for the *Hamming weight* problem. Here, the task is to decide whether a given string $\mathbb{x} \in \{0,1\}^n$ has Hamming weight at least $\alpha(n)$ or is far from it: it has Hamming weight less than $\alpha(n) - \delta(n)$, for a proximity parameter δ.

A proof of proximity for this problem would have been useful in the context of the Twitter acquisition. Twitter's network can be represented as a binary vector $\mathbb{x}$ whose length corresponds to the number of Twitter accounts with a value of 1 indicating the non-spambot users. Twitter would submit a proof of the vector's Hamming weight, and Musk, or any other interested party, could efficiently verify the proof while performing only a few queries. A query to the input vector is translated to the expensive task of determining whether a given user is a spambot, which fuels the desire for small query complexity.

Beyond our motivating example, proofs of proximity for the Hamming weight prob-

lem have many applications, as the primary tool in other proximity tests. For example, testing whether an n-vertex graph contains many k-cliques can be directly reduced to the Hamming weight of a corresponding vector of size n^k (where 1 indicates a k-clique). Furthermore, proofs of proximity for Hamming weight (with one-sided error) can be used to transform standard proof systems for arbitrary languages to achieve perfect completeness (we demonstrate this in Section 1.2).

Framing-free security. There is a subtle but crucial property we need from our probabilistic proof in the form of one-sided error. To motivate this property, we return to the Twitter saga. Suppose that the proof of proximity has two-sided error. This means that even if Twitter generates a proof honestly, a malicious party could find a choice of randomness that makes the verifier reject this proof. Musk could leverage this by presenting such choice of randomness to a (resource-limited) judge, claiming that Twitter is lying, which might lead to the revocation of the acquisition. In order for Twitter to be willing to post a proof of their claims, we must ensure the system is "framing-free", which is obtained when the proximity proof has a one-sided error (perfect completeness). In other words: *One-sided error guarantees framing-free security, where honest parties cannot be accused of wrongdoing.*

On top of the above, there are also concrete benefits in the parameters of protocols with one-sided error. These protocols can be more efficiently amplified compared to their two-sided error counterparts. Repeating a protocol k times maintains the one-sided error property of a protocol and reduces the soundness error from ϵ to ϵ^k. For protocols with two-sided error, the soundness error only reduces to $\epsilon^{\Omega(k)}$, which means that to get the same soundness error, one needs more repetitions (and thus higher query complexity).

A brief history of Hamming weight proximity testing. There are several different proofs of proximity of the Hamming weight problem[1] in various models. Without the aid of a prover (i.e., property testing), known sampling lower bounds (see, e.g., [Gol11, Theorem 2.1], or [BKS01, Theorem 15]) tell us that the query complexity of any property tester for the Hamming weight problem is $\Omega(\min\{n, \delta^{-2}\})$, where n is the vector length and δ is the proximity parameter (with constant soundness error). A simple test achieves this bound but does not have perfect completeness (i.e., it has two-sided error).

In striking contrast to the above bounds, we observe that the query complexity of any property tester with perfect completeness (and without a prover) is significantly higher; specifically, it must be $\Omega(n)$, effectively rendering the test trivial.[2]

[1] The Hamming weight problem in previous work usually refers to the problem of *exact* Hamming weight α, whereas we define the constraint to be *at least* weight α. However, the two problems have (almost) tight reductions between each other, as shown in Appendix B.

[2] Consider $\alpha = 2/3$, and suppose towards contradiction that the query complexity is $q = o(n)$. By soundness with respect to the all-zeroes vector, we know that there exists verifier randomness ρ for which the verifier rejects upon querying only zeros. Construct a vector with all ones except for these q places; then the verifier rejects it with nonzero probability. On the other hand, the vector has weight $1 - q/n$ which is more than α since $q = o(n)$, so by perfect completeness the verifier accepts the vector with probability 1.

In [RVW13], an IP of proximity was given (without perfect completeness), with query and communication complexities $O(\delta^{-1} \cdot \text{polylog}(n))$, and $O(\log n)$ many rounds. Alternatively, they construct a 2-message version of their protocol but with a much higher query and communication complexity of $O(n^{1/3} \cdot \delta^{-2/3} \cdot \text{polylog}(n))$. In [GGR18; RR20b], an IP of proximity for a larger complexity classes was given (which include the Hamming problem) with similar round and communication complexity, with constant query complexity.

A *non-interactive* proof of proximity (MAPs) for the Hamming weight problem was given in [GR18]. They showed that for every constant $\alpha \in (0,1)$ there is a MAP (with two-sided error) for Hamming weight with proof length $\widetilde{O}(n^{\alpha})$, and query complexity $\widetilde{O}(\sqrt{n^{1-\alpha}} \cdot \delta^{-1})$. For example, for $\alpha = 2/3$, the proof length is $\widetilde{O}(n^{2/3})$ and the query complexity is $\widetilde{O}(n^{1/6} \cdot \delta^{-1})$. They also showed that their results can be transformed to have perfect completeness while incurring a poly-logarithmic overhead to the query and proof complexities [GR18, Lemma 4.5]. Applying this transformation to the simple tester (without a prover) yields a one-sided error MAP with proof length $O(\delta^{-4} \cdot \log^2 n \cdot \log(\delta^{-1} \cdot \log n))$ and query complexity $O(\delta^{-4} \cdot \log n \cdot \log(\delta^{-1} \cdot \log n))$.

The work of [AGRR23] studied distribution-free proofs of proximity for the Hamming weight problem, where the verifier receives input samples from an unknown distribution. They showed a distribution-free protocol with perfect completeness, $O(\delta^{-1} \cdot \log n)$ rounds, $O(\delta^{-1} \cdot \log^2 n)$ communication complexity and δ^{-1} samples. [KSY20] studied the Hamming weight of social graphs, where instance samples are given via random walks in the graph. Finally, departing from information-theoretic security, [KR15] introduced the notion of interactive *arguments* of proximity. Roughly, they showed that all P has a 2-message argument with communication and query complexity $o(n)$ (assuming sub-exponentially secure FHE).

PCPs and IOPs of proximity. PCPs of proximity (PCPPs) were studied in [BGHSV06] and [DR04]. They are non-interactive proof of proximity systems where the verifier has *oracle access* to both the input and the given proof. In contrast to MAPs (where the verifier reads the entire proof), the proof string in PCPPs is typically of super-linear length (but the verifier reads only a few bits from it). Quoting [GR18], PCPPs may be thought of as the PCP analog of property testing, whereas MAPs are the NP analog of property testing. We are unaware of explicit works of PCPPs for the Hamming weight problem (beyond general PCPPs that are applicable for all languages in P). Applying a general purpose theorem for PCPPs (e.g., [Mie09]), one can obtain a PCPP for the Hamming weight problem with constant query complexity (for constant distance δ) but with a super-linear proof length and a relatively slow prover (however, still polynomial time).

IOPs of proximity (IOPP) are a combination of IPs and PCPs of proximity [BCS16; RRR16]. Here, the prover and verifier interact in multiple rounds, but the verifier has only oracle access to the prover's messages in addition to its oracle access to the input. IOPs leverage interaction to overcome barriers that arise with PCPs. For instance, known IOPs achieve linear proof length as well as other desirable properties such as fast provers, zero

knowledge, and concrete efficiency [BCGV16; Ben+17; BCGRS17; BBHR18; BCGGHJ17; XZZPS19; BCG20; BCL22; RR20a; ACY22b; ACY23; ACFY24; BN22; RR22]. We are unaware of explicit IOPPs for the Hamming weight language.

One additional advantage of constructing PCPPs and IOPPs is that they serve as the underlying building block for an interactive *arguments* with small communication complexity. For example, one can use the Kilian construction [Kil92] while relying on collision-resistant hash functions to commit to the prover message and only reveal the locations queries by the verifier. This is how hash-based arguments and SNARKs are constructed (see also [Mic00; BCS16; CY20; CY21a; CY21b]).

1.1 Main results

We provide a systematic study of the Hamming weight problem, presenting new protocols in various models (MAP, PCPP, IOPP) with sublinear communication, surpassing all known results for testing proximity to the Hamming weight. We also present new lower bounds, pointing to the limits of this problem. Let Ham_α be the language of all binary vectors of Hamming weight at least α. Recall that without a prover, $\Omega(n)$ queries are required for one-sided error.

In all our results, we distinguish between the proof query complexity (queries performed to the prover messages) and the input query complexity (queries performed to the input). The reason is that, depending on the application, each query might incur different costs. This is exemplified in the Twitter example where a proof query is relatively cheap (a query to a position in some file sitting on a server), while a query to the input is rather expensive (verifying that a specific user is not a spambot). Thus, it is typically most desirable to minimize the input query complexity.

Sublinear proofs of proximity. For each model (MAP, PCPP, IOPP), we give a protocol with one-sided error, sublinear communication, while also providing small query complexity. Focusing on the input query complexity, our MAP has $O(\log n)$ query complexity, the PCPP makes $O(\operatorname{loglog} n)$ queries, and the IOPP makes a single input query (where n is the vector length). The following theorem is an informal summary of these results presented for constant α, constant distance δ, and constant soundness error. For simplicity, in the theorem we hide dependencies on α and δ.

Theorem 1 (Informal)**.** *For every constant $\alpha \in (0, 1]$ there are MAP, PCPP, and IOPP protocols for Ham_α with perfect completeness and parameters summarized below:*

Model	Queries to input	Queries to proof	Proof length	Rounds
MAP	$O(\log n)$	-	$O(\log^2 n)$	-
PCPP	$O(\operatorname{loglog} n)$	$O(\log n \cdot \operatorname{loglog} n)$	$O(n/\log^2 n)$	-
IOPP	1	$O(\log n)$	$O(\log^2 n)$	2

The precise theorem statement and dependencies on all parameters can be found in Theorem 5.4 for the MAP protocol, in Theorem 5.6 for the PCPP protocol, and in Theorem 7.1 for the IOPP protocol.

The MAP described in Theorem 1 improves upon the one described in [GR18] by removing a $\log\log n$ factor; the input complexity is $O(\log n)$ and the proof length is $O(\log^2 n)$, compared to $O(\log n \cdot \log\log n)$ and $O(\log^2 n \cdot \log\log n)$ respectively. Our PCPP improves on this by reducing the input query complexity dramatically to $O(\log\log n)$, while also allowing the verifier to read fewer bits from the prover message. We are unaware of any other PCPP for the Hamming problem beyond the one described in Theorem 1. The IOPP, when compared to the IPP derived from [GGR18] yields an improvement in the number of rounds, which is reduced from $O(\log n)$ to 2, and the verifier only needs to query $O(\log n)$ bits from the prover messages, rather than reading the entire messages of size $\mathrm{polylog}\, n$.[3]

Lower bound. We continue our systematic study with a lower bound for the Hamming weight problem. [GR18] shows a lower bound for MAPs for the Hamming weight problem: roughly speaking, a protocol with proof complexity $\mathsf{l} = \Omega(\log n)$ and query complexity q must satisfy $\mathsf{l} \cdot \mathsf{q} = \Omega(\min\{n, \delta^{-2}\})$.

We give lower bounds for perfectly complete IPPs and IOPPs for the Hamming weight problem regardless of the number of rounds. Our IOPP lower bound applies to protocols in which the verifier is semi-adaptive, meaning that the verifier decides which queries to perform to the i-th prover message based on the first i prover/verifier messages (i.e., including the randomness sampled right after the i-th message). Note that PCPPs and IPPs are special cases of semi-adaptive IOPPs.

Theorem 2 (informal)**.** *For every constant $\alpha \in (0,1)$ the following hold:*

1. *Any perfectly complete IPP for Ham_α with total proof length l and input query complexity $\mathsf{q}_\mathbb{x}$ has $\mathsf{l} = \Omega(\log(n/\mathsf{q}_\mathbb{x}))$.*
2. *Any semi-adaptive perfectly complete IOPP for Ham_α with total length l, input query complexity $\mathsf{q}_\mathbb{x}$ and proof query complexity q_π has $\mathsf{q}_\pi \cdot \log \mathsf{l} = \Omega(\log(n/\mathsf{q}_\mathbb{x}))$.*

The above theorem has the following consequence: IPPs (regardless of the number of rounds) with query complexity $\mathsf{q}_\mathbb{x} = n^{1-\epsilon}$ for any constant $\epsilon > 0$, must have at least a logarithmic proof length. This lower bound implies that our MAP construction in Theorem 1 has proof length that is optimal up to a $O(\log n)$ factor. For (semi-adaptive) IOPPs, it shows that any IOPP with length $\mathrm{polylog}(n)$ and constant input query complexity must have proof query complexity $\Omega(\log(n)/\log\log n)$. The challenge of proving a lower bound for (fully) adaptive IOPPs remains as an open problem.

[3] The protocol from [GGR18] is described with constant query complexity but can be naturally modified to have a single input query [Rot24].

Single input query (SIQ). Our IOPP, as described in Theorem 1, has the remarkable property that, in addition to having a one-sided error, the verifier performs only a *single query to the input*. We denote such protocols as SIQ protocols (single input query). However, the cost of our SIQ-IOPP relative to its PCP counterpart is having additional rounds. Thus, we ask: can we achieve SIQ-PCPPs with sublinear proof length?

We give a negative answer to this question and show that no perfectly complete proof of proximity (for the Hamming weight problem) can simultaneously have a single input query, sublinear length, and be non-interactive.

Theorem 3. *For any $\alpha \in (0.5, 0.77)$, any perfectly complete PCP (or MA) of proximity for Ham_α with input query complexity 1 has proof length $\Omega(n)$.*

On the positive side, we show that with proof length $n+o(n)$, we can construct perfectly complete SIQ-PCPPs with small proof query complexity.

Theorem 4 (Informal)**.** *For every constant $\alpha \in (0, 1]$, there exists a perfectly complete SIQ-PCPP of proximity for Ham_α with proof length $n + O(\log^2 n)$ and proof query complexity $O(\log^2 n)$. (The formal theorem with the precise dependency on all parameters appears in Theorem 6.2.)*

Prover running time. We further strengthen the protocols described in Theorems 1 and 4 by showing efficient algorithms for the honest prover strategies, making our protocols doubly-efficient. In particular, the honest prover in both theorems runs in expected time $O(n \log n)$, where perfect completeness always holds when the prover outputs a message. This holds also for the first message of the IOPP in Theorem 1, and its second message can be computed in deterministic time $\mathrm{polylog}(n)$. We further remark that, given a Nisan–Wigderson type PRG [NW94], the prover in all of the protocols can be made to run in deterministic time $\mathrm{poly}(n)$. All prior works on proofs of proximity for Hamming weight did not explicitly analyze the honest prover running time.

1.2 Application: perfect completeness for PCPs and IOPs

The problem of transforming proof systems with imperfect completeness to ones with perfect completeness was first studied in the context of interactive proofs by [FGMSZ89] and is considered a cornerstone of research into IPs. Perfect completeness for PCPs and IOPs began to be explored only recently, with the goals of improving hardness of approximation results [BV19; ACY22a; ACY22b], and as a tool for proving barriers for proof systems [ABCY22].

We observe that proofs of proximity for Hamming weight can be utilized in this application. Using the techniques developed in the previous sections, we show new ways to transform PCPs and IOPs with nonzero completeness error into ones with perfect completeness. The following theorem is an informal summary of our results, presented for constant

completeness and soundness errors, both for the original proof system, and for the resultant perfectly complete proof system.

Theorem 5 (informal)**.** *Every language L that has a PCP (resp. IOP) with constant completeness error has a perfectly complete PCP (resp. IOP) with parameters given in Table 1. (The formal theorem with the precise dependencies on parameters is given in Section 9.)*

	Model	Queries	Proof length	Rounds
[BV19]	PCP	$\mathsf{q} + O(\mathsf{r})$	$\mathsf{l} + O(2^{\mathsf{r}})$	-
[This work]	PCP	$O(\mathsf{q} \cdot \mathsf{r} + \mathsf{r}^2)$	$\mathsf{l} + O(\mathsf{r}^2)$	-
[This work]	PCP	$O((\mathsf{q} + \mathsf{r}) \cdot \log \mathsf{r})$	$\mathsf{l} + O(2^{\mathsf{r}}/\mathsf{r}^2)$	-
[This work]	PCP	$\mathsf{q} + O(\mathsf{r}^2)$	$\mathsf{l} + 2^{\mathsf{r}} + O(\mathsf{r}^2)$	-
[ACY22a; ACY22b]	IOP	$O(\max\{1, \mathsf{k}/\log n\})$	$\mathrm{poly}(n, \mathsf{l}, \mathsf{r})$	k
[ABCY22]	IOP	$O(\mathsf{q} \cdot \log \mathsf{r} + \mathsf{r} \cdot \log \mathsf{r})$	$O(\mathsf{l} \cdot \mathsf{r} \cdot \log \mathsf{r})$	$\mathsf{k} + 1$
[This work]	IOP	$\mathsf{q} + O(\mathsf{r})$	$O(\mathsf{l} \cdot \mathsf{r})$	$\mathsf{k} + 1$

Table 1: A comparison of our PCP to perfectly complete PCP and IOP to perfectly complete IOP transformations with prior work. Above, q, l, r, and k denote the query complexity, proof length, randomness complexity, and number of rounds of the original PCP/IOP being transformed, and n is the instance size. Each of our results is derived by taking one of our upper bounds an using it to transform a PCP/IOP with imperfect completeness into one with perfect completeness.

2 Techniques

In this section, we give an overview of our techniques. Throughout, we denote $\mathsf{weight}(\mathbb{x}) := \frac{1}{n}\sum_{i\in[n]} \mathbb{x}[i]$ to be the Hamming weight of $\mathbb{x} \in \{0,1\}^n$ and use the shorthand $\mathbb{x}[i+j]$ to mean $\mathbb{x}[i+j \mod n]$. For simplicity, unless stated otherwise we consider all parameters (e.g., α, etc.) apart from n to be constant.

2.1 PCPP for Hamming weight with sublinear proof length

In this section, we sketch the proof of the PCPP of Theorem 1 in which our focus is on minimizing query complexity while maintaining sublinear proof length. We construct a PCPP for Hamming weight α that for vectors of length n has: (1) length $o(n)$ (2) input query complexity $O(\mathrm{loglog}\, n)$, and (3) proof query complexity $O(\log n \cdot \mathrm{loglog}\, n)$. Moreover, the honest prover runs in (expected) time $O(n \log n)$.

Our construction relies on the concept of "good shifts" and is achieved by combining an "outer protocol" and an "inner protocol". We start by defining good shifts, which will be the cornerstone of the honest prover strategies throughout this paper. This technique

is inspired by the beautiful "reverse randomization" method, which can be traced back to Lautemann's proof that BPP is in the polynomial hierarchy [Lau83], and has been useful for other applications as well (e.g., [FGMSZ89; Nao89; DNR04; HNY17; BV22]).

Good shifts. We say that the "shifts" $z_1, \ldots, z_t \in [n]$ are "good" for a vector $\mathbb{x} \in \{0,1\}^n$ if for every $\rho \in [n]$ it holds that the induced vector $\mathbb{x}_\rho := (\mathbb{x}[\rho + z_1], \ldots, \mathbb{x}[\rho + z_t]) \in \{0,1\}^t$ has Hamming weight at least $0.95 \cdot \alpha$. We show that for $t := \Theta(\frac{1}{\alpha} \cdot \log n)$ the following hold:

1. *Large Hamming weight.* For every $\mathbb{x}$ with $\mathsf{weight}(\mathbb{x}) \geq \alpha$, there exist shifts $z_1, \ldots, z_t$ that are good for $\mathbb{x}$, i.e., where for every ρ, it holds that $\mathsf{weight}(\mathbb{x}_\rho) \geq 0.95 \cdot \alpha$. Moreover, these shifts can be found in expected time $O(n \cdot \log n)$.
2. *Small Hamming weight.* For every $\mathbb{x}$ with $\mathsf{weight}(\mathbb{x}) = \alpha - \delta$ where $\delta \in (0, \alpha)$, and any choice of $z_1, \ldots, z_t$:
   (a) $\Pr_\rho[\mathsf{weight}(\mathbb{x}_\rho) \geq 0.95 \cdot \alpha] \leq 1.1 \cdot \frac{\alpha - \delta}{\alpha}$, and
   (b) $\mathbb{E}_\rho[\mathsf{weight}(\mathbb{x}_\rho)] = \alpha - \delta$.

Discussion. Following the definition of good shifts, a natural strategy emerges for verifying the Hamming weight of a vector $\mathbb{x}$: the prover sends good shifts $z_1, \ldots, z_t$, and the verifier needs to check that the shifts are indeed good (with perfect completeness). Recall that in the honest case, the induced vector $\mathbb{x}_\rho$ has large Hamming weight for every ρ, whereas if $\mathbb{x}$ has small Hamming weight, then $\mathbb{x}_\rho$ has small Hamming weight for most choices of ρ. Thus it is natural to sample $\rho \leftarrow [n]$ and check that the vector $\mathbb{x}_\rho := (\mathbb{x}[\rho + z_1], \ldots, \mathbb{x}[\rho + z_t])$ has high Hamming weight by querying $\mathbb{x}$ at all t locations. While perfectly complete and sound, this PCPP has bad parameters: it results in the verifier reading $O(t \cdot \log n) = O(\log^2 n)$ queries from the proof and making $O(t) = O(\log n)$ queries to the input.

In order to lower the query complexities, we would like to apply a PCPP for the claim that all of the vectors in $(\mathbb{x}_\rho)_{\rho \in [n]} \subseteq \{0,1\}^t$ have large Hamming weight. Naively, we could solve this by having the prover supply a separate proof showing that $\mathbb{x}_\rho$ has large Hamming weight for each $\rho \in [n]$. Alas, we cannot afford this since there would be $O(n)$ such proofs; even if each proof was one bit in length, we would miss our target of sublinear length. To overcome this challenge the prover will provide a proof that applies multiple choices of ρ simultaneously.

The protocol. For $a = O(\log^3 n)$,[4] let $(X_s)_{s \in [n/a]}$ be a partition of the induced vectors $(\mathbb{x}_\rho)_{\rho \in [n]}$ into n/a sets of size a. Our protocol, given $\mathbb{x} \in \{0,1\}^n$, is as follows:

1. Prover: Send shifts $z_1, \ldots, z_t$ which are good for $\mathbb{x}$. Then, for every $s \in [n/a]$ write an inner proof π_s claiming that all vectors in X_s have high Hamming weight (i.e., Hamming weight at least $0.95 \cdot \alpha$).

[4] This value for a is chosen for convenience. In the full protocol, it is left as a parameter which allows for tuning properties of the proof.

2. Verifier: Choose $s \leftarrow [n/a]$ uniformly at random. Run the inner proof verifier for the claim that all vectors in X_s have high Hamming weight, and accept if the inner proof verifier accepts.

We analyze the PCPP we constructed and derive properties which, if they are held by the inner protocol, are sufficient for our needs:

1. *Perfect completeness:* Following Item 1, it holds that if $\mathbb{x}$ has high Hamming weight, then, for every s, all vectors in X_s have high Hamming weight. Consequently, if the inner protocol has perfect completeness then this holds also for the final PCPP.

2. *Soundness:* Fix $\mathbb{x} \in \{0,1\}^n$ with Hamming weight $\alpha - \delta$, where $\delta \in (0, \alpha)$, and a prover message $z_1, \ldots, z_t, (\pi_s)_{s \in [n/a]}$. Let $\beta_s := \frac{1}{a} \sum_{i=1}^{a} \mathsf{weight}(X_s[i])$ be the average weight in X_s. By construction:

$$\begin{aligned} \Pr[\mathbf{V} \text{ accepts}] &= \Pr_s [\ \mathbf{V}^{\text{in}} \text{ accepts } X_s \text{ given } \pi_s] \\ &= \sum_{\beta} \Pr_s [\ \beta_s = \beta\ ] \cdot \Pr_s [\ \mathbf{V}^{\text{in}} \text{ accepts } X_s \text{ given } \pi_s \mid \beta_s = \beta] \quad . \end{aligned}$$

   In order to bound the error, it suffices that $\Pr_s [\ \mathbf{V}^{\text{in}} \text{ accepts } X_s \text{ given } \pi_s \mid \beta_s = \beta] \leq \beta \cdot \varepsilon$ for ε that does not depend on s and β (and this is what we will later achieve):

$$\Pr[\mathbf{V} \text{ accepts}] \leq \sum_{\beta} \Pr_s [\ \beta_s = \beta\ ] \cdot \beta \cdot \varepsilon = \mathbb{E}[\beta_s] \cdot \varepsilon = (\alpha - \delta) \cdot \varepsilon \quad .$$

   The final equality follows from Item 2b, which posits that $\mathbb{E}[\beta_s] = \frac{1}{a} \sum_{i=1}^{a} \mathbb{E}_\rho[\mathsf{weight}(\mathbb{x}_\rho)] = \alpha - \delta$. In our construction of the inner protocol, $\varepsilon \approx \frac{1}{\alpha}$, so that the soundness error is approximately $(\alpha - \delta)/\alpha$.

3. *Complexity parameters:* Let l^{in}, $\mathsf{q}^{\text{in}}_{\mathbb{x}}$, and $\mathsf{q}^{\text{in}}_{\pi}$ denote the proof length, input query complexity, and proof query complexity of the inner proof, respectively. Then the PCPP has proof length $t \cdot \log n + \frac{n}{a} \cdot \mathsf{l}^{\text{in}}$, and so to achieve sublinear proof length we require $\mathsf{l}^{\text{in}} = o(a) = o(\log^3 n)$. The input query complexity is $\mathsf{q}^{\text{in}}_{\mathbb{x}}$ and the proof query complexity is $\mathsf{q}^{\text{in}}_{\pi} + O(\mathsf{q}^{\text{in}}_{\mathbb{x}} \cdot \log n)$: each query to X_s induces a query to $\mathbb{x}_\rho[i] = \mathbb{x}[\rho + z_i]$ for some i and ρ which, in turn, induces an input query, and the reading of z_i (which has length $O(\log n)$). Finally, the verifier must read $\mathsf{q}^{\text{in}}_{\pi}$ bits from π_s. Thus, to achieve the parameters of the PCPP described in Theorem 1, we require $\mathsf{q}^{\text{in}}_{\mathbb{x}} = O(\text{loglog}\, n)$ and $\mathsf{q}^{\text{in}}_{\pi} = O(\log n \cdot \text{loglog}\, n)$.

The inner protocol. The inner protocol works on similar ideas to the outer protocol: the prover sends good shifts, and the verifier needs to check that these shifts are, indeed, good. This requires us to change the definition of good shifts to apply also to sets of vectors $X_s = (\mathbb{x}_1, \ldots, \mathbb{x}_a) \subseteq \{0,1\}^t$. Indeed, we show that, provided that $t' := \Theta(\frac{1}{\alpha} \cdot \log(a \cdot t)) = \Theta(\text{loglog}\, n)$, the following hold:

1. *Large Hamming weight.* For every $\mathbb{x}_1, \ldots, \mathbb{x}_a$ such that $\mathsf{weight}(\mathbb{x}_i) \geq \alpha$ for every $i \in [t]$, there exist shifts $z_1, \ldots, z_{t'} \in [t]$ that are good for all the vectors in the set simultaneously. Moreover, these shifts can be found in expected time $O(t \cdot a \cdot \log(t \cdot a)) = \mathrm{polylog}(n)$.
2. *Small average Hamming weight.* For every $\mathbb{x}_1, \ldots, \mathbb{x}_a$ with $\frac{1}{a}\sum_{i\in[a]} \mathsf{weight}(\mathbb{x}_i) = \alpha - \delta$ where $\delta \in (0, \alpha)$, and any choice of $z_1, \ldots, z_{t'} \in [t]$: $\Pr_{i,\rho}[\mathsf{weight}(\mathbb{x}_{i,\rho}) \geq 0.95 \cdot \alpha] \leq 1.1 \cdot \frac{\alpha-\delta}{\alpha}$, where $\mathbb{x}_{i,\rho} := (\mathbb{x}_i[\rho + z_1], \ldots, \mathbb{x}_i[\rho + z_{t'}])$.

This extended definition allows us to construct our inner protocol. The protocol proceeds as follows on input $X_s = (\mathbb{x}_1, \ldots, \mathbb{x}_a) \subseteq \{0,1\}^t$:

1. Prover: Send shifts $z_1, \ldots, z_{t'} \in [t]$ which are good for X_s.
2. Verifier: Read all of $z_1, \ldots, z_{t'}$, choose $i \leftarrow [a]$ and $\rho \leftarrow [t]$ uniformly at random, and check that $\mathsf{weight}(\mathbb{x}_i[\rho + z_1], \ldots, \mathbb{x}_i[\rho + z_{t'}]) \geq 0.95 \cdot \alpha$ by querying $\mathbb{x}_i$ at the appropriate locations.

The length of the proof is $\mathsf{l}^{\mathrm{in}} = t' \cdot \log n = O(\log n \cdot \mathrm{loglog}\, n) = o(a)$. The verifier reads this proof in its entirety, so $\mathsf{q}_\pi^{\mathrm{in}} = \mathsf{l}^{\mathrm{in}} = O(\log n \cdot \mathrm{loglog}\, n)$ bits. The input query complexity is $\mathsf{q}_{\mathbb{x}}^{\mathrm{in}} = t' = O(\mathrm{loglog}\, n)$.

Completeness follows from Item 1 of the adapted definition of good shifts. For soundness, recall that we wanted:

$$\Pr\left[\mathbf{V}^{\mathrm{in}} \text{ accepts } \mid \frac{1}{a}\sum_{i\in[a]} \mathsf{weight}(\mathbb{x}_i) = \beta\right] \leq \beta \cdot \varepsilon \ ,$$

for ε that does not depend on β. The verifier accepts only if $\mathsf{weight}(\mathbb{x}_i[\rho + z_1], \ldots, \mathbb{x}_i[\rho + z_{t'}]) \geq 0.95 \cdot \alpha$. It follows from Item 2 of the adapted definition of good shifts that the probability of this occurring when the average Hamming weight of the vectors in X_s is β is $1.1 \cdot \frac{\beta}{\alpha}$, which concludes the proof of soundness (here, $\varepsilon = \frac{1.1}{\alpha}$).

Finally, we observe that the inner protocol is useful in its own right. In fact, by choosing $a = 1$ (and replacing t with n) we get the MAP for Hamming weight described in Theorem 1.

2.2 SIQ-PCPP for Hamming weight

In this section, we focus on perfectly complete PCPPs for the Hamming weight problem, where the verifier makes a single input query (SIQ). In Section 2.2.1, we sketch the proof of Theorem 3, showing a lower bound on the proof length for SIQ MAPs (which induces a bound also for PCPPs). In Section 2.2, we give a construction of a SIQ PCPP with linear proof size (Theorem 4).

2.2.1 Lower bound

In this section, we sketch the proof of Theorem 3, showing that any perfectly complete MAP for Hamming weight with a single input query must have a large proof length. Let $\alpha = 2/3$ and $\delta = 1/3$.

Consider a MAP with message length l, and input query complexity 1, where for inputs of distance δ from Hamming weight has nontrivial soundness error. Let $\mathsf{Exact\text{-}Ham}_{\alpha,n} \subseteq \mathsf{Ham}_\alpha$ be the set of all vectors of size n that have Hamming weight exactly α. For a prover message π, let S_π be the set of all vectors in $\mathsf{Exact\text{-}Ham}_{\alpha,n}$ for which π is the honest prover message. By an averaging argument, since there are at most 2^{l} different prover messages, there must exist some proof π with $|S_\pi| \geq |\mathsf{Exact\text{-}Ham}_{\alpha,n}|/2^{\mathsf{l}}$. Since each vector in $\mathsf{Exact\text{-}Ham}_{\alpha,n}$ has exactly $(1-\alpha)\cdot n$ zeros, $|\mathsf{Exact\text{-}Ham}_{\alpha,n}| = \binom{n}{(1-\alpha)\cdot n}$. However, the number of vectors that the honest prover can prove with the same proof is small: Claim 1 shows that $|S_\pi| \leq \binom{(1-\alpha+\delta)\cdot n}{(1-\alpha)\cdot n}$. By rearranging the terms and taking a logarithm, we get that $\mathsf{l} \geq \log\binom{n}{(1-\alpha)\cdot n} - \log\binom{(1-\alpha+\delta)\cdot n}{(1-\alpha)\cdot n} = \Omega(n)$ (the final equality follows since $\alpha = 2/3$ and $\delta = 1/3$).

Claim 1. $|S_\pi| \leq \binom{(1-\alpha+\delta)\cdot n}{(1-\alpha)\cdot n}$.

Proof sketch. Let $\mathtt{x}$ be the bitwise AND of all the vectors in S_π and let I_0 be the indices where $\mathtt{x}$ is zero. By the definition of $\mathtt{x}$, for every $\mathtt{v} \in S_\pi$ and $j \in [n]$ where $\mathtt{v}[j] = 0$, it must hold that $j \in I_0$ (i.e., $\mathtt{x}[j] = 0$). Thus, we can bound the size S_π by the number of vectors that have zeroes only within I_0. Since $S_\pi \subseteq \mathsf{Exact\text{-}Ham}_{\alpha,n}$, every $\mathtt{v} \in S_\pi$ has exactly $(1-\alpha)\cdot n$ zeroes. The maximal number of vectors with exactly $(1-\alpha)\cdot n$ zeroes that have zeroes only within I_0 is $\binom{|I_0|}{(1-\alpha)\cdot n}$. It follows that $|S_\pi| \leq \binom{|I_0|}{(1-\alpha)\cdot n}$. We now show that $|I_0| < (1-\alpha+\delta)\cdot n$, which completes the proof.

Assume towards contradiction that $|I_0| \geq (1-\alpha+\delta)\cdot n$, meaning that $\mathtt{x}$ is δ far from Ham_α. By the soundness property of the protocol, there exists some randomness ρ for which the verifier rejects. Let $j \in [n]$ be the index of the single bit of $\mathtt{x}$ queried by the verifier given access to π and randomness ρ. Since $\mathtt{x}$ is the bitwise AND of the vectors in S_π, there exists some vector $\mathtt{v} \in S_\pi$ with $\mathtt{v}[j] = \mathtt{x}[j]$. Fix such vector $\mathtt{v}$. Since j is the only index queried by the verifier when given the proof π and the randomness ρ, the verifier will reject $\mathtt{v}$. This contradicts the perfect completeness of the protocol as $\mathtt{v} \in S_\pi \subseteq \mathsf{Exact\text{-}Ham}_{\alpha,n} \subseteq \mathsf{Ham}_\alpha$, and since π is the honest prover's message for $\mathtt{v}$ (by definition of S_π). □

2.2.2 Upper bound

In this section, we describe our construction of a PCPP for Hamming weight where the verifier makes a single input query, and the proof length is $n + O(\log^2 n)$, as described in Theorem 4.

The protocol. The basic idea underlying how our protocol achieves single input query complexity is that the prover copies the vector $\mathtt{x}$ into the proof and writes a proof that this copy has large Hamming weight. The verifier then checks that the copy has large Hamming weight and that the copy is consistent with $\mathtt{x}$. On input $\mathtt{x} \in \{0,1\}^n$, the protocol proceeds as follows:

1. Prover: Send $\mathbb{x}' := \mathbb{x}$ and additionally send shifts $z_1, \ldots, z_t \in [n]$ which are good for $\mathbb{x}$.
2. Verifier: Read all of $z_1, \ldots, z_t$ and accept if the following checks pass:
   (a) Choose $\rho_1, \ldots, \rho_m \leftarrow [n]$ for $m = O(\log n)$ and check that for every ℓ, the induced vector $\mathbb{x}_{\rho_\ell} := (\mathbb{x}'[\rho_i + z_1], \ldots, \mathbb{x}'[\rho_i + z_t])$ has Hamming weight $\geq 0.95 \cdot \alpha$ by querying $\mathbb{x}'$ at the appropriate locations.
   (b) Choose $r_1, \ldots, r_q \leftarrow [n]$ for $q = O(\log n)$ and query $\mathbb{x}'[r_i]$. If $\mathbb{x}'[r_i] = 0$ for every i, then we consider the check to have passed. Otherwise, let ℓ be the minimal index so that $\mathbb{x}'[r_\ell] = 1$. Check that $\mathbb{x}[r_\ell] = 1$ by querying $\mathbb{x}$.

Analysis. We begin by assessing the complexity parameters. The proof length is $n + t \cdot \log n = n + O(\log^2 n)$, and the verifier makes at most one query to $\mathbb{x}$, and $O(\log^2 n)$ queries to the proof string. Perfect completeness follows from Item 1 of the definition of good shifts and from the fact that the honest vector sets $\mathbb{x}' = \mathbb{x}$. All that remains is to show soundness.

Fix $\mathbb{x} \in \{0,1\}^n$ that is δ-far from Ham_α and a prover message $\pi = (\mathbb{x}', z_1, \ldots, z_t)$. Intuitively, if the Hamming weight of the copied vector is close to the Hamming weight of the input vector, then the verifier's check in Item 2a will fail with high probability. On the other hand, if the Hamming weight of the copied vector is much larger than the Hamming weight of the input vector, then there is a large disparity between $\mathbb{x}$ and $\mathbb{x}'$ so the verifier's check in Item 2b will fail with high probability. We now formalize this intuition.

For every $\rho \in [n]$, let $\mathbb{x}'_\rho := (\mathbb{x}'[\rho_i + z_1], \ldots, \mathbb{x}'[\rho_i + z_t])$, and let $\mathcal{H} := \{\rho \mid \mathsf{weight}(\mathbb{x}'_\rho) \geq 0.95 \cdot \alpha\}$ be the set of all ρ such that the vector $\mathbb{x}'_\rho$ has Hamming weight $\geq 0.95 \cdot \alpha$. We split the analysis into two cases: $|\mathcal{H}| < n/2$, and $|\mathcal{H}| \geq n/2$. If $|\mathcal{H}| < n/2$ then by the definition of $\mathcal{H}$, the verifier's check in Item 2a will pass with probability at most $(1/2)^m = (1/n)^{O(1)}$. If $|\mathcal{H}| \geq n/2$ then the verifier accepts with probability at most $1.1 \cdot \frac{\alpha - \delta}{\alpha} + \frac{1}{n^{O(1)}}$, as exemplified in Lemma 1. Overall, we conclude that the verifier accepts with probability at most $1.1 \cdot \frac{\alpha - \delta}{\alpha} + \frac{1}{n^{O(1)}}$ as described in Theorem 4.

Lemma 1. *If $|\mathcal{H}| \geq n/2$ then the verifier accepts with probability at most $1.1 \cdot \frac{\alpha - \delta}{\alpha} + \frac{1}{n^{O(1)}}$.*

Proof sketch. Let $\beta' := \mathsf{weight}(\mathbb{x}')$ be the Hamming weight of $\mathbb{x}'$. To begin with, we show that $\beta' \geq 0.95 \cdot \alpha/2$ by observing that, by the definition of good shifts (specifically Item 2b), $\mathsf{weight}(\mathbb{x}') = \mathbb{E}_\rho[\mathbb{x}'_\rho]$. Since $|\mathcal{H}| \geq n/2$, at least half of the vectors $\mathbb{x}'_\rho$ have weight at least $0.95 \cdot \alpha$, and so $\mathbb{E}_\rho[\mathbb{x}'_\rho] \geq 0.95 \cdot \alpha/2$.

Define the following events: E_{weight} is the event that the verifier's check in Item 2a passes, E_0 is the event that the verifier's check in Item 2b passes because the verifier read only zeros from $\mathbb{x}'$, and E_1 is the event that the verifier's check in Item 2b passes and the verifier has read a nonzero entry of $\mathbb{x}'$. Using this notation,

$$\Pr[\text{Verifier accepts}] = \Pr[E_{\mathsf{weight}} \wedge (E_0 \vee E_1)]$$

Since the two checks of the verifier are independent, and by using the union-bound:

$$\Pr[E_{\mathsf{weight}} \wedge (E_0 \vee E_1)] = \Pr[E_{\mathsf{weight}}] \cdot \Pr[E_0 \vee E_1] \leq \Pr[E_{\mathsf{weight}}] \cdot (\Pr[E_0] + \Pr[E_1]) \ .$$

We bound the probabilities that E_0 and E_1 occur:

- $\Pr[E_0]$: Since $\beta' \geq 0.95 \cdot \alpha/2$, the probability that all of the samples r_i are to locations where $\mathbb{x}'$ contains 0 is at most: $(1 - 0.95 \cdot \alpha/2)^q = 1/n^{O(1)}$.
- $\Pr[E_1]$: Conditioned on sampling a nonzero index in $\mathbb{x}'$, the smallest such index is distributed uniformly over the nonzero entries of $\mathbb{x}'$. The fraction of ones in $\mathbb{x}'$ is β' and the fraction of ones in $\mathbb{x}$ is $\alpha - \delta$, and so sampling a random nonzero entry in $\mathbb{x}'$ is nonzero in $\mathbb{x}$ with probability at most $(\alpha - \delta)/\beta'$. Consequently: $\Pr[E_1] \leq \Pr[E_1 \mid \text{sampled nonzero location}] \leq (\alpha - \delta)/\beta'$.

Therefore, $\Pr[\text{Verifier accepts}] \leq \Pr[E_{\mathsf{weight}}] \cdot (\Pr[E_0] + \Pr[E_1]) \leq \Pr[E_{\mathsf{weight}}] \cdot (\frac{\alpha-\delta}{\beta'} + \frac{1}{n^{O(1)}})$. We now split the argument into two cases. In both cases, we show that the verifier accepts with probability at most $1.1 \cdot (\alpha - \delta)/\alpha + 1/n^{O(1)}$, which concludes this proof sketch.

1. If $\alpha < 1.1 \cdot \beta'$: then $\Pr[\text{Verifier accepts}] \leq \frac{\alpha-\delta}{\beta'} + \frac{1}{n^{O(1)}} < 1.1 \cdot \frac{\alpha-\delta}{\alpha} + \frac{1}{n^{O(1)}}$.
2. If $\alpha \geq 1.1 \cdot \beta'$: then $\Pr_\rho[\mathsf{weight}(\mathbb{x}'_\rho) \geq 0.95 \cdot \alpha] \leq 1.1 \cdot \frac{\beta'}{\alpha}$ by Item 2a in the properties of good shifts. Therefore, the probability that the verifier's check in Item 2a passes is $\left(1.1 \cdot \frac{\beta'}{\alpha}\right)^m$. Thus, the verifier accepts with probability at most:

$$\Pr[\text{Verifier accepts}] \leq \left(1.1 \cdot \frac{\beta'}{\alpha}\right)^m \cdot \left(\frac{\alpha - \delta}{\beta'} + \frac{1}{n^{O(1)}}\right) \leq 1.1 \cdot \frac{\alpha - \delta}{\alpha} + \frac{1}{n^{O(1)}} \ .$$

□

2.3 A SIQ-IOPP for Hamming weight with sublinear proof length

In this section, we show that by utilizing interaction, perfect completeness, sublinear proof length, and input query complexity 1 are all simultaneously achievable. This is in stark contrast to the non-interactive (i.e., PCPP) case where, as shown in Section 2.2.1, achieving all three properties together is impossible.

Recall that an IOP is a generalization of PCPs, where the prover and verifier interact over multiple rounds. Similarly to a PCP, the verifier is given oracle access to the messages supplied by the prover. We sketch the construction of the IOPP in Theorem 1, showing an IOPP for the Hamming weight problem with perfect completeness, input query complexity 1, four messages, proof length polylog(n) and proof query complexity $O(\log n)$.

Discussion. The IOPP utilizes the ideas developed in Section 2.1. Specifically, recall the initial construction proposed: The prover generates good shifts $z_1, \ldots, z_t$ for $t = \text{polylog}(n)$, the verifier chooses ρ, and needs to verify that $\mathbb{x}_\rho := (\mathbb{x}[\rho + z_1], \ldots, \mathbb{x}[\rho + z_t])$ has at least a $0.95 \cdot \alpha$ fraction of ones. In the non-interactive case, we ran into the problem that we could not write a proof of this fact for every possible $\mathbb{x}_\rho$, i.e., for every choice of ρ. Our observation is that if the protocol is allowed to be interactive, it suffices for the prover to

give an "inner proof" that $\mathtt{x}_\rho$ has high Hamming weight only for the single ρ chosen by the verifier.

Since this inner statement itself has size polylog(n), which is the communication complexity that we are already willing to afford, a simple proof will suffice: the prover will send $\mathtt{x}_\rho$ to the verifier, who will check that it has high Hamming weight. It then checks that $\mathtt{x}_\rho$ matches the restriction of the real vector $\mathtt{x}$. While we could do this by simply comparing the two on a random location, since we are only really interested in the restriction of $\mathtt{x}$ having many nonzero entries, we get better soundness error by choosing a random nonzero location j of $\mathtt{x}_\rho$ and checking that $\mathtt{x}[\rho + z_j] = 1$.

The protocol. Given $\mathtt{x} \in \{0,1\}^n$, the protocol proceeds as follows:

1. Prover: Send good shifts $z_1, \ldots, z_t \in [n]$.
2. Verifier: Choose $\rho \leftarrow [n]$ uniformly at random.
3. Prover: Send $\mathtt{x}_\rho \in \{0,1\}^t$, where in the honest case $\mathtt{x}_\rho := (\mathtt{x}[\rho + z_1], \ldots, \mathtt{x}[\rho + z_t])$.
4. Verifier: Read $\mathtt{x}_\rho$ in its entirety, and (a) check that $\mathsf{weight}(\mathtt{x}_\rho) \geq 0.95 \cdot \alpha$, and (b) choose a random j from the indices where $\mathtt{x}_\rho$ is nonzero, and check that $\mathtt{x}[\rho + z_j] = 1$.

Completeness and soundness follow straightforwardly from the analysis done in Section 2.1 regarding good shifts, and a simple probabilistic argument showing that if the prover did not send the correct $\mathtt{x}_\rho$, then it will be caught by the verifier with high probability.

2.4 A lower bound for IPPs and semi-adaptive IOPPs

In this section, we discuss the proof of Theorem 2, showing a trade-off between the length, input query complexity and proof query complexity of IPPs and *semi-adaptive* IOPPs. IPPs are a special case of semi-adaptive IOPPs, and so in this overview we consider semi-adaptive IOPPs unless stated otherwise. In the full proof a minor optimization is utilized to give better bounds for IPPs.

We prove our lower-bound in two steps: (1) in Section 2.4.1 we introduce a communication complexity problem which we call HitOne, and prove a lower bound for it, and (2) in Section 2.4.2 we show that any semi-adaptive IOPP for Hamming weight with one-sided error can be used as a strategy to solve the HitOne problem. Together, these introduce bounds on the parameters of the IOPP.

Semi-adaptive IOPPs. A k-round IOPP is *semi-adaptive* if the locations of the verifier's queries made to the prover's i-th message π_i (also known as the verifier's *view* of this oracle) depend only on $\rho_1, \ldots, \rho_i$ and the verifier's view of the prover's messages π_1 through π_i.[5] See Section 8 for a formal definition.

[5] Adaptivity with respect to π_i is allowed: the verifier's j-th query π_i can depend on the previous $j-1$ queries to made to π_i.

2.4.1 The HitOne problem

In the HitOne problem, Alice, given a binary vector $\mathbb{x} \in \{0,1\}^n$ with at least $\alpha(n)$ fraction of ones, must communicate to Bob the location of a single 1 (for this sketch we consider constant α). In more detail, Alice is given $\mathbb{x}$ and outputs a message m. Bob then reads this message and makes q queries to $\mathbb{x}$. The goal is for Bob to query a nonzero location of $\mathbb{x}$ (with probability 1) while minimizing the size of the message m and the number of queries made to $\mathbb{x}$ (indeed, the task is trivial if $|m| = \log n$ or $q > (1-\alpha) \cdot n$).

We show that $|m| = \Omega(\log(n/q))$. To see why, suppose towards contradiction that $m = o(\log(n/q))$. We construct a vector of length n with Hamming weight α for which Bob queries only zeroes, which contradicts the correctness of the protocol. For every one of the $2^m = o(n/q)$ possible verifier messages, set the q locations queried by Bob to be 0. Set the rest of the vector to be all ones. The vector contains at most $2^m \cdot q = o(n)$ zeroes, and so has an α-fraction of ones (recall that in this sketch, we are assuming that α is constant). On the other hand, by how we defined the vector, no matter what message Alice sends, Bob will always query the vector at locations that all contain zeroes.

2.4.2 IOPP to HitOne.

We show how to transform any semi-adaptive IOPP $(\mathbf{P}, \mathbf{V})$ for Hamming weight α with perfect completeness into a strategy for HitOne (as in the rest of this technical overview, we consider constant α). Specifically, we show that given an IOPP for Hamming weight with perfect completeness, it can be converted into a strategy for HitOne where Alice sends a message of length $O(\mathsf{q}_\pi \cdot \log \mathsf{l})$ and Bob makes $\mathsf{q}_\mathbb{x}$ queries, where q_π, l and $\mathsf{q}_\mathbb{x}$ are the proof query complexity, length and input query complexity of the IOPP respectively. When put together with the lower-bound for HitOne described in Section 2.4.1, we conclude that $O(\mathsf{q}_\pi \cdot \log \mathsf{l}) > \log(n/\mathsf{q}_\mathbb{x})$. For constant $\mathsf{q}_\mathbb{x}$ and polylogarithmic proof length, we conclude that $\mathsf{q}_\pi = \Omega(\log n / \operatorname{loglog} n)$.

In this section, fix $\mathbb{x} \in \{0,1\}^n$ to be a vector with at least $\alpha \cdot n$ ones for the HitOne problem. We describe the transformation for 4-message IOPPs for Hamming weight of $\alpha \cdot n$. This can be readily generalized for any number of messages.

Warm up. We start with a transformation where Alice's message length is linear in the verifier's randomness complexity. We define notions of *useful* random strings: (a) (ρ_1, ρ_2) are useful if for $\pi_1 := \mathbf{P}(\mathbb{x})$ and $\pi_2 := \mathbf{P}(\mathbb{x}, \rho_1)$ it is the case that the IOPP verifier rejects the all zeroes vector, i.e., $\mathbf{V}^{\vec{0},\pi_1,\pi_2}(\rho_1, \rho_2) = 0$, and (b) ρ_1 is useful if there exists ρ_2 such that (ρ_1, ρ_2) are useful. The definition of useful strings is exemplified by the following claim:

Claim 2. *If* $\mathbf{V}^{\vec{0},\pi_1,\pi_2}(\rho_1, \rho_2) = 0$ *where* $\pi_1 := \mathbf{P}(\mathbb{x})$ *and* $\pi_2 := \mathbf{P}(\mathbb{x}, \rho_1)$*, then* $\mathbf{V}^{\mathbb{x},\pi_1,\pi_2}(\rho_1, \rho_2)$ *queries* $\mathbb{x}$ *at a nonzero location.*

Proof sketch. By the perfect completeness of the IOPP, we have that $\mathbf{V}^{\mathbb{x},\pi_1,\pi_2}(\rho_1, \rho_2) = 1$ since π_1 and π_2 were generated honestly with respect to $\mathbb{x}$. On the other hand by the claim

statement, it holds that $\mathbf{V}^{\vec{0},\pi_1,\pi_2}(\rho_1,\rho_2) = 0$. The only difference between the executions is the existence of ones in the vector $\mathbb{x}$, and these are only accessed via verifier queries. Therefore, $\mathbf{V}$ must query $\mathbb{x}$ at a nonzero location. □

Claim 2 yields a natural strategy for HitOne: Alice computes $\pi_1 := \mathbf{P}(\mathbb{x})$, finds the (lexicographically) smallest ρ_1 that is useful, computes $\pi_2 := \mathbf{P}(\mathbb{x},\rho_1)$, and chooses the smallest ρ_2 so that (ρ_1,ρ_2) are useful (such a choice exists since ρ_1 is useful). Finally she outputs as her message $m = (\pi_1,\rho_1,\pi_2,\rho_2)$. Bob runs $\mathbf{V}^{\mathbb{x},\pi_1,\pi_2}(\rho_1,\rho_2)$ making the same queries to $\mathbb{x}$ as made by $\mathbf{V}$.

Alice's message has length $O(\mathsf{r}+\mathsf{l})$ and Bob makes $\mathsf{q}_{\mathbb{x}}$ queries, where r, l and $\mathsf{q}_{\mathbb{x}}$ are the randomness, length, and input query complexity of the IOPP respectively. As mentioned in Section 2.4.1, the HitOne problem is trivial if Alice's message is allowed to have length $\log n$. Both r and l are commonly at least logarithmic in n, and so we need to reduce the dependency on r and l. We lower the dependency on l by observing that $\mathbf{V}$ reads π_1 and π_2 only at a few locations. It therefore suffices for Alice to send Bob the views w_1 and w_2 of π_1 and π_2 respectively their stead (where each view contains both the query locations and the values read from its respective proof), getting us to length $O(\mathsf{r}+\mathsf{q}_\pi\cdot\log\mathsf{l})$ where q_π is the proof query complexity of the IOPP.

We are left with the goal of reducing the dependency on r. In fact, we will completely eliminate it by removing ρ_1 and ρ_2 from Alice's message, and having Bob *infer* them given only w_1 and w_2.

Second attempt. We would like for Bob to mimic the way that Alice chooses ρ_1 and ρ_2. In order to do so, naively Bob would have to compute π_1 and π_2, which requires knowledge of $\mathbb{x}$ that Bob does not have. While Bob does not have access to π_1 and π_2, he knows that the randomness chosen by Alice is consistent with w_1 and w_2, i.e., ρ_1 and ρ_2 never cause $\mathbf{V}$ to query outside of w_1 and w_2. Thus we have the following strategy for Bob: choose the smallest ρ_1 that is consistent with w_1 and w_2 where there exists a consistent ρ_2 such that $\mathbf{V}^{\vec{0},\mathsf{w}_1,\mathsf{w}_2}(\rho_1,\rho_2) = 0$, where, by oracle access to w_1 and w_2 we mean that Bob emulates the verifier's access to π_1 and π_2 using the information in w_1 and w_2. Since ρ_1 and ρ_2 are consistent with $\mathsf{w}_1,\mathsf{w}_2$ the verifier only queries inside the views, and so this operation is well-defined. This choice of ρ_1 immediately also gives Bob a choice of a consistent ρ_2.

Alas, this does guarantee that (ρ_1,ρ_2) are useful due to a circular dependency: Bob's choice of ρ_1 depends on w_2, whereas the honestly generated w_2 depends on ρ_1 (since $\pi_2 := \mathbf{P}(\mathbb{x},\rho_1)$). In order to exemplify this we give a (contrived) example of this issue. Consider an IOPP where the honest proof $\pi_2 := \mathbf{P}(\mathbb{x},\rho_1)$ contains at its first index the first bit of ρ_1. Following the interaction, the verifier queries $\pi_2[1]$ and checks that $\rho_1[1] = \pi_2[1]$. If this does not hold, the verifier immediately rejects without querying $\mathbb{x}$.

Alice chooses ρ_1^{A} and ρ_2^{A} as above and uses them to generate $(\mathsf{w}_1,\mathsf{w}_2)$. For any ρ_1^{B} and ρ_2^{B} that Bob may choose where $\rho_1^{\mathsf{B}}[1] \neq \rho_1^{\mathsf{A}}[1]$, it holds that $\mathbf{V}^{\vec{0},\mathsf{w}_1,\mathsf{w}_2}(\rho_1^{\mathsf{B}},\rho_2^{\mathsf{B}}) = 0$. Since ρ_1^{B}

is inconsistent with w_2, the verifier does not query $\mathbb{x}$ and, consequently, Bob will also not query $\mathbb{x}$ (let alone at a nonzero location).

To resolve this issue, we need to remove this circular dependency.

The transformation. To resolve the circular dependency, we choose ρ_1 using a property that is stronger than being useful: we choose ρ_1 if, given $\pi_1 := \mathbf{P}(\mathbb{x})$, for *every* π_2 there exists some ρ_2 so that $\mathbf{V}^{\vec{0},\pi_1,\pi_2}(\rho_1,\rho_2) = 0$. We show that this definition suffices, beginning with the protocol:

- Alice, given $\mathbb{x}$:
  1. Compute $\pi_1 := \mathbf{P}(\mathbb{x})$.
  2. Let ρ_1 be the (lexicographically) smallest string so that for every π_2' there exists ρ_2' such that $\mathbf{V}^{\vec{0},\pi_1,\pi_2'}(\rho_1,\rho_2') = 0$.
  3. Compute $\pi_2 := \mathbf{P}(\mathbb{x},\rho_1)$.
  4. Let ρ_2 be the smallest string so that $\mathbf{V}^{\vec{0},\pi_1,\pi_2}(\rho_1,\rho_2) = 0$.
  5. Send $(\mathsf{w}_1,\mathsf{w}_2)$, which are $\mathbf{V}$'s views of π_1 and π_2 (respectively) in the execution $\mathbf{V}^{\mathbb{x},\pi_1,\pi_2}(\rho_1,\rho_2)$.
- Bob, given $(\mathsf{w}_1,\mathsf{w}_2)$ and oracle access to $\mathbb{x}$:
  1. Let ρ_1 be the smallest string that is consistent with w_1 and for every π_2' there exists ρ_2' such that $\mathbf{V}^{\vec{0},\mathsf{w}_1,\pi_2'}(\rho_1,\rho_2') = 0$.
  2. Let ρ_2 be the smallest string that is consistent with w_2 so that $\mathbf{V}^{\vec{0},\mathsf{w}_1,\mathsf{w}_2}(\rho_1,\rho_2) = 0$.
  3. Run $\mathbf{V}^{\mathbb{x},\mathsf{w}_1,\mathsf{w}_2}(\rho_1,\rho_2)$ making the same queries that it makes to $\mathbb{x}$.

Alice sends $O(\mathsf{q}_\pi \cdot \log \mathsf{l})$ bits to Bob, who makes $\mathsf{q}_\mathbb{x}$ queries.Theorem 2 follows by applying this transformation to the bound derived in Section 2.4.1. If the IOPP is an IPP (i.e., $\mathsf{q}_\pi = \mathsf{l}$), then the verifier's view contains the entire proof, so Alice does not need to send indices in the proof and her message length can be decreased to l. This optimization yields the improved bound for IPPs described in Theorem 2. We sketch the proof showing that Bob must query a nonzero index of $\mathbb{x}$.

Lemma 2. *Bob queries a nonzero index of* $\mathbb{x}$.

Proof sketch. We show that Alice is able to find (ρ_1,ρ_2), and that Bob derives the same (ρ_1,ρ_2). Since these strings are such that $\mathbf{V}^{\vec{0},\pi_1,\pi_2}(\rho_1,\rho_2) = \mathbf{V}^{\vec{0},\mathsf{w}_1,\mathsf{w}_2}(\rho_1,\rho_2) = 0$, it follows by Claim 2 that Bob queries $\mathbb{x}$ at a nonzero location.

- *Alice finds some* (ρ_1,ρ_2). We first show that Alice will have a choice of ρ_1: indeed, suppose towards contradiction that for every ρ_1 there exists π_2' such that for every ρ_2' it holds that $\mathbf{V}^{\vec{0},\pi_1,\pi_2'}(\rho_1,\rho_2') = 1$. If this is the case, then a malicious Prover could convince the verifier to accept the all-zeroes vector with probability 1 by sending π_1, getting challenge ρ_1, and then following the strategy to get a π_2' that is accepted by the verifier for every

ρ_2'. This contradicts the soundness of the IOPP. Once ρ_1 has been chosen, $\pi_2 := \mathbf{P}(\mathbb{x}, \rho_1)$ is defined. Then, since ρ_1 was chosen, it must be the case that there exists ρ_2 for Alice to choose where $\mathbf{V}^{\vec{0},\pi_1,\pi_2}(\rho_1, \rho_2) = 0$.

- *Bob chooses the same* (ρ_1, ρ_2). We show that Alice and Bob agree on ρ_1. Agreement on ρ_2 follows by a similar argument. Bob goes over ρ_1^* in lexicographic order. We show that Bob does not choose $\rho_1^* < \rho_1$ (this is where we will use the fact that the IOPP is semi-adaptive) and that when it reaches $\rho_1^* = \rho_1$ it chooses this string.

  - $\rho_1^* < \rho_1$: Suppose towards contradiction that Bob chooses $\rho_1^* < \rho_1$. Since Alice did not choose ρ_1^*, it holds that there exists π_2' such that for every ρ_2, $\mathbf{V}^{\pi_1,\pi_2'}(\rho_1^*, \rho_2) = 1$. Furthermore, since Bob chose ρ_1^*, it holds that ρ_1^* is consistent with w_1. Since the IOPP is semi-adaptive, the view of the verifier of π_1 depends only on its first randomness ρ_1^*. Thus, ρ_1 and ρ_1^* induce the same view w_1 from π_1, and they do so regardless of π_2' and ρ_2'. We conclude that there exists π_2' such that for every ρ_2' it holds that $\mathbf{V}^{\mathsf{w}_1,\pi_2'}(\rho_1^*, \rho_2') = \mathbf{V}^{\pi_1,\pi_2'}(\rho_1^*, \rho_2') = 1$. This is a contradiction to the fact that, since Bob has chosen ρ_1^*, it holds that for every π_2' there exists ρ_2' so that $\mathbf{V}^{\mathsf{w}_1,\pi_2'}(\rho_1^*, \rho_2') = 0$.
  - $\rho_1^* = \rho_1$: Since Alice chose ρ_1, it holds that for every π_2' there exists a ρ_2' so that $\mathbf{V}^{\pi_1,\pi_2'}(\rho_1, \rho_2') = 1$. Moreover, by definition ρ_1 is consistent with w_1. It follows that for every π_2' there exists ρ_2' so that $\mathbf{V}^{\mathsf{w}_1,\pi_2'}(\rho_1, \rho_2') = 0$. Therefore Bob will choose $\rho_1^* = \rho_1$ when it is reached.

□

2.5 Application: perfect completeness for PCPs and IOPs

In this section, we show, as an application of our main results, how to transform PCPs and IOPs for arbitrary languages with two-sided error into ones with perfect completeness.

Perfect completeness for PCPs. Consider a PCP system for a language L with completeness error c and soundness error s where the verifier uses r bits of randomness. We reduce the completeness error to 0 using any of the PCPPs (or the MAP) for the Hamming weight problem described in previous sections.

Given an instance x and the honest prover's proof π, we can define a binary vector $\mathbb{x}$ of length 2^{r} where at index $\rho \in \{0,1\}^{\mathsf{r}}$ the vector $\mathbb{x}$ is equal to 1 if and only if the PCP verifier accepts given x, randomness ρ and oracle access to π. If $x \in L$ then $\mathbb{x}$ has at least $(1-\mathsf{c}) \cdot 2^{\mathsf{r}}$ ones, and if $x \notin L$ then $\mathbb{x}$ has at most $\mathsf{s} \cdot 2^{\mathsf{r}}$ ones. Given a perfectly complete PCPP (or MAP) for asserting that $\mathbb{x}$ has at least a $1-\mathsf{c}$ fraction of ones, we produce a PCP for L with perfect completeness: The new PCP contains the original PCP proof and the PCPP proof that $\mathbb{x}$ has many ones. The new PCP verifier runs the PCPP verifier, where in order to query $\mathbb{x}$ at ρ, the verifier runs the original PCP verifier on randomness ρ, and

outputs the PCP verifier decision as the value in $\mathtt{x}[\rho]$. Perfect completeness and soundness follow from the completeness and soundness of the PCPP.

Perfect completeness for IOPs. One would hope that the above approach for PCPs would also work to eliminate completeness error in IOPs. Unfortunately, this does not seem to be the case: defining a static $\mathtt{x}$ relying on verifier randomness in all rounds combined seems incompatible with the reliance of IOPs on interaction to achieve soundness. Due to this difficulty, we do not give a generic transformation for achieving perfect completeness in IOPs given an IOPP for the Hamming weight problem. Nonetheless, we show that the IOPP described in Section 2.3 can be adapted to transform any IOP with two-side error into one with perfect completeness.

3 Preliminaries

For a vector $\mathbb{x} \in \{0,1\}^n$ and an index $i \in \mathbb{N}$, we let $\mathbb{x}[i] := \mathbb{x}[i \mod n]$. For interactive (oracle) algorithms $\mathbf{A}$ and $\mathbf{B}$, we denote by $\langle \mathbf{A}(a), \mathbf{B}(b) \rangle (c)$ the random variable describing the output of $\mathbf{B}$ following the interaction between $\mathbf{A}$ and $\mathbf{B}$, where $\mathbf{A}$ is given private input a, $\mathbf{B}$ is given private input b and both parties are given joint input c. We define a function $f : \mathbb{N} \to (0,1]$ to be *computable in linear time* if the time to compute $f(x)$ is linear in the size of the binary representation of x. Moreover, for any two functions $f, f' : \mathbb{N} \to (0,1]$, we say that $f < f'$ if for any $x \in \mathbb{N}$, $f(x) < f'(x)$.

3.1 Hamming weight problem and Hamming distance

In this paper, we consider the relative hamming weight of bit vectors:

Definition 3.1 ((Relative) Hamming weight)**.** *The relative Hamming weight of a bit vector $\mathbb{x} \in \{0,1\}^n$, denoted $\mathsf{weight}(\mathbb{x})$, is the fraction of ones in $\mathbb{x}$:*

$$\mathsf{weight}(\mathbb{x}) = \frac{1}{n} \cdot |\{i \in [n] \mid \mathbb{x}[i] = 1\}| \quad .$$

The main language that we consider in this paper is α-Hamming-weight, which consists of all bit vectors of weight at least α:

Definition 3.2 (α-Hamming-weight language)**.** *For $\alpha : \mathbb{N} \to (0,1]$, the α-Hamming-weight language, Ham_α, is the set of all bit vectors with Hamming weight at least $\alpha(\cdot)$, where α is a function on the size of the vector:*

$$\mathsf{Ham}_\alpha := \bigcup_{n \in \mathbb{N}} \{\mathbb{x} \mid \mathbb{x} \in \{0,1\}^n \wedge \mathsf{weight}(\mathbb{x}) \geq \alpha(n)\} \quad .$$

We define a language that k-Ham_α contains all lists of k vectors that all have Hamming weight α. The language is defined so that 1-$\mathsf{Ham}_\alpha \equiv \mathsf{Ham}_\alpha$.

Definition 3.3 ((k,α)-list-Hamming-weight language)**.** *For $k \in \mathbb{N}$ and $\alpha \in (0,1]$, the (k,α)-list-Hamming-weight language, k-Ham_α, is the language of all lists of k vectors of identical length, each of which has Hamming weight at least α:*

$$k\text{-}\mathsf{Ham}_\alpha := \bigcup_{n \in \mathbb{N}} \{\mathbb{x}_1, \ldots, \mathbb{x}_k \in \{0,1\}^n \mid \forall i \in [k],\ \mathsf{weight}(\mathbb{x}_i) \geq \alpha(n)\} \quad .$$

We use Hamming distance as our measure of distance from the α-Hamming-weight language:

Definition 3.4 (Hamming distance)**.** *Let n, k be parameters in $\mathbb{N}$. For any two bit vectors $\mathbb{x}, \mathbb{x}' \in \{0,1\}^n$, denote $\Delta(\mathbb{x}, \mathbb{x}')$ as the Hamming distance between $\mathbb{x}$ and $\mathbb{x}'$. Formally,*

$$\Delta(\mathbb{x}, \mathbb{x}') := \frac{1}{n} \cdot \sum_{i \in [n]} \left| \mathbb{x}[i] - \mathbb{x}'[i] \right| \ .$$

Moreover, for any $\mathbb{x} \in \{0,1\}^n$ and a language $L \subseteq \{0,1\}^n$, denote $\Delta(\mathbb{x}, L)$ as the Hamming distance between $\mathbb{x}$ and L. Formally,

$$\Delta(\mathbb{x}, L) := \min_{\mathbb{x}' \in L \cap \{0,1\}^n} \Delta(\mathbb{x}, \mathbb{x}') \ .$$

3.2 Probabilistic proof systems

In this paper we consider a number of models of proof systems, such as IOPs, PCPs, MA proofs, their "proximity" variants, and variants where the error function may depend arbitrarily on the inputs. We choose to define them through the lens of a general object which we call "generalized IOPs", which includes explicit and implicit inputs, a witness, and arbitrary errors.

A *generalized* k-round (public-coin) IOP [BCS16; RRR16], works as follows. The verifier is given explicit input $\mathbb{y}$ and oracle access to implicit input $\mathbb{x}$. The (honest) prover is additionally given a witness $\mathbb{w}$. In every round $i \in [\mathsf{k}]$, the verifier sends a uniformly random message ρ_i to the prover; then the prover sends a proof string π_i to the verifier. After k rounds of interaction, the verifier reads explicit input $\mathbb{y}$, makes some queries to the implicit input $\mathbb{x}$ and to the proof strings $\pi_1, \ldots, \pi_{\mathsf{k}}$ sent by the prover and then decides if to accept or to reject. The following definition discusses the error parameters of a generalized IOP:

Definition 3.5 (Generalized IOPP)**.** *Let $(\mathbf{P}, \mathbf{V})$ be a tuple where $\mathbf{P}$ is an interactive algorithm, and $\mathbf{V}$ is an interactive oracle algorithm. We say that $(\mathbf{P}, \mathbf{V})$ is a public-coin* ***generalized IOP*** *for a relation $R := \{((\mathbb{x}, \mathbb{y}), \mathbb{w})\}$ with k rounds, completeness error c, and soundness error s if the following holds.*

- ***Completeness.*** *For every $((\mathbb{x}, \mathbb{y}), \mathbb{w}) \in R$,*

$$\Pr_{\rho_1, \ldots, \rho_{\mathsf{k}}} \left[\mathbf{V}^{\mathbb{x}, \pi_1, \ldots, \pi_{\mathsf{k}}}(|\mathbb{x}|, \mathbb{y}, \rho_1, \ldots, \rho_{\mathsf{k}}) = 1 \;\middle|\; \begin{array}{r} \pi_1 \leftarrow \mathbf{P}(\mathbb{x}, \mathbb{y}, \mathbb{w}) \\ \vdots \\ \pi_{\mathsf{k}} \leftarrow \mathbf{P}(\mathbb{x}, \mathbb{y}, \mathbb{w}, \rho_1, \ldots, \rho_{\mathsf{k}}) \end{array} \right] \geq 1 - \mathsf{c}(\mathbb{x}, \mathbb{y}) \ .$$

  If $\mathsf{c}(\mathbb{x}, \mathbb{y}) = 0$ for every $(\mathbb{x}, \mathbb{y}) \in L(R)$, we say that the IOPP has perfect completeness.

- ***Soundness.*** *For every $(\mathbb{x}, \mathbb{y}) \notin L(R)$ and unbounded malicious prover $\tilde{\mathbf{P}}$,*

$$\Pr_{\rho_1, \ldots, \rho_{\mathsf{k}}} \left[\mathbf{V}^{\mathbb{x}, \pi_1, \ldots, \pi_{\mathsf{k}}}(|\mathbb{x}|, \mathbb{y}, \rho_1, \ldots, \rho_{\mathsf{k}}) = 1 \;\middle|\; \begin{array}{r} \pi_1 \leftarrow \tilde{\mathbf{P}} \\ \vdots \\ \pi_{\mathsf{k}} \leftarrow \tilde{\mathbf{P}}(\rho_1, \ldots, \rho_{\mathsf{k}}) \end{array} \right] \leq \mathsf{s}(\mathbb{x}, \mathbb{y}) \ .$$

Above, $L(R) := \{(\mathbb{x}, \mathbb{y}) \mid \exists \mathbb{w}, ((\mathbb{x}, \mathbb{y}), \mathbb{w}) \in R\}$.

In the rest of this paper, we sometimes omit explicitly writing the verifier's input $|\mathbb{x}|$, but this is always assumed to be given to the verifier.

Efficiency measures. We study several efficiency measures. All of these complexity measures are implicitly functions of the instance $(\mathbb{x}, \mathbb{y})$.

- *Rounds* k: The IOP has k rounds of interaction.
- *Proof length* l: the combined number of bits in the proofs π_i.
- *Queries to (implicit) input* $\mathsf{q}_{\mathbb{x}}$: the number of bits read by the verifier from $\mathbb{x}$.
- *Queries to proof* q_π: the number of bits read by the verifier from $\pi_1, \ldots, \pi_{\mathsf{k}}$.
- *Randomness* r: the combined number of bits in the verifier messages ρ_i.
- *Verifier time* vt: $\mathbf{V}$ runs in time vt.
- *Prover time* pt: The prover runs in time pt. In some cases we will have *expected* prover running time, in which case this will be stated explicitly.

We use generalized versions of PCPs and MA proofs:

Definition 3.6 (Generalized PCP and generalized MA)**.** *A* ***generalized probabilistically checkable proof*** *(generalized PCP) is a generalized IOP with no interaction rounds (only a single prover message, after which the verifier may choose random coins). A* ***generalized MA proof*** *is a generalized PCP in which the verifier queries the entire (single) prover message.*

An IOP of proximity is a generalized IOP where the completeness error depends only on the length of the inputs, and the soundness error can be described as a function of the distance of the implicit instance $\mathbb{x}$ from an implicit input in the language. Formally:

Definition 3.7 (Proofs of proximity)**.** *An* ***IOP of proximity*** *(IOPP) with respect to distance function Δ is a generalized IOP where there exist functions c' and s' such that $\mathsf{c}(\mathbb{x}, \mathbb{y}) = \mathsf{c}'(|\mathbb{x}|, \mathbb{y})$ and $\mathsf{s}(\mathbb{x}, \mathbb{y}) = \mathsf{s}'(|\mathbb{x}|, \mathbb{y}, \delta)$ where $\delta := \Delta(\mathbb{x}, L_{\mathbb{y}}(R) \cap \{0,1\}^{|\mathbb{x}|})$ for $L_{\mathbb{y}}(R) := \{\mathbb{x}' \in \{0,1\}^* \mid \exists \mathbb{w}, ((\mathbb{x}', \mathbb{y}), \mathbb{w}) \in R\}$.*

PCPs of proximity (PCPPs) and MA proofs of proximity (MAPs) are similarly defined as variants of generalized PCPs and generalized MA proofs respectively.

Whenever the distance function Δ is not explicitly specified, we implicitly refer to Hamming distance. It is common for proofs of proximity to be defined for relations with no explicit input $\mathbb{y}$ or witness $\mathbb{w}$. Indeed, this is the case for the Hamming relation Ham_α which is the focus of this work. In this case we will omit $\mathbb{y}$ and $\mathbb{w}$ from all notation.

Finally, we define standard IOPs, PCPs and MA proofs:

Definition 3.8 (Standard IOP/PCP/MA proofs)**.** *A (standard) IOP (respectively standard PCP or standard MA proof) is an IOPP (respectively PCPP or MAP proof) for a relation $R := \{((\perp, \mathbb{y}), \mathbb{w})\}$ (i.e., there is no $\mathbb{x}$ in the relation).*

For the standard variants of probabilistic proof systems, we will omit $\mathbb{x}$ from notation, as it is always set to $\perp$. Moreover, we will sometimes denote the input by x and witness by w (instead of $\mathbb{y}$ and $\mathbb{w}$) as is standard for IOPs.

Remark 3.9 (Computability of error functions)**.** In this work we assume unless stated otherwise that c and s are computable in polynomial time given the implicit input, $\mathbb{x}$, explicit input $\mathbb{y}$, and the proximity δ (as defined in Definition 3.7).

3.3 Probabilistic inequalities

We use the multiplicative Chernoff bound.

Theorem 3.10 (Multiplicative Chernoff Bound)**.** *Let $X = \sum_{i\in[n]} X_i$, where $X_1, \ldots, X_n$ are independent random variables in $\{0,1\}$, with $\mathbb{E}[X] = \mu$. Then for any $\epsilon \geq 0$,*

$$\Pr[X \leq (1-\epsilon)\mu] \leq e^{-\left(\epsilon^2\mu/2\right)},$$
$$\Pr[X \geq (1+\epsilon)\mu] \leq e^{-\left(\epsilon^2\mu/3\right)} \ .$$

4 Finding good shifts

In this section, we define the concept of "good" shifts and prove that such shifts can be found efficiently in expected probabilistic time. These good shifts will be helpful for us throughout the paper, generally for showing perfect completeness of protocols and for bounding the (expected) running time of the honest prover.

We define good shifts for a set of vectors:

Definition 4.1 (Good shifts)**.** *For every $n, k, t \in \mathbb{N}$, $\epsilon \in (0,1]$, and for every list of bit vectors $\mathbb{x}_1, \ldots, \mathbb{x}_k \in \{0,1\}^n$ we define the set $\mathsf{Good}_{t,\epsilon}(\mathbb{x}_1, \ldots, \mathbb{x}_k)$ to be the set of shifts $(z_1, \ldots, z_t) \in \{0,1\}^{t \cdot n}$ such that*

$$\forall\, i \in [k], \rho \in [n] \quad \sum_{j \in [t]} \mathbb{x}_i[\rho + z_j] \geq \epsilon \cdot t \ .$$

We give a probabilistic algorithm that, given a list of vectors, outputs a set of good shifts in small (expected) time:

Construction 4.2 (Shift finding algorithm)**.** The algorithm $\mathbf{A}$ is given as input α, η, and $(\mathbb{x}_1, \ldots, \mathbb{x}_k)$. It proceeds as follows:

- Repeat the following until shifts are output:
  1. Sample $z_1, \ldots, z_t \leftarrow [n]$ uniformly at random.
  2. For every $i \in [k]$:
     (a) Set $\mathsf{counter}_i := 0$ and let bucket_i be the all zeros array of size n.
     (b) For every $\rho \in [n]$: if $\mathsf{counter}_i < \alpha \cdot n$ and $x_i[\rho] = 1$ then update: $\mathsf{counter}_i := \mathsf{counter}_i + 1$ and, for every $j \in [t]$, update $\mathsf{bucket}_i[\rho - z_j] := \mathsf{bucket}_i[\rho - z_j] + 1$.
     (c) Check that $\mathsf{bucket}_i[\rho] \geq (\alpha - \eta) \cdot t$ for every $\rho \in [n]$.
  3. Output $z_1, \ldots, z_t$ if the previous checks passed for every $i \in [k]$.

The following lemma shows that $\mathbf{A}$ finds good shifts and gives a bound on its running:

Lemma 4.3. *Fix parameters $n \in \mathbb{N}$, $\alpha \in (0,1]$ and $\eta \in (0, \alpha)$, and let $t := 2 \cdot \log(k \cdot n)/\eta^2$. For every set of bit vectors $\mathbb{x}_1, \ldots, \mathbb{x}_k \in \mathsf{Ham}_\alpha \cap \{0,1\}^n$, there exist $(z_1, \ldots, z_t) \in \mathsf{Good}_{t,\alpha-\eta}(\mathbb{x}_1, \ldots, \mathbb{x}_k)$ and the algorithm $\mathbf{A}(\alpha, \eta, \mathbb{x}_1, \ldots, \mathbb{x}_k)$, described in Construction 4.2, outputs a set of such shifts in expected time $O\left(\frac{\alpha}{\alpha-\eta} \cdot n \cdot k \cdot \log(n \cdot k)\right)$.*

Proof. Fix bit vectors $\mathbb{x}_1, \ldots, \mathbb{x}_k \in \mathsf{Ham}_\alpha$, and for each i let $\mathbb{x}'_i$ be $\mathbb{x}_i$ where all but the first $\alpha \cdot n$ ones are flipped to 0 (note that $\mathsf{weight}(\mathbb{x}'_i) = \alpha \cdot n \leq \mathsf{weight}(\mathbb{x}_i)$). In Claim 4.4 we show that $\mathbf{A}$ outputs shifts if and only if they are good for $(\mathbb{x}'_1, \ldots, \mathbb{x}'_k)$. Then, in Claim 4.5, we show that the probability that a randomly sampled set of shifts is good for $(\mathbb{x}'_1, \ldots, \mathbb{x}'_k)$ is

at least 0.2. Observe that $\mathsf{Good}_{t,\alpha-\eta}(\mathbb{x}'_1,\ldots,\mathbb{x}'_t) \subseteq \mathsf{Good}_{t,\alpha-\eta}(\mathbb{x}_1,\ldots,\mathbb{x}_t)$ since for every i and ρ:

$$\sum_{j\in[t]} \mathbb{x}[\rho+z_j] \geq \sum_{j\in[t]} \mathbb{x}'[\rho+z_j] \ .$$

Therefore, the expected number of times that $z_1,\ldots,z_t$ are sampled until $\mathbf{A}$ outputs $(z_1,\ldots,z_t) \in \mathsf{Good}_{t,\alpha-\eta}(\mathbb{x}_1,\ldots,\mathbb{x}_t)$ is 5.

Each sample takes time $O(t)$. The algorithm then iterates over all of the vectors and their values and, for each vector $\mathbb{x}_i$, updates bucket_i for every j for at most $\alpha\cdot n$ times due to the counter $\mathsf{counter}_i$. Each update takes time $O(t)$. Thus, the computation for each vectors takes time $O(n+\alpha\cdot n\cdot t)$.

Overall, the expected running time of $\mathbf{A}$ is

$$\begin{aligned} O(t+k\cdot(n+\alpha\cdot n\cdot t)) &= O\left(k\cdot(n+\alpha\cdot n\cdot\frac{\log(n\cdot k)}{\alpha-\eta})\right) \\ &= O\left(\frac{\alpha}{\alpha-\eta}\cdot n\cdot k\cdot\log(n\cdot k)\right) \ , \end{aligned}$$

where the second equality holds since $\alpha/(\alpha-\eta)\geq O(1)$.

We now prove our first claim, showing that $\mathbf{A}$ outputs $z_1,\ldots,z_t$ if and only if they are good.

Claim 4.4. $\mathbf{A}$ *outputs the sampled shifts* $z_1,\ldots,z_t$ *if and only if* $(z_1,\ldots,z_t)\in\mathsf{Good}_{t,\alpha-\eta}(\mathbb{x}'_1,\ldots,\mathbb{x}'_k)$.

Proof. Consider a set of shifts $z_1,\ldots,z_t$. For a set index $i\in[k]$, $\mathsf{counter}_i$ begins at 0 and, for every ρ with $\mathbb{x}_i[\rho]=1$, $\mathsf{counter}_i$ is increased by 1. Once $\mathsf{counter}_i=(\alpha-\eta)\cdot n$ (i.e., once the first $(\alpha-\eta)\cdot n$ ones of $\mathbb{x}_i$ are seen), $\mathsf{counter}_i$ and bucket_i do not change. Therefore, the algorithm acts identically for $\mathbb{x}_1,\ldots,\mathbb{x}_k$ and $\mathbb{x}'_1,\ldots,\mathbb{x}'_k$.

Now note that for every ρ with $\mathbb{x}'_i[\rho]=1$, we add 1 to $\mathsf{bucket}_i[\rho-z_j]$ for every $j\in[t]$. Thus

$$\mathsf{bucket}_i[\rho] = |\{j\in[t]:\mathbb{x}'_i[\rho+z_j]=1\}| = \sum_j \mathbb{x}'_i[\rho+z_j] \ .$$

Thus, by the end of the iteration, $\mathsf{bucket}_i[\rho]=\sum_j \mathbb{x}'_i[\rho+z_j]$. The algorithm outputs $z_1,\ldots,z_t$ if and only if for every $i\in[k]$, and every $\rho\in[n]$:

$$\sum_j \mathbb{x}'_i[\rho+z_j] \geq \mathsf{bucket}_i[\rho] \geq (\alpha-\eta)\cdot k \ ,$$

which precisely means that $z_1,\ldots,z_t$ is output if and only if $(z_1,\ldots,z_t)\in\mathsf{Good}_{t,\alpha-\eta}(\mathbb{x}'_1,\ldots,\mathbb{x}'_k)$. □

We now show that by uniformly sampling shifts, one hits a good set with constant probability:

Claim 4.5. $\Pr_{z_1,\ldots,z_t}\left[\ (z_1,\ldots,z_t) \in \mathsf{Good}_{t,\alpha-\eta}(\mathbb{x}'_1,\ldots,\mathbb{x}'_k)\ \right] > 0.2$.

Proof. Recall that

$$\Pr_{z_1,\ldots,z_t}\left[\ (z_1,\ldots,z_t) \in \mathsf{Good}_{t,\alpha-\eta}(\mathbb{x}'_1,\ldots,\mathbb{x}'_k)\ \right] = \Pr_{z_1,\ldots,z_t}\left[\ \forall\, i \in [k], \rho \in [n] \ \sum_{j\in[t]} \mathbb{x}'_i[\rho + z_j] \geq (\alpha-\eta)\cdot t\ \right] \ .$$

Notice that for every i and ρ:

$$\Pr_{z_j \leftarrow [n]}[\mathbb{x}'_i[\rho + z_j] = 1] = \Pr_{z_j \leftarrow [n]}[\mathbb{x}'_i[z_j] = 1] \ .$$

Thus, by applying the union bound, we have that:

$$\begin{aligned}\Pr_{z_1,\ldots,z_t}\left[\ \exists\, \rho \in [n].\ \sum_{j\in[t]} \mathbb{x}'_i[\rho + z_j] < (\alpha-\eta)\cdot t\ \right] &\leq \sum_{\rho\in[n]}\left(\Pr_{z_1,\ldots,z_t}\left[\ \sum_{j\in[t]} \mathbb{x}'_i[\rho + z_j] < (\alpha-\eta)\cdot t\ \right]\right) \\ &= n \cdot \Pr_{z_1,\ldots,z_t}\left[\ \sum_{j\in[t]} \mathbb{x}'_i[z_j] < (\alpha-\eta)\cdot t\ \right] \ .\end{aligned}$$

Notice that for every j:

$$\mathbb{E}_{z_j}[\mathbb{x}'_i[z_j]] = \Pr_{z_j}[\mathbb{x}'_i[z_j] = 1] = (\alpha-\eta)\cdot t \ .$$

Thus, by applying the Chernoff bound with $\epsilon := \alpha - \eta$, we have that

$$\Pr_{z_1,\ldots,z_t}\left[\ \sum_{j\in[t]} \mathbb{x}'_i[z_j] < (\alpha-\eta)\cdot t\ \right] \leq e^{-\frac{\eta^2}{2\alpha}\cdot t} \ ,$$

Therefore,

$$\Pr_{z_1,\ldots,z_t}\left[\ \exists\, \rho \in [n] \ \sum_{j\in[t]} \mathbb{x}'_i[\rho + z_j] < (\alpha-\eta)\cdot t\ \right] \leq n \cdot e^{-\frac{\eta^2}{2\alpha}\cdot t} \ .$$

By applying the union bound, we have that:

$$\begin{aligned}\Pr_{z_1,\ldots,z_t}\left[\ \exists\, i \in [k], \rho \in [n] \ \sum_{j\in[t]} \mathbb{x}'_i[\rho + z_j] < (\alpha-\eta)\cdot t\ \right] &\leq k \cdot n \cdot e^{-\frac{\eta^2}{2\alpha}\cdot t} \\ &= 2^{\log(k\cdot n)} \cdot e^{-\frac{\log(k\cdot n)}{\alpha}} \\ &\leq (2/e)^{\log(k\cdot n)} \\ &< 0.8 \ ,\end{aligned}$$

where the equality follows from the definition of $t := 2 \cdot \log(k \cdot n)/\eta^2$. Therefore,

$$\Pr_{z_1,\ldots,z_t}\left[\forall\, i \in [k], \rho \in [n] \;\; \sum_{j\in[t]} \mathtt{x}'_i[\rho + z_j] \geq (\alpha - \eta) \cdot t\right] > 0.2 \;\; .$$

□

□

5 Non-interactive proofs for Hamming weight with sublinear communication

In this section, we develop an MA proof and a PCPP with sublinear communication complexity for the Hamming weight problem. We begin by describing the MAP:

Theorem 5.1. *For every $\alpha, \eta : \mathbb{N} \to (0,1]$ such that $0 < \eta < \alpha$ (that are computable in linear time), there exists a perfectly complete MAP for Ham_α with the following parameters,*

MAP for Ham_α	
Soundness error	$\mathsf{s}(\delta) = \frac{\alpha-\delta}{\alpha-\eta}$
Communication length	$2 \cdot \log^2 n/\eta^2$
Queries to input	$2 \cdot \log n/\eta^2$
Randomness	$\log n$
Verifier running time	$O(\log n/\eta^2)$
Prover expected running time	$O\left(\alpha/(\alpha-\eta) \cdot n \cdot \log n\right)$

where $n \in \mathbb{N}$ is the input size, $\alpha := \alpha(n)$, and $\eta := \eta(n)$.

The PCPP is as follows:

Theorem 5.2. *For every $\alpha, \eta : \mathbb{N} \to (0,1]$ such that $0 < \eta < \frac{2}{3} \cdot \alpha$ (that are computable in linear time), there exists a perfectly complete PCPP for Ham_α with the following parameters,*

PCPP for Ham_α	
Soundness error	$\mathsf{s}(\delta) = \frac{\alpha-\delta}{\alpha-1.5\eta}$
Proof length	$O\left(\frac{n}{\log^2 n} \cdot (-\log^2 \eta)/\eta^2\right)$
Queries to input	$O\left((\operatorname{loglog} n - \log \eta)/\eta^2\right)$
Queries to proof	$O\left(\log n \cdot (\operatorname{loglog} n - \log^2 \eta)/\eta^2\right)$
Randomness	$\log n + \operatorname{loglog} n - 2\log \eta + 1$
Verifier running time	$O\left(\log n \cdot (\operatorname{loglog} n - \log^2 \eta)/\eta^2\right)$
Prover expected running time	$O\left(\frac{\alpha}{\alpha-1.5\eta} \cdot n \cdot \log n \cdot (\operatorname{loglog} n - \log \eta)/\eta^2\right)$

where $n \in \mathbb{N}$ is the input size, $\alpha := \alpha(n)$, and $\eta := \eta(n)$.

This section is organized as follows:

- In Section 5.1 we construct a generalized MA for list-Hamming. Theorem 5.1 follows as a corollary from this construction.
- In Section 5.2 we introduce a transformation from generalized PCP for list-Hamming with specific error structure (which the generalized MA constructed in the previous section has) to a PCPP for Hamming.
- In Section 5.3 we construct a PCPP for Hamming by plugging in the result from Section 5.1, which provides a generalized MA for list-Hamming, into the transformation described in Section 5.2. Note that MAs can be used in this transformation since MAs are a specific case of PCPs. This step directly implies Theorem 5.2.

5.1 MA proof of proximity

We construct a generalized MA proof for k-Ham_α with sublinear communication complexity. The resultant generalized MA proof will directly imply an MAP for Ham_α.

Theorem 5.3. *For every $k \in \mathbb{N}$, $\alpha, \eta : \mathbb{N} \to (0,1]$ such that $0 < \eta < \alpha$ (that are computable in linear time), Construction 5.5 yields a perfectly complete generalized MA proof for k-Ham_α with the following parameters:*

Generalized MA for k-Ham_α	
Soundness error	$\frac{1}{k\cdot(\alpha-\eta)} \cdot \sum_{i=1}^{k} \mathsf{weight}(\mathbb{x}_i)$
Proof length	$2 \cdot \log n \cdot \log(k \cdot n)/\eta^2$
Queries to input	$2 \cdot \log(k \cdot n)/\eta^2$
Randomness	$\log(k \cdot n)$
Verifier running time	$O(\log(k \cdot n)/\eta^2)$
Prover expected running time	$O\left(\alpha/(\alpha-\eta) \cdot n \cdot k \cdot \log(n \cdot k)\right)$

where $(\mathbb{x}_1, \ldots, \mathbb{x}_k) \in (\{0,1\}^n)^k$ is the input, $\alpha := \alpha(n)$, and $\eta := \eta(n)$.

Note that for $k = 1$, we have that 1-$\mathsf{Ham}_\alpha \equiv \mathsf{Ham}_\alpha$. Moreover, the soundness error of the protocol is a function of the distance from the language: $\frac{1}{\alpha-\eta} \cdot \mathsf{weight}(\mathbb{x}) = \frac{1}{\alpha-\eta} \cdot (\alpha - \Delta(\mathbb{x}, \mathsf{Ham}_\alpha))$. Therefore, Theorem 5.3, when fixing $k = 1$, directly implies the following theorem.

Theorem 5.4. *For every $\alpha, \eta : \mathbb{N} \to (0,1]$ such that $0 < \eta < \alpha$ (that are computable in linear time), Construction 5.5 yields a perfectly complete oracle MAP for k-Ham_α with the following parameters:*

MAP for Ham_α	
Soundness error	$\mathsf{s}(\delta) = \frac{\alpha-\delta}{\alpha-\eta}$
Proof length	$2 \cdot \log^2 n/\eta^2$
Queries to input	$2 \cdot \log n/\eta^2$
Randomness	$\log n$
Verifier running time	$O(\log n/\eta^2)$
Prover expected running time	$O\left(\alpha/(\alpha-\eta) \cdot n \cdot \log n\right)$

where $n \in \mathbb{N}$ is the input size, $\alpha := \alpha(n)$, and $\eta := \eta(n)$.

Theorem 5.3 follows from the construction below:

Construction 5.5. Let $t := 2 \cdot \log(k \cdot n)/\eta^2$. The prover $\mathbf{P}$ receives as input bit vector $\mathbb{x}_1, \ldots, \mathbb{x}_k \in \{0,1\}^n$, while the verifier $\mathbf{V}$ has oracle access to the vector $\mathbb{x}_1, \ldots, \mathbb{x}_k$. They interact as follows.

- $\mathbf{P}(\mathbb{x}_1, \ldots, \mathbb{x}_k)$: The prover sends $z_1, \ldots, z_t \in [n]$.

- $\mathbf{V}^{\mathbb{x}_1,\ldots,\mathbb{x}_k}(n', z_1, \ldots, z_t)$:
  1. Set $n := n'/k$.
  2. Choose $i \leftarrow [k]$, $\rho \leftarrow [n]$ uniformly.
  3. Query $\mathbb{x}_i[\rho + z_1], \ldots, \mathbb{x}_i[\rho + z_t]$.
  4. Accept if $\mathsf{weight}(\mathbb{x}_i[\rho + z_1], \ldots, \mathbb{x}_i[\rho + z_t]) \geq (\alpha - \eta)$ and reject otherwise.

Proof of Theorem 5.3. We analyze completeness and soundness and then describe the complexity measures of the PCPP.

Completeness. Fix bit vectors $(\mathbb{x}_1, \ldots, \mathbb{x}_k) \in k\text{-}\mathsf{Ham}_\alpha$, i.e., for every $i \in [k]$, $\mathsf{weight}(\mathbb{x}_i) \geq \alpha$. By Lemma 4.3, there exists a series of shifts $z_1, \ldots, z_t$ such that for every i, ρ there are at least $(\alpha - \eta) \cdot t$ indices j where $\mathbb{x}_i[\rho + z_j] = 1$.

Given these shifts, by definition, for every choice of i and ρ, the bit vector $(\mathbb{x}_i[\rho + z_1], \ldots, \mathbb{x}_i[\rho + z_t])$ contains at least $(\alpha - \eta) \cdot t$ ones. Consequently, if it chooses i and ρ, the verifier $\mathbf{V}$ will accept during Item 4. Thus, the honest prover strategy of sending the series of shifts $z_1, \ldots, z_t$ promised by Lemma 4.3 causes the verifier to accept with probability 1.

Soundness. Fix bit vectors $(\mathbb{x}_1, \ldots, \mathbb{x}_k) \notin k\text{-}\mathsf{Ham}_\alpha$, and a prover message $z_1, \ldots, z_t$, and let $\beta := \frac{1}{k} \cdot \sum_{i=1}^{k} \mathsf{weight}(\mathbb{x}_i)$. For every $j \in [t]$, let X_j be the 0/1 random variable such that

$$\Pr[X_j = 1] = \Pr_{i \leftarrow [k], \rho \leftarrow [n]}[\mathbb{x}_i[\rho + z_j] = 1] = \Pr_{i \leftarrow [k], \rho \leftarrow [n]}[\mathbb{x}_i[\rho] = 1] = \frac{1}{k} \cdot \sum_{i=1}^{k} \mathsf{weight}(\mathbb{x}_i) = \beta \ ,$$

and observe that $\mathbb{E}_{i,\rho}[X_j] = \beta$. The verifier accepts when $\sum_{j \in [t]} X_j \geq (\alpha - \eta) \cdot t$. Thus, using Markov's inequality and the linearity of expectation, we conclude:

$$\begin{aligned} \Pr_{i,\rho}[\ \mathbf{V}\ \textit{accepts}\ ] &= \Pr_{i,\rho}\left[\sum_{j \in [t]} X_j \geq (\alpha - \eta) \cdot t\right] \\ &\leq \frac{\mathbb{E}_{i,\rho}[\sum_j X_j]}{(\alpha - \eta) \cdot t} \leq \frac{\sum_j \mathbb{E}_{i,\rho}[X_j]}{(\alpha - \eta) \cdot t} \leq \frac{\beta}{\alpha - \eta} = \frac{1}{k \cdot (\alpha - \eta)} \cdot \sum_{i=1}^{k} \mathsf{weight}(\mathbb{x}_i) \ . \end{aligned}$$

Complexity measures. We analyze the complexity parameters of the MAP.

- *Communication size:* The prover sends $t \cdot \log n = 2 \cdot \log n \cdot \log(k \cdot n)/\eta^2$ bits.
- *Queries to input:* The verifier makes $t = 2 \cdot \log(k \cdot n)/\eta^2$ queries to $\mathbb{x}$.
- *Randomness:* The verifier uses $\log(k \cdot n)$ bits of randomness.
- *Verifier running time:* The verifier runs in time $O(\log k + t) = O(\log(k \cdot n)/\eta^2)$.
- *Prover expected running time:* By Lemma 4.3, the expected running time of the prover is
$$O(\alpha/(\alpha - \eta) \cdot n \cdot k \cdot \log(n \cdot k)) \ .$$

□

5.2 PCP for list-Hamming to PCPP for Hamming

We construct a PCPP for Hamming from a (generalized) PCP for list-Hamming.

Theorem 5.6. *Suppose that for every $\alpha', \eta' : \mathbb{N} \to (0,1]$ such that $0 < \eta' < \alpha'$ (that are computable in linear time), there is perfectly complete generalized PCP $(\mathbf{P}_{\mathsf{PCP}}, \mathbf{V}_{\mathsf{PCP}})$ for k'-$\mathsf{Ham}_{\alpha'}$ such that for input $(\mathbb{x}_1, \ldots, \mathbb{x}_{k'}) \in (\{0,1\}^{n'})^{k'}$ has soundness error of the form $\epsilon(\alpha', \eta') \cdot \frac{1}{k'} \cdot \sum_{i=1}^{k'} \mathsf{weight}(\mathbb{x}_i)$. Then for every $a \in \mathbb{N}$, $\alpha, \eta : \mathbb{N} \to (0,1]$ such that $0 < \eta' < \alpha'$ (that are computable in linear time), Construction 5.7 yields a PCPP $(\mathbf{P}_{\mathsf{PCPP}}, \mathbf{V}'_{\mathsf{PCPP}})$ for Ham_α with the following parameters:*

Generalized PCP $(\mathbf{P}, \mathbf{V})$ for k'-$\mathsf{Ham}_{\alpha'}$	
Soundness error	$\epsilon(\alpha, \eta) \cdot \frac{1}{k'} \cdot \sum_{i=1}^{k'} \mathsf{weight}(\mathbb{x}_i)$
Proof length	$\mathsf{l}_{\mathbf{P}} := \mathsf{l}_{\mathbf{P}}(n', k', \eta')$
Queries to input	$\mathsf{q}_{\mathbb{x}} := \mathsf{q}_{\mathbb{x}}(n', k', \eta')$
Queries to proof	$\mathsf{q}_\pi := \mathsf{q}_\pi(n', k', \eta')$
Randomness	$\mathsf{r} := \mathsf{r}(n', k', \eta')$
Verifier running time	$\mathsf{vt} := \mathsf{vt}(n', k', \eta')$
Prover expected running time	$\mathsf{pt} := \mathsf{pt}(n', k', \alpha', \eta')$

$\longrightarrow$

PCPP $(\mathbf{P}', \mathbf{V}')$ for Ham_α	
Soundness error	$\mathsf{s}(\delta, \alpha, \eta) = \epsilon(\alpha', \eta') \cdot (\alpha - \delta)$
Proof length	$n' \cdot \log n + \frac{n}{a} \cdot \mathsf{l}_{\mathbf{P}}$
Queries to input	$\mathsf{q}_{\mathbb{x}}$
Queries to proof	$\log n \cdot \mathsf{q}_{\mathbb{x}} + \mathsf{q}_\pi$
Randomness	$\log(n/a) + \mathsf{r}$
Verifier running time	$\mathsf{vt} + O(\mathsf{q}_{\mathbb{x}})$
Prover expected running time	$O\left(\left(\frac{1}{\eta^2} + \frac{\alpha}{\alpha - \frac{\eta}{2}}\right) \cdot n \cdot \log n + \frac{n}{a} \cdot \mathsf{pt}\right)$

where $n' = 2 \cdot \log n / \eta^2(n)$, $k' = a$, $\eta' = \eta(n)/2$, $\alpha' = \alpha(n) - \eta(n)/2$.

Construction 5.7. Let $t := 8 \cdot \log n / \eta^2$, and for every $s \in [n/a]$ let $I_s := ((s-1) \cdot a + 1, \ldots, s \cdot a)$ be a list of a indices. Let $(\mathbf{P}_{\mathsf{PCP}}, \mathbf{V}_{\mathsf{PCP}})$ be a perfectly complete PCP for k'-$\mathsf{Ham}_{\alpha'}$ with vectors of size $n' := t$, and $\eta' := \eta/2$. The prover $\mathbf{P}_{\mathsf{PCPP}}$ receives as input the bit vector $\mathbb{x}$, while the verifier $\mathbf{V}_{\mathsf{PCPP}}$ has oracle access to the bit vector $\mathbb{x}$. They interact as follows.

- $\mathbf{P}_{\mathsf{PCPP}}(\mathbb{x})$:
  1. Set bit vectors $z_1, \ldots, z_t \in [n]$.
  2. For every $\rho \in [n]$, set $\mathbb{x}_\rho := (\mathbb{x}[\rho + z_1], \ldots, \mathbb{x}[\rho + z_t])$.
  3. For every $s \in [n/a]$, set $X_s := (\mathbb{x}_\rho)_{\rho \in I_s}$, and compute $\pi_s := \mathbf{P}_{\mathsf{PCP}}(X_s)$.
  4. Output $\big((z_1, \ldots, z_t), (\pi_1, \ldots, \pi_{n/a})\big)$.
- $\mathbf{V}^{\mathbb{x}, \pi}_{\mathsf{PCPP}}(n)$:

0. *Notation:*
   (a) For every $\rho \in [n]$, let $\mathbb{x}_\rho := (\mathbb{x}[\rho + z_1], \ldots, \mathbb{x}[\rho + z_t])$.
   (b) For every $s \in [n/a]$, let $X_s := (\mathbb{x}_\rho)_{\rho \in I_s}$.
   (Note that the verifier does not compute the above.)
1. Parse $\pi := ((z_1, \ldots, z_t), (\tilde{\pi}_1, \ldots, \tilde{\pi}_{n/a}))$.
2. Choose $s \leftarrow [n/a]$ uniformly.
3. Emulate $\mathbf{V}_{\mathsf{PCP}}^{X_s, \tilde{\pi}_s}(a \cdot t)$, where for every input query $\mathbb{x}_j[i]$, query z_i, $\mathbb{x}[I_s[j] + z_i]$, and answer accordingly.
4. Accept if and only if $\mathbf{V}^{X_s, \tilde{\pi}_s}(a \cdot t)$ accepts.

Completeness. Fix bit vector $\mathbb{x} \in \mathsf{Ham}_\alpha$. By Lemma 4.3 with $\frac{\eta}{2}$, there exists a series of shifts $z_1, \ldots, z_t$ such that for every ρ there are at least $(\alpha - \frac{\eta}{2}) \cdot t$ indices j where $\mathbb{x}[\rho + z_j] = 1$. The honest prover $\mathbf{P}'$ uses these shifts. Therefore, for every ρ, the bit vector $\mathbb{x}_\rho = (\mathbb{x}[\rho + z_1], \ldots, \mathbb{x}[\rho + z_t])$ has Hamming weight $\mathsf{weight}(\mathbb{x}_\rho) \geq \alpha - \frac{\eta}{2}$. By the completeness of the generalized PCP protocol for a-$\mathsf{Ham}_{\alpha'}$ where $\alpha' = \alpha - \frac{\eta}{2}$, all the proofs π_s will be accepted by the verifier $\mathbf{V}'$ with probability 1.

Soundness. Fix bit vector $\mathbb{x} \notin \mathsf{Ham}_\alpha$. For every $j \in [t]$:

$$\Pr_{\rho \leftarrow [n]}[\mathbb{x}_\rho[j] = 1] = \Pr_{\rho \leftarrow [n]}[\mathbb{x}[\rho + z_j] = 1] = \Pr_{\rho \leftarrow [n]}[\mathbb{x}[\rho] = 1] = \mathsf{weight}(\mathbb{x}) \ .$$

Observe that $\mathbb{E}_{\rho \leftarrow [n]}[\mathsf{weight}(\mathbb{x}_\rho)] = \mathsf{weight}(\mathbb{x})$. The following claim shows that the probability of sampling a list of vectors X_s with high average weight is small.
Let $W = a \cdot t$ be the maximal number of ones in X_S, and let $\epsilon := \epsilon(\alpha', \eta')$. By the law of total probability we have that:

$$\begin{aligned}
&\Pr[\langle \tilde{\mathbf{P}}(\mathbb{x}), \mathbf{V}^{\mathbb{x}} \rangle = 1] \\
&\quad = \sum_{w=0}^{W} \Pr_s \left[\frac{1}{a} \cdot \sum_{i=1}^{a} \mathsf{weight}(X_s[i]) = \frac{w}{W} \right] \cdot \Pr \left[\langle \tilde{\mathbf{P}}(\mathbb{x}), \mathbf{V}^{\mathbb{x}} \rangle = 1 \,\middle|\, \tfrac{1}{a} \cdot \textstyle\sum_{i=1}^{a} \mathsf{weight}(X_s[i]) = \frac{w}{W} \right] \\
&\quad = \sum_{w=0}^{W} \Pr_s \left[\frac{1}{a} \cdot \sum_{i=1}^{a} \mathsf{weight}(X_s[i]) = \frac{w}{W} \right] \cdot \epsilon \cdot \frac{w}{W} \\
&\quad = \epsilon \cdot \mathbb{E}_s \left[\frac{1}{a} \cdot \sum_{i=1}^{a} \mathsf{weight}(X_s[i]) \right] \\
&\quad = \epsilon \cdot \frac{1}{a} \cdot \sum_{i=1}^{a} \mathbb{E}_s \left[\, \mathsf{weight}(X_s[i]) \, \right] \\
&\quad = \epsilon \cdot \mathbb{E}_{i \leftarrow [a], s \leftarrow [n/a]} \left[\, \mathsf{weight}(X_s[i]) \, \right] \\
&\quad = \epsilon \cdot \mathbb{E}_{\rho \leftarrow [n]} \left[\, \mathsf{weight}(\mathbb{x}_\rho) \, \right] \\
&\quad = \epsilon \cdot \mathsf{weight}(\mathbb{x}) = \epsilon \cdot (\alpha - \Delta(\mathsf{Ham}_\alpha, \mathbb{x})) \ ,
\end{aligned}$$

where the second equality is by the soundness of the underlying PCP protocol for a-$\mathsf{Ham}_{\alpha'}$.

Complexity measures. We analyze the complexity parameters of the new PCPP.

- *Proof length:* The proof length is
$$t \cdot \log n + \frac{n}{a} \cdot \mathsf{l}_{\mathbf{P}} = n' \cdot \log n + \frac{n}{a} \cdot \mathsf{l}_{\mathbf{P}} \ .$$
- *Queries to input:* The verifier makes $\mathsf{q}_{\mathbb{x}}$ queries to $\mathbb{x}$.
- *Queries to proof:* The verifier makes $\log n \cdot \mathsf{q}_{\mathbb{x}} + \mathsf{q}_{\pi}$ queries to the proof.
- *Randomness:* The verifier uses $\log(\frac{n}{a}) + \mathsf{r}$ bits of randomness.
- *Verifier running time:* The verifier runs in time $\mathsf{vt} + O(\mathsf{q}_{\mathbb{x}})$.
- *Prover expected running time:* By Lemma 4.3 with the parameter $\frac{\eta}{2}$, the expected time takes to generate the shifts is $O\left(\alpha/(\alpha - \frac{\eta}{2}) \cdot n \cdot \log n\right)$. Therefore, the overall expected running time of the prover is
$$\begin{aligned} &O\left(\frac{\alpha}{\alpha - \frac{\eta}{2}} \cdot n \cdot \log n + k \cdot n \cdot t + \frac{n}{a} \cdot \mathsf{pt}\right) \\ &\leq O\left(\frac{\alpha}{\alpha - \frac{\eta}{2}} \cdot n \cdot \log n + \frac{1}{\eta^2} \cdot n \cdot \log n + \frac{n}{a} \cdot \mathsf{pt}\right) \\ &= O\left(\left(\frac{\alpha}{\alpha - \frac{\eta}{2}} + \frac{1}{\eta^2}\right) \cdot n \cdot \log n + \frac{n}{a} \cdot \mathsf{pt}\right) \ . \end{aligned}$$

5.3 PCP of proximity

The following theorem follows by plugging in Theorem 5.3 into Theorem 5.6.

Theorem 5.8. *For every $\alpha, \eta : \mathbb{N} \to (0, 1]$ such that $\eta \in (0, \alpha)$ (that are computable in linear time), there exists a perfectly complete PCPP $(\mathbf{P}, \mathbf{V})$ for Ham_{α} with the following parameters:*

PCPP $(\mathbf{P}, \mathbf{V})$	
Completeness error	0
Soundness error	$\mathsf{s}(\delta, \alpha, \eta) = \frac{\alpha - \delta}{\alpha - \eta}$
Proof length	$O\left(\frac{n}{\eta^2 \cdot \log^2 n} \cdot (-\log^2 \eta)\right)$
Queries to input	$O\left((\mathrm{loglog}\, n - \log \eta)/\eta^2\right)$
Queries to proof	$O\left(\log n \cdot (\mathrm{loglog}\, n - \log^2 \eta)/\eta^2\right)$
Randomness	$\log n + \mathrm{loglog}\, n - 2 \log \eta + 1$
Verifier running time	$O\left(\log n \cdot (\mathrm{loglog}\, n - \log^2 \eta)/\eta^2\right)$
Prover expected running time	$O\left(\frac{\alpha}{\alpha - \eta} \cdot n \cdot \log n \cdot (\mathrm{loglog}\, n - \log \eta)/\eta^2\right)$

where $n \in \mathbb{N}$ is the input size, $\alpha := \alpha(n)$, and $\eta := \eta(n)$.

The proof of this theorem appears in Appendix A.

6 SIQ-PCPP for Hamming weight

In this section, we explore SIQ-PCPP: PCPPs where the verifier makes a single query to its input.

In Section 6.1 we show that any SIQ-PCPP must have large proof length:

Corollary 6.1. *For $\alpha \in (0.5, 0.77)$, any perfectly complete MA proof of proximity for Ham_α that, for inputs of length n and distance $\delta = 1 - \alpha$ has soundness error smaller than 1, message length l, and input query complexity 1 has:*

$$\mathsf{l} = \Omega(n) \ .$$

Proof. We plug in constants $\alpha = 2/3$ and $\delta = 1 - \alpha$ to Theorem 6.3 and use the approximation $\log \binom{n}{\gamma \cdot n} = n \cdot H(\gamma) - \frac{1}{2} \cdot \log(2\pi \cdot n \cdot \gamma \cdot (1-\gamma)) + O(\frac{1}{n})$ where H is the binary entropy function. This gives us

$$\begin{aligned}
\mathsf{l} &> \log \binom{n}{(1-\alpha) \cdot n} - \log \binom{2 \cdot (1-\alpha) \cdot n}{(1-\alpha) \cdot n} \\
&= \log \binom{n}{(1-\alpha) \cdot n} - \log \binom{2 \cdot (1-\alpha) \cdot n}{1/2 \cdot 2 \cdot (1-\alpha) \cdot n} \\
&> n \cdot H(1-\alpha) - (1-\alpha) \cdot n \cdot H(1/2) + O(\log n) \\
&= \Omega(n) \ ,
\end{aligned}$$

where the final equality holds for $\alpha \in (0.5, 0.77)$. □

In Section 6.2 we construct a SIQ-PCPP, resulting in the following theorem:

Theorem 6.2. *For every $\alpha, \eta : \mathbb{N} \to (0, 1]$ such that $0 < \eta < \alpha$ (that are computable in linear time), there exists a perfectly complete PCPP $(\mathbf{P}, \mathbf{V})$ for Ham_α with the following parameters:*

PCPP $(\mathbf{P}, \mathbf{V})$	
Soundness error	$\mathsf{s}(\delta) = \frac{\alpha - \delta}{\alpha - \eta} + \frac{1}{n^2}$
Proof length	$n + 2 \cdot \log^2 n/\eta^2$
Queries to input	1
Queries to proof	$2 \cdot \log n \cdot (3 \cdot \log n/\eta^2 + 2/(\alpha - \eta))$
Randomness	$2 \cdot \log^2 n \cdot (1 + 2/(\alpha - \eta))$
Verifier running time	$O(\log^2 n/\eta^2 + \log n/(\alpha - \eta))$
Prover expected running time	$O\left(\alpha/(\alpha - \eta) \cdot n \cdot \log n\right)$

where $n \in \mathbb{N}$ is the input size, $\alpha := \alpha(n)$, and $\eta := \eta(n)$.

6.1 Lower bound

In this section we show that any one-round interactive proof of proximity for Ham_α in which the verifier makes a single query to its input must have large proof length. Observe that, since a PCP is a restricted case of one-round IPs where the verifier does not read the prover's entire message, it follows that any PCP for this problem must have large proof length.

Theorem 6.3. *Any perfectly complete MA proof of proximity for Ham_α that on inputs of length n and Hamming distance δ has soundness error smaller than 1, message length l, and input query complexity 1 has:*

$$\mathsf{l} > \log \binom{n}{(1-\alpha)\cdot n} - \log \binom{(1-\alpha+\delta)\cdot n}{(1-\alpha)\cdot n} \ .$$

Proof. Denote by $(\mathbf{P}, \mathbf{V})$ the MA proof of proximity. Let S be the set of vectors of Hamming weight exactly α. Notice that $|S| = \binom{n}{(1-\alpha)\cdot n}$. For a prover message π let $S_\pi := \{\mathbb{x} \in S \mid \pi := \mathbf{P}(\mathbb{x})\}$ be the set of vectors for which π is the honest prover message. By an averaging argument, since there are at most 2^{l} different prover messages, there must exist some proof π with $|S_\pi| \geq |S|/2^{\mathsf{l}}$. Henceforth fix π to be such a prover message.

Let the vector $\mathbb{x}$ be the bitwise-AND of all of the vectors in S_π, and $I := \{i \in [n] \mid \mathbb{x}[i] = 0\}$ be the set of indices for which $\mathbb{x}[i] = 0$. We show that $\mathbb{x}$ must be nonzero at many locations:

Claim 6.4. $|I| < (1-\alpha+\delta)\cdot n$.

Proof. Suppose towards contradiction (of the perfect completeness of the protocol) that $|I| \geq (1-\alpha+\delta)\cdot n$, meaning the $\Delta(\mathbb{x}, \mathsf{Ham}_\alpha) \geq \delta$. It follows from the soundness property of the MA proof that

$$\Pr_\rho[\mathbf{V}^{\mathbb{x}}(\pi;\rho) = 1] < 1 \ .$$

This means that there exists some choice of verifier randomness ρ such that $\mathbf{V}^{\mathbb{x}}(\pi;\rho) = 0$. Let $j \in [n]$ be the single index of $\mathbb{x}$ queried by $\mathbf{V}$ when given oracle access to π and this fixed randomness ρ. Since $\mathbb{x}$ is the bitwise-AND of vectors in S_π, it follows that there exists $u \in S_\pi$ with $u[j] = \mathbb{x}[j]$. Fix such a vector u.

Then, since j is the only index queried by the verifier, $\mathbf{V}^u(\pi;\rho) = \mathbf{V}^{\mathbb{x}}(\pi;\rho) = 0$. It follows that

$$\Pr_\rho[\mathbf{V}^u(\pi;\rho) = 1] < 1 \ .$$

Recall that, by the definition of S_π, u has Hamming weight exactly α and $\pi = \mathbf{P}(u)$, and so perfect completeness holds for u. We therefore have a contradiction to the perfect completeness of the protocol. □

Consider a vector $u \in S_\pi$. By the definition of $\mathbb{x}$, any index j with $u[j] = 0$ must belong to the set I. Moreover, since $u \in S$, u has exactly $(1-\alpha) \cdot n$ zeroes. Thus the total number of such vectors is $\binom{|I|}{(1-\alpha)\cdot n}$ (i.e., since we must choose $(1-\alpha) \cdot n$ locations out of the I indices to be 0 for each vector).

Putting all of this together, we have

$$\binom{n}{(1-\alpha)\cdot n}/2^{\mathsf{l}} \leq |S_\pi| \leq \binom{|I|}{(1-\alpha)\cdot n} < \binom{(1-\alpha+\delta)\cdot n}{(1-\alpha)\cdot n} ,$$

where the final inequality follows from Claim 6.4. The theorem follows by reordering the expressions and taking a logarithm. □

6.2 Upper bound

Construction 6.5. Define $t := 2 \cdot \log n/\eta^2$, $q := 4 \cdot \log n/(\alpha - \eta)$, and $m := 2 \cdot \log n$. The prover $\mathbf{P}$ receives as input bit vector $\mathbb{x} \in \{0,1\}^n$, while the verifier $\mathbf{V}$ has oracle access to the vector $\mathbb{x}$. They interact as follows.

- $\mathbf{P}(\mathbb{x})$:
  1. Generate $z_1, \ldots, z_t \in [n]$.
  2. Set $\mathbb{x}' := \mathbb{x}$.
  3. Output $\pi := (\mathbb{x}', z_1, \ldots, z_t)$.
- $\mathbf{V}^{\mathbb{x},\pi}$:
  1. Parse $\pi := (\mathbb{x}', z_1, \ldots, z_t)$.
  2. Query $z_1, \ldots, z_t \in [n]$.
  3. Sample $\rho_1, \ldots, \rho_m \leftarrow [n]$, and $r_1, \ldots, r_q \leftarrow [n]$.
  4. If $\exists \ell \in [m] :$ $\mathsf{weight}(\mathbb{x}'[\rho_\ell + z_1], \ldots, \mathbb{x}'[\rho_\ell + z_t]) < (\alpha - \eta)$, then reject.
  5. Query $\mathbb{x}'[r_i]$ for every $i \in [q]$. If there exists index where $\mathbb{x}'[r_i] = 1$ then query $\mathbb{x}[r^*]$ and reject if $\mathbb{x}[r^*] \neq \mathbb{x}'[r^*]$, where $r^* := r_\ell$ and ℓ is the minimal index with $\mathbb{x}'[r_\ell] = 1$ (if no such index exists, then skip this check).
  6. Otherwise, accept.

Completeness. Fix a vector $\mathbb{x} \in \mathsf{Ham}_\alpha$, i.e., $\mathsf{weight}(\mathbb{x}) \geq \alpha$. By Lemma 4.3 with $k = 1$, there exists a series of shifts $z_1, \ldots, z_t$ such that for every ρ there are at least $(\alpha - \eta) \cdot t$ indices j where $\mathbb{x}[\rho + z_j] = 1$. The honest prover $\mathbf{P}$ uses these shifts. For every $\rho \in [n]$, the bit vector $(\mathbb{x}'[\rho + z_1], \ldots, \mathbb{x}'[\rho + z_t])$ generated by $\mathbf{V}$ contains at least $(\alpha - \eta) \cdot t$ ones. Consequently, $\mathbf{V}$ will not reject upon reading the vectors in Item 4. Moreover, we have that $\mathbb{x}' = \mathbb{x}$, and therefore, $\mathbf{V}$ will accept during Item 5. Thus, the honest prover strategy of sending this series of shifts $z_1, \ldots, z_t$ and then following the protocol as prescribed causes the verifier to accept with probability 1.

Soundness. Fix a malicious prover $\tilde{\mathbf{P}}$, bit vectors $\mathbb{x} \notin \mathsf{Ham}_\alpha$, and a proof $\tilde{\pi} := (\mathbb{x}', z_1, \ldots, z_t)$ sent by $\tilde{\mathbf{P}}$. Let $\delta := \Delta(\mathbb{x}, \mathsf{Ham}_\alpha)$. Note that $\delta = \alpha - \mathsf{weight}(\mathbb{x})$.

For every $\rho \in [n]$, let $\mathbb{v}_\rho := (\mathbb{x}'[\rho + z_1], \ldots, \mathbb{x}'[\rho + z_t])$. For the prover and verifier interaction $\langle \tilde{\mathbf{P}}(\mathbb{x}), \mathbf{V}^{\mathbb{x}} \rangle$, we define the following events:

$$\begin{aligned} E_{\mathsf{weight}} &:= [\forall \ell \in [m],\ \mathsf{weight}(\mathbb{v}_{\rho_\ell}) \geq (\alpha - \eta)] \ , \\ E_0 &:= \left[\forall \ell \in [q],\ \mathbb{x}'[r_\ell] = 0\right] \ , \\ E_1 &:= \left[\exists \ell \in [q] \text{ s.t. } \mathbb{x}'[r_\ell] = 1 \wedge \mathbb{x}[r^*] = \mathbb{x}'[r^*] = 1\right] \ . \end{aligned}$$

Note that,

$$\begin{aligned} \Pr[\langle \tilde{\mathbf{P}}(\mathbb{x}), \mathbf{V}^{\mathbb{x}} \rangle = 1] &= \Pr[E_{\mathsf{weight}} \wedge (E_0 \vee E_1)] \\ &= \Pr[E_{\mathsf{weight}}] \cdot \Pr[E_0 \vee E_1] \ . \end{aligned} \tag{1}$$

where the second equality is since E_{weight} is independent of E_0 and E_1. We define a set of randomness as follows:

$$\mathcal{H} := \{\rho \mid \mathsf{weight}(\mathbb{v}_\rho) \geq (\alpha - \eta)\} \ .$$

We bound Equation 1 by splitting into two cases:

- By Claim 6.6: if $|\mathcal{H}| < n/2$ then the verifier accepts with probability at most $\frac{1}{n^2}$.
- By Claim 6.7: if $|\mathcal{H}| \geq n/2$ then the verifier accepts with probability at most $\frac{\alpha - \delta}{\alpha - \eta} + \frac{1}{n^2}$.

Together,

$$\begin{aligned} \Pr[\langle \tilde{\mathbf{P}}(\mathbb{x}), \mathbf{V}^{\mathbb{x}} \rangle &\leq \max \left\{ \frac{1}{n^2}, \frac{\alpha - \delta}{\alpha - \eta} + \frac{1}{n^2} \right\} \\ &= \frac{\alpha - \delta}{\alpha - \eta} + \frac{1}{n^2} \ . \end{aligned}$$

We now show that if the set $\mathcal{H}$ is small, then the verifier rejects with high probability.

Claim 6.6. *If $|\mathcal{H}| < n/2$ then* $\Pr[\langle \tilde{\mathbf{P}}(\mathbb{x}), \mathbf{V}^{\mathbb{x}} \rangle = 1] < \frac{1}{n^2}$.

Proof. If $|\mathcal{H}| < n/2$ then:

$$\begin{aligned} \Pr[E_{\mathsf{weight}}] &= \Pr_{\rho_1, \ldots, \rho_m} [\forall \ell \in [m],\ \mathsf{weight}(\mathbb{v}_{\rho_\ell}) \geq (\alpha - \eta)] \\ &= \prod_{\ell \in [m]} \Pr_{\rho_\ell} [\mathsf{weight}(\mathbb{v}_{\rho_\ell}) \geq (\alpha - \eta)] \\ &= \prod_{\ell \in [m]} \frac{|\mathcal{H}|}{n} < \left(\frac{1}{2}\right)^m = \frac{1}{n^2} \ . \end{aligned}$$

Thus, plugging this back in to Equation 1, if $|\mathcal{H}| < n/2$ then

$$\Pr[\langle \tilde{\mathbf{P}}(\mathbb{x}), \mathbf{V}^{\mathbb{x}} \rangle = 1] \leq \Pr[E_{\mathsf{weight}}] < \frac{1}{n^2}.$$

□

The following claim bounds the probability that the verifier accepts when the set $\mathcal{H}$ is large:

Claim 6.7. *If* $|\mathcal{H}| \geq n/2$ *then* $\Pr[\langle \tilde{\mathbf{P}}(\mathbb{x}), \mathbf{V}^{\mathbb{x}} \rangle = 1] < \frac{\alpha-\delta}{\alpha-\eta} + \frac{1}{n^2}$.

Proof. We separately bound $\Pr[E_0], \Pr[E_1]$, and $\Pr[E_{\mathsf{weight}}]$, and then use them to bound the probability that the verifier accepts by plugging them into Equation 1.

Denote by β and β' the Hamming weights of $\mathbb{x}$ and $\mathbb{x}'$ respectively. Observe that,

$$\beta := \alpha - \delta = \mathsf{weight}(\mathbb{x}) \ ,$$

$$\beta' := \mathsf{weight}(\mathbb{x}') \ .$$

- $\Pr[E_0]$: we first lower bound the probability of the verifier to sample $\mathbb{x}'[r] = 1$,

$$\begin{aligned}
\Pr_r[\mathbb{x}'[r] = 1] &= \Pr_{i,r}[\mathbb{x}'[r + z_i] = 1] \\
&\geq \Pr_r[r \in \mathcal{H}] \cdot \Pr_{i,r}[\mathbb{x}'[r + z_i] = 1 \mid r \in \mathcal{H}] \\
&= \frac{|\mathcal{H}|}{n} \cdot \Pr_{i,r}[\mathbb{v}_r[i] = 1 \mid \mathsf{weight}(\mathbb{v}_r) \geq (\alpha - \eta)] \\
&\geq \frac{\alpha - \eta}{2} \ .
\end{aligned}$$

Therefore,

$$\begin{aligned}
\Pr[E_0] &= \Pr_{r_1,\ldots,r_q}[\forall \ell \in [q], \ \mathbb{x}'[r_\ell] = 0] \\
&= \prod_{\ell \in [q]} \Pr_{r_\ell}[\mathbb{x}'[r_\ell] = 0] \\
&= \prod_{\ell \in [q]} \Pr_r[\mathbb{x}'[r] = 0] \\
&\leq \prod_{\ell \in [q]} \left(1 - \frac{\alpha - \eta}{2}\right) \\
&= \left(1 - \frac{\alpha - \eta}{2}\right)^{4 \cdot \log n/(\alpha - \eta)} \\
&\leq \left(\frac{1}{e}\right)^{2 \cdot \log n} \leq \frac{1}{n^2} \ . \qquad (2)
\end{aligned}$$

- $\Pr[E_1]$:

$$\begin{aligned}\Pr[E_1] &= \Pr_{r_1,\ldots,r_q}\left[\exists \ell \in [q] \text{ s.t. } \mathbb{x}'[r_\ell] = 1 \wedge \mathbb{x}[r^*] = \mathbb{x}'[r^*] = 1\right] \\ &\leq \Pr_{r_1,\ldots,r_q}\left[\mathbb{x}[r^*] = \mathbb{x}'[r^*] = 1 \mid \exists \ell \in [q] \text{ s.t. } \mathbb{x}'[r_\ell] = 1\right] \\ &= \Pr_{r}\left[\mathbb{x}[r] = \mathbb{x}'[r] = 1 \mid \mathbb{x}'[r] = 1\right] \\ &= \frac{|\mathbb{x}|}{|\mathbb{x}'|} = \frac{\beta}{\beta'} \ . \end{aligned} \tag{3}$$

- $\Pr[E_{\mathsf{weight}}]$: For every $j \in [t]$, let X_j be the $0/1$ random variable such that

$$\Pr[X_j = 1] = \Pr_{\rho \leftarrow [n]}\left[\mathbb{x}[\rho + z_j] = 1\right] \ .$$

  Notice that $\mathbb{E}_\rho[X_j] \leq \beta'$ for every $j \in [t]$. In what follows we show that the probability that there are $(\alpha - \eta) \cdot t$ shifts z_j for which $\tilde{\mathbf{P}}$ is able to convince $\mathbf{V}$ is small. By utilizing Markov's inequality and the linearity of expectation, we derive:

$$\begin{aligned}\Pr[E_{\mathsf{weight}}] &= \Pr_{\rho_1,\ldots,\rho_m}[\forall \ell \in [m],\ \mathsf{weight}(\mathbb{v}_{\rho_\ell}) \geq (\alpha - \eta)] \\ &= \prod_{\ell \in [m]} \Pr_{\rho_\ell}[\mathsf{weight}(\mathbb{v}_{\rho_\ell}) \geq (\alpha - \eta)] \\ &= \prod_{\ell \in [m]} \Pr_{\rho}[\mathsf{weight}(\mathbb{v}_{\rho}) \geq (\alpha - \eta)] \\ &= \prod_{\ell \in [m]} \Pr_{\rho}\left[\sum_{j \in [t]} X_j \geq (\alpha - \eta) \cdot t\right] \\ &\leq \left(\frac{\mathbb{E}[\sum_j X_j]}{(\alpha - \eta) \cdot t}\right)^m \\ &\leq \left(\frac{\sum_j \mathbb{E}[X_j]}{(\alpha - \eta) \cdot t}\right)^m \\ &\leq \min\left\{\left(\frac{\beta'}{\alpha - \eta}\right)^m, 1\right\} \ . \end{aligned} \tag{4}$$

By plugging Equations 2 to 4 into Equation 1 we get that if $|\mathcal{H}| \geq n/2$ then:

$$\begin{aligned}\Pr[\langle \tilde{\mathbf{P}}(\mathbb{x}), \mathbf{V}^{\mathbb{x}} \rangle = 1] &= \Pr[E_{\mathsf{weight}}] \cdot \Pr[E_0 \vee E_1] \\ &\leq \Pr[E_{\mathsf{weight}}] \cdot (\Pr[E_0] + \Pr[E_1]) \\ &\leq \min\left\{\left(\frac{\beta'}{\alpha - \eta}\right)^m, 1\right\} \cdot \left(\frac{1}{n^2} + \frac{\beta}{\beta'}\right) \ . \end{aligned}$$

To bound the above expression, we split into the following two cases.

- $\beta' < \alpha - \eta$: In this case, $\left(\frac{\beta'}{\alpha-\eta}\right)^m < 1$. Therefore,

$$\begin{aligned}
&\min\left\{\left(\frac{\beta'}{\alpha-\eta}\right)^m, 1\right\} \cdot \left(\frac{1}{n^2} + \frac{\beta}{\beta'}\right) \\
&\leq \left(\frac{\beta'}{\alpha-\eta}\right) \cdot \left(\frac{1}{n^2} + \frac{\beta}{\beta'}\right) \\
&\leq \frac{\beta}{\alpha-\eta} + \frac{1}{n^2} \ .
\end{aligned}$$

- $\beta' \geq \alpha - \eta$: In this case,

$$\begin{aligned}
&\min\left\{\left(\frac{\beta'}{\alpha-\eta}\right)^m, 1\right\} \cdot \left(\frac{1}{n^2} + \frac{\beta}{\beta'}\right) \\
&= \frac{\beta}{\beta'} + \frac{1}{n^2} \\
&\leq \frac{\beta}{\alpha-\eta} + \frac{1}{n^2} \ .
\end{aligned}$$

Overall, we get that if $|\mathcal{H}| \geq n/2$ then,

$$\begin{aligned}
\Pr[\langle \tilde{\mathbf{P}}(\mathbb{x}), \mathbf{V}^{\mathbb{x}} \rangle = 1] &\leq \min\left\{\left(\frac{\beta'}{\alpha-\eta}\right)^m, 1\right\} \cdot \left(\frac{1}{n^2} + \frac{\beta}{\beta'}\right) \\
&\leq \frac{\beta}{\alpha-\eta} + \frac{1}{n^2} \\
&= \frac{\alpha-\delta}{\alpha-\eta} + \frac{1}{n^2} \ .
\end{aligned}$$

□

Complexity measures. We analyze the complexity parameters of the SIQ-PCPP.

- **Proof length:** The proof length is $n + \log n \cdot t = n + 2 \cdot \log^2 n/\eta^2$.
- **Queries to input:** The verifier makes at most 1 query to $\mathbb{x}$.
- **Queries to proof:** The queries to proof in each step is as follows,
  - In Item 2 , $\log n \cdot t$ queries.
  - In Item 4, $m \cdot t$ queries.
  - In Item 5, q queries.

Overall, the verifier makes the following number of queries to the proof,

$$\begin{aligned}\log n \cdot t + q + m \cdot t &= (\log n + m) \cdot t + q \\ &= (\log n + 2 \cdot \log n) \cdot 2 \cdot \log n \cdot \frac{1}{\eta^2} + 4 \cdot \frac{\log n}{(\alpha - \eta)} \\ &= 6 \cdot \log^2 n \cdot \frac{1}{\eta^2} + 4 \cdot \frac{\log n}{(\alpha - \eta)} \\ &= 2 \cdot \log n \cdot \left(3 \cdot \log n \cdot \frac{1}{\eta^2} + \frac{2}{(\alpha - \eta)}\right) \quad .\end{aligned}$$

- **Randomness:** The overall randomness that the verifier uses is

$$\begin{aligned}m \cdot \log n + q \cdot \log n &= (m + q) \cdot \log n \\ &= \left(2 \cdot \log n + 4 \cdot \frac{\log n}{\alpha - \eta}\right) \cdot \log n \\ &= 2 \cdot \log^2 n \cdot \left(1 + \frac{2}{\alpha - \eta}\right) \quad .\end{aligned}$$

- **Verifier running time:** The verifier time in each step is as follows,
  - In Item 2, $O(t)$ time.
  - In Item 3, $O(m + q)$ time.
  - In Item 4, $O(m \cdot t)$ time.
  - In Item 5, $O(q)$ time.

  Overall, the verifier runs in time $O(m \cdot t + q) = O(\log^2 n/\eta^2 + \log n/(\alpha - \eta))$.

- **Prover expected running time:** By Lemma 4.3, we get that the prover runs in expected time
  $O\left(\alpha/(\alpha - \eta) \cdot n \cdot \log n\right)$.

7 SIQ-IOPP for Hamming weight

In this section, we show an IOP of proximity for the Ham_α problem.

Theorem 7.1. *For every $\alpha, \eta : \mathbb{N} \to (0,1]$ such that $0 < \eta < \alpha$ (that are computable in linear time), Construction 7.2 yields a perfectly complete public-coin IOPP for Ham_α with the following parameters:*

IOPP $(\mathbf{P}, \mathbf{V})$	
Soundness error	$\mathsf{s}(\delta) = \frac{\alpha-\delta}{\alpha-\eta}$
Rounds	2
Proof length	$O(\log^2 n/\eta^2)$
Queries to input	1
Queries to proof	$O(\log n/\eta^2)$
Randomness	$O(\log n + \log 1/\eta)$
Verifier running time	$O(\log n/\eta^2)$
Prover expected running time	$O\left(\left(n \cdot \alpha/(\alpha-\eta) + 1/\eta^2\right) \cdot \log n\right)$

where $n \in \mathbb{N}$ is the input size, $\alpha := \alpha(n)$, and $\eta := \eta(n)$.

The protocol is described below:

Construction 7.2. Let $t := 2 \cdot \log n/\eta^2$. The prover $\mathbf{P}$ receives as input a of bit vector $\mathtt{x} \in \{0,1\}^n$, while the verifier $\mathbf{V}$ has oracle access to the vector $\mathtt{x}$. They interact as follows.

- $\mathbf{P}$: Send $z_1, \ldots, z_t \in [n]$ to the verifier.
- $\mathbf{V}$: Choose $\rho \leftarrow [n]$ uniformly and send it to the prover.
- $\mathbf{P}$: Send $b_1, \ldots, b_t \in \{0,1\}$ as a non-oracle message where $b_j := \mathtt{x}[\rho + z_j]$.
- $\mathbf{V}$: Receive $b_1, \ldots, b_t \in \{0,1\}$ as a non-oracle message. Let $S := \{j \in [t] \mid b_j = 1\}$ be the indices where b_j equals 1. Sample $j \leftarrow S$ uniformly at random. Accept if and only if both of the following checks pass:
  1. Check that $|S| > (\alpha - \eta) \cdot t$.
  2. Query z_j and check that $\mathtt{x}[\rho + z_j] = 1$ by querying $\mathtt{x}$ at the appropriate location.

Proof of Theorem 7.1. We analyze completeness and soundness and then describe the complexity measures of the IOPP.

Completeness. Fix a vector $\mathtt{x} \in \mathsf{Ham}_\alpha$, i.e., $\mathsf{weight}(\mathtt{x}) \geq \alpha$. By Lemma 4.3 with $k = 1$, there exists a series of shifts $z_1, \ldots, z_t$ such that for every ρ there are at least $(\alpha-\eta)\cdot t$ indices j where $\mathtt{x}[\rho + z_j] = 1$. The honest prover $\mathbf{P}$ uses these shifts. For every $\rho \in [n]$, the bit vector $(b_1, \ldots, b_t) = (\mathtt{x}[\rho+z_1], \ldots, \mathtt{x}[\rho+z_t])$ generated by $\mathbf{V}$ contains at least $(\alpha-\eta)\cdot t$ ones. Consequently, $\mathbf{V}$ will not reject upon checking in Item 1 that there are at least $(\alpha - \eta) \cdot t$ indices j in which $b_j = 1$. Moreover, we have that $(b_1, \ldots, b_t) = (\mathtt{x}[\rho + z_1], \ldots, \mathtt{x}[\rho + z_t])$,

and therefore, $\mathbf{V}$ will accept during Item 2. Thus, the honest prover strategy of sending this series of shifts $z_1, \ldots, z_t$ and then following the protocol as prescribed causes the verifier to accept with probability 1.

Soundness. Fix a malicious prover $\tilde{\mathbf{P}}$ and a vector $\mathbb{x} \notin \mathsf{Ham}_\alpha$, and denote $\delta := \Delta(\mathbb{x}, \mathsf{Ham}_\alpha)$. Let $z_1, \ldots, z_t$ be the first message output by $\tilde{\mathbf{P}}$. For every $j \in [t]$, let X_j be the 0/1 random variable that is equal to 1 if and only if $\mathbb{x}[\rho + z_j] = 1$. Let $X \equiv \sum_{j=1}^{t} X_j$ be the random variable representing the number of X_j-s set to 1.

We begin by showing that if exactly w of the X_j random variables are set to 1 then the prover manages to convince the verifier with probability $\frac{w}{t \cdot (\alpha - \eta)}$:

Claim 7.3. *For every $w \in \{0, \ldots, t\}$:*

$$\Pr\left[\langle \tilde{\mathbf{P}}, \mathbf{V}^{\mathbb{x}} \rangle = 1 \,\middle|\, X = w \right] = \frac{w}{t \cdot (\alpha - \eta)} \ .$$

Proof. Since $X = \sum_{j \in [t]} X_j = w$, we have that the verifier $\mathbf{V}$ is able to be convinced by $\tilde{\mathbf{P}}$ for at w of the indices $j \in [t]$. In order for $\tilde{\mathbf{P}}$ to cause $\mathbf{V}$ to accept, the weight of the vector $(b_1, \ldots, b_t)$ sent by $\tilde{\mathbf{P}}$ must be at least $\alpha - \eta$. Consequently, the verifier accepts with probability at most $\frac{w}{t \cdot (\alpha - \eta)}$: the probability of sampling one of the w indices for which the prover can cause the verifier to accept out of the $(\alpha - \eta) \cdot t$ bits that must be set to 1. □

Notice now that $\mathbb{E}[X_j] = \alpha - \delta$ for every j, and so $\mathbb{E}[X] = (\alpha - \delta) \cdot t$. Furthermore, notice that $\mathbb{E}[X] = \sum_{w=0}^{t} w \cdot \Pr[X = w]$. By applying Claim 7.3 we have:

$$\begin{aligned}
\Pr[\langle \tilde{\mathbf{P}}, \mathbf{V}^{\mathbb{x}} \rangle = 1] &= \sum_{w=0}^{t} \Pr[X = w] \cdot \Pr\left[\langle \tilde{\mathbf{P}}, \mathbf{V}^{\mathbb{x}} \rangle = 1 \,\middle|\, X = w \right] \\
&= \sum_{w=0}^{t} \Pr[X = w] \cdot \frac{w}{t \cdot (\alpha - \eta)} \\
&= \frac{1}{t \cdot (\alpha - \eta)} \cdot \sum_{w=0}^{t} w \cdot \Pr[X = w] \\
&= \frac{\alpha - \delta}{\alpha - \eta} \ .
\end{aligned}$$

Complexity measures.

- *Proof length:* The proof length is $t \cdot \log n + t = O(\log^2 n / \eta^2)$.
- *Queries to input:* The verifier makes 1 query to $\mathbb{x}$.
- *Queries to proof:* The verifier makes $t + \log n = O(\log n / \eta^2)$ queries to the proof.

- *Randomness:* The verifier uses $\log n + O(\log |S|) = \log n + O(\log t) = \log n + O(\log\log n + \log 1/\eta)$ bits of randomness.
- *Verifier running time:* The verifier runs in time $O(t) = O(\log n/\eta^2)$.
- *Prover expected running time:* By Lemma 4.3, we get that computing $z_1, \ldots, z_t$ can be done in expected time $O\left(\alpha/(\alpha-\eta) \cdot n \cdot \log n\right)$. Therefore, the prover runs in expected time,

$$\begin{aligned} & O\left(\frac{\alpha}{\alpha-\eta} \cdot n \cdot \log n + t\right) \\ & = O\left(\frac{\alpha}{\alpha-\eta} \cdot n \cdot \log n + \frac{1}{\eta^2} \cdot \log n\right) \\ & = O\left(\left(\frac{\alpha}{\alpha-\eta} \cdot n + \frac{1}{\eta^2}\right) \cdot \log n\right) \quad . \end{aligned}$$

□

8 Lower bound for IOPPs

In this section, we bound the number of proof queries made by the verifier in an IOPP for Ham_α.

Theorem 8.1. *Let $(\mathbf{P}, \mathbf{V})$ be a perfectly complete semi-adaptive IOPP (see Definition 8.8) for Ham_α where $\alpha \in (0, 1 - \mathsf{q}_\mathbb{x}/n)$ that on inputs of length n and Hamming distance δ has soundness error smaller than 1, total length l, $\mathsf{q}_\mathbb{x}$ input queries and q_π queries to the prover messages. Then $\mathsf{q}_\pi \cdot (1 + \log \mathsf{l}) > \log\left(\frac{(1-\alpha)\cdot n}{\mathsf{q}_\mathbb{x}}\right)$.*

Moreover, if the IOPP is an IPP, then $\mathsf{l} > \log\left(\frac{(1-\alpha)\cdot n}{\mathsf{q}_\mathbb{x}}\right)$.[6]

Plugging in a constant α, we get the following bound:

Corollary 8.2. *For every constant $\alpha \in (0, 1)$, every perfectly complete semi-adaptive IOPP for Ham_α with total length $\mathsf{l} = \mathrm{polylog}(n)$ where the verifier makes $\mathsf{q}_\mathbb{x} = O(1)$ input queries and q_π queries to the prover messages has $\mathsf{q}_\pi = \Omega(\log(n)/\log\log n)$.*

Theorem 8.1 is proved by reducing a perfectly complete IOPP for Ham_α to a perfectly correct protocol for HitOne_α problem defined below (Lemma 8.5), and a communication-complexity lower bound for HitOne_α (Lemma 8.4).

Definition 8.3. *A protocol for the HitOne_α problem is pair $(\mathbf{A}, \mathbf{B})$ where $\mathbf{A}$ is a deterministic algorithm and $\mathbf{B}$ is a deterministic oracle algorithm. The aim of the protocol is for $\mathbf{A}$ to give $\mathbf{B}$ information that allows it to query a bit vector $\mathbb{x}$ at a nonzero location. $\mathbf{A}$ and $\mathbf{B}$ interact in the following way:*

1. *$\mathbf{A}(\mathbb{x})$: outputs a message $m \in \{0,1\}^\mathsf{l}$.*
2. *$\mathbf{B}^\mathbb{x}(m)$: makes q queries to $\mathbb{x}$.*

*We say that the protocol is **perfectly correct** if for every vector $\mathbb{x} \in \{0,1\}^n$ with $\mathsf{weight}(\mathbb{x}) \geq \alpha$: $\mathbf{B}^\mathbb{x}(\mathbf{A}(\mathbb{x}))$ queries $\mathbb{x}$ at an index $i \in [n]$ with $\mathbb{x}[i] = 1$.*

8.1 A lower-bound for perfectly correct protocols for HitOne_α

In this section we show a lower-bound on the length of perfectly correct protocols for HitOne_α.

Lemma 8.4. *Let $(\mathbf{A}, \mathbf{B})$ be a perfectly correct protocol for HitOne_α for $\alpha \in (0, 1 - \mathsf{q}/n]$ where, for vectors $\mathbb{x} \in \{0,1\}^n$ with $\mathsf{weight}(\mathbb{x}) \geq \alpha$, $\mathbf{A}(\mathbb{x})$ sends a message m of length l and $\mathbf{B}^\mathbb{x}(m)$ makes at most q queries to $\mathbb{x}$. Then $\mathsf{l} > \log\left(\frac{(1-\alpha)\cdot n}{\mathsf{q}}\right)$.*

[6] Observe that any IPP is semi-adaptive since the verifier always reads the prover's messages in their entirety.

Proof. Suppose towards contradiction that $\mathsf{l} \leq \log\left(\frac{(1-\alpha)\cdot n}{\mathsf{q}}\right)$. For any message m, let Q_m be the indices of $\mathbb{x}$ queried by $\mathbf{B}^{\mathbb{x}}(m)$ and set $Q := \bigcup_{m\in\{0,1\}^{\mathsf{l}}} Q_m$. Observe that $|Q| \geq \mathsf{q}\cdot 2^{\mathsf{l}}$ since each of the 2^{l} messages m can cause $\mathbf{B}$ to query a different set of q locations. Define a vector $\mathbb{x} \in \{0,1\}^n$ as follows:

$$\mathbb{x}[i] := \begin{cases} 0 & i \in Q \\ 1 & o.w. \end{cases}$$

By definition, and by the assumption that $\mathsf{l} \leq \log\left(\frac{(1-\alpha)\cdot n}{\mathsf{q}}\right)$ we have

$$\mathsf{weight}(\mathbb{x}) = 1 - \frac{|Q|}{n} \geq 1 - \frac{\mathsf{q}\cdot 2^{\mathsf{l}}}{n} \geq \alpha \ .$$

Moreover, by construction, no matter what message m it receives, $\mathbf{B}$ queries only zeroes. This contradicts the perfect correctness of the protocol $(\mathbf{A}, \mathbf{B})$. □

8.2 Perfectly complete IOPP for Hamming to perfectly correct protocol for HitOne_α

In this section we show how to transform a perfectly complete IOPP for the Hamming problem into a perfectly correct protocol for HitOne.

Lemma 8.5. *Let $(\mathbf{P}, \mathbf{V})$ be a perfectly complete public-coin semi-adaptive IOPP for Ham_α. Then Construction 8.10 yields a perfectly correct protocol $(\mathbf{A}, \mathbf{B})$ for HitOne with the following parameters:*

Perfectly complete IOPP for Ham_α	
Completeness error	0
Soundness error	< 1
Total proof length	l
Queries to proofs	q_π
Queries to vector	$\mathsf{q}_\mathbb{x}$

$\longrightarrow$

Perfectly correct protocol for HitOne_α	
Message length	$\mathsf{q}_\pi \cdot (1 + \log \mathsf{l})$
Queries	$\mathsf{q}_\mathbb{x}$

Moreover, if the IOPP is an IPP then the message length is reduced to l.

We define the view of an oracle algorithm to be the set of bits that the algorithm reads from the oracle at each round, where each bit is represented as a pair of index location and bit value. In the general case, where the algorithm interacts with multiple oracles, the algorithm's view is defined as a vector of sets. Each set within the vector corresponds to the bits read from a distinct oracle.

Definition 8.6 (View of oracle algorithm)**.** *Let A be a k-oracle algorithm, $o_1, \ldots, o_\mathsf{k}$ be oracles, and let x be an input value. The view of $\mathsf{A}^{o_1,\ldots,o_\mathsf{k}}(x)$ is defined as follows:*

$$\mathsf{View}\left(\mathsf{A}^{o_1,\ldots,o_\mathsf{k}}(x)\right) := (Q_\ell, \mathbf{a}_\ell)_{\ell\in[\mathsf{k}]} \ ,$$

where Q_ℓ is the set of queries that $\mathsf{A}^{o_1,\ldots,o_\mathsf{k}}(x)$ makes to o_ℓ, and $\mathbf{a}_\ell[j] := o_\ell(j)$ for each query $j \in Q_\ell$.

Given the view $\mathsf{w} := (Q, \mathbf{a})$ of an oracle, we let $\langle \mathsf{w} \rangle$ be the function that, on input j outputs $\mathbf{a}[j]$ if $j \in Q$ and $\perp$ otherwise.

The following fact shows that an oracle algorithm has the same view when given an oracle o as when rerun with the restricted view of o.

Fact 8.7. *Let A be an oracle algorithm, o be an oracle, x be an input value and $\mathsf{w} := \mathsf{View}(\mathsf{A}^o(x))$. Then $\mathsf{w} = \mathsf{View}(\mathsf{A}^{\langle \mathsf{w} \rangle}(x))$ and for every y with $\mathsf{w} = \mathsf{View}(\mathsf{A}^{\langle \mathsf{w} \rangle}(y))$ it holds that:*

$$\mathsf{View}(\mathsf{A}^{\langle \mathsf{w} \rangle}(y)) = \mathsf{View}(\mathsf{A}^o(y)) \ .$$

An IOP verifier is semi-adaptive if the locations it queries its ℓ-th oracle are independent of the $(\ell+1)$-th prover message onwards. This can be described formally as follows:

Definition 8.8 (Semi-adaptive IOP verifier)**.** *Let $(\mathbf{P}, \mathbf{V})$ be an IOPP for Ham_α. We say that $\mathbf{V}$ is **semi-adaptive** if there exist algorithms $\mathbf{V}_1, \ldots, \mathbf{V}_\mathsf{k}, \mathbf{Dec}$ such that for every vector $\mathbb{x}$ and full transcript $\mathsf{tr} := (\pi_1, \rho_1, \ldots, \pi_\mathsf{k}, \rho_\mathsf{k})$ the verifier's decision phase can be rewritten as follows:*

1. *For every $\ell \in [\mathsf{k}]$, compute $\mathsf{w}_\ell := \mathsf{View}\left(\mathbf{V}_\ell^{\pi_\ell}(\vec{\mathsf{w}}_1, \ldots, \vec{\mathsf{w}}_{\ell-1}, \rho_1, \ldots, \rho_\ell)\right)$.*
2. *Output $\mathbf{Dec}^{\mathbb{x}}(\vec{\mathsf{w}}_1, \ldots, \vec{\mathsf{w}}_\mathsf{k}, \rho_1, \ldots, \rho_\mathsf{k})$.*

We define a set of "useful" randomness, which will be used extensively in our transformation.

Definition 8.9 (Useful randomness)**.** *Let $\mathbf{V} = (\mathbf{V}_1, \ldots, \mathbf{V}_\mathsf{k}, \mathbf{Dec})$ be a semi-adaptive verifier and $\mathsf{w}_1, \ldots, \mathsf{w}_i$ be views. We define a set $\mathsf{Useful}(\mathsf{w}_1, \ldots, \mathsf{w}_i)$ to be all sets of randomness $(\rho_1, \ldots, \rho_i)$ such that*

$$\forall \pi_{i+1}, \exists \rho_{i+1} \ldots, \forall \pi_\mathsf{k}, \exists \rho_\mathsf{k} \ \vec{\mathbf{Dec}}^{\vec{0}}(\vec{\mathsf{w}}_1, \ldots, \vec{\mathsf{w}}_\mathsf{k}, \rho_1, \ldots, \rho_\mathsf{k}) = 0 \ ,$$

where $\forall \ell \in \{i+1, \ldots, \mathsf{k}\}$ $\mathsf{w}_\ell := \mathsf{View}(\mathbf{V}_\ell^{\pi_\ell}(\vec{\mathsf{w}}_1, \ldots, \vec{\mathsf{w}}_{\ell-1}, \rho_1, \ldots, \rho_\ell))$.

We now construct a protocol $(\mathbf{A}, \mathbf{B})$ for HitOne_α.

Construction 8.10. Let $(\mathbf{P}, \mathbf{V})$ be a perfectly complete public-coin IOPP for Ham_α for $\alpha \in (0,1)$ with semi-adaptive verifier $\mathbf{V} := (\mathbf{V}_1, \ldots, \mathbf{V}_\mathsf{k}, \mathbf{Dec})$. The protocol $(\mathbf{A}, \mathbf{B})$ for HitOne_α is as follows:

- $\mathbf{A}(\mathbb{x})$:
  1. For $\ell = 1$ to k:

(a) Compute $\pi_\ell := \mathbf{P}(\mathbb{x}, \rho_1^{\mathsf{A}}, \ldots, \rho_{\ell-1}^{\mathsf{A}})$.

(b) Find the lexicographically first $\rho_\ell^{\mathsf{A}} \in \{0,1\}^{\mathsf{r}}$ such that for $\mathsf{w}_\ell := \mathsf{View}(\mathbf{V}_\ell^{\pi_\ell}(\vec{\mathsf{w}}_1, \ldots, \vec{\mathsf{w}}_{\ell-1}, \rho_1^{\mathsf{A}}, \ldots, \rho_\ell^{\mathsf{A}}))$, we have that $(\rho_1^{\mathsf{A}}, \ldots, \rho_\ell^{\mathsf{A}}) \in \mathsf{Useful}(\mathsf{w}_1, \ldots, \mathsf{w}_\ell)$. (If no such ρ_ℓ^{A} exists then abort.)

(c) Set $\mathsf{w}_\ell := \mathsf{View}(\mathbf{V}_\ell^{\pi_\ell}(\vec{\mathsf{w}}_1, \ldots, \vec{\mathsf{w}}_{\ell-1}, \rho_1^{\mathsf{A}}, \ldots, \rho_\ell^{\mathsf{A}}))$.

2. Output $m := (\mathsf{w}_1, \ldots, \mathsf{w}_{\mathsf{k}})$.

- $\mathbf{B}^{\mathbb{x}}(m)$:

1. Parse $m := (\mathsf{w}_1, \ldots, \mathsf{w}_{\mathsf{k}})$.
2. For $\ell = 1$ to k,

(a) Find the lexicographically first $\rho_\ell^{\mathsf{B}} \in \{0,1\}^{\mathsf{r}}$ such that the following two conditions hold,

i. $\mathsf{w}_\ell = \mathsf{View}(\mathbf{V}_\ell^{\langle \mathsf{w}_\ell \rangle}(\vec{\mathsf{w}}_1, \ldots, \vec{\mathsf{w}}_{\ell-1}, \rho_1^{\mathsf{B}}, \ldots, \rho_\ell^{\mathsf{B}}))$.

ii. $(\rho_1^{\mathsf{B}}, \ldots, \rho_\ell^{\mathsf{B}}) \in \mathsf{Useful}(\mathsf{w}_1, \ldots, \mathsf{w}_\ell)$.

(If no such ρ_ℓ^{B} exists then abort.)

3. Run $\mathbf{Dec}^{\mathbb{x}}(\mathsf{w}_1, \ldots, \mathsf{w}_{\mathsf{k}}, \rho_1^{\mathsf{B}}, \ldots, \rho_{\mathsf{k}}^{\mathsf{B}})$ making any queries it makes to $\mathbb{x}$.

Proof. We first show that the protocol $(\mathbf{A}, \mathbf{B})$ is perfectly correct. Following this, we analyze the complexity parameters of the protocol.

Perfect correctness. Fix $\mathbb{x}$ with $\mathsf{weight}(\mathbb{x}) \geq \alpha$. We start by proving that $\mathbf{A}$ never aborts (Claim 8.11), meaning that it manages to fix randomness $(\rho_1^{\mathsf{A}}, \ldots, \rho_{\mathsf{k}}^{\mathsf{A}})$ during its execution, and outputs $m := (\mathsf{w}_1, \ldots, \mathsf{w}_{\mathsf{k}})$. Then, we prove that $\mathbf{B}$ chooses the same randomness as $\mathbf{A}$, i.e., $\mathbf{B}$ never aborts and chooses random strings $(\rho_1^{\mathsf{B}}, \ldots, \rho_{\mathsf{k}}^{\mathsf{B}}) = (\rho_1^{\mathsf{A}}, \ldots, \rho_{\mathsf{k}}^{\mathsf{A}})$ (Claim 8.12). Finally, we prove that $\mathbf{Dec}^{\mathbb{x}}(\mathsf{w}_1, \ldots, \vec{\mathsf{w}}_{\mathsf{k}}, \rho_1^{\mathsf{A}}, \ldots, \rho_{\mathsf{k}}^{\mathsf{A}})$ must query $\mathbb{x}$ at location i with $\mathbb{x}[i] = 1$ (Claim 8.13). Putting all of this together, we have that $\mathbf{B}$ always queries $\mathbb{x}$ at some nonzero location, as required.

Claim 8.11. $\mathbf{A}$ *never aborts.*

Proof. We prove that $\mathbf{A}$ never aborts by showing (by induction) that for every $i \in \{0, \ldots, \mathsf{k}-1\}$, if $(\rho_1^{\mathsf{A}}, \ldots, \rho_{i-1}^{\mathsf{A}}) \in \mathsf{Useful}(\mathsf{w}_1, \ldots, \mathsf{w}_{i-1})$, then for all π_i (and specifically for $\pi_i = \pi_i^{\mathsf{A}}$), there exists ρ_i^{A} such that $(\rho_1^{\mathsf{A}}, \ldots, \rho_i^{\mathsf{A}}) \in \mathsf{Useful}(\mathsf{w}_1, \ldots, \mathsf{w}_i)$, where $\mathsf{w}_i := \mathsf{View}(\mathbf{V}_i^{\pi_i}(\mathsf{w}_1, \ldots, \mathsf{w}_{i-1}, \rho_1^{\mathsf{A}}, \ldots, \rho_\ell^{\mathsf{A}}))$. It follows that in every iteration, $\mathbf{A}$ will have a choice of ρ_i^{A}, and so it will not abort.

Base case. For $i = 0$, using λ to denote the empty string, we need to show that $\lambda \in \mathsf{Useful}(\lambda)$. Assume towards contradiction that $\lambda \notin \mathsf{Useful}(\lambda)$. By definition this means that

$$\exists \tilde{\pi}_1, \forall \rho_1 \ldots, \exists \tilde{\pi}_{\mathsf{k}}, \forall \rho_{\mathsf{k}}\ \mathbf{Dec}^{\vec{0}}(\mathsf{w}_1, \ldots, \mathsf{w}_{\mathsf{k}}, \rho_1, \ldots, \rho_{\mathsf{k}}) = 1\ ,$$

where $\forall \ell \in [\mathsf{k}]\ \mathsf{w}_\ell := \mathsf{View}(\mathbf{V}_\ell^{\tilde{\pi}_\ell}(\mathsf{w}_1, \ldots, \mathsf{w}_{\ell-1}, \rho_1, \ldots, \rho_\ell))$. Therefore, there exists an unbounded malicious prover $\tilde{\mathbf{P}}$ such that,

$$\Pr_{\rho_1,\ldots,\rho_\mathsf{k}}\left[\mathbf{Dec}^{\vec{0}}(\mathsf{w}_1,\ldots,\mathsf{w}_\mathsf{k},\rho_1,\ldots,\rho_\mathsf{k})=1 \;\middle|\; \begin{array}{r} \tilde{\pi}_1 \leftarrow \tilde{\mathbf{P}} \\ \vdots \\ \tilde{\pi}_\mathsf{k} \leftarrow \tilde{\mathbf{P}}(\rho_1,\ldots,\rho_{\mathsf{k}-1}) \\ \forall \ell \in [\mathsf{k}]\ \mathsf{w}_\ell := \mathsf{View}(\mathbf{V}_\ell^{\tilde{\pi}_\ell}(\mathsf{w}_1,\ldots,\mathsf{w}_{\ell-1},\rho_1,\ldots,\rho_{\ell-1})) \end{array}\right] = 1 \ .$$

Or equivalently,

$$\Pr_{\rho_1,\ldots,\rho_\mathsf{k}}\left[\mathbf{V}^{\vec{0},\tilde{\pi}_1,\ldots,\tilde{\pi}_\mathsf{k}}(\rho_1,\ldots,\rho_\mathsf{k})=1 \;\middle|\; \begin{array}{r} \tilde{\pi}_1 \leftarrow \tilde{\mathbf{P}} \\ \vdots \\ \tilde{\pi}_\mathsf{k} \leftarrow \tilde{\mathbf{P}}(\rho_1,\ldots,\rho_{\mathsf{k}-1}) \end{array}\right] = 1 \ ,$$

which contradicts the fact that the IOPP has soundness error smaller than 1.

Induction step. Fix $i > 0$, and suppose that $(\rho_1^{\mathsf{A}},\ldots,\rho_{i-1}^{\mathsf{A}}) \in \mathsf{Useful}(\mathsf{w}_1,\ldots,\mathsf{w}_{i-1})$. By definition:

$$\forall \pi_i, \exists \rho_i \ldots, \forall \pi_\mathsf{k}, \exists \rho_\mathsf{k}\ \mathbf{Dec}^{\vec{0}}\left(\mathsf{w}_1,\ldots,\mathsf{w}_\mathsf{k},\rho_1^{\mathsf{A}},\ldots,\rho_{i-1}^{\mathsf{A}},\rho_i,\ldots,\rho_\mathsf{k}\right)=1 \ ,$$

where $\forall \ell \in \{i,\ldots,\mathsf{k}\}$, $\mathsf{w}_\ell := \mathsf{View}(\mathbf{V}_\ell^{\pi_\ell}(\mathsf{w}_1,\ldots,\mathsf{w}_{\ell-1},\rho_1^{\mathsf{A}},\ldots,\rho_{i-1}^{\mathsf{A}},\rho_i,\ldots,\rho_\ell))$. Fix $\pi_i := \pi_i^{\mathsf{A}} := \mathbf{P}(\mathbb{x},\rho_1^{\mathsf{A}},\ldots,\rho_{i-1}^{\mathsf{A}})$, as done by $\mathbf{A}$. By unraveling the above expression, we get that,

$$\exists \rho_i, \forall \pi_{i+1}, \exists \rho_{i+1} \ldots, \forall \pi_\mathsf{k}, \exists \rho_\mathsf{k}\ \mathbf{Dec}^{\vec{0}}\left(\mathsf{w}_1,\ldots,\mathsf{w}_\mathsf{k},\rho_1^{\mathsf{A}},\ldots,\rho_{i-1}^{\mathsf{A}},\rho_i,\ldots,\rho_\mathsf{k}\right)=1 \ ,$$

where $\forall \ell \in \{i,\ldots,\mathsf{k}\}$, $\mathsf{w}_\ell := \mathsf{View}(\mathbf{V}_\ell^{\pi_\ell}(\mathsf{w}_1,\ldots,\mathsf{w}_{\ell-1},\rho_1^{\mathsf{A}},\ldots,\rho_{i-1}^{\mathsf{A}},\rho_i,\ldots,\rho_\ell))$.

Fixing ρ_i^{A} such that the above holds, it immediately follows that $(\rho_1^{\mathsf{A}},\ldots,\rho_\ell^{\mathsf{A}}) \in \mathsf{Useful}(\mathsf{w}_1,\ldots,\mathsf{w}_i)$ where $\mathsf{w}_i := \mathsf{View}(\mathbf{V}_\ell^{\pi_i^{\mathsf{A}}}(\mathsf{w}_1,\ldots,\mathsf{w}_{i-1},\rho_1^{\mathsf{A}},\ldots,\rho_\ell^{\mathsf{A}}))$, as required. □

Claim 8.12. $\mathbf{B}$ *never aborts and* $(\rho_1^{\mathsf{A}},\ldots,\rho_\mathsf{k}^{\mathsf{A}}) = (\rho_1^{\mathsf{B}},\ldots,\rho_\mathsf{k}^{\mathsf{B}})$.

Proof. We show by induction that for every $i \in \{0,\ldots,\mathsf{k}\}$ we have that $(\rho_1^{\mathsf{A}},\ldots\rho_i^{\mathsf{A}}) = (\rho_1^{\mathsf{B}},\ldots,\rho_i^{\mathsf{B}})$ (which also implies that $\mathbf{B}$ did not abort in the i-th iteration). For $i = 0$, this is trivially true. For the inductive step, we assume that the claim is true for every index smaller than $i > 0$, and we prove for i.

To prove that $\rho_i^{\mathsf{B}} = \rho_i^{\mathsf{A}}$ we need to show that (1) for every $\rho_i < \rho_i^{\mathsf{A}}$, at least one of the conditions in Item 2a does not hold, and (2) for $\rho_i = \rho_i^{\mathsf{A}}$, both of the conditions in Item 2a hold.

- $\rho_i < \rho_i^{\mathsf{A}}$: Assume towards contradiction that there exists $\rho_i < \rho_i^{\mathsf{A}}$ such that both of the conditions in Item 2a hold, i.e.,

  1. $\mathsf{w}_i = \mathsf{View}(\mathbf{V}_i^{\langle \mathsf{w}_i \rangle}(\mathsf{w}_1,\ldots,\mathsf{w}_{i-1},\rho_1^{\mathsf{B}},\ldots,\rho_{i-1}^{\mathsf{B}},\rho_i))$, and

2. $(\rho_1^{\mathsf{B}}, \ldots, \rho_{i-1}^{\mathsf{B}}, \rho_i) \in \mathsf{Useful}(\mathsf{w}_1, \ldots, \mathsf{w}_i)$.

It follows that

$$\begin{aligned}
\mathsf{w}_i &= \mathsf{View}(\mathbf{V}_i^{\langle \mathsf{w}_i \rangle}(\mathsf{w}_1, \ldots, \mathsf{w}_{i-1}, \rho_1^{\mathsf{B}}, \ldots, \rho_{i-1}^{\mathsf{B}}, \rho_i)) \\
&= \mathsf{View}(\mathbf{V}_i^{\langle \mathsf{w}_i \rangle}(\mathsf{w}_1, \ldots, \mathsf{w}_{i-1}, \rho_1^{\mathsf{A}}, \ldots, \rho_{i-1}^{\mathsf{A}}, \rho_i)) \qquad (5) \\
&= \mathsf{View}(\mathbf{V}_i^{\pi_i^{\mathsf{A}}}(\mathsf{w}_1, \ldots, \mathsf{w}_{i-1}, \rho_1^{\mathsf{A}}, \ldots, \rho_{i-1}^{\mathsf{A}}, \rho_i)) \ , \qquad (6)
\end{aligned}$$

where Equation 5 follows from the inductive assumption, and Equation 6 follows from Fact 8.7.

However, since $(\rho_1^{\mathsf{B}}, \ldots, \rho_{i-1}^{\mathsf{B}}, \rho_i) \in \mathsf{Useful}(\mathsf{w}_1, \ldots, \mathsf{w}_i)$, this contradicts the fact that $\mathbf{A}$ chooses the minimal ρ_i^{A} for which the derived random string is useful for the derived list of views.

- $\rho_i = \rho_i^{\mathsf{A}}$: We need to show that,

  1. $\mathsf{w}_i = \mathsf{View}(\mathbf{V}_i^{\langle \mathsf{w}_i \rangle}(\mathsf{w}_1, \ldots, \mathsf{w}_{i-1}, \rho_1^{\mathsf{B}}, \ldots, \rho_{i-1}^{\mathsf{B}}, \rho_i^{\mathsf{A}}))$.
  2. $(\rho_1^{\mathsf{B}}, \ldots, \rho_{i-1}^{\mathsf{B}}, \rho_i^{\mathsf{A}}) \in \mathsf{Useful}(\mathsf{w}_1, \ldots, \mathsf{w}_i)$.

  Or equivalently, by the inductive assumption, since $(\rho_1^{\mathsf{B}}, \ldots, \rho_{i-1}^{\mathsf{B}}) = (\rho_1^{\mathsf{A}}, \ldots, \rho_{i-1}^{\mathsf{A}})$,

  1. $\mathsf{w}_i = \mathsf{View}(\mathbf{V}_i^{\langle \mathsf{w}_i \rangle}((\mathsf{w}_1, \ldots, \mathsf{w}_{i-1}, \rho_1^{\mathsf{A}}, \ldots, \rho_{i-1}^{\mathsf{A}}, \rho_i^{\mathsf{A}}))$.
  2. $(\rho_1^{\mathsf{A}}, \ldots, \rho_{i-1}^{\mathsf{A}}, \rho_i^{\mathsf{A}}) \in \mathsf{Useful}(\mathsf{w}_1, \ldots, \mathsf{w}_i)$.

  By the construction of $\mathbf{A}$ and by Fact 8.7:

  $$\mathsf{w}_i = \mathsf{View}(\mathbf{V}_i^{\pi_i^{\mathsf{A}}}(\mathsf{w}_1, \ldots, \mathsf{w}_{i-1}, \rho_1^{\mathsf{A}}, \ldots, \rho_{i-1}^{\mathsf{A}}, \rho_i^{\mathsf{A}})) = \mathsf{View}(\mathbf{V}_i^{\langle \mathsf{w}_i \rangle}(\mathsf{w}_1, \ldots, \mathsf{w}_{i-1}, \rho_1^{\mathsf{A}}, \ldots, \rho_{i-1}^{\mathsf{A}}, \rho_i^{\mathsf{A}})) \ ,$$

  and so Item 1 holds. The correctness of Item 2 is implied by the construction of $\mathbf{A}$.

□

Claim 8.13. $\mathbf{Dec}^{\mathbb{x}}(\mathsf{w}_1, \ldots, \mathsf{w}_{\mathsf{k}}, \rho_1^{\mathsf{A}}, \ldots, \rho_{\mathsf{k}}^{\mathsf{A}})$ *queries* $\mathbb{x}$ *at a nonzero location.*

Proof. Since $(\rho_1^{\mathsf{A}}, \ldots, \rho_{\mathsf{k}}^{\mathsf{A}}) \in \mathsf{Useful}(\mathsf{w}_1, \ldots, \mathsf{w}_{\mathsf{k}})$, it follows that $\mathbf{Dec}^{\vec{0}}(\mathsf{w}_1, \ldots, \mathsf{w}_{\mathsf{k}}, \rho_1^{\mathsf{A}}, \ldots, \rho_{\mathsf{k}}^{\mathsf{A}}) = 0$. On the other hand, by perfect completeness of the IOPP we have that:

$$\Pr_{\rho_1, \ldots, \rho_{\mathsf{k}}} \left[\mathbf{Dec}^{\mathbb{x}}(\mathsf{w}_1, \ldots, \mathsf{w}_{\mathsf{k}}, \rho_1, \ldots, \rho_{\mathsf{k}}) = 1 \;\middle|\; \begin{array}{r} \pi_1 \leftarrow \mathbf{P}(\mathbb{x}) \\ \vdots \\ \pi_{\mathsf{k}} \leftarrow \mathbf{P}(\mathbb{x}, \rho_1, \ldots, \rho_{\mathsf{k}-1}) \\ \forall \ell \in [\mathsf{k}] \ \mathsf{w}_\ell := \mathsf{View}(\mathbf{V}_\ell^{\pi_\ell}(\mathsf{w}_1, \ldots, \mathsf{w}_{\ell-1}, \rho_1, \ldots, \rho_{\ell-1})) \end{array} \right] = 1 \ .$$

Observe that $\mathbf{A}$ computes w_ℓ exactly as computed in the above process. It follows that

$$\mathbf{Dec}^{\mathbb{x}}(\mathsf{w}_1, \ldots, \mathsf{w}_\mathsf{k}, \rho_1^{\mathsf{A}}, \ldots, \rho_\mathsf{k}^{\mathsf{A}}) = 1 \ .$$

Since the only difference between $\mathbf{Dec}^{\mathbb{x}}(\mathsf{w}_1, \ldots, \mathsf{w}_\mathsf{k}, \rho_1^{\mathsf{A}}, \ldots, \rho_\mathsf{k}^{\mathsf{A}})$ and $\mathbf{Dec}^{\vec{0}}(\mathsf{w}_1, \ldots, \mathsf{w}_\mathsf{k}, \rho_1^{\mathsf{A}}, \ldots, \rho_\mathsf{k}^{\mathsf{A}})$ are the locations of $\mathbb{x}$ in which $\mathbb{x}$ is nonzero and yet their outputs are different, it must hold that $\mathbf{Dec}^{\mathbb{x}}(\mathsf{w}_1, \ldots, \mathsf{w}_\mathsf{k}, \rho_1^{\mathsf{A}}, \ldots, \rho_\mathsf{k}^{\mathsf{A}})$ queries $\mathbb{x}$ at a nonzero location. □

Complexity measures. We analyze the complexity measures of the resulting protocol $(\mathbf{A}, \mathbf{B})$.

- *Message length.* Alice sends Bob a message m containing a list of index-bit pairs matching the query complexity of the IOPP $(\mathbf{P}, \mathbf{V})$ to the prover messages. Thus, the message length of the protocol is $\mathsf{q}_\pi \cdot (1 + \log \mathsf{l})$. Observe that if the IOPP is, in fact, an IPP, i.e., $\mathsf{q}_\pi = \mathsf{l}$, then the verifier's view is the entire proof, and so we do not need the indices, meaning that we get message length l.
- *Query complexity.* Bob makes at most $\mathsf{q}_{\mathbb{x}}$ queries to its oracle $\mathbb{x}$.

□

9 Application: perfect completeness for PCPs and IOPs

In this section, we show how to apply our techniques to transform probabilistic proof systems with nonzero completeness error into ones with perfect completeness. While following theorems are described for PCPPs and IOPPs, recall that (standard) PCPs and IOPs are a subset of their proximity variants.

- In Section 9.1 we show Theorem 9.3 which uses a perfectly complete PCPP for Ham to correct completeness errors in PCPs of proximity. By plugging in the PCPPs developed in Sections 5 and 6 into Theorem 9.3 we have the following corollary:

  Corollary 9.1. *For every relation R that has a PCPP with the following parameters and completeness error c, and for every $\eta \in (0, 1-\mathsf{c})$, R has a* perfectly complete *PCPP with the following parameters:*

PCPP for R	
Completeness error	c
Soundness error	s
Proof length	l
Queries to input	$\mathsf{q}_{\mathbb{x}}$
Queries to proof	q_{π}
Randomness	r
Verifier running time	vt

↓

Perfectly complete PCPP for R	Theorem 5.4	Theorem 5.8	Theorem 6.2
Soundness error	$\frac{\mathsf{s}}{1-\mathsf{c}-\eta}$	$\frac{\mathsf{s}}{1-\mathsf{c}-1.5\eta}$	$\frac{\mathsf{s}}{1-\mathsf{c}-\eta} + \frac{1}{n^2}$
Proof length	$\mathsf{l} + 2 \cdot \mathsf{r}^2/\eta^2$	$\mathsf{l} + O(2^{\mathsf{r}}/\mathsf{r}^2 \cdot (-\log^2 \eta)/\eta^2)$	$\mathsf{l} + 2^{\mathsf{r}} + 2 \cdot \mathsf{r}^2/\eta^2$
Queries to input	$2 \cdot \mathsf{q}_{\mathbb{x}} \cdot \mathsf{r}/\eta^2$	$O\left(\mathsf{q}_{\mathbb{x}} \cdot (\log \mathsf{r} - \log \eta)/\eta^2\right)$	$\mathsf{q}_{\mathbb{x}}$
Queries to proof	$O(\mathsf{q}_{\pi} \cdot \mathsf{r}/\eta^2 + \mathsf{r}^2/\eta^2)$	$O(\mathsf{q}_{\pi} \cdot (\log \mathsf{r} - \log \eta)/\eta^2 + \mathsf{r} \cdot (\log \mathsf{r} - \log^2 \eta)/\eta^2)$	$\mathsf{q}_{\pi} + O(\mathsf{r}^2/\eta^2 + \mathsf{r}/(1-\mathsf{c}-\eta))$
Randomness	r	$\mathsf{r} + \log \mathsf{r} - O(\log \eta)$	$O(\mathsf{r}^2/(1-\mathsf{c}-\eta))$

- In Section 9.2 we prove Theorem 9.3, which uses the techniques developed in previous sections to transform any IOPP into an IOPP with perfect completeness:

  Theorem 9.2. *Let R be a relation with a IOPP with nonzero completeness error c. Then for every $\eta \in (0, 1-\mathsf{c})$, R has a perfectly complete IOPP with the following parameters:*

IOPP for R	
Completeness error	c
Soundness error	s
Rounds	k
Proof length	l
Queries to input	$\mathsf{q}_{\mathbb{x}}$
Queries to proof	q_{π}
Randomness	r
Verifier running time	vt

$\longrightarrow$

Perfectly complete IOPP for R	
Completeness error	0
Soundness error	$\frac{\mathsf{s}}{1-\mathsf{c}-\eta}$
Rounds	$\mathsf{k}+1$
Proof length	$O(\mathsf{l}\cdot\mathsf{r}/\eta^2)$
Queries to input	$\mathsf{q}_{\mathbb{x}}$
Queries to proof	$\mathsf{q}_{\pi}+O(\mathsf{r}/\eta^2)$
Randomness	$\mathsf{r}+O(\log(\mathsf{r}/\eta))$
Verifier running time	$\mathsf{vt}+O(\mathsf{r}/\eta^2)$

9.1 Perfect completeness for PCPPs

In this section we show that any PCP of proximity for the gap-Hamming problem can be used to transform imperfectly complete PCPs to PCPs with perfect completeness.

Theorem 9.3. *Let R be a relation with a PCPP with nonzero completeness error c and randomness complexity r. Then, given a perfectly complete PCPP for $\mathsf{Ham}_{1-\mathsf{c}}$ for instances of size 2^{r}, Construction 9.4 yields a perfectly complete PCPP for R with the following parameters:*

PCPP for R	
Completeness error	c
Soundness error	s
Proof length	l
Queries to input	$\mathsf{q}_{\mathbb{x}}$
Queries to proof	q_{π}
Randomness	r
Verifier running time	vt

$+$

Perfectly complete PCPP for $\mathsf{Ham}_{1-\mathsf{c}}$	
Completeness error	0
Soundness error	$\mathsf{s}_{\mathsf{Ham}}(\delta)$
Proof length	$\mathsf{l}_{\mathsf{Ham}}$
Queries to input	$\mathsf{q}_{\mathbb{x},\mathsf{Ham}}$
Queries to proof	$\mathsf{q}_{\pi,\mathsf{Ham}}$
Randomness	$\mathsf{r}_{\mathsf{Ham}}$
Verifier running time	$\mathsf{vt}_{\mathsf{Ham}}$

$\longrightarrow$

Perfectly complete PCPP for R	
Completeness error	0
Soundness error	$\mathsf{s}_{\mathsf{Ham}}(1-\mathsf{c}-\mathsf{s})$
Proof length	$\mathsf{l}+\mathsf{l}_{\mathsf{Ham}}$
Queries to input	$\mathsf{q}_{\mathbb{x},\mathsf{Ham}}\cdot\mathsf{q}_{\mathbb{x}}$
Queries to proof	$\mathsf{q}_{\mathbb{x},\mathsf{Ham}}\cdot\mathsf{q}_{\pi}+\mathsf{q}_{\pi,\mathsf{Ham}}$
Randomness	$\mathsf{r}_{\mathsf{Ham}}$
Verifier running time	$\mathsf{q}_{\mathbb{x},\mathsf{Ham}}\cdot\mathsf{vt}+\mathsf{vt}_{\mathsf{Ham}}$

Construction 9.4. Let $(\mathbf{P},\mathbf{V})$ be a PCPP for R with completeness error c, and $(\mathbf{P}_{\mathsf{Ham}},\mathbf{V}_{\mathsf{Ham}})$ be a perfectly complete PCPP for $\mathsf{Ham}_{1-\mathsf{c}}$. On explicit input x and implicit input w the protocol executes as follows:

1. $\mathbf{P}'$:

   (a) Compute $\pi := \mathbf{P}(x,w)$.

(b) Let $\mathtt{x} \in \{0,1\}^{2^{\mathsf{r}}}$ be the vector such that $\mathtt{x}[\rho] = 1$ if and only if $\mathbf{V}^{w,\pi}(x;\rho) = 1$ for $\rho \in \{0,1\}^{\mathsf{r}}$.
(c) Compute $\pi_{\mathsf{Ham}} := \mathbf{P}_{\mathsf{Ham}}(\mathtt{x})$.
(d) Output $\pi' := (\pi, \pi_{\mathsf{Ham}})$.

2. $\mathbf{V}'$:

(a) Choose $\rho_{\mathsf{Ham}} \leftarrow \{0,1\}^{\mathsf{r}}$.
(b) Run $\mathbf{V}_{\mathsf{Ham}}^{\mathtt{x},\pi_{\mathsf{Ham}}}(\rho_{\mathsf{Ham}})$ where every query ρ to $\mathtt{x}$ is answered by executing $\mathbf{V}^{w,\pi}(x;\rho)$ (by making the appropriate queries to w and π) and handing $\mathbf{V}_{\mathsf{Ham}}$ the output of $\mathbf{V}$.
(c) Accept if and only if $\mathbf{V}_{\mathsf{Ham}}$ accepts.

Proof of Theorem 9.3. We prove completeness, then soundness, and finally analyze complexity measures.

Completeness. Fix $(x,w) \in R$. We show that the honest prover $\mathbf{P}'$ makes $\mathbf{V}'$ accept given x and oracle access to w with probability 1. Let $(\pi, \pi_{\mathsf{Ham}}) := \mathbf{P}'(x,w)$ be the proof output by $\mathbf{P}'$ and let $\mathtt{x}$ be the vector defined by the prover in Item 1b. By definition, since the PCPP $(\mathbf{P},\mathbf{V})$ has completeness error c:

$$\Pr[\mathtt{x}[\rho] = 1 \mid \rho \leftarrow \{0,1\}^{\mathsf{r}}] = \Pr[\,\mathbf{V}^{w,\pi}(x;\rho) = 1 \mid \rho \leftarrow \{0,1\}^{\mathsf{r}}\,] \geq 1 - \mathsf{c} \ .$$

Therefore $\mathtt{x} \in \mathsf{Ham}_{1-\mathsf{c}}$. Thus, since $(\mathbf{P}_{\mathsf{Ham}}, \mathbf{V}_{\mathsf{Ham}})$ is a perfectly complete PCPP for $\mathsf{Ham}_{1-\mathsf{c}}$, and since $\mathbf{V}'$ emulates the vector $\mathtt{x}$ for $\mathbf{V}_{\mathsf{Ham}}$, we conclude that

$$\Pr\left[\,\mathbf{V}'^{w,\pi'}(x) = 1\,\right] = \Pr_{\rho_{\mathsf{Ham}}}[\,\mathbf{V}_{\mathsf{Ham}}^{\mathtt{x},\pi_{\mathsf{Ham}}}(\rho_{\mathsf{Ham}}) = 1\,] = 1 \ .$$

Soundness. Fix $(x,w) \notin R$ and let $\pi' := (\pi, \pi_{\mathsf{Ham}})$ be a proof string. We show that

$$\Pr\left[\,\mathbf{V}'^{w,\pi'}(x) = 1\,\right] \leq \mathsf{s}_{\mathsf{Ham}} \ .$$

Let $\mathtt{x} \in \{0,1\}^{2^{\mathsf{r}}}$ be the vector such that $\mathtt{x}[\rho] = 1$ if and only if $\mathbf{V}^{w,\pi}(x;\rho) = 1$ for $\rho \in \{0,1\}^{\mathsf{r}}$. Since $(\mathbf{P},\mathbf{V})$ has soundness error s:

$$\Pr[\,\mathtt{x}[\rho] = 1 \mid \rho \leftarrow \{0,1\}^{\mathsf{r}}\,] = \Pr[\,\mathbf{V}^{w,\pi}(x;\rho) = 1 \mid \rho \leftarrow \{0,1\}^{\mathsf{r}}\,] \leq \mathsf{s} \ .$$

Therefore $\mathtt{x}$ is a vector with a fraction of at most s ones. Thus, since $(\mathbf{P}_{\mathsf{Ham}}, \mathbf{V}_{\mathsf{Ham}})$ is a PCPP for $\mathsf{Ham}_{1-\mathsf{c}}$ with soundness error $\mathsf{s}_{\mathsf{Ham}}(\delta)$, and since $\mathbf{V}'$ emulates the vector $\mathtt{x}$ for $\mathbf{V}_{\mathsf{Ham}}$, we conclude that

$$\Pr\left[\,\mathbf{V}'^{w,\pi'}(x) = 1\,\right] = \Pr_{\rho_{\mathsf{Ham}}}[\,\mathbf{V}_{\mathsf{Ham}}^{\mathtt{x},\pi_{\mathsf{Ham}}}(\rho_{\mathsf{Ham}}) = 1\,] \leq \mathsf{s}_{\mathsf{Ham}}(\Delta(\mathtt{x},\mathsf{Ham}_{1-\mathsf{c}})) \leq \mathsf{s}_{\mathsf{Ham}}(1-\mathsf{c}-\mathsf{s}) \ .$$

Complexity measures. We analyze the complexity parameters of the new PCPP.

- *Proof length.* The proof length is $\mathsf{l} + \mathsf{l}_{\mathsf{Ham}}$.
- *Queries to input.* The verifier makes $\mathsf{q}_{\mathbb{x}}$ queries to w for every one of the $\mathsf{q}_{\mathbb{x},\mathsf{Ham}}$ queries that $\mathbf{V}_{\mathsf{Ham}}$ makes to $\mathbb{x}$, for a total of $\mathsf{q}_{\mathbb{x},\mathsf{Ham}} \cdot \mathsf{q}_{\mathbb{x}}$.
- *Queries to proof.* The verifier makes $\mathsf{q}_{\pi,\mathsf{Ham}}$ queries to π_{Ham}, and q_{π} queries to π for every one of the $\mathsf{q}_{\mathbb{x},\mathsf{Ham}}$ queries that $\mathbf{V}_{\mathsf{Ham}}$ makes to $\mathbb{x}$, for a total of $\mathsf{q}_{\mathbb{x},\mathsf{Ham}} \cdot \mathsf{q}_{\pi} + \mathsf{q}_{\pi,\mathsf{Ham}}$ to $\pi' := (\pi, \pi_{\mathsf{Ham}})$.
- *Randomness.* The verifier uses $\mathsf{r}_{\mathsf{Ham}}$ bits of randomness in order to execute $\mathbf{V}_{\mathsf{Ham}}$.
- *Verifier running time.* The verifier makes a single invocation of $\mathbf{V}_{\mathsf{Ham}}$, taking time $\mathsf{vt}_{\mathsf{Ham}}$, in which every one of the $\mathsf{q}_{\mathbb{x},\mathsf{Ham}}$ queries made to $\mathbb{x}$ translates to computing $\mathbf{V}$ in time vt, for a total of $\mathsf{q}_{\mathbb{x},\mathsf{Ham}} \cdot \mathsf{vt} + \mathsf{vt}_{\mathsf{Ham}}$.

□

9.2 Perfect completeness for IOPPs

In this section we prove Theorem 9.2, showing how to achieve perfect completeness for IOPPs using the techniques developed in Section 7. We detail the construction:

Construction 9.5. Let $(\mathbf{P}, \mathbf{V})$ be an IOPP for R where, for convenience, we assume that the protocol $(\mathbf{P}, \mathbf{V})$ begins with a verifier message and let $t := 2 \cdot \mathsf{r}/\eta^2$. Below, $\oplus$ denotes the bitwise binary XOR function. The prover $\mathbf{P}'$ receives as input (x, w) for R, while the verifier $\mathbf{V}'$ receives x explicitly and has oracle access to w. They interact as follows.

- $\mathbf{P}'$: Send $z_1, \ldots, z_t \in \{0,1\}^{\mathsf{r}}$ where $z_i := (z_{i,1}, \ldots, z_{i,\mathsf{k}})$ for $z_{i,j} \in \{0,1\}^{\mathsf{r}_j}$ to the verifier. (where r_j is the number of random bits chosen by $\mathbf{V}$ in round j)
- For $j = 1$ to k:
  1. $\mathbf{V}'$: Choose $\rho_j \leftarrow \{0,1\}^{\mathsf{r}_j}$.
  2. $\mathbf{P}'$: For every $i \in [t]$ send $\pi_{i,j}$. In the honest case $\pi_{i,j} := \mathbf{P}(x, w, \rho_1 \oplus z_{i,1}, \ldots, \rho_j \oplus z_{i,j})$.
- $\mathbf{P}'$: Send $b_1, \ldots, b_t \in \{0,1\}$ as a non-oracle message where $b_i = 1$ if and only if $\mathbf{V}^w(x; \rho_1 \oplus z_{i,1}, \ldots, \rho_{\mathsf{k}} \oplus z_{i,\mathsf{k}}) = 1$.
- $\mathbf{V}'$: Receive $b_1, \ldots, b_t \in \{0,1\}$ as a non-oracle message. Let $S := \{i \in [t] \mid b_i = 1\}$ be the indices where b_i equals 1. Sample $i \leftarrow S$ uniformly at random. Accept if and only if both of the following checks pass:
  1. Check that $|S| > (1 - \mathsf{c} - \eta) \cdot t$.
  2. Query z_i and check that $\mathbf{V}^{w,\pi_{i,1},\ldots,\pi_{i,\mathsf{k}}}(x; \rho_1 \oplus z_{i,1}, \ldots, \rho_{\mathsf{k}} \oplus z_{i,\mathsf{k}}) = 1$ by running $\mathbf{V}$.

Proof of Theorem 9.2. We prove completeness, then soundness, and finally analyze complexity measures. Completeness and soundness are an almost identical as in the proof of Theorem 7.1, edited where appropriate.

Completeness. Fix $(x, w) \in R$. We show that the protocol has perfect completeness when interacting on input (x, w). Consider the vector $\mathbb{x} \in \{0,1\}^{2^{\mathsf{r}}}$ where $\mathbb{x}[(\rho_1, \ldots, \rho_{\mathsf{k}})] = 1$ if and only if $\mathbf{V}^{w,\pi_1,\ldots,\pi_{\mathsf{k}}}(x; \rho_1, \ldots, \rho_{\mathsf{k}}) = 1$ where $\pi_j := \mathbf{P}(x, w, \rho_1, \ldots, \rho_j)$. Since the IOPP $(\mathbf{P}, \mathbf{V})$ has completeness error c, $\mathsf{weight}(\mathbb{x}) \geq 1 - \mathsf{c}$. We augment the definition of good shifts to, rather than being defined with respect to "$+ \mod n$", to a definition with bitwise XOR "$\oplus$". We observe that *Lemma* 4.3 holds also for this definition by an identical proof. By applying Lemma 4.3, with $k = 1$, there exist shifts $(z_1, \ldots, z_t)$ that are good for $\mathbb{x}$, i.e., where:

$$\forall (\rho_1, \ldots, \rho_{\mathsf{k}}) \in \{0,1\}^{\mathsf{r}}, \ \sum_{i \in [t]} (\rho_1, \ldots, \rho_{\mathsf{k}}) \oplus (z_{i,1}, \ldots, z_{i,\mathsf{k}})] \geq (1 - \mathsf{c} - \eta) \cdot t \ ,$$

where, above $z_{i,j} \in \{0,1\}^{\mathsf{r}_j}$. By definition of the bitwise-XOR, we can rewrite this as:

$$\forall (\rho_1, \ldots, \rho_{\mathsf{k}}) \in \{0,1\}^{\mathsf{r}}, \ \sum_{i \in [t]} (\rho_1 \oplus z_{i,1}, \ldots, \rho_{\mathsf{k}} \oplus z_{i,\mathsf{k}})] \geq (1 - \mathsf{c} - \eta) \cdot t \ .$$

In other words, there exist shifts $z_1, \ldots, z_t$ such that for every $\rho_1, \ldots, \rho_{\mathsf{k}}$ there are at least $(1 - \mathsf{c} - \eta) \cdot t$ indices i for which $\mathbf{V}^{w,\pi_1,\ldots,\pi_{\mathsf{k}}}(x; \rho_1 \oplus z_{i,1}, \ldots, \rho_{\mathsf{k}} \oplus z_{i,j}) = 1$ where $\pi_i := \mathbf{P}(x, w, \rho_1 \oplus z_{i,1}, \ldots, \rho_j \oplus z_{i,j})$.

Supposing the honest prover $\mathbf{P}'$ uses these shifts as its first message, the bit vector $b_1, \ldots, b_t$ generated by $\mathbf{P}'$ contains at least $(1 - \mathsf{c} - \eta) \cdot t$ ones. Consequently, $\mathbf{V}'$ will not reject upon reading the vector in Item 1. Moreover, since $\mathbf{P}'$ generates its proof messages honestly with respect to the shifted random strings, for every i sampled by $\mathbf{V}'$, we have $\mathbf{V}^{w,\pi_{i,1},\ldots,\pi_{i,\mathsf{k}}}(x; \rho_1 \oplus z_{i,1}, \ldots, \rho_{\mathsf{k}} \oplus z_{i,j}) = 1$, and so $\mathbf{V}'$ will not reject at Item 2. Thus, the honest prover strategy of sending this series of shifts $z_1, \ldots, z_t$ and then following the protocol as prescribed causes $\mathbf{V}'$ to accept with probability 1.

Soundness. Fix a malicious prover $\tilde{\mathbf{P}}'$ and an input $(x, w) \notin R$. Let $z_1, \ldots, z_t$ be the first message output by $\tilde{\mathbf{P}}'$. For every $j \in [t]$, let X_j be the 0/1 random variable that is equal to 1 if and only if $\mathbf{V}^{w,\pi_1,\ldots,\pi_{\mathsf{k}}}(x; \rho_1 \oplus z_{i,1}, \ldots, \rho_{\mathsf{k}} \oplus z_{i,j}) = 1$ where $\pi_i := \tilde{\mathbf{P}}(\rho_1 \oplus z_{i,1}, \ldots, \rho_j \oplus z_{i,j})$. Let $X \equiv \sum_{j=1}^{t} X_j$ be the random variable representing the number of X_j-s set to 1. Observe that since $\rho_1, \ldots, \rho_{\mathsf{k}}$ are uniform and independent, then for every fixed j so are $\rho_1 \oplus z_{i,1}, \ldots, \rho_{\mathsf{k}} \oplus z_{i,\mathsf{k}}$. It therefore follows by soundness of the IOPP $(\mathbf{P}, \mathbf{V})$ that $\mathbb{E}[X_j] \leq \mathsf{s}$, and so $\mathbb{E}[X] = \mathsf{s} \cdot t$.

We begin by showing that if exactly w of the X_j random variables are set to 1 then the prover manages to convince the verifier with probability $\frac{w}{t \cdot (\alpha - \eta)}$:

Claim 9.6. *For every $w \in \{0, \ldots, t\}$:*

$$\Pr\left[\mathbf{V}' \textit{ accepts} \,\middle|\, X = w \ \right] = \frac{w}{t \cdot (1 - \mathsf{c} - \eta)} \ .$$

Proof. Since $X = \sum_{j \in [t]} X_j = w$, we have that the verifier $\mathbf{V}'$ is able to be convinced by $\tilde{\mathbf{P}}'$ for at w of the indices $i \in [t]$. In order for $\tilde{\mathbf{P}}'$ to cause $\mathbf{V}$ to accept, the weight of the vector $(b_1, \ldots, b_t)$ sent by $\tilde{\mathbf{P}}$ must be at least $1 - \mathsf{c} - \eta$. Consequently, the verifier accepts with probability at most $\frac{w}{t \cdot (1 - \mathsf{c} - \eta)}$: the probability of sampling one of the w indices for which the prover can cause the verifier to accept out of the $(1 - \mathsf{c} - \eta) \cdot t$ bits that must be set to 1. □

Observe that $\mathbb{E}[X] = \sum_{w=0}^{t} w \cdot \Pr[X = w]$. By applying Claim 9.6 we have:

$$\begin{aligned}
\Pr[\mathbf{V}' \textit{ accepts}] &= \sum_{w=0}^{t} \Pr[X = w] \cdot \Pr\left[\mathbf{V}' \textit{ accepts} \,\middle|\, X = w\ \right] \\
&= \sum_{w=0}^{t} \Pr[X = w] \cdot \frac{w}{t \cdot (1 - \mathsf{c} - \eta)} \\
&= \frac{1}{t \cdot (1 - \mathsf{c} - \eta)} \cdot \sum_{w=0}^{t} w \cdot \Pr[X = w] \\
&\leq \frac{\mathsf{s}}{1 - \mathsf{c} - \eta}\ .
\end{aligned}$$

Complexity measures. We analyze the complexity parameters of the new PCPP.

- *Proof length.* The proof length is $t \cdot \mathsf{l} + t = O(\mathsf{l} \cdot \mathsf{r}/\eta^2)$.
- *Queries to input.* The verifier makes $\mathsf{q}_{\mathbb{x}}$ to w during the single invocation of $\mathbf{V}$.
- *Queries to proof.* The verifier makes q_π queries to the messages of $\mathbf{P}'$ during the execution of $\mathbf{V}$, and additionally reads the bit vector $b_1, \ldots, b_t$ and $z_i \in \{0,1\}^{\mathsf{r}}$. Thus the verifier makes at most $\mathsf{q}_\pi + t + \mathsf{r} = \mathsf{q}_\pi + O(\mathsf{r}/\eta^2)$ queries to the proof.
- *Randomness.* The verifier uses $\mathsf{r} + \log t = \mathsf{r} + O(\log(\mathsf{r}/\eta))$ bits of randomness.
- *Verifier running time.* The verifier makes a single invocation $\mathbf{V}$ and otherwise runs in time $O(t)$ for a total running time of $\mathsf{vt} + O(\mathsf{r}/\eta^2)$.

□

Acknowledgments

We are grateful to Ron Rothblum for valuable discussions and for directing us to related work.

Gal Arnon is supported in part by a grant from the Israel Science Foundation (no. 2686/20) and by the Simons Foundation Collaboration on the Theory of Algorithmic Fairness. Shany Ben-David is supported by the Israel Science Foundation (Grant no. 2302/22). Eylon Yogev is supported by an Alon Young Faculty Fellowship, by the Israel Science Foundation (Grant no. 2302/22).

References

[ABCY22] Gal Arnon, Amey Bhangale, Alessandro Chiesa, and Eylon Yogev. "A Toolbox for Barriers on Interactive Oracle Proofs". In: *Proceedings of the 20th Theory of Cryptography Conference*. TCC '22. 2022, pp. 447–466.

[ACFY24] Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. *STIR: Reed–Solomon Proximity Testing with Fewer Queries*. Cryptology ePrint Archive, Paper 2024/390. 2024.

[ACY22a] Gal Arnon, Alessandro Chiesa, and Eylon Yogev. "A PCP Theorem for Interactive Proofs". In: *Proceedings of the 41st Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '22. 2022, pp. 64–94.

[ACY22b] Gal Arnon, Alessandro Chiesa, and Eylon Yogev. "Hardness of Approximation for Stochastic Problems via Interactive Oracle Proofs". In: *Proceedings of the 37th Annual IEEE Conference on Computational Complexity*. CCC '22. 2022, 24:1–24:16.

[ACY23] Gal Arnon, Alessandro Chiesa, and Eylon Yogev. "IOPs with Inverse Polynomial Soundness Error". In: *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*. IEEE, 2023, pp. 752–761.

[AGRR23] Hugo Aaronson, Tom Gur, Ninad Rajgopal, and Ron Rothblum. "Distribution-Free Proofs of Proximity". In: *Electron. Colloquium Comput. Complex.* TR23-118 (2023).

[BBHR18] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. "Fast Reed–Solomon Interactive Oracle Proofs of Proximity". In: *Proceedings of the 45th International Colloquium on Automata, Languages and Programming*. ICALP '18. 2018, 14:1–14:17.

[BCG20] Jonathan Bootle, Alessandro Chiesa, and Jens Groth. "Linear-Time Arguments with Sublinear Verification from Tensor Codes". In: *Proceedings of the 18th Theory of Cryptography Conference*. TCC '20. 2020, pp. 19–46.

[BCGGHJ17] Jonathan Bootle, Andrea Cerulli, Essam Ghadafi, Jens Groth, Mohammad Hajiabadi, and Sune K. Jakobsen. "Linear-Time Zero-Knowledge Proofs for Arithmetic Circuit Satisfiability". In: *Proceedings of the 23rd International Conference on the Theory and Applications of Cryptology and Information Security*. ASIACRYPT '17. 2017, pp. 336–365.

[BCGRS17] Eli Ben-Sasson, Alessandro Chiesa, Ariel Gabizon, Michael Riabzev, and Nicholas Spooner. "Interactive Oracle Proofs with Constant Rate and Query Complexity". In: *Proceedings of the 44th International Colloquium on Automata, Languages and Programming*. ICALP '17. 2017, 40:1–40:15.

[BCGV16] Eli Ben-Sasson, Alessandro Chiesa, Ariel Gabizon, and Madars Virza. "Quasilinear-Size Zero Knowledge from Linear-Algebraic PCPs". In: *Proceedings of the 13th Theory of Cryptography Conference*. TCC '16-A. 2016, pp. 33–64.

[BCL22] Jonathan Bootle, Alessandro Chiesa, and Siqi Liu. "Zero-Knowledge IOPs with Linear-Time Prover and Polylogarithmic-Time Verifier". In: *Proceedings of the 41st Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '22. 2022, pp. 275–304.

[BCS16] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. "Interactive Oracle Proofs". In: *Proceedings of the 14th Theory of Cryptography Conference*. TCC '16-B. 2016, pp. 31–60.

[BGHSV06] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. "Robust PCPs of Proximity, Shorter PCPs, and Applications to Coding". In: *SIAM Journal on Computing* 36.4 (2006), pp. 889–974.

[BKS01] Ziv Bar-Yossef, Ravi Kumar, and D. Sivakumar. "Sampling algorithms: lower bounds and applications". In: *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece*. Ed. by Jeffrey Scott Vitter, Paul G. Spirakis, and Mihalis Yannakakis. ACM, 2001, pp. 266–275.

[BN22] Sarah Bordage and Jade Nardi. "Interactive Oracle Proofs of Proximity to Algebraic Geometry Codes". In: *Proceedings of the 37th Annual IEEE Conference on Computational Complexity*. CCC '22. 2022, 30:1–30:45.

[BV19] Mitali Bafna and Nikhil Vyas. "Imperfect Gaps in Gap-ETH and PCPs". In: *34th Computational Complexity Conference, CCC 2019, July 18-20, 2019, New Brunswick, NJ, USA*. Ed. by Amir Shpilka. Vol. 137. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019, 32:1–32:19.

[BV22] Nir Bitansky and Vinod Vaikuntanathan. "A Note on Perfect Correctness by Derandomization". In: *J. Cryptol.* 35.3 (2022), p. 18.

[Ben+17] Eli Ben-Sasson et al. "Computational integrity with a public random string from quasi-linear PCPs". In: *Proceedings of the 36th Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '17. 2017, pp. 551–579.

[CY20] Alessandro Chiesa and Eylon Yogev. "Barriers for Succinct Arguments in the Random Oracle Model". In: *Proceedings of the 18th Theory of Cryptography Conference*. TCC '20. 2020, pp. 47–76.

[CY21a] Alessandro Chiesa and Eylon Yogev. "Subquadratic SNARGs in the Random Oracle Model". In: *Proceedings of the 41st Annual International Cryptology Conference*. CRYPTO '21. 2021, pp. 711–741.

[CY21b] Alessandro Chiesa and Eylon Yogev. "Tight Security Bounds for Micali's SNARGs". In: *Proceedings of the 19th Theory of Cryptography Conference*. TCC '21. 2021, pp. 401–434.

[DNR04] Cynthia Dwork, Moni Naor, and Omer Reingold. "Immunizing Encryption Schemes from Decryption Errors". In: *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*. Ed. by Christian Cachin and Jan Camenisch. Vol. 3027. Lecture Notes in Computer Science. Springer, 2004, pp. 342–360.

[DR04] Irit Dinur and Omer Reingold. "Assignment Testers: Towards a Combinatorial Proof of the PCP Theorem". In: *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*. FOCS '04. 2004, pp. 155–164.

[EKR04] Funda Ergün, Ravi Kumar, and Ronitt Rubinfeld. "Fast approximate probabilistically checkable proofs". In: *Information and Computation* 189.2 (2004), pp. 135–159.

[FGMSZ89] Martin Fürer, Oded Goldreich, Yishay Mansour, Michael Sipser, and Stathis Zachos. "On Completeness and Soundness in Interactive Proof Systems". In: *Advances in Computing Research* 5 (1989), pp. 429–442.

[GGR18] Oded Goldreich, Tom Gur, and Ron D. Rothblum. "Proofs of proximity for context-free languages and read-once branching programs". In: *Inf. Comput.* 261 (2018), pp. 175–201.

[GGR98] Oded Goldreich, Shafi Goldwasser, and Dana Ron. "Property Testing and its Connection to Learning and Approximation". In: *J. ACM* 45.4 (1998), pp. 653–750.

[GR18] Tom Gur and Ron D. Rothblum. "Non-interactive proofs of proximity". In: *Comput. Complex.* 27.1 (2018), pp. 99–207.

[Gol11] Oded Goldreich. "A Sample of Samplers: A Computational Perspective on Sampling". In: *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation - In Collaboration with Lidor Avigad, Mihir Bellare, Zvika Brakerski, Shafi Goldwasser, Shai Halevi, Tali Kaufman, Leonid Levin, Noam Nisan, Dana Ron, Madhu Sudan, Luca Trevisan, Salil Vadhan, Avi Wigderson, David Zuckerman*. Ed. by Oded Goldreich. Vol. 6650. Lecture Notes in Computer Science. Springer, 2011, pp. 302–332.

[HNY17] Pavel Hubácek, Moni Naor, and Eylon Yogev. "The Journey from NP to TFNP Hardness". In: *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*. Ed. by Christos H. Papadimitriou. Vol. 67. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017, 60:1–60:21.

[KR15] Yael Tauman Kalai and Ron D. Rothblum. "Arguments of Proximity - [Extended Abstract]". In: *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*. Ed. by Rosario Gennaro and Matthew Robshaw. Vol. 9216. Lecture Notes in Computer Science. Springer, 2015, pp. 422–442.

[KSY20] Liran Katzir, Clara Shikhelman, and Eylon Yogev. "Interactive Proofs for Social Graphs". In: *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part III*. Ed. by Daniele Micciancio and Thomas Ristenpart. Vol. 12172. Lecture Notes in Computer Science. Springer, 2020, pp. 574–601.

[Kil92] Joe Kilian. "A note on efficient zero-knowledge proofs and arguments". In: *Proceedings of the 24th Annual ACM Symposium on Theory of Computing*. STOC '92. 1992, pp. 723–732.

[Lau83] Clemens Lautemann. "BPP and the Polynomial Hierarchy". In: *Inf. Process. Lett.* 17.4 (1983), pp. 215–217.

[Mic00] Silvio Micali. "Computationally Sound Proofs". In: *SIAM Journal on Computing* 30.4 (2000). Preliminary version appeared in FOCS '94., pp. 1253–1298.

[Mie09] Thilo Mie. "Short PCPPs verifiable in polylogarithmic time with O(1) queries". In: *Annals of Mathematics and Artificial Intelligence* 56 (3 2009), pp. 313–338.

[NW94] Noam Nisan and Avi Wigderson. "Hardness vs Randomness". In: *Journal of Computer and System Sciences* 49.2 (1994), pp. 149–167.

[Nao89] Moni Naor. "Bit Commitment Using Pseudo-Randomness". In: *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*. Ed. by Gilles Brassard. Vol. 435. Lecture Notes in Computer Science. Springer, 1989, pp. 128–136.

[RR20a] Noga Ron-Zewi and Ron Rothblum. "Local Proofs Approaching the Witness Length". In: *Proceedings of the 61st Annual IEEE Symposium on Foundations of Computer Science*. FOCS '20. 2020, pp. 846–857.

[RR20b] Guy N. Rothblum and Ron D. Rothblum. "Batch Verification and Proofs of Proximity with Polylog Overhead". In: *Theory of Cryptography - 18th International Conference, TCC 2020, Durham, NC, USA, November 16-19, 2020, Proceedings, Part II*. Ed. by Rafael Pass and Krzysztof Pietrzak. Vol. 12551. Lecture Notes in Computer Science. Springer, 2020, pp. 108–138.

[RR22] Noga Ron-Zewi and Ron D. Rothblum. "Proving as Fast as Computing: Succinct Arguments with Constant Prover Overhead". In: *Proceedings of the 54th ACM Symposium on the Theory of Computing*. STOC '22. 2022, pp. 1353–1363.

[RRR16] Omer Reingold, Ron Rothblum, and Guy Rothblum. "Constant-Round Interactive Proofs for Delegating Computation". In: *Proceedings of the 48th ACM Symposium on the Theory of Computing*. STOC '16. 2016, pp. 49–62.

[RS96] Ronitt Rubinfeld and Madhu Sudan. "Robust Characterizations of Polynomials with Applications to Program Testing". In: *SIAM Journal on Computing* 25.2 (1996), pp. 252–271.

[RVW13] Guy N. Rothblum, Salil P. Vadhan, and Avi Wigderson. "Interactive proofs of proximity: delegating computation in sublinear time". In: *Proceedings of the 45th ACM Symposium on the Theory of Computing*. STOC '13. 2013, pp. 793–802.

[Rot24] Ron Rothblum. Private communication. 2024.

[XZZPS19] Tiancheng Xie, Jiaheng Zhang, Yupeng Zhang, Charalampos Papamanthou, and Dawn Song. "Libra: Succinct Zero-Knowledge Proofs with Optimal Prover Computation". In: *Proceedings of the 39th Annual International Cryptology Conference*. CRYPTO '19. 2019, pp. 733–764.

A Proof of Theorem 5.8

Proof. Let $(\mathbf{P}, \mathbf{V})$ be the PCPP obtained by Theorem 5.3 with efficiency parameters which we denote by $(\mathsf{l}_{\mathbf{P}}, \mathsf{q}_{\mathbb{x}}, \mathsf{q}_{\pi}, \mathsf{r}, \mathsf{vt})$ and soundness error of the form $(\alpha - \delta) \cdot \epsilon(\alpha, \eta)$. Applying Theorem 5.6 to $(\mathbf{P}, \mathbf{V})$ with parameters $a := \log^3 n$, we get a new PCPP $(\mathbf{P}', \mathbf{V}')$ with efficiency parameters $(\mathsf{l}'_{\mathbf{P}}, \mathsf{q}'_{\mathbb{x}}, \mathsf{q}'_{\pi}, \mathsf{r}', \mathsf{vt}')$ and soundness error ε', which we compute below.

- *Soundness error:*

$$\begin{aligned}\mathsf{s}(\delta, \alpha, \eta) &= \epsilon\left(\alpha - \frac{\eta}{2}, \frac{\eta}{2}\right) \cdot (\alpha - \delta) \\ &= \frac{\alpha - \delta}{(\alpha - \frac{\eta}{2}) - \frac{\eta}{2}} \\ &= \frac{\alpha - \delta}{\alpha - \eta} \ .\end{aligned}$$

- *Proof length:*

$$\mathsf{l}'_{\mathbf{P}}(n, \eta) = 2 \cdot \log^2 n \cdot \frac{1}{\eta^2} + \frac{n}{\log^3 n} \cdot \mathsf{l}\left(2 \cdot \log n \cdot \frac{1}{\eta^2}, \log^3 n, \frac{\eta}{2}\right) \ .$$

Note that,

$$\begin{aligned}\mathsf{l}\left(2 \cdot \log n \cdot \frac{1}{\eta^2}, \log^3 n, \frac{\eta}{2}\right) &= 2 \cdot \log\left(2 \cdot \log n \cdot \frac{1}{\eta^2}\right) \cdot \log\left(\log^3 n \cdot (2 \cdot \log n \cdot \frac{1}{\eta^2})\right) \cdot \left(\frac{2}{\eta}\right)^2 \\ &= O\left(\log\left(\log n \cdot \frac{1}{\eta}\right) \cdot \log\left(\log n \cdot \frac{1}{\eta}\right) \cdot \frac{1}{\eta^2}\right) \\ &= O\left(\left((\mathrm{loglog}\, n)^2 - \log^2 \eta\right) \cdot \frac{1}{\eta^2}\right) \ .\end{aligned}$$

Therefore,

$$\begin{aligned}\mathsf{l}'(n, \eta) &= 2 \cdot \log^2 n \cdot \frac{1}{\eta^2} + \frac{n}{\log^3 n} \cdot O\left(\left((\mathrm{loglog}\, n)^2 - \log^2 \eta\right) \cdot \frac{1}{\eta^2}\right) \\ &= O\left(\frac{1}{\eta^2} \cdot \log^2\left(\frac{1}{\eta}\right) \cdot \frac{n}{\log^2 n}\right) \ .\end{aligned}$$

- *Queries to input:*

$$\begin{aligned}\mathsf{q}'_{\mathbb{x}}(n,\eta) &= \mathsf{q}_{\mathbb{x}}\left(2\cdot\log n\cdot\frac{1}{\eta^2},\log^3 n,\frac{\eta}{2}\right)\\ &= 2\cdot\log\left(\log^3 n\cdot\left(4\cdot\log n\cdot\frac{1}{\eta^2}\right)\right)\cdot\left(\frac{2}{\eta}\right)^2\\ &= O\left(\log\left(\log n\cdot\frac{1}{\eta}\right)\cdot\frac{1}{\eta^2}\right)\\ &= O\left((\mathrm{loglog}\, n-\log\eta)\cdot\frac{1}{\eta^2}\right)\ .\end{aligned}$$

  Note that,

$$\mathsf{q}_{\mathbb{x}}\left(2\cdot\log n\cdot\frac{1}{\eta^2},\log^3 n,\frac{\eta}{2}\right)=O\left((\mathrm{loglog}\, n-\log\eta)\cdot\frac{1}{\eta^2}\right)\ . \tag{7}$$

- *Queries to proof:*

$$\mathsf{q}'_{\pi}(n,\eta)=\log n\cdot\mathsf{q}_{\mathbb{x}}\left(2\cdot\log n\cdot\frac{1}{\eta^2},\log^3 n,\frac{\eta}{2}\right)+\mathsf{q}_{\pi}\left(2\cdot\log n\cdot\frac{1}{\eta^2},\log^3 n,\frac{\eta}{2}\right)$$

  Note that,

$$\begin{aligned}\mathsf{q}_{\pi}\left(2\cdot\log n\cdot\frac{1}{\eta^2},\log^3 n,\frac{\eta}{2}\right) &= \log\left(2\cdot\log n\cdot\frac{1}{\eta^2}\right)\cdot 2\cdot\log\left(\log^3 n\cdot\left(2\cdot\log n\cdot\frac{1}{\eta^2}\right)\right)\cdot\left(\frac{2}{\eta}\right)^2\\ &= O\left(\log\left(\log n\cdot\frac{1}{\eta}\right)\cdot\log\left(\log n\cdot\frac{1}{\eta}\right)\cdot\frac{1}{\eta^2}\right)\\ &= O\left(\left((\mathrm{loglog}\, n)^2-\log^2\eta\right)\cdot\frac{1}{\eta^2}\right)\ .\end{aligned}$$

  Therefore, by combining the above equations with Equation 7,

$$\begin{aligned}\mathsf{q}'_{\pi}(n,\eta) &= \log n\cdot O\left((\mathrm{loglog}\, n-\log\eta)\cdot\frac{1}{\eta^2}\right)+O\left(\left((\mathrm{loglog}\, n)^2-\log^2\eta\right)\cdot\frac{1}{\eta^2}\right)\\ &\leq O\left(\log n\cdot\left(\mathrm{loglog}\, n-\log^2\eta\right)\cdot\frac{1}{\eta^2}\right)\ .\end{aligned}$$

- *Randomness:*

$$\begin{aligned}\mathsf{r}'(n,\eta) &= \log\left(\frac{n}{\log^3 n}\right)+\mathsf{r}\left(2\cdot\log n\cdot\frac{1}{\eta^2},\log^3 n,\frac{\eta}{2}\right)\\ &= \log n-\log\left(\log^3 n\right)+\log\left(\log^3 n\cdot\left(2\cdot\log n\cdot\frac{1}{\eta^2}\right)\right)\\ &= \log n+\mathrm{loglog}\, n-2\log\eta+1\ .\end{aligned}$$

- *Verifier running time:*

$$\mathsf{vt}'(n,\eta) = \mathsf{vt}\left(2\cdot\log n\cdot\frac{1}{\eta^2},\log^3 n,\frac{\eta}{2}\right) + O\left(\log n\cdot\mathsf{q}_{\mathbb{x}}\left(2\cdot\log n\cdot\frac{1}{\eta^2},\log^3 n,\frac{\eta}{2}\right)\right) \ .$$

Note that,

$$\begin{aligned}\mathsf{vt}\left(2\cdot\log n\cdot\frac{1}{\eta^2},\log^3 n,\frac{\eta}{2}\right) &= O\left(\log(2\cdot\log n\cdot\frac{1}{\eta^2})\cdot\log(\log^3 n\cdot(2\cdot\log n\cdot\frac{1}{\eta^2}))\cdot\left(\frac{2}{\eta}\right)^2\right)\\ &= O\left(\log(\log n\cdot\frac{1}{\eta})\cdot\log(\log n\cdot\frac{1}{\eta})\cdot\frac{1}{\eta^2}\right)\\ &= O\left(\left((\mathrm{loglog}\, n)^2-\log^2\eta\right)\cdot\frac{1}{\eta^2}\right) \ .\end{aligned}$$

Therefore, by combining the above equations with Equation 7,

$$\begin{aligned}\mathsf{vt}'(n,\eta) &= O\left(\log n\cdot(\mathrm{loglog}\, n-\log\eta)\cdot\frac{1}{\eta^2}\right)+O\left(\left((\mathrm{loglog}\, n)^2-\log^2\eta\right)\cdot\frac{1}{\eta^2}\right)\\ &\le O\left(\log n\cdot\left(\mathrm{loglog}\, n-\log^2\eta\right)\cdot\frac{1}{\eta^2}\right) \ .\end{aligned}$$

- *Prover expected running time:*

$$\mathsf{pt}'(n,\alpha,\eta) = O\left(\left(\frac{1}{\eta^2}+\frac{\alpha}{\alpha-\frac{\eta}{2}}\right)\cdot n\cdot\log n+\frac{n}{\log^3 n}\cdot\mathsf{pt}\left(2\cdot\log n\cdot\frac{1}{\eta^2},\log^3 n,\alpha-\frac{\eta}{2},\frac{\eta}{2}\right)\right) \ .$$

Note that,

$$\begin{aligned}&\mathsf{pt}\left(2\cdot\log n\cdot\frac{1}{\eta^2},\log^3 n,\alpha-\frac{\eta}{2},\frac{\eta}{2}\right)\\ &= O\left(\frac{\alpha-\frac{\eta}{2}}{\alpha-\eta}\cdot 2\cdot\log n\cdot\frac{1}{\eta^2}\cdot\log^3 n\cdot\log\left(2\cdot\log n\cdot\frac{1}{\eta^2}\cdot\log^3 n\right)\right)\\ &= O\left(\frac{\alpha-\frac{\eta}{2}}{\alpha-\eta}\cdot\log^4 n\cdot\frac{1}{\eta^2}\cdot\log\left(2\cdot\log^4 n\cdot\frac{1}{\eta^2}\right)\right)\\ &= O\left(\frac{\alpha-\frac{\eta}{2}}{\alpha-\eta}\cdot\log^4 n\cdot\frac{1}{\eta^2}\cdot(\mathrm{loglog}\, n-\log\eta)\right) \ .\end{aligned}$$

Therefore, by combining the above equations,

$$\begin{aligned}\mathsf{pt}'(n,\alpha,\eta) &= O\left(\left(\frac{1}{\eta^2}+\frac{\alpha}{\alpha-\frac{\eta}{2}}\right)\cdot n\cdot\log n+\frac{\alpha-\frac{\eta}{2}}{\alpha-\eta}\cdot\frac{n}{\log^3 n}\cdot\log^4 n\cdot\frac{1}{\eta^2}\cdot(\mathrm{loglog}\, n-\log\eta)\right)\\ &\le O\left(\frac{1}{\eta^2}\cdot\frac{\alpha}{\alpha-\eta}\cdot n\cdot\log n\cdot(\mathrm{loglog}\, n-\log\eta)\right) \ .\end{aligned}$$

where the inequality is since $\frac{\alpha}{\alpha-\eta}\ge\max\left\{\frac{\alpha-\frac{\eta}{2}}{\alpha-\eta},\frac{\alpha}{\alpha-\frac{\eta}{2}}\right\}$.

□

B Hamming to exact Hamming

Our results are defined for the Hamming weight problem where vectors in the language are those with Hamming weight *at least* α. Some of the results in the literature refer to the *exact* Hamming problem, where vectors in the language have Hamming weight exactly α. We show that our results solve both problems with (roughly) the same parameters. This is done by a general reduction from the exact Hamming weight problem to our notion.

We begin with a formal definition of the exact Hamming weight problem.

Definition B.1 (Exact α-Hamming-weight language)**.** *For $\alpha \in [0,1]$, the exact α-Hamming-weight language,* $\mathsf{Exact\text{-}Ham}_\alpha$*, is the set of all bit vectors with Hamming weight* exactly α*:*

$$\mathsf{Exact\text{-}Ham}_\alpha := \{\mathbb{x} \in \{0,1\}^* \mid \mathsf{weight}(\mathbb{x}) = \alpha\} \ .$$

The following lemma shows that IOPPs for Ham_α and $\mathsf{Ham}_{1-\alpha}$ can be combined to generate an IOPP for $\mathsf{Exact\text{-}Ham}_\alpha$.

Lemma B.2. *Suppose there are a perfectly complete IOPP for* Ham_α *and a public-coin perfectly complete IOPP for* $\mathsf{Ham}_{1-\alpha}$*. Then is a public-coin perfectly complete IOPP for* $\mathsf{Exact\text{-}Ham}_\alpha$ *with the following parameters:*

IOPP for Ham_α	
Soundness error	s
Rounds	k
Proof length	l
Queries to vector	$\mathsf{q}_\mathbb{x}$
Queries to proof	q_π
Randomness	r
Verifier running time	vt
Prover expected running time	pt

$+$

IOPP for $\mathsf{Ham}_{1-\alpha}$	
Soundness error	s'
Rounds	k'
Proof length	l'
Queries to vector	$\mathsf{q}'_\mathbb{x}$
Queries to proof	q'_π
Randomness	r'
Verifier running time	vt'
Prover expected running time	pt'

$\longrightarrow$

IOPP for $\mathsf{Exact\text{-}Ham}_\alpha$	
Soundness error	$\frac{1}{2} + \frac{\max\{\mathsf{s}(\delta), \mathsf{s}'(\delta)\}}{2}$
Rounds	$\max\{\mathsf{k}, \mathsf{k}'\}$
Proof length	$\mathsf{l} + \mathsf{l}'$
Queries to vector	$\max\{\mathsf{q}_\mathbb{x}, \mathsf{q}'_\mathbb{x}\}$
Queries to proof	$\max\{\mathsf{q}_\pi, \mathsf{q}'_\pi\}$
Randomness	$\mathsf{r} + \mathsf{r}' + 1$
Verifier running time	$\mathsf{vt} + \mathsf{vt}' + O(1)$
Prover expected running time	$\mathsf{pt} + \mathsf{pt}' + O(1)$

Moreover, if the IOPPs for Ham_α *and* $\mathsf{Ham}_{1-\alpha}$ *are PCPPs, then so is the IOPP for* $\mathsf{Exact\text{-}Ham}_\alpha$.

Proof sketch. We assume without loss of generality that the IOPP verifier for Ham_α and the IOPP verifier for $\mathsf{Ham}_{1-\alpha}$ makes all queries after the last prover message (this can be assumed since the protocol is public coin).

The new IOPP protocol runs the IOPP for Ham_α on vector $\mathbb{x}$ and the IOPP for $\mathsf{Ham}_{1-\alpha}$ on vector $\bar{\mathbb{x}}$ (i.e., $\mathbb{x}$ with all its bits flipped) up to the last prover message, where each execution is done independently and in parallel. For the last step, the new verifier samples bit $b \leftarrow \{0,1\}$ uniformly at random. If $b = 1$, then the new verifier continues the execution of the verifier for Ham_α (by querying the proof and input vector at the appropriate locations) and answer accordingly. If $b = 0$, then the new verifier continues the execution of the verifier for $\mathsf{Ham}_{1-\alpha}$ and answer accordingly. The complexity parameters follow immediately from the construction. We turn to completeness and soundness.

For completeness, if $\mathbb{x} \in \mathsf{Exact\text{-}Ham}_\alpha$ then by definition $\mathsf{weight}(\mathbb{x}) = \alpha$. It follows that $\mathsf{weight}(\mathbb{x}) = \alpha$ and $\mathsf{weight}(\bar{\mathbb{x}}) = 1-\alpha$. Therefore the honest prover can convince the verifier in both IOPPs with probability 1.

For proximity soundness, suppose that $\delta := \Delta(\mathbb{x}, \mathsf{Exact\text{-}Ham}_\alpha) > 0$. We have two cases.

- $\mathsf{weight}(\mathbb{x}) = \alpha - \delta$*:* In this case, the prover will convince the verifier if (1) the new verifier sampled $b = 0$, or (2) the new verifier sampled $b = 1$ and then the Ham_α verifier accepted the proof. Since b is sampled uniformly at random and independently of the Ham_α verifier, we get that the new verifier accepts the proof with probability $\leq \frac{1}{2} + \frac{1}{2} \cdot \mathsf{s}(\delta)$.
- $\mathsf{weight}(\mathbb{x}) = \alpha + \delta$*:* In this case, $\mathsf{weight}(\bar{\mathbb{x}}) = 1-\alpha-\delta$. Therefore, the prover will convince the verifier if (1) the new verifier sampled $b = 1$, or (2) the new verifier sampled $b = 0$ and then the $\mathsf{Ham}_{1-\alpha}$ verifier accepted the proof. Since b is sampled uniformly at random and independently of the Ham_α verifier, we get that the new verifier accepts the proof with probability $\leq \frac{1}{2} + \frac{1}{2} \cdot \mathsf{s}'(\delta)$.

Overall, the prover will convince the verifier with probability at most

$$\begin{aligned} &\max\left\{\frac{1}{2} + \frac{1}{2} \cdot \mathsf{s}(\delta), \frac{1}{2} + \frac{1}{2} \cdot \mathsf{s}'(\delta)\right\} \\ &= \frac{1}{2} + \frac{\max\{\mathsf{s}(\delta), \mathsf{s}'(\delta)\}}{2} \ . \end{aligned}$$

□