Electronic Colloquium on Computational Complexity, Report No. 24 (2024)

Strong Batching for Non-Interactive Statistical Zero-Knowledge

Changrui Mu[*] Shafik Nassar[†] Ron D. Rothblum[‡] Prashant Nalini Vasudevan[§]

February 14, 2024

Abstract

A zero-knowledge proof enables a prover to convince a verifier that $x \in S$, without revealing anything beyond this fact. By running a zero-knowledge proof k times, it is possible to prove (still in zero-knowledge) that k separate instances $x_1, \ldots, x_k$ are all in S. However, this increases the communication by a factor of k. Can one do better? In other words, is (non-trivial) zero-knowledge *batch verification* for S possible?

Recent works by Kaslasi et al. (TCC 2020, Eurocrypt 2021) show that any problem possessing a *non-interactive statistical* zero-knowledge proof (**NISZK**) has a non-trivial statistical zero-knowledge batch verification protocol. Their results had two major limitations: (1) to batch verify k inputs of size n each, the communication in their batch protocol is roughly $\text{poly}(n, \log k) + O(k)$, which is better than the naive cost of $k \cdot \text{poly}(n)$ but still scales linearly with k, and, (2) the batch protocol requires $\Omega(k)$ rounds of interaction.

In this work we remove both of these limitations by showing that any problem in **NISZK** has a *non-interactive* statistical zero-knowledge batch verification protocol with communication $\text{poly}(n, \log k)$.

[*]National University of Singapore. Email: changrui.mu@u.nus.edu.
[†]UT Austin. Email: shafik@cs.utexas.edu.
[‡]Technion. Email: rothblum@cs.technion.ac.il.
[§]National University of Singapore. Email: prashant@comp.nus.edu.sg.

ISSN 1433-8092

Contents

1 Introduction

Zero-knowledge proofs, introduced in the groundbreaking work of Goldwasser, Micali, and Rackoff [GMR89], allow a prover to convince a verifier that a given statement "$x \in S$" is true, without revealing anything beyond its validity. Since their inception, zero-knowledge proofs have had a profound impact on cryptography, complexity theory, and more generally throughout theoretical computer science. Remarkably, these proof-systems are now being used in practical systems as well.[1]

In this work, we study *batch verification* of zero-knowledge proofs: assuming that S has a zero-knowledge proof, can one prove, still in zero-knowledge, that $x_1, \dots, x_k$ all belong to S? The immediate answer to this question is yes – one can simply prove separately that each $x_i \in S$, and the resulting protocol only has a linear in k loss in zero-knowledge error. What we ask however, is whether there is a protocol that can do so with much shorter communication.

We focus on the setting of *statistical zero-knowledge (**SZK**) proofs* – these are proof-systems in which both the soundness and zero-knowledge properties hold in a strong information-theoretic sense. Statistical zero-knowledge proofs are known for most of the commonly studied problems in cryptography and are closely related to constructions of encryption and signature schemes. In particular, the study of batch verification of zero-knowledge proofs is motivated by their enabling of batch proofs that public-keys, ciphertexts or signatures are well-formed, and more generally, for better understanding the rich structure[2] of **SZK**.

The question of batch verification for statistical zero-knowledge proofs was raised in a recent pair of works by Kaslasi *et al.* [KRR$^+$20, KRV21]. These works showed that every problem possessing a *non-interactive* **SZK** proof, has an interactive **SZK** proof-system for batch verification, with non-trivial communication complexity. Recall that non-interactive statistical zero-knowledge proofs (**NISZK**) [SCPY98, GSV99], similarly to their computational counterparts [BFM88], are defined in the *common random string* model, in which all parties have access to a common random string (aka a CRS).[3]

Thus, [KRR$^+$20, KRV21] construct **SZK** batch verification protocols for every problem in **NISZK**. However, their results suffer from some important drawbacks. First, the communication complexity of their protocol is (up to poly-logarithmic factors) $\mathrm{poly}(n) + O(k)$. This is better than the naive protocol which has communication $\mathrm{poly}(n) \cdot k$, but the improvement is still limited. We call a batching protocol achieving such communication a *weak batching protocol*, since, ideally, we would like the dependence on k to be much smaller. Second, while the starting point is a problem that has a *non-interactive* **SZK** proof, the resulting batch protocol is highly interactive, requiring $\Omega(k)$ rounds of interaction, which can be exorbitant for large values of k.

Our Results. In this work, we improve on the results of [KRR$^+$20, KRV21] and construct a *strong* batch verification protocol in which the communication only grows *poly-logarithmically with* k. Furthermore, the resulting protocol is non-interactive (in the CRS model).

[1]See https://zkproof.org and references therein.

[2]See Vadhan's thesis [Vad99] for further background.

[3]As is typically done in the statistical setting (see [GSV99]), we focus on the case that the CRS is a *uniform* random string (rather than the related common "reference" string model, which is sometimes considered in the computational setting).

Theorem 1.1 (Batch Proofs for **NISZK**). *Suppose* $\Pi \in \mathbf{NISZK}$ *and* $k = k(n) \in \mathbb{N}$ *such that* $k(n) \leq 2^{n^{0.01}}$*, where* n *denotes the length of a single instance of* Π*. Then,* $\Pi^{\otimes k}$ *has an* **NISZK** *protocol in which the communication complexity and the length of the common random string is* $\mathrm{poly}\big(n, \log k\big)$*. The completeness, soundness, and zero-knowledge errors are all negligible in* n *and* k*, and the verifier runs in time* $\mathrm{poly}(n, k)$*.*

Here and throughout, $\Pi^{\otimes k}$ denotes the set of k-tuples of inputs $(x_1, \ldots, x_k)$ (of equal length), all of which belong to Π.[4] We remark that a $\mathrm{poly}(n)$ dependence in the communication complexity is inevitable, even when $k = 1$, assuming the existence of a sub-exponentially hard problem in **NISZK** (this follows from known limitations on laconic provers [GH98, GVW02]).

In addition, our protocol is significantly simpler than those in [KRR$^+$20, KRV21]. The main technical observations underlying it are that:

1. Hash functions with bounded independence (specifically 4-wise independence suffices) preserve very specific types of entropies, and,
2. A cascade of such hash functions can be derandomized while still preserving this behaviour.

We elaborate on these points next.

1.1 Technical Overview

In this section, we provide an overview of our construction of batch **NISZK** protocols for any problem in **NISZK**.

Batching Protocol for Permutations. The starting point of our protocol is the same as those of [KRR$^+$20, KRV21]. In particular, [KRR$^+$20] first demonstrate a very simple batching protocol for a specific promise problem in **NISZK**, denoted PERM, of checking whether a given *length-preserving* circuit $C : \{0,1\}^n \to \{0,1\}^n$ is a permutation (these are the YES instances of the problem) or a 2-to-1 function (these are the NO instances). Hereon, we will use the notation $N = 2^n$. Overloading notation, we will use C to also represent the distribution induced by evaluating the circuit on a uniformly random input.

A straightforward **NISZK** protocol for (a single instance of) PERM is to have the CRS contain a random string $r \in \{0,1\}^n$ and the proof is an $x \in \{0,1\}^n$ such that $C(x) = r$. This simple protocol clearly has perfect completeness and soundness error $1/2$ (which can be amplified by repetition). The protocol is perfectly zero-knowledge since the simulator can just sample x at random and output $(x, r = C(x))$. The communication complexity and CRS length are both n, corresponding to the input and output length of C – significantly, these are otherwise independent of the size of C.

This protocol can be easily extended to prove that circuits $C_1, \ldots, C_k$ all belong to PERM as follows. The CRS is again a uniformly random $r \in \{0,1\}^n$, but now the proof is a string $x \in \{0,1\}^n$ such that $(C_k \circ C_{k-1} \circ \cdots \circ C_1)(x) = r$ (here $\circ$ denotes composition of functions). Completeness and zero-knowledge are as before. As for soundness, observe that even if one of the C_i's is 2-to-1, then the composed circuit has an image of size at most $2^n/2$ and so with probability at least $1/2$, the CRS string r is sampled outside the image and so a suitable preimage x does not exist.

[4] In the technical sections we refer to promise problems, in which the formal definition of $\Pi^{\otimes k}$ requires that all k inputs satisfy the promise of Π, see Section 2.

Approximate Injectivity. If the problem PERM were **NISZK**-complete, we would be done – given k instances of any **NISZK** problem Π, we could reduce each of them to an instance of PERM, and run the above batch **NISZK** protocol. Unfortunately, PERM is not known to be **NISZK**-complete, and in fact seems unlikely to be, as it has a *perfect* zero-knowledge proof.

Still, [KRR$^+$20, KRV21] identify a closely related problem that they show to be **NISZK**-complete. This problem is called *Approximate Injectivity*, denoted by $\mathsf{AI}_{L,\delta}$, and is specified by two parameters L and δ.[5] In the $\mathsf{AI}_{L,\delta}$ problem, the instance is a circuit $C : \{0,1\}^n \to \{0,1\}^m$, where $m \geq n$, and:

- YES instances are circuits that are injective on all except a δ-fraction of inputs; that is, for all except $\delta \cdot N$ elements $x \in \{0,1\}^n$, there is no $x' \neq x$ such that $C(x) = C(x')$.
- NO instances are circuits where for all except a δ-fraction of inputs x, there are at least L elements x' such that $C(x) = C(x')$.

They then show that, even for sub-exponential values of L and δ, the problem $\mathsf{AI}_{L,\delta}$ is **NISZK**-complete.

For this overview, it will be convenient to focus first on an exact variant of $\mathsf{AI}_{L,\delta}$ that we will call *Exact Injectivity*, denoted EI_L. Here YES instances are required to be *fully* injective, whereas NO instances are exactly L-to-1. We will describe how to construct a batch **NISZK** protocol for this problem, and later describe how to make this work with just approximate injectivity.

Batch Protocol for Exact Injectivity. We would now like to design an **NISZK** protocol that distinguishes between the YES case where input circuits $C_1, \ldots, C_k : \{0,1\}^n \to \{0,1\}^m$ are all injective, and the NO case where at least one of them is highly non-injective (i.e., it is L-to-1). Simply composing the circuits as we did in the protocol for PERM does not work, even syntactically, since the co-domain of each circuit C_i is larger than the domain of C_{i+1}. Since it is in general unclear how to injectively map the co-domain of the former to the domain of the latter, a natural approach is to perform this mapping at random.

Thus, consider selecting hash functions $h_1, \ldots, h_k : \{0,1\}^m \to \{0,1\}^n$ from a suitable hash function family (e.g., of bounded independence), and applying these in between consecutive applications of the C_i's. This defines the following "chain" circuit:

$$C(x) = (h_k \circ C_k \circ \cdots \circ h_2 \circ C_2 \circ h_1 \circ C_1)(x).$$

Before continuing, looking ahead, a major problem with this approach is that even if each hash function requires merely a constant number of bits to represent it, and if the hash functions are chosen independently, then they must be communicated between the parties, which implies that communication (or CRS length) still scales linearly with k. Getting around this requires an entirely separate set of techniques that we describe towards the end of this section. For now, we ignore this issue and focus on simply constructing a **NISZK** protocol with short prover to verifier communication.

Does Naive Hashing Work? Clearly, if even one of the C_i's is L-to-1, then similarly to the case of PERM, the size of the range of the chain circuit C is at most N/L. So in the NO case we

[5]The **NISZK** hardness of AI also follows from the instance dependent universal one-way hashing constructed in the earlier work of Ong and Vadhan [OV08].

have what we want, but what about the YES case? Can we argue that if $C_1, \dots, C_k$ are all injective then C is also injective (with high probability)? Unfortunately, the answer is negative. Even if the functions h_i were chosen completely at random, by the birthday bound, a very large number of collisions is likely to occur. Let alone further compositions, the expected size of the range of even $(h_1 \circ C_1)$ is only $(1 - (1 - 1/N)^N) \cdot N \approx (1 - 1/e) \cdot N$, which is a constant factor smaller than N. These collisions from the h_i's skew the output distribution of C, and after a large $(k \gg n)$ number of compositions it is unclear whether we will be able to distinguish between the YES and NO cases. Indeed, even if each composition decreases the entropy of the resulting circuit by merely $1/p(n)$ for some polynomial p, after $k \gg p$ steps, the two cases become indistinguishable.

Prior work [KRR+20, KRV21] handled this using a delicate interactive protocol in which information is only gradually revealed both to the prover and the verifier. This gradual process handled each circuit in the chain, in order, via a constant-round interactive protocol. In each round, the collisions coming out of the corresponding hash h_i were "nipped at the bud" (i.e., immediately when they appeared). This approach led to a total of $\Omega(k)$ rounds, which, in particular, also meant that the communication complexity was $\Omega(k)$.

At this point we depart from the [KRR+20, KRV21] approach. We show that as a matter of fact, in the YES case, even though there are many collisions in C, its output distribution still has higher entropy (for a particular notion of entropy) than in the NO case. Given such a gap between YES and NO instances, we then construct an **NISZK** protocol that takes advantage of this. We find the fact that we can control the output distribution of the circuit even after a huge number of compositions (i.e., even if k is sub-exponential in n) quite surprising, and we elaborate on this below.

The Range of C. To see why C stands any chance of having high entropy when all the C_i's are injective, it is useful to think about the size of its range. As observed earlier, even if C_1 is injective and h_1 is a completely random function, the expected size of the range of $(h_1 \circ C_1)$ is, with good probability, close to $(1 - 1/e) \cdot N$. This is computed as follows:

- There are N elements $y \in \{0,1\}^m$ that are in the range of C_1, since it is injective.
- For any $z \in \{0,1\}^n$, the probability there exists a y in the image of C_1 with $h_1(y) = z$ is:
$$\left(1 - (1 - 1/N)^N\right) \approx 1 - 1/e.$$
- The expected number of z's in the range of $(h_1 \circ C_1)$ is thus $\approx (1 - 1/e) \cdot N$.

It can also be shown that the size of this range concentrates around this expectation. The expected size of the range of $(h_2 \circ C_2 \circ h_1 \circ C_1)$ will also be a constant factor smaller than $(1 - 1/e) \cdot N$. If this trend continues, then after k compositions there is no hope that C will have a large range.

Fortunately, it does not. Suppose that for some i and hash functions $h_1, \dots, h_{i-1}$, the function $(C_i \circ h_{i-1} \circ \dots \circ C_1)$ has range of size $S \ll N$. Then, the expected size of the range of $(h_i \circ C_i \circ \cdots)$ when h_i is completely random is computed as:

- For any $z \in \{0,1\}^n$, the probability there exists a y in the image of $(C_i \circ \cdots)$ with $h_i(y) = z$ can be approximated using the Taylor series as:
$$\left(1 - (1 - 1/N)^S\right) \approx \frac{S}{N} - O\left(\frac{S^2}{N^2}\right).$$

- The expected number of z's in the range of $(h_i \circ C_i \circ \cdots)$ is thus $\approx S - O(S^2/N)$.

So if (S^2/N) is smaller than $o(S)$, the size of the range does *not* shrink by a constant factor. In other words, as we keep composing with $(h_2 \circ C_2)$, $(h_3 \circ C_3)$, etc., the size of the range of the composed circuit might quickly drop to $o(N)$, but after that the rate of its decrease goes down, and the size of the range nearly stabilises. By careful arguments along these lines, it can in fact be shown that, with high probability, the size of the range of C is at least $\Omega(N/k)$.

Entropies of C. The arguments so far indicate that if all the C_i's are injective, the range of C is of size at least $\Omega(N/k)$, whereas if even one of them is L-to-1, it is at most N/L. If L is much larger than k (as can be arranged[6]), there is a significant gap between these numbers. While this is encouraging, it is insufficient for an **NISZK** protocol. The problem of distinguishing between circuits whose range is large and ones whose range is small is, in fact, **NP**-hard[7], and thus unlikely to have an **SZK** protocol (let alone an **NISZK** one).

We do, however, know how to construct **NISZK** protocols that distinguish between circuits whose output distribution has large entropy, and circuits with a small range. To help with precision, we define the following notions of entropy of a distribution D. Below, D_x denotes the probability mass placed on the element x by D.

- *Max Entropy:* $H_0(D) = \log |\mathrm{Supp}(D)|$.
- *Shannon Entropy:* $H_1(D) = \sum_x -D_x \log D_x$.

Circuit C having a small range corresponds to $H_0(C)$ being small. Following the work of Goldreich *et al.* [GSV99], we know **NISZK** protocols that can distinguish between circuits with large and small *Shannon* entropies. As the max entropy is always larger than the Shannon entropy, this immediately gives us an **NISZK** protocol that can distinguish between circuits with large Shannon entropy and those with small range.

So if we can show that in the YES case the composed circuit C has high Shannon entropy, we would be done. We show this by proving a stronger statement. We consider the following notion of entropy that is a lower bound on the Shannon entropy[8]:

- *Rényi Entropy:* $H_2(D) = -\log \sum_x D_x^2$.

We show that in the YES case (i.e., when $C_1, \ldots, C_k$ are injective), C actually has large Rényi entropy. We consider this notion of entropy for two reasons. First, the quantity inside the log above (i.e., $\sum_x D_x^2$, aka the collision probability of D) is simpler and easier to work with. The more important reason, however, is the following. Eventually, we are going to derandomise the construction of C so that we don't need $\Omega(k)$ bits to describe all the functions h_i. We show that the derandomization procedure we use more-or-less preserves the Rényi entropy of C. It is not at all clear, however, whether the process preserves the Shannon entropy. So it would not have been sufficient to show that C has high Shannon entropy, and we do need it to have Rényi entropy.

[6]Recall that $k \le 2^{n^{\varepsilon}}$, for some small $\varepsilon > 0$, and as noted earlier, the problem $\mathsf{AI}_{L,\delta}$ is **NISZK**-hard [KRV21] even for some sub-exponential values of L.

[7]This **NP**-hardness can be shown by reducing from SAT. Given a SAT formula ϕ, construct a circuit that takes input (x, y) where $x, y \in \{0,1\}^n$, and outputs 0^n if $\phi(x) = 0$, and y otherwise. If ϕ is not satisfiable, the size of the range of this circuit is 1, whereas if it has even one satisfying assignment, it is 2^n. So an algorithm that can distinguish between these two cases can be used to solve SAT.

[8]Technically, this is only one of a family of measures called Rényi entropies, of which the Shannon entropy is also one. We simply refer to this as Rényi entropy for convenience.

Preservation of Rényi Entropy. Next we describe how we bound the Rényi entropy of C, arguing in terms of its collision probability, denoted by $\mathbf{cp}(C)$. Note that $H_2(D) = -\log(\mathbf{cp}(D))$. For each $i \in [0,k]$, define the following distribution:

$$D_i \equiv (h_i \circ C_i \circ \cdots \circ h_1 \circ C_1)(x),$$

where x is uniformly random. To show that the Rényi entropy of C is high if all the C_i's are injective, we proceed inductively, and show that C's collision probability is small. First, D_0 is simply the uniform distribution over $\{0,1\}^n$, and its collision probability is $1/N$. We then show that for each $i \in [k]$:

$$\mathop{\mathbf{E}}_{h_i}[\mathbf{cp}(D_i)] \leq \mathbf{cp}\big(C_i(D_{i-1})\big) + \frac{1}{N} = \mathbf{cp}(D_{i-1}) + \frac{1}{N},$$

where the inequality follows from the law of total expectation and the pairwise independence of h_i, and the equality follows from the fact that C_i is injective. This shows that the expected collision probability of $D_k \equiv C$ is at most $(k+1)/N$. We then similarly bound the variances of the $\mathbf{cp}(D_i)$'s, and inductively use concentration bounds to show that, with high probability, the collision probability of D_k is not much larger than $O(k/N)$. The above bound on the expectation can be shown as long as the h_i's are drawn from a family that is pairwise-independent. And for the bound on the variance, it is sufficient that they are 4-wise independent.

So the Rényi entropy of C, with high probability over the choice of $h_1, \ldots, h_k$, is not much less than $(n - \log k)$. If at least one of the C_i's was L-to-1, then C has a range of size at most N/L, and thus max entropy of at most $(n - \log L)$. So as long as $k \ll L$, there is a gap between these bounds and we have an **NISZK** protocol that distinguishes between these cases. For any $k < 2^{n^{o(1)}}$, there is a setting of $L \gg k$ and δ for which $\mathsf{AI}_{L,\delta}$ is **NISZK**-hard, and so we can support batching of k instances of any **NISZK** problem with this approach.

Dealing with Approximate Injectivity. So far, however, we have ignored the fact that in the actual **NISZK**-complete problem $\mathsf{AI}_{L,\delta}$, the circuits are only *approximately* injective or L-to-1. In the NO case, the approximation is not an issue as it only slightly increases the size of the range of the circuit. In the YES case, however, we need to be careful.

To be more precise, recall that YES instances of $\mathsf{AI}_{L,\delta}$ are circuits where up to a δ fraction of inputs may not be mapped injectively. When this happens, the relation $\mathbf{cp}\big(C_i(D_{i-1})\big) = \mathbf{cp}(D_{i-1})$ that we used to inductively bound the collision probabilities of D_i breaks down – composition with C_i does not necessarily preserve collision probabilities any more. For instance, suppose C_1 is such that it maps $(1-\delta)N$ inputs injectively, and maps all of the remaining δN inputs to 0^m. The collision probability of $C_1(x)$ is now at least δ^2, which could already be much larger than $O(k/N)$ and $O(L/N)$.

However, while the collision probability of $C_1(x)$ is not small, it is, in fact, close to another distribution whose collision probability is small. Consider a function $\hat{C}_1$ that satisfies the following two properties:

- $\hat{C}_1 : \{0,1\}^n \to \{0,1\}^m$ is injective.
- For any x that C_1 maps injectively, $\hat{C}_1(x) = C_1(x)$.

Fix any such function. Note that the statistical distance between $C_1(x)$ and $\hat{C}_1(x)$ is at most δ – the probability mass from all the inputs on which C_1 and $\hat{C}_1$ might disagree. And also, the collision probability of $\hat{C}_1(x)$ is $1/N$, as $\hat{C}_1$ is injective.

Recall that our earlier approach was to show that the collision probability of D_k is small. While we cannot hope for this any more following the above observations, we may still endeavour to show that D_k is *close to* a distribution whose collision probability is small. This would aso be sufficient for constructing an **NISZK** protocol, using a simple reduction to the Statistical Difference from Uniform (SDU) problem (also complete for **NISZK** [GSV99]) by hashing and using the Leftover Hash Lemma.

So far using the closeness of C_1 and $\hat{C}_1$, for any h_1, we can show the following bound on statistical distance from the data processing inequality:

$$\Delta(h_1 \circ C_1, h_1 \circ \hat{C}_1) \leq \delta.$$

So $D_1 \equiv (h_1 \circ C_1)$ is indeed close to a distribution whose collision probability is small. To take this argument further, simlarly define $\hat{C}_i$ for the other circuits C_i, and define the following corresponding distributions:

$$\hat{D}_i \equiv (h_i \circ \hat{C}_i \circ \cdots \circ h_1 \circ \hat{C}_1)(x).$$

We have shown above that for any choice of h_1, the distributions D_1 and $\hat{D}_1$ are close. We would like to argue that for any choices of $h_1, \ldots, h_k$, the distributions D_k and $\hat{D}_k$ are close. It is not straightforward to argue this for further compositions, however. Ideally, we would like to also say, for instance, that for any h_1 and h_2, the distributions of $(h_2 \circ C_2 \circ h_1 \circ \hat{C}_1)$ and $(h_2 \circ \hat{C}_2 \circ h_1 \circ \hat{C}_1)$ are close, to enable a hybrid argument where we slowly replace the C_i's with the $\hat{C}_i$'s. This is, however, not true. Consider an h_1 that maps all inputs to an x on which C_2 is not injective. Then, the distance between $(C_2 \circ h_1 \circ \hat{C}_1)$ and $(\hat{C}_2 \circ h_1 \circ \hat{C}_1)$ can be very large.

Pathological cases like this can be avoided if the distribution of $(h_1 \circ \hat{C}_1)$ has high entropy. Roughly, if this distribution has high entropy, then it cannot place too much probability mass on the elements on which C_2 and $\hat{C}_2$ differ. Observe that the distance between $(C_2 \circ h_1 \circ \hat{C}_1)$ and $(\hat{C}_2 \circ h_1 \circ \hat{C}_1)$ is upper bounded by this probability mass. Using such arguments, we can show that if $\hat{D}_1 \equiv (h \circ \hat{C}_1)$ has high Rényi entropy, then the distance between D_2 and $\hat{D}_2$ is small.

Note that for any choice of $h_1, \ldots, h_k$, the entropy of any $\hat{D}_i$ is at least that of $\hat{D}_k$, as entropy cannot be increased by composition with deterministic functions. We can then proceed inductively to show that for any hash functions $h_1, \ldots, h_k$ for which $\hat{D}_k$ has high Rényi entropy, the distance between D_k and $\hat{D}_k$ is small.

That is, for any such choice of hash functions (which happens with high probability), C is close to a distribution that has high Rényi entropy. This implies, in particular, that with high probability C will be close to a distribution that has high Rényi entropy. This latter statement seems sufficient for our purposes, but is not. We will actually need the former stronger statement to perform the derandomization of C as discussed next.

Derandomizing the Reduction. What we have so far is a randomized reduction from $\mathsf{AI}_{L,\delta}^{\otimes k}$ to a problem in which the YES instances are circuits that are close to having high Rényi entropy, and NO instances are circuits that have low max entropy. As noted earlier, although throughout the previous discussion we have assumed that the hash functions $h_1, \ldots, h_k$ are truly random, it suffices to use 4-wise independent hash functions to bound the collision probability. This still yields a reduction that uses a lot of randomness. Namely, since we need to sample k independent 4-wise independent hash functions, the randomness grows linearly with k. Recall that our goal is to construct an **NISZK** protocol with a CRS of size $\mathrm{poly}(n, \log k)$. In our protocol, both the verifier and prover run the reduction using randomness from the CRS, therefore we cannot afford to sample k independent hash functions for this reduction.

It is natural to try to reduce the randomness by using correlated hash functions, but one has to be extremely careful as in each step we apply the hash function to a potentially correlated input.

We explain how to overcome this problem in the exact injectivity case, since that captures the main idea of the derandomization. We recall that our reduction outputs a chain

$$C(x) = (h_k \circ C_k \circ \cdots \circ h_2 \circ C_2 \circ h_1 \circ C_1)(x).$$

alternating between the input circuits and the random hash functions. We can view the description of the hash functions $h_1, \ldots, h_k$ as additional input to the circuit C and denote

$$C_{h_1,\ldots,h_k}(x) = (h_k \circ C_k \circ \cdots \circ h_2 \circ C_2 \circ h_1 \circ C_1)(x).$$

Completeness of our protocol relies on the guarantee that if all of the circuits $C_1, \ldots, C_k$ are injective then with high probability over $h_1, \ldots, h_k$, the circuit $C_{h_1,\ldots,h_k}$ would have a small collision probability.

Once an input x is fixed, the circuit C can be modeled as a small-width *Read Once Branching Program* (ROBP), with $h_1, \ldots, h_k$ as the inputs. The key observation is that the collision probability of C can therefore also be computed using a small-width ROBP. Hence, we can use Nisan's [Nis92] pseudorandom generator (PRG) for ROBP to sample the $h_1, \ldots, h_k$, while nearly preserving the collision probability, and thus keeping the same guarantee even when sampling pseudorandom $h_1, \ldots, h_k$. We remark that here we crucially use the fact that the collision probability can be computed via a local process (namely sampling two inputs and checking for a collision). This does not seem to be the case for other notions of entropy (e.g., Shannon entropy), for which we do not know a similar derandomization.

In more detail, for any two inputs x_1, x_2, we can define a ROBP $M_{x_1,x_2}(h_1, \ldots, h_k)$ of length k and width 2^{2n} that outputs 1 if and only if $C_{h_1,\ldots,h_k}(x_1) = C_{h_1,\ldots,h_k}(x_2)$. The collision probability can thus be written as

$$\mathbf{cp}(C_{h_1,\ldots,h_k}) = \Pr_{x_1,x_2 \leftarrow \{0,1\}^n}\left[M_{x_1,x_2}(h_1, \ldots, h_k) = 1\right].$$

Using the linearity of expectation, the expected collision probability over the choice of $h_1, \ldots, h_k$ is

$$\mathop{\mathbf{E}}_{h_1,\ldots,h_k}\left[\mathbf{cp}(C_{h_1,\ldots,h_k})\right] = \mathop{\mathbf{E}}_{x_1,x_2 \leftarrow \{0,1\}^n}\left[\Pr_{h_1,\ldots,h_k}\left[M_{x_1,x_2}(h_1, \ldots, h_k) = 1\right]\right].$$

If we employ a PRG for ROBP, with error ε, the collision probability increases by at most ε more than if $h_1, \ldots, h_k$ were sampled uniformly at random. Since the seed length in Nisan's PRG only depends logarithmically on ε, we can afford to use $\varepsilon = 2^{-\Omega(n)}$ and so the collision probability is indeed preserved up to very small factors. Thus, using Nisan's PRG reduces the seed length to $\mathrm{poly}(n, \log k)$, as desired.

The Protocol. To summarize, given k instances of any **NISZK** problem Π, both the verifier and the prover first reduce the instances to $\mathsf{AI}_{L,\delta}$ instances $C_1, \ldots, C_k : \{0,1\}^n \to \{0,1\}^m$. Then they utilize $\mathrm{poly}(n, \log k)$ bits from CRS as the seed for Nisan's PRG. The output of the PRG is then used to sample 4-wise independent hash functions, denoted as $h_1, \dots, h_k : \{0,1\}^m \to \{0,1\}^n$. These functions are used to construct the chain circuit $C(x) = (h_k \circ C_k \circ \cdots \circ h_2 \circ C_2 \circ h_1 \circ C_1)(x)$.

In the YES case (i.e. when all instances are YES instances of Π), with probability all but negligible in n, k, there exists a $\hat{C}$ such that:

- $\hat{C}$ has high Rényi entropy.
- The distribution of C is very close to that of $\hat{C}$.

As a consequence, C can be reduced to a YES instance of SDU by hashing its output with an appropriate pairwise-independent hash function. In the NO case (i.e. some instances are NO instances of Π), the max entropy $H_0(C)$ will be small. As a result, applying the same reduction on C will yield a NO instance of SDU.

The prover and verifier then run the **NISZK** protocol for SDU on this instance. This uses an additional poly(n) bits from the CRS and poly(n) bits of communication. Here we crucially use the fact that the communication of the **NISZK** protocol for SDU only depends on the input/output size of the circuit, rather than the size of its description $\big($as in our case the former is poly(n) whereas the latter is $k \cdot \text{poly}(n)\big)$. The negligible error associated with the reduction is incorporated into the completeness error of the protocol.

1.2 Related Works

The two closest relevant works, already mentioned above are [KRR+20, KRV21], where the former constructed an *honest-verifier* **SZK** protocol for batch verification of **NISZK**, whereas the latter constructed a *malicious-verifier* (and public-coin) protocol. We remark that our **NISZK** protocol can be transformed into an interactive public-coin (malicious verifier) **SZK** protocol using standard transformations [GSV98].

If one drops the zero-knowledge requirement and merely strives for short communication, batch verification is possible for every problem in **NP** (or more generally **PSPACE**), using the **IP** = **PSPACE** Theorem [LFKN92, Sha92]. That protocol however has an exponential-time prover. Reingold, Rothblum and Rothblum [RRR21, RRR18, RR20] constructed batch verification protocol for every problem in **UP** (i.e., **NP** problems in which YES instances have a unique witness) in which the honest prover runs in polynomial-time given the witnesses. Curiously, this line of work also started with a protocol achieveing weak batching [RRR21] (i.e., with an additive linear dependence on k) and gradually improved to a poly-logarithmic dependence on k [RRR18, RR20].

A separate and exciting line of work has construted non-interactive *computationally sound* batch verification protocols for all of **NP** (aka batch arguments or BARGs for short) [BHK17, CJJ21a, KVZ21, CJJ21b, WW22, DGKV22, PP22, KLVW23, CGJ+23]. In contrast to the **NISZK** setting, these results focus on protocols for all of **NP** but only offer computational soundness and rely on unproven cryptographic assumptions such as LWE. A recent work by Bitansky *et al.* [BKP+23] has shown that different notions of batch proofs automatically yield hiding properties such as witness indistinguishibilty and, under suitable cryptographic assumptions, even full fledged zero-knowledge.

Lastly, we mention the work of Goel *et al.* [GHKS23] who construct efficient zero-knowledge proofs for *disjunctions*, whereas batch verification can be viewed as zero-knowledge proofs for *conjuctions*; and the work of Brakerski *et al.* [BBK+23] who construct computationally sound protocols for *monotone policies* within **NP**.

1.3 Discussion and Open Problems

Theorem 1.1 introduces a *non-interactive* batching verification protocol for the **NISZK** class, achieving substantial communication efficiency compared to independent executions. This is an encouraging indication for the possibility of batch verification in zero-knowledge proofs. Below are some natural open problems for future investigation:

1. Theorem 1.1 gives a *non-interactive* batching verification protocol for problems in **NISZK**. The most pressing open question is whether a similar result holds for **SZK** – namely, for every $\Pi \in$ **SZK** does there exist an **SZK** proof for $\Pi^{\otimes k}$ with communication $\mathrm{poly}(n, \log k)$? Or, alternatively, one that features less strigent, yet non-trivial, communication such as sub-linear dependence on k?

2. As highlighted in [KRR+20], one avenue of research focuses on prover efficiency. It is known that problems in **SZK** $\cap$ **NP** have **SZK** protocol where the prover is efficient given an **NP**-witness [NV06]. All the current batch protocols for **NISZK** proceed by reducing to **NISZK**-complete problems and thus do not preserve this efficiency. Is it possible to construct batch protocols even for **NISZK** $\cap$ **NP** that preserve prover efficiency?

3. Is it possible to improve the multiplicative overhead in our construction to be a fixed constant? That is, can we achieve communication $O(c)+\mathrm{polylog}(n, k)$, where c is the communication for a single instance? This might necessitate avoiding the complete problems, since the reduction introduces a polynomial overhead (or, alternatively, achieving the result only for a limited class of problems).

   Pushing things even further, can we push the constant to be close to 1 (aka a "rate-1" batch-proof)? Results of this flavor have been recently achieved in the computational setting [PP22, DGKV22].

4. Our protocol shows an efficient closure property of **NISZK** for conjunctions. What about more general efficient forms of closure: given a formula $\phi : \{0,1\}^k \to \{0,1\}$, does there exist an **SZK** protocol for $\phi(b_1, \dots, b_k)$, where $b_i = 1 \leftrightarrow x_i \in S$, with sublinear in k communication?

2 Preliminaries

For any $N \in \mathbb{N}$, we denote the set of numbers $\{1, \dots, N\}$ by $[N]$. For convenience, we may write a boolean circuit C with n input bits and m output bits as $C : [N] \to [M]$ where $N = 2^n, M = 2^m$. For any circuit $C : [N] \to [M]$ and any set $S \subseteq [N]$, we denote by $C(S)$ the set of images of inputs in S, that is $C(S) = \{C(x) : x \in S\}$. For an element $y \in [M]$, we denote by $C^{-1}(y)$ the set of preimages of y. For any set S, we denote by U_S the uniform distribution over S. For any positive integer n, we denote by U_n the uniform distribution of $\{0,1\}^n$.

A *promise problem* is a pair $\Pi = (Y, N)$ of disjoint sets (i.e., $Y \cap N = \emptyset$). We use $x \in \Pi$ to denote that $x \in Y \cup N$ and say that x satisfies the promise. We denote by $\texttt{YES}(\Pi) = Y$ and refer to this set as the "YES" instances and by $\texttt{NO}(\Pi) = N$ the "NO" instances.

Definition 2.1. *Let Π be a promise problem, and let $k = k(n) \in \mathbb{N}$. We define the promise problem $\Pi^{\otimes k}$ where*

$$\texttt{YES}\left(\Pi^{\otimes k}\right) = \left\{(x_1, \dots, x_k) \in \left(\texttt{YES}(\Pi)\right)^k : |x_1| = \dots = |x_k|\right\}$$

and

$$\mathtt{NO}\big(\Pi^{\otimes k}\big) = \Big\{(x_1, \ldots, x_k) \in \Pi^k \;:\; |x_1| = \cdots = |x_k|\Big\} \Big\backslash \mathtt{YES}\big(\Pi^{\otimes k}\big).$$

2.1 Probability Theory Background

Lemma 2.2 (Chebyshev's inequality)**.** *Let X be a random variable. Then, for every $\alpha > 0$:*

$$\Pr\big[|X - \mathbf{E}[X]| \geq \alpha\big] \leq \frac{\mathbf{Var}[X]}{\alpha^2}.$$

Definition 2.3 (Statistical Distance)**.** *The* statistical distance *between two distributions X and Y over a finite domain D is defined as*

$$\Delta(X, Y) = \max_{S \subseteq U}(X(S) - Y(S)) = \frac{1}{2}\sum_{u \in U}|X(u) - Y(u)|.$$

Fact 2.4. *Let D be a domain and X be a distribution over D. If $|\mathrm{Supp}(X)| < \delta \cdot |D|$ then $\Delta(X, U_D) > 1 - \delta$.*

Definition 2.5. *The* collision probability *of a distribution X is defined as $\mathbf{cp}(X) = \Pr_{x,x' \leftarrow X}[x = x']$.*

Definition 2.6. *Consider a random variable X over domain D. For any $x \in D$, denote by $p_x = \Pr[X = x]$. We recall the following notions of entropy of X:*

- *Max Entropy: $H_0(X) = \log(|\{x \mid p_x \neq 0\}|)$.*
- *Shannon Entropy: $H_1(X) = -\sum_{x \in D} p_x \log(p_x)$.*
- *Renyi Entropy: $H_2(X) = -\log\left(\sum_{x \in D} p_x^2\right) = -\log(\mathbf{cp}(X))$.*

Technically, all of the above (as well as the notion of "min-entropy" which we do not use in this work) are usually referred to as Renyi entropies of various orders. For convenience, we use this term only for H_2 and the terms above for the others.

Fact 2.7. *For any random variable X, we have: $H_2(X) \leq H_1(X) \leq H_0(X)$.*

2.2 Hash Functions with Bounded Independence

Definition 2.8. *(ℓ-wise Independent Hash Functions). For $\ell = \ell(n) \in \mathbb{N}$ and $m = m(n) \in \mathbb{N}$, a family of functions $F = (F_n)_{n \in \mathbb{N}}$, where $F_n = \{f : \{0,1\}^m \to \{0,1\}^n\}$ is called ℓ-wise independent if for every $n \in \mathbb{N}$ and every ℓ distinct domain elements $x_1, x_2, \ldots, x_\ell \in \{0,1\}^m$, and every $y_1, y_2, \ldots, y_\ell \in \{0,1\}^n$, it holds that:*

$$\Pr_{f \overset{\$}{\leftarrow} F_n}\left[\bigwedge_{i=1}^{\ell} f(x_i) = y_i\right] = 2^{-\ell \cdot n}.$$

Lemma 2.9 (See, e.g., [Vad12, Section 3.5.5])**.** *For every $\ell = \ell(n) \in \mathbb{N}$ and $m = m(n) \in \mathbb{N}$, there exists a family of ℓ-wise independent hash functions $F(\ell) = \{f : \{0,1\}^m \to \{0,1\}^n\}$ where a random function from $F_{n,m}$ can be selected using $O(\ell \cdot \max(n, m))$ random bits, and given a description of $f \in F(\ell)$ and $x \in \{0,1\}^m$, the value $f(x)$ can be computed in time $\mathrm{poly}(n, m, \ell)$.*

Lemma 2.10 (Leftover Hash Lemma [HILL99], see also [Vad12, Section 6.2])**.** *For any polynomial $k = k(n)$ and $\varepsilon = \varepsilon(n) \in (0,1)$, if $\mathcal{H} = \{h : \{0,1\}^n \to \{0,1\}^m\}$ is a family of pairwise independent hash functions such that $m = k - 2\log(1/\varepsilon)$, then for any distribution X over $\{0,1\}^n$ such that $\mathbf{cp}(X) < 2^{-k}$ it holds that*

$$\Delta\Big(\big(H, H(X)\big), \big(H, U_m\big)\Big) \leq \varepsilon,$$

where H distributed uniformly over $\mathcal{H}$.

Remark 2.11. *The Leftover Hash Lemma is typically described with respect to sources that have large* min-entropy*. However, examining the proof given in [Vad12, Section 6.2] shows that an upper bound on the collision probability suffices.*

3 Non-Interactive Statistical Zero-knowledge

In this section, we present the formal definitions of **NISZK** and some of its complete problems that are useful in our work. In Section 3.2, we define a new problem that is more directly useful, and prove that it is complete for **NISZK**.

Definition 3.1 (**NISZK**)**.** *Let $c = c(n) \in [0,1]$, $s = s(n) \in [0,1]$ and $z = z(n) \in [0,1]$. A* non-interactive statistical zero-knowledge proof (**NISZK**) *with completeness error c, soundness error s and zero-knowledge error z for a promise problem Π, consists of a probabilistic polynomial-time verifier V, a computationally unbounded prover P and a polynomial $\ell = \ell(n)$ such that the following properties hold:*

- ***Completeness:*** *For any $x \in \mathtt{YES}_n(\Pi)$:*

  $$\Pr_{r \leftarrow \{0,1\}^{\ell(|x|)}}[\mathsf{V}(x, r, \pi) \text{ accepts}] \geq 1 - c(n),$$

  where $\pi = \mathsf{P}(x, r)$.

- ***Soundness:*** *For any $x \in \mathtt{NO}_n(\Pi)$:*

  $$\Pr_{r \leftarrow \{0,1\}^{\ell(|x|)}}[\exists \pi^* \text{ s.t. } \mathsf{V}(x, r, \pi^*) \text{ accepts}] \leq s(n),$$

- ***Zero Knowledge**: There exists a probabilistic polynomial-time algorithm* Sim *(called the simulator) such that for any $x \in \mathtt{YES}_n(\Pi)$:*

  $$\Delta\Big(\big(U_\ell, \mathsf{P}(x, U_\ell)\big), \mathsf{Sim}(x)\Big) \leq z(n),$$

  where U_ℓ denotes a random variable distributed uniformly over $\{0,1\}^{\ell(n)}$.

Unless otherwise specified, we assume by default that $c(\cdot), s(\cdot)$ and $z(\cdot)$ are negligible in the input size, and say that Π has an **NISZK** protocol if the latter holds. We further use **NISZK** to denote the class of all such promise problems.

We note that parallel repetition of **NISZK** reduces the completeness and soundness errors at an exponential rate, while increasing the zero-knowledge error at only a linear rate.[9]

[9]We remark that this property does not hold for interactive zero-knowledge proofs [FS90, GK96]. The reason that it works for **NISZK** is that in **NISZK** the verifier cannot cheat as there is no interaction [BSMP91].

3.1 Complete Problems

We recall some known complete problems for **NISZK** that are useful in our work.

Definition 3.2 (Statistical Difference from Uniform [GSV99]). *The* Statistical Difference from Uniform *problem, denoted* SDU*, is a promise problem defined by the following sets:*

$$\begin{aligned} \texttt{YES}_n(\mathsf{SDU}) &= \{\textit{circuit } C \,:\, \Delta(C, U_n) < 1/n\}\,; \\ \texttt{NO}_n(\mathsf{SDU}) &= \{\textit{circuit } C \,:\, \Delta(C, U_n) > 1 - 1/n\}\,, \end{aligned}$$

where C is a circuit that outputs n bits. The size *of an instance C is its output length n.*

Goldreich *et al.* [GSV99] showed that SDU is **NISZK**-complete, as shown by the following lemma.

Lemma 3.3 (SDU is **NISZK**-complete [GSV99]). *The promise problem* SDU *is complete for* **NISZK**. *Moreover, the* **NISZK** *protocol for* SDU *only needs black-box access to the instance circuit; and for a parameter s that is any polynomial in the input and output sizes of the circuit in the instance, there are such protocols with the following properties:*

- *the communication complexity and the length of the common random string are* $\mathrm{poly}(s)$*,*
- *the completeness, soundness, and zero-knowledge errors are* 2^{-s}*.*

We note that the "moreover" part is not stated explicitly in [GSV99], but follows by examining their proof.

Definition 3.4 (Approximate Injectivity [KRR+20, KRV21]). *For any $L = L(n) \in \mathbb{N}$ and any $\delta = \delta(n) \in [0,1]$, the* Approximate Injectivity problem $\mathsf{AI}_{L,\delta}$ *is a promise problem defined as follows over circuits taking n bits of input and outputting $3n$ bits:*

$$\begin{aligned} \texttt{YES}_n(\mathsf{AI}_{L,\delta}) &= \left\{\textit{circuit } C \,:\, \Pr_{x \leftarrow \{0,1\}^n}\left[\left|C^{-1}(C(x))\right| > 1\right] \le \delta\right\}; \\ \texttt{NO}_n(\mathsf{AI}_{L,\delta}) &= \left\{\textit{circuit } C \,:\, \Pr_{x \leftarrow \{0,1\}^n}\left[\left|C^{-1}(C(x))\right| < L\right] \le \delta\right\}. \end{aligned}$$

Lemma 3.5 (AI is **NISZK**-hard [KRR+20, KRV21], see also [OV08, Theorem 2]). *For any $L(n) < 2^{n^{0.1}}$ and non-increasing $\delta(n) > 2^{-n^{0.1}}$, the problem $\mathsf{AI}_{L,\delta}$ is* **NISZK***-hard.*

Remark 3.6. *In [KRR+20, KRV21], the definition of the* AI *problem does not restrict the output length of the circuits to $3n$ as Definition 3.4 does. This restricted version, however, is equivalent in complexity to the unrestricted version. An instance $C : \{0,1\}^n \to \{0,1\}^m$ of the unrestricted* AI *problem can be reduced to an instance $\hat{C}$ of the restricted version as follows:*

- *If $m > 3n$, $\hat{C}$ takes as input $(x, y) \in \{0,1\}^{n+(m-3n)/2}$ and outputs $(C(x), y)$.*
- *If $m < 3n$, $\hat{C}(x)$ outputs $C(x)$ padded with $(3n - m)$ zeroes.*

Neither of these transformations change its membership in $\texttt{YES}(\mathsf{AI})$ *or* $\texttt{NO}(\mathsf{AI})$*, and the resulting circuits are of size $O(m + n)$ larger than that of C.*

3.2 Smooth Entropy Approximation

We start by defining smoothened versions of the various entropy measures defined in Section 2, again slightly modified from usual convention to fit our application.

Definition 3.7 (Smooth Entropy [RW04]). *For any $\varepsilon \geq 0$, the* ε-smooth Rényi entropy *of a random variable X is defined as:*

$$H_2^{\varepsilon}(X) = \max_{Y \in \mathcal{B}_{\varepsilon}(X)} H_2(Y),$$

where $\mathcal{B}_{\varepsilon}(X)$ is the set of all distributions within statistical distance ε of X.

We define the following variant of the Entropy Approximation problem [GSV99] using the smooth Rényi and max entropies. Similar problems can be defined with other entropy measures as well [DGRV11]. We show in Lemma 3.9 below that this problem has an **NISZK** protocol, and that it is, in fact, **NISZK**-complete, as stated in Theorem 3.10.

Definition 3.8 (Smooth Entropy Approximation). *For any $\varepsilon = \varepsilon(n) \in [0,1]$, the* ε-Smooth Entropy Approximation *problem, denoted by $\mathsf{SEA}_{\varepsilon}$, is a promise problem defined by the following sets:*

$$\begin{aligned} \mathtt{YES}_n(\mathsf{SEA}_{\varepsilon}) &= \{(C,k) \mid H_2^{\varepsilon}(C) \geq k+1\}\,; \\ \mathtt{NO}_n(\mathsf{SEA}_{\varepsilon}) &= \{(C,k) \mid H_0(C) \leq k-1\}\,, \end{aligned}$$

where C is a circuit that takes n bits as input and outputs $m \leq 3n$ bits, and k is a positive real number that is at most the output length of C. The input and output sizes of an instance (C,k) refer to the input and output lengths of C, respectively.

Lemma 3.9 (**NISZK** Protocol for SEA). *Consider any $m = m(n)$ and $\varepsilon = \varepsilon(n)$ such that $\varepsilon(n) < o\left(1/\max(n,m) \cdot \log^2 m\right)$. Then, $\mathsf{SEA}_{\varepsilon}$ has an* **NISZK** *protocol where, for any instance (C,k) with input and output lengths n and m, respectively, the communication complexity and the length of the common random string are $\mathrm{poly}(n,m)$. The completeness, soundness, and zero-knowledge errors of this protocol are all $2^{-\Omega(\max(n,m))}$.*

Proof. We show a reduction from SEA to SDU that preserves the input and output lengths of the circuit up to a $\mathrm{poly}(n,m)$ blowup. The rest follows from Lemma 3.3. For any m and $k \leq m$, let $H_{m,k} = \left\{h : \{0,1\}^m \to \{0,1\}^k\right\}$ be the pairwise-independent family of hash functions promised by Lemma 2.9, where each hash function is described by $O(\max(m,k)) = O(m)$ bits.

Consider an instance (C,k) of $\mathsf{SEA}_{\varepsilon,g}$ where the input length of C is n and its output length is m. Construct the circuit C' that corresponds to $20\log m$ copies of C evaluated independently. Its input length is $n' = n \cdot (20\log m)$, and its output length is $m' = m \cdot (20\log m)$. Similarly, let $k' = k \cdot (20\log m)$. The reduction, on input (C,k), outputs a circuit $\widehat{C}$ that works as follows:

- It takes as input a description h of a hash function in $H_{m',k'}$ and an $x \in \{0,1\}^{n'}$.
- It outputs $\left(h, h\big(C'(x)\big)\right)$.

The output length of $\widehat{C}$ is $\hat{m} = O(m') + k' < O(\max(n',m'))$. Its input length is also $O(\max(n',m'))$.

Suppose $(C,k) \in \mathtt{YES}(\mathsf{SEA}_{\varepsilon})$. That is, $H_2^{\varepsilon}(C) \geq k+1$, and thus $H_2^{\varepsilon'}(C') \geq k' + 20\log m$, where $\varepsilon' = \varepsilon \cdot (20\log m)$. This implies that there is a distribution Y that is at most ε'-far from C' that has

$\mathbf{cp}(Y) \leq 2^{-(k'+20\log m)}$. Let H denote the random variable corresponding to a uniformly random $h \in H_{m',k'}$. By the leftover hash lemma (Lemma 2.10), the statistical distance between $(H, H(Y))$ and (H, U_k) is at most $2^{-(20\log m)/2}$. Thus, the distance between $(H, H(C))$ and (H, U_k) is at most $\varepsilon' + 2^{-10\log m} < 1/\hat{m}$. So $\widehat{C} \in \texttt{YES}(\mathsf{SDU})$.

Suppose $(C, k) \in \texttt{NO}(\mathsf{SEA}_\varepsilon)$. That is, $H_0(C) \leq k - 1$. This means that C has support of size at most 2^{k-1}, and C' has support of size at most $2^{k'-20\log m}$. This implies that the support size of $(H, H(C'))$ is at most $|H_{m,k}| \cdot 2^{k'-20\log m} = 2^{-20\log m} \cdot 2^{\hat{m}}$. This implies that the distance of $(H, H(C'))$ from (H, U_k) is at least $(1 - 2^{-20\log m}) \geq (1 - 1/\hat{m})$. Thus, $\widehat{C} \in \texttt{NO}(\mathsf{SDU})$. □

Theorem 3.10 (SEA is **NISZK**-complete)**.** *For any $\varepsilon(n) \in \left(2^{-n^{0.1}}, o(1/n)\right)$, the problem SEA_ε is* **NISZK***-complete under randomized reductions.*

Proof of Theorem 3.10. By Lemma 3.9, for any $\varepsilon(n) < o(1/n)$, we know that SEA_ε is contained in **NISZK**. We now show a reduction from $\mathsf{AI}_{L,\delta}$. Specifically, for any $L = L(n)$ and $\delta = \delta(n)$, with $\varepsilon(n) = \delta(n)$, we claim that if C is a YES instance of $\mathsf{AI}_{L,\delta}$, then $(C, n-1)$ is a YES instance of SEA_ε, and the same for NO instances.

Suppose $C : \{0,1\}^n \to \{0,1\}^{3n}$ is a YES instance of $\mathsf{AI}_{L,\delta}$. Consider any injective function $\hat{C}$ on the same domain and co-domain that agrees with C on all inputs on which C is injective. Then, the statistical distance between the output distributions of C and $\hat{C}$ is at most δ. Further, the Rényi entropy of $\hat{C}$ is n. So $H_2^\varepsilon(C) = n$. Thus, $(C, n-1)$ is a YES instance of SEA_ε.

Suppose C is a NO instance of $\mathsf{AI}_{L,\delta}$. Then, the size of its range is at most $(1-\delta)\cdot(N/L) + \delta N \leq N \cdot (\delta + 1/L)$. So $H_0(C) \leq n - \log L + \log(1 + \delta L)$. As long as $L \geq 8$ and $\delta L \leq 1$, this is at most $(n-2)$, and $(C, n-1)$ is a NO instance of SEA_ε for any ε.

Therefore, for any $\varepsilon(n) \in (2^{-n^{0.1}}, 1/8)$, choosing $\delta(n) = \varepsilon(n)$ and $L(n) = 1/\varepsilon(n)$, we get a reduction to SEA_ε from $\mathsf{AI}_{L,\delta}$, which is hard for **NISZK**. So for such ε, the problem SEA_ε is also hard for **NISZK** (Lemma 3.5). This completes the proof of the theorem. □

4 Derandomizing Batch Reductions

In this section, we set up a framework for derandomizing the specific kinds of randomized reductions that we perform in Section 5. These reductions result in a sequential composition of circuits alternated with randomly chosen hash functions from some hash family. We show that sampling these functions using a PRG for read-once branching programs (rather than sampling them all uniformly at random from the family) preserves certain properties of this composed circuit that are important to our reduction. We start by formalizing this type of reduction, using what we call a "randomized chaining circuit". Subsequently, Lemma 4.4 states that certain properties of such circuits related to thir Rényi entropy are preserved upon derandomization.

Definition 4.1 (Randomized Chaining Circuit)**.** *Let $R : \{0,1\}^r \times \{0,1\}^m \to \{0,1\}^n$ be a Boolean circuit family and denote $R_\rho(y) := R(\rho, y)$ for each $(r, y) \in \{0,1\}^r \times \{0,1\}^m$, and let $C_1, \ldots, C_k : \{0,1\}^n \to \{0,1\}^m$ be a sequence of Boolean circuits. A circuit $\bar{C} : \{0,1\}^n \times \{0,1\}^{rk} \to \{0,1\}^n$ of the following form is called a* randomized chaining circuit*:*

$$\bar{C}(x, \rho_1, \ldots, \rho_k) = (R_{\rho_k} \circ C_k \circ \ldots \circ R_{\rho_1} \circ C_1)(x).$$

The parameter k is called the chain length*.*

Definition 4.2 (Read-Once Branching Program)**.** *A* read-once branching program *(ROBP) is a directed acyclic multi-graph where the vertices are organized into a grid of $\ell+1$ layers indexed in by $0,\dots,\ell$, with w vertices in each layer internal, a single start vertex in layer 0. Every vertex in layers $0,\dots,\ell-1$ has exactly d outgoing (possibly parallel) edges to vertices in the next layer, with each edge having a unique label in $[d]$. Every vertex in layer ℓ is additionally labeled with an output in $[M]$. On input $(x_1,\dots,x_\ell)\in[d]^\ell$, a ROBP $B:[d]^\ell\to[M]$ computes by successively taking each edge label x_i from layer $i-1$ to layer i. The output $B(x_1,\dots,x_\ell)$ is the label of the vertex it reaches in layer ℓ.*

The parameters w,ℓ and d are called the width, length *and* branching factor *of the ROBP respectively.*

In a seminal work, Nisan [Nis92] constructed a PRG for read-once branching programs. His result is typically stated for programs of braching factor 2 but can be easily generalized to larger branching factors (e.g., by emulating the latter program by the former, while acconting for the increase in length and width).

Lemma 4.3 (PRG for Branching Programs [Nis92])**.** *For any $\ell,w,d\in\mathbb{N}$ and any $\varepsilon\in(0,1)$, there exists a function $G:\{0,1\}^{seed}\to[d]^\ell$ where $seed=\log(\ell)\cdot\log(\ell\cdot w\cdot d/\varepsilon)\cdot\mathrm{poly}(\log\log(d))$ such that for any read-once branching program $B:[d]^\ell\to\{0,1\}$ with length ℓ, width w, branching factor d and binary output, it holds that*

$$\left|\Pr[B(U_{[d]^\ell})=1]-\Pr\left[B\big(G(U_{seed})\big)=1\right]\right|\le\varepsilon,$$

where $U_{[d]^\ell}$ and U_{seed} represent uniform distribution over $[d]^\ell$ and $\{0,1\}^{seed}$, respectively. Furthermore, G is computable in $\mathrm{poly}(\ell,seed)$ time.

Lemma 4.4. *For any $n,r,k\in\mathbb{N}$ and $t\in\mathbb{R}$, suppose $C:\{0,1\}^n\times\{0,1\}^{rk}\to\{0,1\}^n$ is a randomized chaining circuit such that:*

$$\Pr_{\rho\leftarrow\{0,1\}^{rk}}[H_2(C_\rho)>t]>1-\delta.$$

Let $G:\{0,1\}^{seed}\to\{0,1\}^{rk}$ be the PRG guaranteed by Lemma 4.3 for Branching Programs with length k, width 2^{2n}, and error ε. Then, for any $s>0$,

$$\Pr_{\rho\leftarrow G(U_{seed})}[H_2(C_\rho)>t-s]>1-\delta',$$

where $\delta'=\frac{1}{2^s}+(\delta+\varepsilon)\cdot 2^{t-s}$.

Proof. Denote by c_ρ the collision probability of the circuit C_ρ. Let P denote the random variable corresponding to the distribution of c_ρ when ρ is sampled from $\{0,1\}^{rk}$, and let $\hat{P}$ denote the same when ρ is sampled from $G(U_s)$. We will show that the expectations of these two variables are very close, and use that to bound the probability that $\hat{P}$ is too large (thus making the Rényi entropy small).

Note that $H_2(C_\rho)>t$ implies that $c_\rho<2^{-t}$. We bound the expectation of P as follows:

$$\mathbf{E}[P]\le(1-\delta)\cdot 2^{-t}+\delta\cdot 1\le 2^{-t}+\delta. \tag{1}$$

Recall that C_ρ is a chaining circuit of the form $(R_{\rho_k}\circ C_k\circ\cdots\circ R_{\rho_1}\circ C_1)(x)$, where $\rho=(\rho_1,\dots,\rho_k)$, with $C_i:\{0,1\}^n\to\{0,1\}^m$ and $R_{\rho_i}:\{0,1\}^m\to\{0,1\}^n$. For any $x_1,x_2\in\{0,1\}^n$, define a read-once branching program M_{x_1,x_2} with length k that gets as input $(\rho_1,\dots,\rho_k)$, reads a single ρ_i at a time, and works as follows:

- M_{x_1,x_2} starts with a state labelled (x_1, x_2).
- At layer $i \in [0, k-1]$, if it is at a state (y_1, y_2) and receives input ρ_{i+1}, it moves to the state $\Big(R_{\rho_{i+1}}(C_{i+1}(y_1)), R_{\rho_{i+1}}(C_{i+1}(y_2))\Big)$ in layer $(i+1)$.
- At layer k, it accepts iff it is at a state (y_1, y_2) such that $y_1 = y_2$.

Note that the length of M_{x_1,x_2} is k, and its width is 2^{2n}. Essentially, M_{x_1,x_2}, on input ρ, computes $C_\rho(x_1)$ and $C_\rho(x_2)$, and accepts if and only if they are equal. Thus, we can express the collision probability of C_ρ as follows:

$$c_\rho = \Pr_{x_1,x_2 \leftarrow \{0,1\}^n}[C_\rho(x_1) = C_\rho(x_2)] = \Pr_{x_1,x_2 \leftarrow \{0,1\}^n}[M_{x_1,x_2}(\rho) = 1].$$

We can write the expectations of P and $\hat{P}$ as follows:

$$\begin{aligned}\mathbf{E}[P] = \mathop{\mathbf{E}}_{\rho \leftarrow \{0,1\}^{rk}}[c_\rho] &= \mathop{\mathbf{E}}_{\rho \leftarrow \{0,1\}^{rk}}\left[\Pr_{x_1,x_2 \leftarrow \{0,1\}^n}[M_{x_1,x_2}(\rho) = 1]\right] \\ &= \mathop{\mathbf{E}}_{x_1,x_2 \leftarrow \{0,1\}^n}\left[\Pr_{\rho \leftarrow \{0,1\}^{rk}}[M_{x_1,x_2}(\rho) = 1]\right].\end{aligned}$$

Similarly, for $\hat{P}$,

$$\begin{aligned}\mathbf{E}\left[\hat{P}\right] = \mathop{\mathbf{E}}_{\rho \leftarrow G(U_s)}[c_\rho] &= \mathop{\mathbf{E}}_{x_1,x_2 \leftarrow \{0,1\}^n}\left[\Pr_{\rho \leftarrow G(U_s)}[M_{x_1,x_2}(\rho) = 1]\right] \\ &\le \mathop{\mathbf{E}}_{x_1,x_2 \leftarrow \{0,1\}^n}\left[\Pr_{\rho \leftarrow \{0,1\}^{rk}}[M_{x_1,x_2}(\rho) = 1]\right] + \varepsilon \\ &= \mathbf{E}[P] + \varepsilon, \end{aligned} \tag{2}$$

where the inequality follows from Lemma 4.3, as M_{x_1,x_2} is indeed an ROBP that satisfies the required conditions. Combining Eqs. (1) and (2), we have that:

$$\mathbf{E}\left[\hat{P}\right] \le \mathbf{E}[P] + \varepsilon \le 2^{-t} + \delta + \varepsilon.$$

Thus, by the Markov bound:

$$\Pr\left[\hat{P} > 2^{-(t-s)}\right] \le \frac{2^{-t} + \delta + \varepsilon}{2^s \cdot 2^{-t}} = \frac{1 + (\delta + \varepsilon) \cdot 2^t}{2^s}.$$

□

5 Batching AI by Direct Composition

In this section, we show how to reduce k size-n instances of AI to a single instance of the Smooth Entropy Approximation (SEA) problem. Crucially, the length of inputs and outputs of the circuit in the resulting SEA instance is still only n. The **NISZK** protocol for this SEA instance (Lemma 3.9) then gives a batch protocol for AI, and thus any problem in **NISZK**.

Theorem 5.1. *Consider functions $k(n)$, $L(n)$, and $\delta(n) \leq 1/L(n)$, and let $\varepsilon(n) = (\delta(n)^{1/2} \cdot k(n) \cdot L(n)^{1/2})$. There is a* $\text{poly}(n, k)$ *algorithm that, given $k = k(n)$ circuits $(C_1, \ldots, C_k)$, each taking n input bits, outputs a tuple (C, t) such that:*

- *If all of the C_i's are* `YES` *instances of $\mathsf{AI}_{L,\delta}$, then except with probability $O(k(n)^3/L(n))$, the instance (C, t) is a* `YES` *instance of SEA_ε.*
- *If some C_i is a* `NO` *instance of $\mathsf{AI}_{L,\delta}$, then (C, t) is a* `NO` *instance of SEA_ε.*

Further, the input and output lengths of C are both n, and the algorithm uses $n \cdot \text{poly}(\log k, \log n)$ bits of randomness.

Before proceeding to the proof, we restate and prove our main theorem about batching **NISZK** proofs.

Theorem 1.1 (Batch Proofs for **NISZK**). *Suppose $\Pi \in$* **NISZK** *and $k = k(n) \in \mathbb{N}$ such that $k(n) \leq 2^{n^{0.01}}$, where n denotes the length of a single instance of Π. Then, $\Pi^{\otimes k}$ has an* **NISZK** *protocol in which the communication complexity and the length of the common random string is* $\text{poly}\big(n, \log k\big)$. *The completeness, soundness, and zero-knowledge errors are all negligible in n and k, and the verifier runs in time* $\text{poly}(n, k)$.

Proof of Theorem 1.1. Given k instances of Π of size n, set $L = \max(2^{\log^3(n)}, k^{\log\log k})$, $\delta = 1/L^4$, and $\varepsilon = (\delta^{1/2} \cdot k \cdot L^{1/2})$. Note that we still have $L < 2^{n^{0.1}}$ and $\delta > 2^{-n^{0.1}}$ as needed for the **NISZK**-hardness of $\mathsf{AI}_{L,\delta}$ (Lemma 3.5), while also satisfying the conditions required by Theorem 5.1. Further, ε and (k^3/L) are both negligible in both k and n. The prover and verifier in our **NISZK** protocol run as follows:

1. Using Lemma 3.5, reduce the k instances of Π respectively to k instances of $\mathsf{AI}_{L,\delta}$ of size $\text{poly}(n, \log k)$ each.[10]
2. Reduce these k instances of $\mathsf{AI}_{L,\delta}$ to a single instance of SEA_ε with input and output length $\text{poly}(n, \log k)$, using the reduction promised by Theorem 5.1. Here, both the prover and verifier will use $\text{poly}(n, \log k)$ bits from the common random string as the randomness for the reduction. The probability that the reduction fails is at most $O(k^3/L)$ for YES instances, and 0 for NO instances.
3. Run the **NISZK** protocol for SEA_ε (with input and output lengths $\text{poly}(n)$) promised by Lemma 3.9. This protocol has all errors bounded by $2^{-n^{\Omega(1)}}$, negligible in k and n.

Completeness, soundness, and zero-knowledge errors are negligible in k and n following those of the protocol for SEA, and using the fact that the reduction to SEA only fails with negligible probability. Overall, the length of the CRS is $\text{poly}(n, \log k)$ and the communication complexity of the protocol is $\text{poly}(n, \log k)$. □

We now present, in Figure 1, the reduction that establishes Theorem 5.1. Then we state Lemmas 5.2 and 5.3 about its properties. These lemmas together imply Theorem 5.1, as we show below. We present the proof of Lemma 5.3 immediately after, and prove Lemma 5.2 in Section 5.1. For the rest of the section we adopt the notation $N = 2^n$ and $L = 2^\ell$.

[10] The $\log(k)$ factor comes from our setting of the parameter L of $\mathsf{AI}_{L,\delta}$, see [KRV21, Lemma 4.3]. It is important to note, however, that given that $k(n) \leq 2^{n^{0.01}}$, it holds $\text{poly}(n, \log k) = \text{poly}(n)$

Reduction from $\mathsf{AI}_{L,\delta}^{\otimes k}$ to SEA

Input: $C_1, \ldots, C_k$, where each C_i is a circuit $C_i : \{0,1\}^n \to \{0,1\}^{3n}$ (k instances of $\mathsf{AI}_{L,\delta}$)

Output: (C, t), where C is a circuit $C : \{0,1\}^n \to \{0,1\}^n$ (one instance of SEA)

Ingredients:

- $\mathcal{H} = \left\{h : \{0,1\}^{3n} \to \{0,1\}^n\right\}$ is a family of efficient 4-wise independent hash functions as guaranteed to exist by Lemma 2.9. A random function from this family is selected using $d = O(n)$ random bits. For any string $r \in \{0,1\}^d$, we will denote by h_r the function in $\mathcal{H}$ selected by this string.
- $G : \{0,1\}^s \to (\{0,1\}^d)^k$ is the PRG for Branching Programs promised by Lemma 4.3, with the parameters there instantiated as: $\ell = k$, $w = 2^{2n}$, $D = 2^d = 2^{O(n)}$, and $\varepsilon = 2^{-5n}$. This implies a seed length of $s = (n + \log(k)) \cdot \log(k) \cdot \mathrm{polylog}(n)$.

Procedure:

1. Sample random $\rho \leftarrow \{0,1\}^s$.
2. Compute $(r_1, \ldots, r_k) \leftarrow G(\rho)$, where each $r_i \in \{0,1\}^d$.
3. Define circuit C as $C(x) = (h_{r_k} \circ C_k \circ h_{r_{k-1}} \circ C_{k-1} \circ \cdots \circ h_{r_1} \circ C_1)(x)$.
4. Output (C, t), where $t = (n - \log L + 2)$.

Figure 1: Reducing k instances of AI to one instance of SEA

Lemma 5.2. *Let (C, t) be the output of the reduction in Figure 1 on input circuits $(C_1, \ldots, C_k)$. For any L and δ, if all of the C_i's are YES-instance of $\mathsf{AI}_{L,\delta}$, then for any $c > 1$:*

$$\Pr\left[H_2^{\varepsilon}(C) < n - c \cdot \log k\right] < O\left(\frac{1}{k^{c-3}}\right),$$

where $\varepsilon = \delta^{1/2} \cdot k^{1+c/2}$.

Lemma 5.3. *Let (C, t) be the output of the reduction in Figure 1 on input circuits $(C_1, \ldots, C_k)$. For any L and δ such that $\delta \cdot L \leq 1$, if at least one of the C_i's is a NO-instance of $\mathsf{AI}_{L,\delta}$, then:*

$$H_0(C) \leq n - \ell + 1.$$

Proof of Lemma 5.3. Suppose C_{i^*} is a NO-instance of $\mathsf{AI}_{L,\delta}$. This implies that:

$$|\mathrm{Supp}(C_{i^*})| \leq (1-\delta) \cdot N \cdot \frac{1}{L} + \delta \cdot N \cdot 1 \leq N \cdot \left(\frac{1}{L} + \delta\right).$$

So even if all the other C_i's and the hash functions chosen in the reduction are injective, the number of possible images of C is at most this (since merely composing functions cannot increase the support size). So the max entropy of C can be bounded as:

$$\begin{aligned} H_0(C) = \log(|\mathrm{Supp}(C)|) &\leq \log N + \log\left(\frac{1}{L} + \delta\right) \\ &= \log N - \log L + \log(1 + \delta L) \\ &\leq n - \ell + 1. \end{aligned}$$

□

Proof of Theorem 5.1. Let (C, t) be the output of the reduction in Figure 1 on input $(C_1, \ldots, C_k)$. Note that $t = n - \ell + 2$. By Lemma 5.3, if even one of the C_i's is a NO-instance of $\mathsf{AI}_{L,\delta}$, then $H_0(C) \leq t - 1$, and thus (C, t) is a NO-instance of SEA_ε for any ε.

Suppose all the C_i's are YES instances of $\mathsf{AI}_{L,\delta}$. Applying Lemma 5.2 with $c = (\ell - 3)/\log k$, we get that:

$$\Pr\left[H_2^\varepsilon(C) \geq n - \ell + 3\right] > 1 - O\left(\frac{k^3}{L}\right),$$

where $\varepsilon < \delta^{1/2} \cdot k \cdot L^{1/2}$. Thus, (C, t) is a YES-instance of SEA_ε for such a value of ε. The input and output lengths of C and the randomness complexity of the reduction may be verified in a straightforward manner to be n and $\mathrm{poly}(n, \log k)$, respectively. This proves the theorem. □

5.1 Proof of Lemma 5.2

For convenience, we set up the following notation in the context of the reduction in Figure 1. For any circuit C, denote by $inj(C)$ the set of inputs on which C is injective. The input to the reduction are the circuits $C_1, \ldots, C_k : \{0,1\}^n \to \{0,1\}^{3n}$, which are all YES instances of $\mathsf{AI}_{L,\delta}$. The output is (C, t), where $C : \{0,1\}^n \to \{0,1\}^n$. We will denote the process of sampling the hash functions in the reduction by $(h_1, \ldots, h_k) \leftarrow G$ – this indicates first computing $(r_1, \ldots, r_k) \leftarrow G(\rho)$ for a uniformly random ρ, and setting h_i to be h_{r_i}. We will denote sampling k uniformly random hash functions from $\mathcal{H}$ by $(h_1, \ldots, h_k) \leftarrow \mathcal{H}^k$. For any tuple of hash functions $\overline{h} = (h_1, \ldots, h_k)$, we will denote the circuit constructed by using these for the composition by $C_{\overline{h}}$ or $C_{h_1,\ldots,h_k}$. That is,

$$C_{h_1,\ldots,h_k}(x) = (h_k \circ C_k \circ h_{k-1} \circ C_{k-1} \circ \cdots \circ h_1 \circ C_1)(x).$$

The reduction samples $\overline{h} = (h_1, \ldots, h_k) \leftarrow G$ and outputs $C_{\overline{h}}$. Our approach is to show that, with high probability over $\overline{h}$, the output distribution of the circuit $C_{\overline{h}}$ is close to that of a different function $\hat{C}_{\overline{h}} : \{0,1\}^n \to \{0,1\}^n$, which has high Rényi entropy. We start by defining this function. For each C_i, we define its *injective completion*, denoted $\hat{C}_i : \{0,1\}^n \to \{0,1\}^{3n}$, to be the lexicographically smallest function[11] that has the following two properties:

- $\hat{C}_i$ is injective.
- For all $x \in inj(C_i)$, we have $\hat{C}_i(x) = C_i(x)$.

[11] Any function that satisfies the two stated properties will do for our purpose, and the injective completion may be defined to be any such function.

Note that $\hat{C}_i$ always exists because the co-domain of C_i is larger than its domain. For any tuple of hash functions $\overline{h}$, the function $\hat{C}_{\overline{h}}$ is defined as:

$$\hat{C}_{h_1,\ldots,h_k}(x) = (h_k \circ \hat{C}_k \circ h_{k-1} \circ \hat{C}_{k-1} \circ \cdots \circ h_1 \circ \hat{C}_1)(x).$$

The proof now proceeds by showing the following:

1. For $(h_1, \ldots, h_k) \leftarrow G$, with high probability, $\hat{C}_{\overline{h}}$ has high Rényi entropy.
2. For any $\overline{h}$ for which $\hat{C}_{\overline{h}}$ has high Rényi entropy, the distribution of $C_{\overline{h}}$ is close to that of $\hat{C}_{\overline{h}}$.

Together, these imply that with $\overline{h} \leftarrow G$, with high probability, $C_{\overline{h}}$ has high smooth Rényi entropy, which proves the lemma. We now state these claims formally, show how to use them to prove the lemma, and then prove the claims.

Proposition 5.4. *For any $s \in (3, n)$, we have:*

$$\Pr_{(h_1,\ldots,h_k)\leftarrow G}\left[H_2(\hat{C}_{h_1,\ldots,h_k}) < n - \log k - s\right] < O\left(\frac{k^2}{2^s}\right).$$

Proposition 5.5. *For any $(h_1, \ldots, h_k)$ for which $H_2(\hat{C}_{h_1,\ldots,h_k}) \geq t$, we have:*

$$\Delta(\hat{C}_{h_1,\ldots,h_k}, C_{h_1,\ldots,h_k}) \leq k \cdot \delta^{1/2} \cdot 2^{(n-t)/2}.$$

Proof of Lemma 5.2. Setting $s = (c-1) \cdot \log k$, we get the following from Proposition 5.4:

$$\Pr_{(h_1,\ldots,h_k)\leftarrow G}\left[H_2(\hat{C}_{h_1,\ldots,h_k}) \geq n - c \cdot \log k\right] > 1 - O\left(\frac{1}{k^{c-3}}\right).$$

For any $(h_1, \ldots, h_k)$ for which the above event happens, Proposition 5.5 implies that:

$$\Delta(\hat{C}_{h_1,\ldots,h_k}, C_{h_1,\ldots,h_k}) \leq \delta^{1/2} \cdot k^{1+c/2},$$

which proves the lemma. □

5.1.1 Proof of Proposition 5.4

We prove the claim by first showing a similar bound when the hash functions are sampled completely at random, and then derandomizing this using G.

Claim 5.6. $\Pr_{(h_1,\ldots,h_k)\leftarrow \mathcal{H}^k}\left[H_2(\hat{C}_{h_1,\ldots,h_k}) < n - \log k - 3\right] < O\left(\frac{k^3}{2^n}\right)$.

Proof of Proposition 5.4. Note that $\hat{C}_{h_1,\ldots,h_k}$ is a chaining circuit (as in Definition 4.1), and so we can use the derandomization techniques from Section 4 to derandomize Claim 5.6. Specifically, applying Lemma 4.4 with $\hat{C}_{h_1,\ldots,h_k}$ as the chaining circuit, with $t = n - \log k - 3$, $\delta = O(k^3/2^n)$, and $\varepsilon = 2^{-5n}$ (as in our reduction), we get the following conclusion:

$$\begin{aligned}\Pr_{(h_1,\ldots,h_k)\leftarrow G}\left[H_2(\hat{C}_{h_1,\ldots,h_k}) < n - \log k - 3 - (s-3)\right] &< \frac{1}{2^{s-3}} + \left(O\left(\frac{k^3}{2^n}\right) + \frac{1}{2^{5n}}\right) \cdot 2^{n - \log k - s} \\ &= O\left(\frac{k^2}{2^s}\right).\end{aligned}$$

□

Recall that $\mathcal{H}$ is a 4-wise independent family of hash functions mapping $\{0,1\}^{3n}$ to $\{0,1\}^n$. We prove Claim 5.6 by showing that most functions from $\mathcal{H}$ nearly preserve the collision probability of their input distribution. Given any distribution D over $\{0,1\}^{3n}$, denote by $h(D)$ the distribution obtained by applying the function h to a sample from D. We first show the following claim, use it to prove Claim 5.6, and then complete its proof.

Claim 5.7. *If $H_2(D) \geq t$ and $\mathcal{H}$ is a 4-wise independent family of hash functions, then:*

$$\Pr_{h \leftarrow \mathcal{H}}\left[H_2\big(h(D)\big) < t - \frac{2^{t+2}}{2^n}\right] \leq \frac{4}{2^{2t-n}}.$$

This claim is only interesting when $t > n/2$. In our applications of it, we will be using values of t that are very close to n, and it gives rather strong bounds.

Proof of Claim 5.6. For $i \in [0,k]$, define distribution D_i as the output distribution of $(h_i \circ \hat{C}_i \circ \cdots \circ h_1 \circ \hat{C}_1)(x)$ when x is uniformly random. We will prove the claim by induction on the D_i's. To start with, note that D_0 is the uniform distribution over $\{0,1\}^n$, and so $H_2(D_0) = n$.

For any i, since $\hat{C}_i$ is injective, we have $H_2(\hat{C}_i(D_{i-1})) = H_2(D_{i-1})$. Applying Claim 5.7 with any $t \in [n - \log k - 3, n - \log k - 2]$, we get that if $H_2(\hat{C}_i(D_{i-1})) = H_2(D_{i-1}) > t$, then $H_2(D_i) < (t - 1/k)$ with probability at most $2^8 \cdot k^2/2^n$. Starting from D_0 and $t = (n - \log k - 2)$ and applying this iteratively, and using a union bound, we get that the probabilty that $H_2(D_k) < n - \log k - 3$ is at most $O(k^3/2^n)$, as needed. □

Proof of Claim 5.7. Our approach will be to show that the expected collision probability of $h(D)$ is not much larger than $\mathbf{cp}(D)$, and that its variance is small. We can then bound the probability that $\mathbf{cp}(h(D))$ is much larger than $\mathbf{cp}(D)$ using Chebyshev's inequality (Lemma 2.2). Recall that $\mathcal{H}$ is a set of 4-wise independent hash functions whose co-domain is of size $N = 2^n$.

We set up the following notation:

- Denote by $c = \mathbf{cp}(D)$ denote the collision probability of D.
- For any $h \in \mathcal{H}$, denote by Q_h the collision probability of $h(D)$.

The hypothesis of the claim implies that:

$$c \leq 2^{-t}. \tag{3}$$

By the definition of collision probability, we have for any h:

$$Q_h = \Pr_{x_1,x_2 \leftarrow D}\left[h(x_1) = h(x_2)\right].$$

Its expectation can be calculated as follows:

$$\begin{aligned}\mathop{\mathbf{E}}_{h \leftarrow \mathcal{H}}\left[Q_h\right] &= \Pr_{h,x_1,x_2}\left[h(x_1) = h(x_2)\right] \\ &= \Pr\left[x_1 = x_2\right] \cdot 1 + \Pr\left[x_1 \neq x_2\right] \cdot \Pr\left[h(x_1) = h(x_2) \mid x_1 \neq x_2\right] \\ &= c + \frac{1-c}{N}, \end{aligned} \tag{4}$$

where in the last equality we used the fact that $\mathcal{H}$ is a pairwise independent family. Next we calculate its second moment. If Q_h is the collision probability of $h(D)$, then Q_h^2 is the probabilty that when two pairs of samples from $h(D)$ are picked, both pairs are colliding. Thus, we have:

$$\begin{aligned}\mathbf{E}\left[Q_h^2\right] &= \Pr_{h,x_1,x_2,x_3,x_4}\left[h(x_1)=h(x_2)\wedge h(x_3)=h(x_4)\right]\\ &= \sum_i \Pr\left[E_i\right]\cdot\Pr\left[h(x_1)=h(x_2)\wedge h(x_3)=h(x_4)\mid E_i\right], \qquad (5)\end{aligned}$$

where E_i's are any set of disjoint events whose union is the entire sample space. We will employ a set of such events E_i as follows, and in each case we will bound the following quantities:

$$p_i = \Pr\left[E_i\right] \quad \text{and } q_i = \Pr\left[h(x_1)=h(x_2)\wedge h(x_3)=h(x_4)\mid E_i\right].$$

Throughout the following analysis, we use the fact that h is from a family of 4-wise independent hash functions, and $\mathbf{cp}(D)=c$.

- $E_1 \equiv \big((x_1=x_2)\wedge(x_3=x_4)\big)$: In this case, hashes are always equal. The probability this event happens is simply the square of the collision probability. So we have:

$$\begin{aligned}p_1 &= c^2;\\ q_1 &= 1.\end{aligned}$$

- $E_2 \equiv \big((x_1=x_2)\wedge(x_3\neq x_4)$: In this case, $h(x_1)$ is always equal to $h(x_2)$, and the event $h(x_3)=h(x_4)$ is independent of this. So we have:

$$\begin{aligned}p_2 &= c\cdot(1-c);\\ q_2 &= \frac{1}{N}.\end{aligned}$$

- $E_3 \equiv \big((x_1\neq x_2)\wedge(x_3=x_4)\big)$: The probabilities here are the same as for E_2.

- $E_4 \equiv (x_1\neq x_2)\wedge(x_3\neq x_4)\wedge(\{x_1,x_2\}=\{x_3,x_4\})$: In this case, either $x_1=x_3$ and $x_2=x_4$, or the other way round. Further, the events $h(x_1)=h(x_2)$ and $h(x_3)=h(x_4)$ are the same. We have:

$$\begin{aligned}p_4 &\leq \Pr\left[x_1=x_3\wedge x_2=x_4\right]+\Pr\left[x_1=x_4\wedge x_2=x_3\right]=2c^2;\\ q_4 &= \frac{1}{N}.\end{aligned}$$

- $E_5 \equiv (x_1\neq x_2)\wedge(x_3\neq x_4)\wedge(\{x_1,x_2\}\neq\{x_3,x_4\})$: In this case, it could be that all the x_i's are distinct, but it could also be that $x_1=x_3$ and $x_2\neq x_4$ (or the other way round, etc.). In any case, the events $h(x_1)=h(x_2)$ and $h(x_3)=h(x_4)$ are always independent due to the 4-wise independence of $\mathcal{H}$. We have:

$$\begin{aligned}p_5 &\leq 1;\\ q_5 &= \frac{1}{N^2}.\end{aligned}$$

With the above analysis and Eq. (5), we can bound the second moment of Q_h as:

$$\mathbf{E}\left[Q_n^2\right] \leq c^2 + \frac{2c(1-c)}{N} + \frac{2c^2}{N} + \frac{1}{N^2}.$$

The variance of Q_h can now be bounded as:

$$\begin{aligned}
\mathbf{Var}\left[Q_h^2\right] &= \mathbf{E}\left[Q_h^2\right] - \mathbf{E}\left[Q_h\right]^2 \\
&\leq \left(c^2 + \frac{2c(1-c)}{N} + \frac{2c^2}{N} + \frac{1}{N^2}\right) - \left(c + \frac{1-c}{N}\right)^2 \\
&= c^2 + \frac{2c(1-c)}{N} + \frac{2c^2}{N} + \frac{1}{N^2} - c^2 - \left(\frac{1-c}{N}\right)^2 - \frac{2c(1-c)}{N} \\
&= \frac{2c^2}{N} + \frac{1}{N^2} - \left(\frac{1}{N^2} + \frac{c^2}{N^2} - \frac{2c}{N^2}\right) \\
&\leq \frac{2c^2}{N} + \frac{2c}{N^2} \\
&\leq \frac{4c^2}{N}, \qquad (6)
\end{aligned}$$

where in the last inequality we used the fact that $c \geq 1/N$. From Eqs. (3), (4) and (6), we get:

$$\mathbf{E}\left[Q_h\right] \leq 2^{-t} + \frac{1}{N};$$
$$\mathbf{Var}\left[Q_h\right] \leq \frac{2^{-(2t-2)}}{N}.$$

Applying Chebychev's inequality (Lemma 2.2),

$$\Pr_h\left[Q_h > 2^{-t} + \frac{2}{N}\right] \leq \frac{2^{-(2t-2)}}{N} \cdot N^2 = \frac{4}{2^{2t-n}}.$$

Using the fact that $\log_2(1+x) \leq 2x$, we have:

$$\begin{aligned}
\Pr\left[H_2(h(D)) < t - \frac{2^{t+2}}{2^n}\right] &\leq \Pr\left[H_2(h(D)) < t - \log\left(1 + \frac{2^{t+1}}{N}\right)\right] \\
&= \Pr\left[Q_h > 2^{-t} + \frac{2}{N}\right] \\
&\leq \frac{4}{2^{2t-n}},
\end{aligned}$$

which proves the claim. $\square$

5.1.2 Proof of Proposition 5.5

Fix some $h_1, \ldots, h_k$ for which $H_2(\hat{C}_{h_1,\ldots,h_k}) \geq t$. To help with the proof, we define the following sets of distributions for $i \in [0, k]$, with x chosen uniformly at random:

$$D_i = (h_i \circ C_i \circ \cdots \circ h_1 \circ C_1)(x).$$
$$\hat{D}_i = (h_i \circ \hat{C}_i \circ \cdots \circ h_1 \circ \hat{C}_1)(x).$$

Here, D_0 and $\hat{D}_0$ are both the uniform distribution over $\{0,1\}^n$, and D_k and $\hat{D}_k$ are the distributions whose distance we need to bound to show the claim. The hypothesis of the claim is that $H_2(\hat{D}_k) \geq t$. We will prove the claim inductively, with the identity of D_0 and $\hat{D}_0$ as the base case, and using the following claims for the inductive steps.

Claim 5.8. *For any $i \in [k]$, we have:*

$$\Delta\left(D_i, \hat{D}_i\right) \leq \Delta\left(C_i(D_{i-1}), \hat{C}_i(\hat{D}_{i-1})\right).$$

Proof. D_i and $\hat{D}_i$ are sampled, respectively, by applying the same function h_i to a sample from $C_i(D_{i-1})$ and $\hat{C}_i(\hat{D}_{i-1})$. Thus, the claim follows from the data processing inequality. □

Claim 5.9. *For any $i \in [k]$, we have:*

$$\Delta\left(C_i(D_{i-1}), \hat{C}_i(\hat{D}_{i-1})\right) \leq \Delta\left(D_{i-1}, \hat{D}_{i-1}\right) + \delta^{\frac{1}{2}} \cdot 2^{\frac{n-t}{2}}.$$

Proof. We start with the observation that the hypothesis of the claim – $H_2(\hat{C}_{h_1,\ldots,h_k}) \geq t$ – also implies that for all $i \in [0,k]$, we have $H_2(\hat{D}_i) \geq t$. This is because all the h_i's and $\hat{C}_i$'s are fixed deterministic functions, and applying them can only decrease the entropy of a distribution.

For any $x \in inj(C_i)$, by definition, $C_i(x) = \hat{C}_i(x)$. Thus, we have:

$$\Delta\left(C_i(\hat{D}_{i-1}), \hat{C}_i(\hat{D}_{i-1})\right) \leq \Pr_{x \leftarrow \hat{D}_{i-1}}\left[x \notin inj(C_i)\right]. \tag{7}$$

Denote the quantity in the right-hand side above by p, which we will now bound. As C_i is a YES instance of $\mathsf{AI}_{L,\delta}$, the number of x's not in $inj(C_i)$ is at most δN. The least possible collision probability achievable for $\hat{D}_{i-1}$ with a probability mass of p on this set is achieved when the mass is distributed uniformly across it. Thus, due to the contribution to collision probability from this set alone, we get:

$$\mathbf{cp}(\hat{D}_{i-1}) \geq \left|\overline{inj(C_i)}\right| \cdot \left(\frac{p}{\left|\overline{inj(C_i)}\right|}\right)^2 \geq \frac{p^2}{\delta N}.$$

On the other hand, we know that $\mathbf{cp}(\hat{D}_{i-1}) \leq 2^{-t}$. This gives us:

$$p \leq (2^{-t} \cdot \delta N)^{1/2} = \delta^{1/2} \cdot 2^{(n-t)/2}. \tag{8}$$

Next, by the data processing inequality, we have:

$$\Delta\left(C_i(D_{i-1}), C_i(\hat{D}_{i-1})\right) \leq \Delta\left(D_{i-1}, \hat{D}_{i-1}\right). \tag{9}$$

Putting together Eqs. (7) to (9) and using the triangle inequality gives us the claim. □

Proof of Proposition 5.5. Starting with the fact that $\Delta(D_0, \hat{D}_0) = 0$ and applying Claims 5.8 and 5.9 inductively k times proves the claim. □

Acknowledgements

We thank Or Keret for helpful comments.

Shafik Nassar and Ron Rothblum were funded by the European Union (ERC, FASTPROOF, 101041208). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them. Shafik Nassar is also supported in part by NSF CNS-1908611, CNS-2318701, and CNS-2140975.

Prashant Nalini Vasudevan is supported by the National Research Foundation, Singapore, under its NRF Fellowship programme, award no. NRF-NRFF14-2022-0010.

References

[BBK+23] Zvika Brakerski, Maya Farber Brodsky, Yael Tauman Kalai, Alex Lombardi, and Omer Paneth. SNARGs for monotone policy batch NP. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part II*, volume 14082 of *Lecture Notes in Computer Science*, pages 252–283. Springer, 2023.

[BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In Janos Simon, editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 103–112. ACM, 1988.

[BHK17] Zvika Brakerski, Justin Holmgren, and Yael Tauman Kalai. Non-interactive delegation and batch NP verification from standard computational assumptions. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 474–482. ACM, 2017.

[BKP+23] Nir Bitansky, Chethan Kamath, Omer Paneth, Ron Rothblum, and Prashant Nalini Vasudevan. Batch proofs are statistically hiding. *Electron. Colloquium Comput. Complex.*, TR23-077, 2023.

[BSMP91] Manuel Blum, Alfredo Santis, Silvio Micali, and Giuseppe Persiano. Noninteractive zero-knowledge. *SIAM J. Comput.*, 20(6):1084–1118, dec 1991.

[CGJ+23] Arka Rai Choudhuri, Sanjam Garg, Abhishek Jain, Zhengzhong Jin, and Jiaheng Zhang. Correlation intractability and SNARGs from sub-exponential DDH. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part IV*, volume 14084 of *Lecture Notes in Computer Science*, pages 635–668. Springer, 2023.

[CJJ21a] Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. Non-interactive batch arguments for NP from standard assumptions. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part IV*, volume 12828 of *Lecture Notes in Computer Science*, pages 394–423. Springer, 2021.

[CJJ21b] Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. SNARGs for P from LWE. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 68–79. IEEE, 2021.

[DGKV22] Lalita Devadas, Rishab Goyal, Yael Kalai, and Vinod Vaikuntanathan. Rate-1 non-interactive arguments for batch-NP and applications. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 1057–1068. IEEE, 2022.

[DGRV11] Zeev Dvir, Dan Gutfreund, Guy N. Rothblum, and Salil P. Vadhan. On approximating the entropy of polynomial mappings. In Bernard Chazelle, editor, *Innovations in Computer Science - ICS 2011, Tsinghua University, Beijing, China, January 7-9, 2011. Proceedings*, pages 460–475. Tsinghua University Press, 2011.

[FS90] U. Feige and A. Shamir. Witness indistinguishable and witness hiding protocols. In *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing*, STOC '90, page 416–426, New York, NY, USA, 1990. Association for Computing Machinery.

[GH98] Oded Goldreich and Johan Håstad. On the complexity of interactive proofs with bounded communication. *Information Processing Letters*, 67(4):205–214, 1998.

[GHKS23] Aarushi Goel, Mathias Hall-Andersen, Gabriel Kaptchuk, and Nicholas Spooner. Speed-stacking: Fast sublinear zero-knowledge proofs for disjunctions. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part II*, volume 14005 of *Lecture Notes in Computer Science*, pages 347–378. Springer, 2023.

[GK96] Oded Goldreich and Hugo Krawczyk. On the composition of zero-knowledge proof systems. *SIAM Journal on Computing*, 25(1):169–192, 1996.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.

[GSV98] Oded Goldreich, Amit Sahai, and Salil P. Vadhan. Honest-verifier statistical zero-knowledge equals general statistical zero-knowledge. In Jeffrey Scott Vitter, editor, *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, pages 399–408. ACM, 1998.

[GSV99] Oded Goldreich, Amit Sahai, and Salil Vadhan. Can statistical zero knowledge be made non-interactive? or on the relationship of SZK and NISZK. In Michael Wiener, editor, *Advances in Cryptology — CRYPTO' 99*, pages 467–484, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

[GVW02] Oded Goldreich, Salil Vadhan, and Avi Wigderson. On interactive proofs with a laconic prover. *Comput. Complex.*, 11(1/2):1–53, jun 2002.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[KLVW23] Yael Kalai, Alex Lombardi, Vinod Vaikuntanathan, and Daniel Wichs. Boosting batch arguments and RAM delegation. In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 1545–1552. ACM, 2023.

[KRR$^+$20] Inbar Kaslasi, Guy N. Rothblum, Ron D. Rothblum, Adam Sealfon, and Prashant Nalini Vasudevan. Batch verification for statistical zero knowledge proofs. In Rafael Pass and Krzysztof Pietrzak, editors, *Theory of Cryptography - 18th International Conference, TCC 2020, Durham, NC, USA, November 16-19, 2020, Proceedings, Part II*, volume 12551 of *Lecture Notes in Computer Science*, pages 139–167. Springer, 2020.

[KRV21] Inbar Kaslasi, Ron D. Rothblum, and Prashant Nalini Vasudevan. Public-coin statistical zero-knowledge batch verification against malicious verifiers. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part III*, volume 12698 of *Lecture Notes in Computer Science*, pages 219–246. Springer, 2021.

[KVZ21] Yael Tauman Kalai, Vinod Vaikuntanathan, and Rachel Yun Zhang. Somewhere statistical soundness, post-quantum security, and SNARGs. In Kobbi Nissim and Brent Waters, editors, *Theory of Cryptography - 19th International Conference, TCC 2021, Raleigh, NC, USA, November 8-11, 2021, Proceedings, Part I*, volume 13042 of *Lecture Notes in Computer Science*, pages 330–368. Springer, 2021.

[LFKN92] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *J. ACM*, 39(4):859–868, 1992.

[Nis92] Noam Nisan. Pseudorandom generators for space-bounded computation. *Comb.*, 12(4):449–461, 1992.

[NV06] Minh-Huyen Nguyen and Salil P. Vadhan. Zero knowledge with efficient provers. In Jon M. Kleinberg, editor, *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 287–295. ACM, 2006.

[OV08] Shien Jin Ong and Salil P. Vadhan. An equivalence between zero knowledge and commitments. In Ran Canetti, editor, *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19-21, 2008*, volume 4948 of *Lecture Notes in Computer Science*, pages 482–500. Springer, 2008.

[PP22] Omer Paneth and Rafael Pass. Incrementally verifiable computation via rate-1 batch arguments. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 1045–1056. IEEE, 2022.

[RR20] Guy N. Rothblum and Ron D. Rothblum. Batch verification and proofs of proximity with polylog overhead. In Rafael Pass and Krzysztof Pietrzak, editors, *Theory of Cryptography - 18th International Conference, TCC 2020, Durham, NC, USA, November 16-19, 2020, Proceedings, Part II*, volume 12551 of *Lecture Notes in Computer Science*, pages 108–138. Springer, 2020.

[RRR18] Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Efficient batch verification for UP. In Rocco A. Servedio, editor, *33rd Computational Complexity Conference, CCC 2018, June 22-24, 2018, San Diego, CA, USA*, volume 102 of *LIPIcs*, pages 22:1–22:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[RRR21] Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Constant-round interactive proofs for delegating computation. *SIAM J. Comput.*, 50(3), 2021.

[RW04] R. Renner and S. Wolf. Smooth Renyi entropy and applications. In *International Symposium onInformation Theory, 2004. ISIT 2004. Proceedings.*, pages 233–, 2004.

[SCPY98] Alfredo De Santis, Giovanni Di Crescenzo, Giuseppe Persiano, and Moti Yung. Image density is complete for non-interactive-SZK (extended abstract). In Kim Guldstrand Larsen, Sven Skyum, and Glynn Winskel, editors, *Automata, Languages and Programming, 25th International Colloquium, ICALP'98, Aalborg, Denmark, July 13-17, 1998, Proceedings*, volume 1443 of *Lecture Notes in Computer Science*, pages 784–795. Springer, 1998.

[Sha92] Adi Shamir. IP = PSPACE. *J. ACM*, 39(4):869–877, 1992.

[Vad99] Salil Pravin Vadhan. *A study of statistical zero-knowledge proofs*. PhD thesis, Massachusetts Institute of Technology, 1999.

[Vad12] Salil P. Vadhan. Pseudorandomness. *Foundations and Trends® in Theoretical Computer Science*, 7(1–3):1–336, 2012.

[WW22] Brent Waters and David J. Wu. Batch arguments for NP and more from standard bilinear group assumptions. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part II*, volume 13508 of *Lecture Notes in Computer Science*, pages 433–463. Springer, 2022.