Electronic Colloquium on Computational Complexity, Report No. 28 (2026)

Weak Zero-Knowledge and One-Way Functions

Rohit Chatterjee* Yunqi Li† Prashant Nalini Vasudevan‡

February 18, 2026

Abstract

We study the implications of the existence of weak Zero-Knowledge (ZK) protocols for worst-case hard languages. These are protocols that have completeness, soundness, and zero-knowledge errors (denoted ϵ_c, ϵ_s, and ϵ_z, respectively) that might not be negligible. Under the assumption that there are worst-case hard languages in NP, we show the following:

1. If all languages in NP have NIZK proofs or arguments satisfying $\epsilon_c + \epsilon_s + \epsilon_z < 1$, then One-Way Functions (OWFs) exist.

   This covers all possible non-trivial values for these error rates. It additionally implies that if all languages in NP have such NIZK proofs and ϵ_c is negligible, then they also have NIZK proofs where all errors are negligible. Previously, these results were known under the more restrictive condition $\epsilon_c + \sqrt{\epsilon_s} + \epsilon_z < 1$ [Chakraborty et al., CRYPTO 2025].

2. If all languages in NP have k-round public-coin ZK proofs or arguments satisfying $\epsilon_c + \epsilon_s + (2k-1) \cdot \epsilon_z < 1$, then OWFs exist.

3. If, for some constant k, all languages in NP have k-round public-coin ZK proofs or arguments satisfying $\epsilon_c + \epsilon_s + k \cdot \epsilon_z < 1$, then infinitely-often OWFs exist.

* rochat@nus.edu.sg. Department of Computer Science, National University of Singapore.
† yunqili@comp.nus.edu.sg. Department of Computer Science, National University of Singapore.
‡ prashvas@nus.edu.sg. Department of Computer Science, National University of Singapore.

ISSN 1433-8092

Contents

1 Introduction **1**
- 1.1 Technical Overview . . . 4

2 Preliminaries **12**
- 2.1 Indistinguishability . . . 13
- 2.2 Circuits and Oracles . . . 15
- 2.3 One-Way Functions . . . 15
- 2.4 Zero-Knowledge Protocols . . . 17
  - 2.4.1 Non-Interactive Zero-Knowledge . . . 17
  - 2.4.2 Public-Coin Zero-Knowledge . . . 18

3 Auxiliary-Input One-Way Functions **19**
- 3.1 Reductions from NIZK . . . 20
- 3.2 Reductions from Public-Coin ZK . . . 23
  - 3.2.1 Proof of Claim 3.1 . . . 25
- 3.3 Reductions from Constant-Round Public-Coin ZK . . . 30
  - 3.3.1 Proof of Claim 3.3 . . . 34

4 One-Way Functions **43**
- 4.1 One-Sided Average-Case Hardness from ai-OWF . . . 44
- 4.2 OWF from One-Sided Average-Case Hardness . . . 44
  - 4.2.1 NIZK and Public-Coin ZK . . . 44
  - 4.2.2 Constant-Round Public-Coin ZK . . . 47

A On Randomized Verification **51**

1 Introduction

The notion of Zero-Knowledge (ZK) protocols is a vital part of modern cryptography. These are interactive proof systems in which a *prover* proves to a *verifier* the validity of a given statement, with the additional guarantee that even a possibly cheating verifier cannot obtain any secrets that might have been used in the proof. More precisely, such protocols guarantee *soundness*, i.e. cheating provers cannot prove false statements, and *zero knowledge*, i.e. a cheating verifier cannot glean anything beyond the validity of the statement over the course of the protocol.

Hardness from Zero-Knowledge One natural question arising in the context of relating zero-knowledge to other cryptographic notions is that of which other cryptographic primitives are implied by it. This was first studied in the work of Ostrovsky [Ost91], who showed that a statistical ZK proof system for any average-case hard language implies the existence of One-Way Functions (OWFs). This was later generalized by Ostrovsky and Wigderson [OW93] to computational ZK proofs, and they showed in addition that such proofs for even a worst-case hard language implies a weaker form of OWFs called auxiliary-input OWFs.

Building on these, the recent work of Hirahara and Nanashima [HN24] showed that if all languages in NP have computational ZK proofs (or even arguments, which only have computational soundness) and NP is hard in the worst-case, then OWFs exist.

Weak Zero-Knowledge The above results all work with proof systems where the completeness, soundness, and zero-knowledge errors are guaranteed to be negligible. There are, however, a number of natural and useful ZK protocols, such as the commonly taught protocols for 3-Coloring and Graph Non-isomorphism, that do not natively have negligible error rates. Such *Weak ZK* protocols may have completeness error ϵ_c, soundness error of ϵ_s, and zero-knowledge error ϵ_z, that may be as large as a constant number. Along the same lines as above, it is natural and important to study the power of such protocols as well.

Amplifying Weak NIZKs The work of Goyal et al. [GJS19] was a first step in this direction, investigating the power of weak *Non-Interactive ZK* (NIZK) arguments. NIZKs consider a non-interactive setting where the prover and verifier have access to a common random string, and the protocol only involves a single prover message, following which the verifier decides whether to accept or reject. They showed that weak NIZKs with negligible ε_c satisfying $\epsilon_s + \epsilon_z < 1 - \delta$ for any non-zero constant δ can be amplified to a standard NIZK argument with negligible errors if we additionally have access to a sub-exponentially secure Public Key Encryption (PKE) scheme.

Bitansky and Geier [BG24] improved this result to show that standard PKE suffices for this amplification. They also showed that if the NIZK system is a proof (i.e., has

statistical soundness), then amplification is possible assuming only OWFs. Applebaum and Kachlon [AK25] improved on this to allow for δ to be as small as an inverse-polynomial function.

Hardness from Weak ZK Seeking to reduce the assumptions needed for such amplification, Chakraborty et al. [CHK25] showed that in certain settings, weak NIZKs can be used to derive OWFs. Specifically, they show that if all languages in NP have NIZK arguments satisfying $\epsilon_c + \sqrt{\epsilon_s} + \epsilon_z < 1$, then OWFs exist under just the worst-case assumption that $\mathsf{NP} \not\subseteq \mathsf{ioP/poly}$[1]. They then combined this with the amplification results of [BG24, AK25] to show that under the additional hypotheses that these are NIZK proofs with negligible ε_c, they could get NIZK proofs with negligible errors for all of NP.

While this result helps characterize the hardness of a broad class of weak NIZKs, it is still somewhat unsatisfactory as it does not cover *all* possible non-trivial weak NIZK parameters. As noted in [CHK25], the setting $\epsilon_c + \epsilon_s + \epsilon_z \geq 1$ is not meaningful for NIZK protocols. Thus, to complete the picture, what is required is to understand the implications of any weak NIZK protocol satisfying $\epsilon_c + \epsilon_s + \epsilon_z < 1$.

Another important question that has not been studied in this recent line of work is that of the complexity of weak interactive ZK protocols. The earlier implications of ZK shown in [OW93, HN24] work for interactive ZK protocols with negligible errors. But the extent of their validity for weak ZK protocols was not understood.

Our Results Our first result addresses the first question above, extending the construcion of OWFs from NIZKs to the most general parameters, under the same assumptions as in prior work.

Theorem 1.1 (Informally, Theorem 4.1)**.** *If $\mathsf{NP} \not\subseteq \mathsf{ioP/poly}$, and every language in NP has an $(\epsilon_c, \epsilon_s, \epsilon_z)$-NIZK proof (or argument) with $\epsilon_c + \epsilon_s + \epsilon_z < 1$, then one-way functions exist.*

Similar to [CHK25], we can in turn use the amplification results of [BG24, AK25] to obtain amplification of NIZK proofs with near-perfect completeness, from the most general setting of the remaining errors.

Corollary 1.2 (NIZK Amplification)**.** *If $\mathsf{NP} \not\subseteq \mathsf{ioP/poly}$, and every language in NP has an $(\epsilon_c, \epsilon_s, \epsilon_z)$-NIZK proof with $\epsilon_c + \epsilon_s + \epsilon_z < 1$ and negligible ϵ_c, then every language in NP has a NIZK proof with negligible errors. Here, the soundness of the NIZK proofs is required to be adaptive.*[2]

[1] That is, there are no polynomial-size circuit families for NP, even if they are only required to be correct infinitely often

[2] In the rest of the paper, we exclusively use the weaker non-adaptive definition of soundness for NIZK protocols. This only strengthens our results, as we use NIZKs to construct other things. Here, the stronger adaptive notion of soundness is needed. See, e.g., [BG24] for the definition of this notion.

We also address the second question raised above for *public-coin* protocols, where the verifier's messages consist solely of uniform random bits.

Theorem 1.3 (Informally, Theorem 4.2)**.** *If* $\mathsf{NP} \not\subseteq \mathsf{ioP/poly}$*, and every language in* NP *has a* t*-message* $(\epsilon_c, \epsilon_s, \epsilon_z)$*-public-coin ZK proof (or argument) with* $\epsilon_c + \epsilon_s + (t-1) \cdot \epsilon_z < 1$*, then one-way functions exist.*

In the above case, the techniques of [OW93] alone would have resulted in the condition being $\epsilon_c + \epsilon_s + t \cdot \epsilon_z < 1$ instead. For constant-round protocols, we improve this condition much more significantly, though in this case we only obtain infinitely-often one-way functions. Below, a *round* refers to one pair of messages in the protocol – one from the verifier and its response from the prover.

Theorem 1.4 (Informally, Theorem 4.3)**.** *If* $\mathsf{NP} \not\subseteq \mathsf{P/poly}$*, and for some constant* k*, every language in* NP *has a* k*-round* $(\epsilon_c, \epsilon_s, \epsilon_z)$*-public-coin ZK proof (or argument) with* $\epsilon_c + \epsilon_s + k \cdot \epsilon_z < 1$*, then infinitely-often one-way functions exist.*

Our results apply to protocols with computational (weak) zero-knowledge and computational (weak) soundness (i.e., arguments), and thus capture the most general class of such protocols.

Open problems Our work leaves open interesting questions around the power of weak zero knowledge systems. We mention some of these below.

- An obvious question is if our analysis can be carried over to the setting of private-coin weak ZK protocols – the standard ZK to OWF implications hold for such protocols as well, and it is of interest to achieve parity here in the weak ZK setting.
- Our final result works for better parameters but only implies *infinitely often* OWFs. This is a limitation of our approach, and it is interesting to improve this to yield standard OWFs.
- A related improvement is to also get improved error parameters for super-constant-round protocols. This will also require new tools or approaches.
- Additionally, it is an exciting problem to consider what other, possibly stronger cryptographic primitives weak ZK or more generally even standard ZK protocols may imply.

Paper outline We continue with a technical overview of our results and proofs in Section 1.1. Section 2 contains our definitions and notation. The first stage of our results are covered in Section 3, which shows how the various kinds of weak ZK protocols we consider yield (variants of) auxiliary input one-way functions. The final implications to one-way functions are shown in Section 4.

1.1 Technical Overview

In this section, we provide a high-level overview of the main ideas and techniques behind our results. We start by reviewing the construction of [OW93] with their analysis for non-interactive ZKs. We then introduce our improved construction and outline the ideas that enable improvement. Further, we find that our approach naturally generalizes to a broader setting – specifically that of public-coin ZK protocols, which we will discuss later.

Throughout, we use the notation $\mathcal{U}$ to denote the uniform distribution over strings whose length will be clear from the context.

Non-interactive zero-knowledge We first consider non-interactive ZK (NIZK) arguments. A NIZK argument for a language $\mathcal{L}$ allows a prover to produce a single proof π to certify that an input $x \in \mathcal{L}$ while preserving zero-knowledge. Specifically, given a uniformly random string r, also known as the *common reference string*, a polynomial-time prover holding with a valid NP witness w for x computes a proof $\pi \leftarrow \mathsf{P}(x, w; r)$; upon receiving the prover's message, the verifier computes $a \leftarrow \mathsf{V}(x; r, \pi)$ to decide whether $x \in \mathcal{L}$.

The protocol is required to satisfy completeness and computational soundness, where the errors are correspondingly denoted by ϵ_c and ϵ_s. Besides these, it also satisfies computational zero-knowledge. In particular, there is a polynomial-time simulator Sim that on input x generates a distribution (r, π) such that no polynomial-time distinguisher can distinguish between this and the (r, π) from the actual protocol with advantage greater than the zero-knowledge error ϵ_z. For simplicity, we assume that the NIZK protocol has perfect completeness ($\epsilon_c = 0$) and we have a deterministic verification algorithm V.

The Ostrovsky-Wigderson approach The key observation underlying [OW93] is that an inverter for the NIZK simulator can be used to construct a distinguisher to decide the language, contradicting its hardness.

Suppose the language $\mathcal{L}$ that has the NIZK argument $(\mathsf{P}, \mathsf{V}, \mathsf{Sim})$ is worst-case hard. Let Sim be the simulator that runs on the input x and randomness ρ, and outputs the common reference string r and a proof π. The candidate hard-to-invert function f_x is defined to be

$$\begin{aligned} f_x(\rho): \quad & (r, \pi) \leftarrow \mathsf{Sim}(x; \rho) \\ & \text{output } r \end{aligned}$$

Since the randomness is treated explicitly as part of the input, the above construction is deterministic and therefore the function is well defined.

Towards a contradiction, assume that there are no auxiliary-input one-way functions. In fact, suppose that there is an adversary $\mathcal{A}$ that perfectly inverts f_x distributionally – that is, given y, $\mathcal{A}$ samples a uniformly random pre-image $f_x^{-1}(y)$. It is known that distributional OWFs imply OWFs [IL89], so the only loss of generality here is the assumption that the

inverter is perfect, but this is not difficult to remove at the cost of an inverse polynomial loss in parameters.

We have that the joint distributions

$$(\mathcal{A}(f_x(\mathcal{U})), f_x(\mathcal{U})) \approx (\mathcal{U}, f_x(\mathcal{U}))$$

are close, where the left-hand side represents the distribution of first computing f_x on random inputs and applying $\mathcal{A}$, while the right-hand side denotes the distribution of sampling a random input r and then outputting $(r, f_x(r))$.

The algorithm $\mathcal{D}$ that decides $\mathcal{L}$ is as follows. Given input x, it invokes $\mathcal{A}$ to decide whether $x \in \mathcal{L}$ or not: it samples $r \leftarrow \mathcal{U}$, runs $\hat{\rho} \leftarrow \mathcal{A}(r)$, then computes $(\hat{r}, \hat{\pi}) \leftarrow \mathsf{Sim}(x; \hat{\rho})$, and accepts if and only if $\mathsf{V}(x; r, \hat{\pi}) = 1$. Note that under our assumptions, we will always have $r = \hat{r}$. We now analyze the performance of $\mathcal{D}$ in the two possible cases.

1. When $x \in \mathcal{L}$, completeness implies that when (r, π) is generated following the protocol, $\mathsf{V}(x; r, \pi) = 1$ holds with probability 1. Zero-knowledge guarantees that the probability that $\mathsf{V}(x; r, \pi) = 1$ when $(r, \pi) \leftarrow \mathsf{Sim}(x; \mathcal{U})$ is at least $1 - \epsilon_z$.

   In the algorithm $\mathcal{D}(x)$, we run V on (r, π) generated from $\mathsf{Sim}(x; \mathcal{A}(\mathcal{U}))$. When the inverter $\mathcal{A}$ is run on r sampled from $\mathsf{Sim}(x; \mathcal{U})$, the resulting inverse is uniformly random (due to the definition of f_x). Again by zero-knowledge, the uniform distribution of r is ϵ_z-indistinguishable from the distribution of r sampled by $\mathsf{Sim}(x; \mathcal{U})$. So the distribution of $\mathcal{A}(\mathcal{U})$ is also ϵ_z-indistinguishable from uniform. This implies that the distribution of $\mathsf{Sim}(x; \mathcal{A}(\mathcal{U}))$ is ϵ_z-indistinguishable from $\mathsf{Sim}(x; \mathcal{U})$. Altogether, we have

   $$\Pr[\mathcal{D}(x) = 1] \geq 1 - 2\epsilon_z.$$

2. When $x \notin \mathcal{L}$, soundness ensures that for any efficient method of generating π for random r, the probability that V accepts is bounded by ϵ_s, and so

   $$\Pr[\mathcal{D}(x) = 1] \leq \epsilon_s.$$

Consequently, if $\epsilon_s < 1 - 2\epsilon_z$, the inverter $\mathcal{A}$ can be used to determine whether x is in $\mathcal{L}$. If such an inverter works for every x, then we can decide the language, contradicting its hardness. Thus, the family of functions $\{f_x\}$ must be a family of auxiliary-input one-way functions.

Improving the condition on errors More recently, such analysis has seen further progress. In particular, this bound was improved by [CHK25], where it was shown that $\sqrt{\epsilon_s} + \epsilon_z < 1$ suffices. This was shown with sophisticated arguments using a one-sided version of universal approximation, where the inverter is used to estimate the probabilities of certain outputs.

Our starting point is the observation that with rather simple but careful arguments, it is feasible to relax this bound to $\epsilon_s + \epsilon_z < 1$, which is the most general it can be. We avoid paying for the zero-knowledge error twice by involving the verification procedure inside the candidate one-way function. Our one-way function is as follows

$$\begin{aligned} f_x(\rho): \quad & (r, \pi) \leftarrow \mathsf{Sim}(x; \rho) \\ & a \leftarrow \mathsf{V}(x; r, \pi) \\ & \text{output } (r, a) \end{aligned}$$

Suppose again that there is a near-perfect polynomial-time inverter $\mathcal{A}$ for f_x (it need not be a distributional inverter). That is,

$$\Pr_{(r,a) \leftarrow f_x(\mathcal{U})} [f_x(\mathcal{A}(r, a)) = (r, a)] \approx 1,$$

The algorithm $\mathcal{D}$ for $\mathcal{L}$ can be constructed as follows. Given input x, sample r uniformly at random, compute $\hat{\rho} \leftarrow \mathcal{A}(r, 1)$, and accept if and only if this is a valid pre-image of $(r, 1)$ under f_x – that is, iff $f_x(\hat{\rho}) = (r, 1)$.

1. For $x \in \mathcal{L}$, consider the following procedure $g(r, \pi)$: it computes $a \leftarrow \mathsf{V}(x; r, \pi)$ and outputs (r, a). When (r, π) is sampled from the simulator $\mathsf{Sim}(x; \mathcal{U})$, the distribution of $g(r, \pi)$ is equivalent to $f_x(\mathcal{U})$. When (r, π) follows the protocol view, the distribution of $g(r, \pi)$ is the same as $(r, 1)$ in $\mathcal{D}$. So by ZK and the data processing inequality, these distributions are ϵ_z-indistinguishable.

   Further, when (r, π) is sampled from the simulator, the inverter $\mathcal{A}$ will almost always find a valid pre-image of (r, a). Therefore, given $(r, 1)$ as in $\mathcal{D}$, it finds a valid pre-image with overall probability at least $\approx 1 - \epsilon_z$.

   $$\Pr[\mathcal{D}(x) = 1] = \Pr_{r \leftarrow \mathcal{U}} [f_x(\mathcal{A}(r, 1)) = (r, 1)] \geq 1 - \epsilon_z.$$

2. For $x \notin \mathcal{L}$, whenever a valid pre-image $\hat{\rho}$ for $(r, 1)$ is found, the corresponding $(r, \pi) \leftarrow \mathsf{Sim}(x; \hat{\rho})$ is accepted by V. As both $\mathcal{A}$ and Sim are efficient algorithms, soundness guarantees that a valid inverse cannot be found with probability more than ϵ_s. So

   $$\Pr[\mathcal{D}(x) = 1] \leq \epsilon_s.$$

Therefore, following the same remaining arguments as earlier, $\epsilon_s + \epsilon_z < 1$ suffices to imply the existence of auxiliary-input one-way functions.

The key idea behind our improvement is that we implicitly utilize the verification V while restricting the inverter to output a good pre-image corresponding to a valid proof. Thus as long as the inverter succeeds, there is no need to perform an additional explicit verification. This saves us the extra zero-knowledge error penalty.

The distinguisher: an alternate view Our analysis reveals that in both the [OW93] construction and our new one, these distinguishers can be regarded as providing efficient simulations of the protocol, where the original honest prover algorithm is replaced with an efficient algorithm without the witness and V serves to certify correctness. For example, the Ostrovsky-Wigderson distinguisher admits an equivalent formulation as a protocol between a prover $\tilde{\mathsf{P}}$ and the verifier V. The prover $\tilde{\mathsf{P}}$ performs: on randomness r, find the message $\tilde{\pi}$ corresponding to the simulator randomness $\tilde{\rho}$, where $\tilde{\rho}$ is found efficiently by the inverter. Our construction yields a similar prover as well. The only difference is the modification to the inverter since the correctness of a pre-image ensures the validity of corresponding proof.

Multiple-round public-coin zero-knowledge Our approach naturally extends to the setting of multiple-round public-coin zero-knowledge protocols. Such a protocol allows interaction between the prover and the verifier, where all of the verifier's communication consists of uniform random bits. In particular, all its randomness is public and available to the prover.

For the sake of exposition, we assume below perfect completeness with soundness error ϵ_s and zero-knowledge error ϵ_z. By adapting our approach for the NIZK case, we obtain that the existence a k-round public-coin ZK protocol for a hard language $\mathcal{L}$ with parameters $\epsilon_s + (2k-1)\epsilon_z < 1$ implies the existence of one-way functions. We use the term *round* to denote one interaction where the verifier sends a random challenge and the prover responds with a proof message.

Better bounds for constant rounds Moreover, the bound can be further improved to $\epsilon_s + k \cdot \epsilon_z < 1$ using a more sophisticated argument when k is only a constant.

For illustration, we focus here on the simplest setting in which the language $\mathcal{L}$ admits a two-round public-coin ZK protocol. Here both parties share a common input x while the honest prover additionally holds a witness w. In the first round, the verifier first samples a random string r_1 and the prover replies with a message $\pi_1 \leftarrow \mathsf{P}_1(x, w; r_1)$. In the second round, the verifier sends another random string r_2 and the prover responds with $\pi_2 \leftarrow \mathsf{P}(x, w; r_1, \pi_1, r_2)$. Finally, the verifier checks the transcript by computing $0/1 \leftarrow \mathsf{V}(x; r_1, \pi_1, r_2, \pi_2)$, where 1 denotes acceptance.

The zero-knowledge condition implies the existence of a simulator Sim that on input x and randomness ρ, outputs a transcript (r_1, π_1, r_2, π_2); for any $x \in \mathcal{L}$ and a valid witness w, $\mathsf{Sim}(x)$ is ϵ_z-indistinguishable from the protocol view.

$$\Delta(\mathsf{Sim}(x); \langle \mathsf{P}, \mathsf{V} \rangle(x, w)) \leq \epsilon_z.$$

A recursive approach In the following, we will follow a recursive approach to construct our candidate one-way functions. More precisely, we begin by defining a function, and any algorithm that breaks the one-wayness of this function will be incorporated into the

construction of a second function. If the second one is not one-way either, then the resulting inverters can be combined to decide the language. Accordingly, we first define a function $f_{2,x}$ as follows:

$$\begin{aligned} f_{2,x}(\rho): \quad & (r_1, \pi_1, r_2, \pi_2) \leftarrow \mathsf{Sim}(x; \rho) \\ & a \leftarrow \mathsf{V}(x; r_1, \pi_1, r_2, \pi_2) \\ & \text{output } (r_1, \pi_1, r_2, a) \end{aligned}$$

Assume that there is a poly-time algorithm $\mathcal{A}_2$ that inverts $f_{2,x}$

$$\Pr_{(r_1,\pi_1,r_2,a)\leftarrow f_{2,x}(\mathcal{U})} [f_{2,x}(\mathcal{A}_2(x; r_1, \pi_1, r_2, a)) = (r_1, \pi_1, r_2, a)] \approx 1. \tag{1}$$

As in the NIZK case, the deciding procedure is essentially equivalent to efficiently simulating the protocol by constructing an efficient prover $\tilde{\mathsf{P}}$: this queries the inverter $\mathcal{A}$ on input $(r, 1)$ and generates the prover's message π accordingly. For the second round of the protocol, we define the strategy $\tilde{\mathsf{P}}_2$ similarly.

$$\begin{aligned} \tilde{\mathsf{P}}_2(x; r_1, \pi_1, r_2): \quad & \tilde{\rho} \leftarrow \mathcal{A}_2(x; r_1, \pi_1, r_2, 1) \\ & (\tilde{r}_1, \tilde{\pi}_1, \tilde{r}_2, \tilde{\pi}_2) \leftarrow \mathsf{Sim}(x; \tilde{\rho}) \\ & \text{output } \tilde{\pi}_2 \end{aligned}$$

We measure the performance of $\tilde{\mathsf{P}}_2$ formally by the following quantity:

$$\mathsf{Succ}_3(x; r_1, \pi_1, r_2) = 1[f_{2,x}(\mathcal{A}_2(x; r_1, \pi_1, r_2, 1)) = (r_1, \pi_1, r_2, 1)]$$

where $\mathcal{A}_2$ is assumed to be deterministic for simplicity. The value of Succ_3 represents the probability that $\mathcal{A}_2$ finds a valid pre-image with respect to $f_{2,x}$. This provides a lower bound for the acceptance probability of the protocol conditioned on the first 3 messages being (r_1, π_1, r_2), since $\tilde{\pi}_2$ is valid as long as $\mathcal{A}_2$ finds a valid pre-image of $f_{2,x}$.

$$f_{2,x}(\tilde{\rho}) = (r_1, \pi_1, r_2, 1) \Rightarrow \mathsf{V}(x; r_1, \pi_1, r_2, \tilde{\pi}_2) = 1.$$

Since the verifier samples r_2 uniformly, we can equivalently lower bound the success probability of this prover by the following expression (where the first two messages are (r_1, π_1) and $\tilde{\mathsf{P}}_2$ is the second round strategy of the prover):

$$\mathsf{Succ}_2(x; r_1, \pi_1) = \mathop{\mathbb{E}}_{r_2\leftarrow\mathcal{U}} [\mathsf{Succ}_3(x; r_1, \pi_1, r_2)]$$

Now consider the protocol $\langle \mathsf{P}, \mathsf{V} \rangle$ that replaces the prover algorithm in the second round with $\tilde{\mathsf{P}}_2$. We can establish an upper bound for the completeness error of this modified protocol.

Denote by $D_{2,x}$ the distribution that samples (r_1, π_1, r_2, π_2) from the protocol $\langle \mathsf{P}, \mathsf{V} \rangle(x, w)$, sets $a \leftarrow \mathsf{V}(x; r_1, \pi_1, r_2, \pi_2)$ and outputs (r_1, π_1, r_2, a). By the perfect completeness of $\langle \mathsf{P}_w, \mathsf{V} \rangle$, $a = 1$ always holds. It follows by the data processing inequality that, for $x \in \mathcal{L}$

$$\Delta(f_{2,x}(\mathcal{U}); D_{2,x}) \leq \Delta(\mathsf{Sim}(x); \langle \mathsf{P}, \mathsf{V} \rangle(x)) \leq \epsilon_z.$$

Combining above with our assumption (1), the acceptance probability is at least

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\substack{r_1 \leftarrow \mathcal{U} \\ \pi_1 \leftarrow \mathsf{P}_1(x,w;r_1)}} [\mathsf{Succ}_2(x; r_1, \pi_1)] \\
&= \Pr_{(r_1,\pi_1,r_2,a) \leftarrow D_{2,x}} [f_{2,x}(\mathcal{A}_2(x; r_1, \pi_1, r_2, a)) = (r_1, \pi_1, r_2, a)] \\
&\geq \Pr_{(r_1,\pi_1,r_2,a) \leftarrow f_{2,x}(\mathcal{U})} [f_{2,x}(\mathcal{A}_2(x; r_1, \pi_1, r_2, a)) = (r_1, \pi_1, r_2, a)] - \epsilon_z \\
&\geq 1 - \epsilon_z. \qquad (2)
\end{aligned}$$

The above implies that when the prover applies $\mathsf{P}_1(x, w)$ and $\tilde{\mathsf{P}}_2(x)$ in each round, respectively, the acceptance probability is at least $1 - \epsilon_z$ when $x \in \mathcal{L}$. However, because the witness for P_1 is unavailable to our deciding strategy, we need a different polynomial-time algorithm as a replacement to obtain an efficient simulation for the protocol.

We now turn to building our first-round strategy. A natural attempt is to define a new function $f_{1,x}(\rho)$ analogously to $f_{2,x}$ and apply the inverter to produce the message π_1, where $f_{1,x}(\rho)$ can be defined as follows: sample $(r_1, \pi_1, r_2, \pi_2) \leftarrow \mathsf{Sim}(x; \rho)$, $a \leftarrow \mathsf{V}(x; r_1, \pi_1, r_2, \pi_2)$, and output (r_1, a) (or only output r_1). However, our analysis of the resulting prover algorithms shows that the best achievable result is $\epsilon_s + 3\epsilon_z < 1$, which matches $\epsilon_s + (2k - 1)\epsilon_z < 1$ for $k = 2$. We refer the reader to Section 3.2 for the details.

Instead, we introduce a new construction of $f_{1,x}$ that leads to an improved bound. After fixing $\tilde{\mathsf{P}}_2$, our objective is to find an efficient way $\tilde{\mathsf{P}}_1$ to generate π_1 that maximizes the value $\mathsf{Succ}_2(x; r_1, \pi_1)$ optimally for $x \in \mathcal{L}$. The value of Succ_2 can be efficiently approximated up to any arbitrary inverse-polynomial error due to our assumption on $\mathcal{A}_2$ and standard concentration arguments. For convenience, we ignore the accuracy issue here and assume that we are able to compute Succ_2 precisely in polynomial-time. The key is to instead have the new function compute Succ_2 for the relevant partial transcript. More precisely, we define:

$$\begin{aligned}
f_{1,x}(\rho): \quad & (r_1, \pi_1, r_2, \pi_2) \leftarrow \mathsf{Sim}(x; \rho) \\
& \text{output } (r_1, \mathsf{Succ}_2(x; r_1, \pi_1))
\end{aligned}$$

Suppose now that there is an efficient algorithm $\mathcal{A}_1$

$$\Pr_{(r_1,a) \leftarrow f_{1,x}(\mathcal{U})} [f_{1,x}(\mathcal{A}_1(x; r_1, a)) = (r_1, a)] \approx 1.$$

We are now ready to formally specify $\tilde{\mathsf{P}}_1$, which works as follows:

$$\begin{aligned}\tilde{\mathsf{P}}_1(x; r_1): \quad & \tilde{a} \leftarrow \arg\max_a \{a \cdot \mathbf{1}[f_{1,x}(\mathcal{A}_1(x; r_1, a)) = (r_1, a)\}] \\ & \tilde{\rho} \leftarrow \mathcal{A}_1(x; r_1, \tilde{a}) \\ & \tilde{\pi}_1 \leftarrow \mathsf{Sim}(\tilde{\rho}) \\ & \text{output } \tilde{\pi}_1\end{aligned}$$

where we take the support size of a to be polynomial, enabling us to efficiently iterate over all possible values and find the maximum $\tilde{a}$. In fact, in general one can efficiently find an approximate maximum value instead, which suffices.

On input $(x; r_1)$, the prover $\tilde{\mathsf{P}}_1$ goes through all possible values of a, and selects the highest one on which $\mathcal{A}_1$ inverts $f_{1,x}$ successfully. Note that when $\mathcal{A}_1$ finds a correct pre-image of (r_1, a), it implies that there exists a consistent π_1 with value $\mathsf{Succ}_2(x; r_1, \pi_1) = a$; so $\tilde{\mathsf{P}}_1$ takes the maximal a and outputs its associated prover message.

With the construction of $\tilde{\mathsf{P}}_1$ and $\tilde{\mathsf{P}}_2$, it remains to analyze the performance of the resulting protocol $\langle \tilde{\mathsf{P}}, \mathsf{V} \rangle$. More specifically, we are interested in

$$\mathop{\mathbb{E}}_{\substack{r_1 \leftarrow \mathcal{U}, \pi_1 \leftarrow \tilde{\mathsf{P}}_1(x; r_1) \\ r_2 \leftarrow \mathcal{U}, \pi_2 \leftarrow \tilde{\mathsf{P}}_2(x; r_1, \pi_1, r_2)}} [\mathsf{V}(x; r_1, \pi_1, r_2, \pi_2)] \tag{3}$$

Recall that

$$\begin{aligned}(3) &\geq \mathop{\mathbb{E}}_{\substack{r_1 \leftarrow \mathcal{U} \\ \pi_1 \leftarrow \tilde{\mathsf{P}}_1(x; r_1)}} [\mathsf{Succ}_2(x; r_1, \pi_1)] \\ &= \mathop{\mathbb{E}}_{r_1 \leftarrow \mathcal{U}} \left[\max_a \{a \cdot \mathbf{1}[f_{1,x}(\mathcal{A}_1(x; r_1, a)) = (r_1, a)]\}\right] \end{aligned} \tag{4}$$

where the last equality is ensured since by definition of $\tilde{\mathsf{P}}_1(x; r_1)$, the Succ_2 estimate a associated with π_1 is the largest such value with respect to which $\mathcal{A}_1$ can succeed, i.e.,

$$\mathsf{Succ}_2(x; r_1, \pi_1) = \max_a \{a \cdot \mathbf{1}[f_{1,x}(\mathcal{A}_1(x; r_1, a)) = (r_1, a)]\}.$$

Now observe that, for any r_1 and any particular a^*

$$\max_a \{a \cdot \mathbf{1}[f_{1,x}(\mathcal{A}_1(x; r_1, a)) = (r_1, a)]\} \geq a^* \cdot \mathbf{1}[f_{1,x}(\mathcal{A}_1(x; r_1, a^*)) = (r_1, a^*)]. \tag{5}$$

Let a^* above be sampled as $a^* \leftarrow \mathsf{Succ}_2(x; r_1, \pi_1)$ where $\pi_1 \leftarrow \mathsf{P}_1(x, w; r_1)$. We now investigate the new completeness error to obtain a lower bound for (4) when $x \in \mathcal{L}$. Note

that the value of Succ_2 falls in the range $[0,1]$, thus we have

$$\begin{aligned}(4) &\geq \mathop{\mathbb{E}}_{\substack{r_1\leftarrow\mathcal{U},\pi_1\leftarrow\mathsf{P}_1(x,w;r_1)\\ a\leftarrow\mathsf{Succ}_2(x;r_1,\pi_1)}} [a\cdot\mathbf{1}[f_{1,x}(\mathcal{A}_1(x;r_1,a))=(r_1,a)]]\\ &\geq \mathop{\mathbb{E}}_{\substack{r_1\leftarrow\mathcal{U},\pi_1\leftarrow\mathsf{P}_1(x,w;r_1)\\ a\leftarrow\mathsf{Succ}_2(x;r_1,\pi_1)}} [a] - \mathop{\mathbb{E}}_{\substack{r_1\leftarrow\mathcal{U},\pi_1\leftarrow\mathsf{P}_1(x,w;r_1)\\ a\leftarrow\mathsf{Succ}_2(x;r_1,\pi_1)}} [\mathbf{1}[f_{1,x}(\mathcal{A}_1(x;r_1,a))\neq(r_1,a)]]\\ &\geq 1-\epsilon_z - \Pr_{\substack{r_1\leftarrow\mathcal{U},\pi_1\leftarrow\mathsf{P}_1(x,w;r_1)\\ a\leftarrow\mathsf{Succ}_2(x;r_1,\pi_1)}} [f_{1,x}(\mathcal{A}_1(x;r_1,a))\neq(r_1,a)]\,, \end{aligned} \tag{6}$$

where the first inequality is obtained by the observation (5) and the last inequality holds by (2). Now define the distribution $D_{1,x}$: sample $r_1\leftarrow\mathcal{U}$ and $\pi_1\leftarrow\mathsf{P}_1(x,w;r_1)$, and finally output $(r_1,\mathsf{Succ}_2(x;r_1,\pi_1))$.

For $x\in\mathcal{L}$, by the data processing inequality and zero-knowledge condition

$$\Delta(f_{1,x}(\mathcal{U});D_{1,x})\leq\Delta(\mathsf{Sim}(x);\langle\mathsf{P}_w,\mathsf{V}\rangle(x))\leq\epsilon_z.$$

Since we assume that $\mathcal{A}_1$ inverts the function almost perfectly, we get

$$\begin{aligned}&\Pr_{\substack{r_1\leftarrow\mathcal{U},\pi_1\leftarrow\mathsf{P}_1(x,w;r_1)\\ a\leftarrow\mathsf{Succ}_2(x;r_1,\pi_1)}} [f_{1,x}(\mathcal{A}_1(x;r_1,a))\neq(r_1,a)]\\ &\leq \Pr_{(r_1,a)\leftarrow f_{1,x}(\mathcal{U})} [f_{1,x}(\mathcal{A}_1(x;r_1,a))\neq(r_1,a)]+\epsilon_z\\ &\leq \epsilon_z. \end{aligned} \tag{7}$$

Therefore, by combining (3), (6) and (7), we derive that for $x\in\mathcal{L}$

$$\mathop{\mathbb{E}}_{\substack{r_1\leftarrow\mathcal{U},\pi_1\leftarrow\tilde{\mathsf{P}}_1(x;r_1)\\ r_2\leftarrow\mathcal{U},\pi_2\leftarrow\tilde{\mathsf{P}}_2(x;r_1,\pi_1,r_2)}} [\mathsf{V}(x;r_1,\pi_1,r_2,\pi_2)]\geq 1-2\epsilon_z.$$

On the other hand, soundness of the original protocol $\langle\mathsf{P}_w,\mathsf{V}\rangle$ ensures that, for $x\notin\mathcal{L}$

$$\mathop{\mathbb{E}}_{\substack{r_1\leftarrow\mathcal{U},\pi_1\leftarrow\tilde{\mathsf{P}}_1(x;r_1)\\ r_2\leftarrow\mathcal{U},\pi_2\leftarrow\tilde{\mathsf{P}}_2(x;r_1,\pi_1,r_2)}} [\mathsf{V}(x;r_1,\pi_1,r_2,\pi_2)]\leq\epsilon_s.$$

When $\epsilon_s+2\epsilon_z$ is noticeably less than 1, we thus obtain an efficient algorithm that decides if $x\in\mathcal{L}$, which yields the desired result.

One catch here is that while this approach is conceptually complete, it only gives us a construction of an infinitely often one-way function. This is because in our approach, we will require that the adversaries $\mathcal{A}_1$ and $\mathcal{A}_2$ must both succeed in their inversion so that we successfully decide on an instance. This means that the sequence of input lengths on which our assumed inverters work must overlap when we aim for a contradiction - and the formal negation for this only implies infinitely-often hardness. See Section 4.2.2 and Remark 3.3 for details.

Handling randomized verification in NIZKs We note that it requires non-trivial techniques to derive a similar bound for the errors of NIZK protocols when the verification step V is randomized. Note that it is feasible to deal with the randomized case by adapting the approach in the multi-round public-coin case. We briefly describe the construction for the NIZK case below. The construction of candidate function is analogous to $f_{1,x}$ defined before. In particular,

$$\begin{aligned} f_x(\rho): \quad & (r,\pi) \leftarrow \mathsf{Sim}(x;\rho) \\ & \text{output } (r, \mathbb{E}[\mathsf{V}(x;r,\pi)]) \end{aligned}$$

We also assume for now that one can compute this expected value deterministically in polynomial time. Given an efficient inverter $\mathcal{A}$, we can define an efficient prover strategy that runs (without any witness) as follows:

$$\begin{aligned} \tilde{\mathsf{P}}(x;r): \quad & \tilde{a} \leftarrow \arg\max_a \{a \cdot \mathbf{1}[f_x(\mathcal{A}(x;r,a)) = (r,a)\}] \\ & \tilde{\rho} \leftarrow \mathcal{A}(x;r,\tilde{a}) \\ & \tilde{\pi} \leftarrow \mathsf{Sim}(x;\tilde{\rho}) \\ & \text{output } \tilde{\pi} \end{aligned}$$

From a similar argument, for $x \in \mathcal{L}$, the acceptance probability of $\langle \tilde{\mathsf{P}}, \mathsf{V} \rangle$ is at least

$$\begin{aligned} & \mathop{\mathbb{E}}_{\substack{r_1 \leftarrow \mathcal{U}, \pi \leftarrow \mathsf{P}(x,w;r) \\ a \leftarrow \mathbb{E}[\mathsf{V}(x;r,\pi)]}} [a \cdot \mathbf{1}[f_x(\mathcal{A}(x;r,a)) = (r,a)]] \\ \geq\ & 1 - \Pr_{\substack{r \leftarrow \mathcal{U}, \pi \leftarrow \mathsf{P}(x,w;r) \\ a \leftarrow \mathbb{E}[\mathsf{V}(x;r,\pi)]}} [f_x(\mathcal{A}(x;r,a)) \neq (r,a)] \\ \geq\ & 1 - \epsilon_z. \end{aligned}$$

For $x \notin \mathcal{L}$, soundness holds with error probability ϵ_s. Therefore, we conclude that the condition $\epsilon_s + \epsilon_z < 1$ suffices when the verification is randomized as well.

2 Preliminaries

Notations Denote by $\mathcal{U}_\ell$ the uniform distribution over the length-ℓ binary strings $\{0,1\}^\ell$. For a language $\mathcal{L}$, let $\mathcal{L}_n = \mathcal{L} \cap \{0,1\}^n$ be the set of all the length-n strings in the language. For $k \in \mathbb{N}$, denote $[k] = \{1,\ldots,k\}$. We define a distribution ensemble $\mathcal{X} = \{X_n\}_{n\in\mathbb{N}}$ to be a collection of distributions where each X_n is defined over $\{0,1\}^{m(n)}$ where $m : \mathbb{N} \to \mathbb{N}$ is some arithmetic function. We say a function ν is negligible if for every polynomial p there exists an $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$, $\mu(n) < \frac{1}{p(n)}$. Similarly, we call a function μ noticeable if there exists a polynomial p and $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$, $\mu(n) \geq \frac{1}{p(n)}$.

For $\epsilon_1 = \epsilon_1(n), \epsilon_2 = \epsilon_2(n)$, we say $\epsilon_1 <_n \epsilon_2$ (or $\epsilon_2 >_n \epsilon_1$) to represent the noticeable gap between ϵ_1 and ϵ_2, if there exists a polynomial p such that for all sufficiently large $n \in \mathbb{N}$, we have

$$\epsilon_1(n) + \frac{1}{p(n)} < \epsilon_2(n),$$

which implies asymptotically, there is an inverse-polynomial gap between ϵ_1 and ϵ_2.

We use the Hoeffding bound, stated as follows.

Lemma 2.1 (Hoeffding's inequality)**.** *Suppose we have independent random variables $X_1, \ldots, X_q$ with support $[0,1]$. Let $X = \sum_{i \in [q]} X_i$, then the following holds*

$$\Pr\left[|X - \mathbb{E}X| > t \cdot q\right] \leq 2e^{-2t^2 q}.$$

2.1 Indistinguishability

Assume that we have two distributions X and Y defined over a common universe U. We first consider statistical indistinguishability.

Definition 2.1 (Statistical Distance)**.** The *statistical distance* between distributions X and Y (defined over the support of X, denoted by $\mathsf{Supp}(X)$) is defined as

$$\Delta_s(X;Y) = \frac{1}{2} \sum_{u \in \mathsf{Supp}(X)} \big|\Pr[X = u] - \Pr[Y = u]\big|.$$

The following properties hold.

Lemma 2.2 (Triangle Inequality)**.** *For any three distributions X, Y and Z, we have*

$$\Delta_s(X;Z) \leq \Delta_s(X;Y) + \Delta_s(Y;Z).$$

Lemma 2.3 (Data Processing Inequality)**.** *For any two probability distributions X, Y (on a common universe U), and any (possibly randomized) process f, we have*

$$\Delta_s(f(X); f(Y)) \leq \Delta_s(X;Y).$$

Suppose that we have a given distinguishing algorithm D to distinguish between X and Y, taking inputs in U and outputting a bit to indicate whether it identifies a given input as being sampled from X or Y. Define the *distinguishing advantage* $\mathsf{Adv}_D(X,Y)$ of D as:

$$\mathsf{Adv}_D(X;Y) = \left| \Pr_{x \leftarrow X}[D(x) = 1] - \Pr_{y \leftarrow Y}[D(y) = 1] \right|.$$

In fact, the statistical distance implicitly provides an upper bound on the advantage that any distinguisher can obtain, that is

$$\Delta_s(X;Y) = \max_D \mathsf{Adv}_D(X;Y)$$

where D can be any possible algorithm. Usually, for a constant ϵ, when $\Delta_s(X;Y) < \epsilon$, X and Y are said to be ϵ-statistically indistinguishable. Next, we formally define the statistical indistinguishability asymptotically.

Definition 2.2 (Statistical Indistinguishability)**.** Consider a function $\epsilon : \mathbb{N} \to [0,1]$, and distribution ensembles $\mathcal{X} = \{X_n\}_{n\in\mathbb{N}}$ and $\mathcal{Y} = \{Y_n\}_{n\in\mathbb{N}}$. We say that $\mathcal{X}$ and $\mathcal{Y}$ are *ϵ-statistically indistinguishable* (or have statistical distance at most ϵ), denoted by $\Delta_s(\mathcal{X};\mathcal{Y}) \leq \epsilon$, if for all $n \in \mathbb{N}$, we have $\Delta_s(X_n;Y_n) \leq \epsilon(n)$.

Notice that we do not impose any computational constraint on the distinguisher above, thus the statistical notion implies that even computationally unbounded algorithms cannot achieve an advantage better than ϵ. It is also natural to restrict attention to polynomial-time algorithms. Next, we define computational indistinguishability.

Definition 2.3 (Computational Indistinguishability)**.** Consider a function $\epsilon : \mathbb{N} \to [0,1]$, and distribution ensembles $\mathcal{X} = \{X_n\}_{n\in\mathbb{N}}$ and $\mathcal{Y} = \{Y_n\}_{n\in\mathbb{N}}$. If for every non-uniform probabilistic polynomial-time algorithm D, there is an $n_D \in \mathbb{N}$ such that for all $n \geq n_D$ we have

$$\mathsf{Adv}_D(X_n;Y_n) \leq \epsilon(n),$$

then we say that $\mathcal{X}$ and $\mathcal{Y}$ are *ϵ-computationally indistinguishable* (with respect to polynomial-time algorithms). We denote this by

$$\Delta_c(\mathcal{X};\mathcal{Y}) \leq \epsilon.$$

In the course of our technical arguments, for the sake of simplicity, we often make statements of the form $\Delta_c(X_n;Y_n) \leq \epsilon(n)$. These statements and their implications are to be interpreted in the asymptotic sense, as holding for all large enough n rather than for all n.

The definition ensures that $\Delta_c(\mathcal{X};\mathcal{Y}) \leq \Delta_s(\mathcal{X};\mathcal{Y})$. Versions of the triangle inequality and data processing inequality hold computational indistinguishability as well. The former is easily implied by essentially a hybrid argument. We state it formally as follows.

Lemma 2.4 (Triangle Inequality)**.** *For any three distribution ensembles $\mathcal{X}$, $\mathcal{Y}$, $\mathcal{Z}$, we have*

$$\Delta_c(\mathcal{X};\mathcal{Y}) \leq \epsilon_1, \Delta_c(\mathcal{Y};\mathcal{Z}) \leq \epsilon_2 \Rightarrow \Delta_c(\mathcal{X};\mathcal{Z}) \leq \epsilon_1 + \epsilon_2.$$

Lemma 2.5 (Data Processing Inequality)**.** *For any distribution ensembles $\mathcal{X}, \mathcal{Y}$ and any probabilistic polynomial-time procedure f, we have*

$$\Delta_c(\mathcal{X};\mathcal{Y}) \leq \epsilon \Rightarrow \Delta_c(f(\mathcal{X});f(\mathcal{Y})) \leq \epsilon.$$

This is a simple consequence of the observation that any such efficient function f can simply be run on top of samples from $\mathcal{X}$ or $\mathcal{Y}$ and then fed into a distinguisher for $f(\mathcal{X})$ and $f(\mathcal{Y})$.

We slightly abuse the notation by writing $\Delta_c(\mathcal{X};\mathcal{Z}) \leq \Delta_c(\mathcal{X};\mathcal{Y}) + \Delta_c(\mathcal{Y};\mathcal{Z})$ and $\Delta_c(f(\mathcal{X});f(\mathcal{Y})) \leq \Delta_c(\mathcal{X};\mathcal{Y})$, by which we mean the properties defined above.

2.2 Circuits and Oracles

We define notions of circuits and functions computed with respect to certain oracles.

Definition 2.4 (Oracle-Aided Circuits and Algorithms)**.** Let $\mathcal{O} : \{0,1\}^* \to \{0,1\}^*$ be an oracle (an arbitrary function). An oracle circuit (or algorithm) C with respect to $\mathcal{O}$, denoted by $C^{\mathcal{O}}$ is a circuit (or algorithm) where in addition to standard operations, C also has oracle gates (or oracle operations) where it can make a query to $\mathcal{O}$, and expect its output as response.

Definition 2.5 (Oracle-Aided Functions)**.** Let $\mathcal{O} : \{0,1\}^* \to \{0,1\}^*$ be an oracle (an arbitrary function). An oracle aided function f with respect to $\mathcal{O}$, denoted $f^{\mathcal{O}}$, is a function computable by a deterministic oracle-aided algorithm $A^{\mathcal{O}}$.

Remark 2.1. *When $\mathcal{O}$ is a randomized algorithm, we view the oracle gates for $\mathcal{O}$ as deterministic ones that take an additional randomness as input.*

2.3 One-Way Functions

In the following, we present the definition of one-way functions, along with several weaker variants, which will serve as intermediate steps in our later proofs.

Definition 2.6 (One-Way Function, OWF)**.** For m_1, m_2 being polynomials, a function family $\mathcal{F} = \{f_n : \{0,1\}^{m_1(n)} \to \{0,1\}^{m_2(n)}\}_{n\in\mathbb{N}}$ is said to be a *One-Way Function (OWF)* if $\mathcal{F}$ is efficiently computable and for every non-uniform PPT algorithm $\mathcal{A}$ there is a negligible function $\nu(\cdot)$ such that for all large enough $n \in \mathbb{N}$,

$$\Pr_{x\leftarrow\mathcal{U}_{m_1(n)}}\left[f_n(\mathcal{A}(f_n(x))) = f_n(x)\right] \leq \nu(n).$$

Definition 2.7 (Weak One-Way Function)**.** For m_1, m_2 being polynomials, a function family $\mathcal{F} = \{f_n : \{0,1\}^{m_1(n)} \to \{0,1\}^{m_2(n)}\}_{n\in\mathbb{N}}$ is said to be a *Weak One-Way Function* if $\mathcal{F}$ is efficiently computable and for every non-uniform PPT algorithm $\mathcal{A}$, there is a polynomial p such that for all large enough $n \in \mathbb{N}$,

$$\Pr_{x\leftarrow\mathcal{U}_{m_1(n)}}\left[f_n(\mathcal{A}(f_n(x))) = f_n(x)\right] \leq 1 - \frac{1}{p(n)}.$$

Definition 2.8 (Distributional One-Way Function, dOWF)**.** For m_1, m_2 being polynomials, a function family $\mathcal{F} = \{f_n : \{0,1\}^{m_1(n)} \to \{0,1\}^{m_2(n)}\}_{n\in\mathbb{N}}$ is a *Distributional One-Way Function (dOWF)* if $\mathcal{F}$ is efficiently computable and for every non-uniform PPT algorithm A, there is a polynomial p such that for all large enough n, the following two distributions:

- $X_n : \left\{(x, f(x)) : x \leftarrow \mathcal{U}_{m_1(n)}\right\}$

- $Y_n : \{(\mathcal{A}(f(x)), f(x)) : x \leftarrow \mathcal{U}_{m_1(n)}\}$

satisfy the following

$$\Delta_s(X_n; Y_n) > \frac{1}{p(n)}.$$

Remark 2.2. *In our arguments, we will consider adversaries against distributional one-way functions. We will refer to such an adversary $\mathcal{A}$ as inverting the function in a distributional sense (or distributionally inverting it as shorthand), with deviation say $\frac{1}{q(n)}$ (where $q(\cdot)$ is a polynomial), to mean that $\Delta_s\big((x, f(x)); (\mathcal{A}(f(x), f(x)))\big) \leq \frac{1}{q(n)}$ for uniformly sampled $x \leftarrow \mathcal{U}_{m_1(n)}$.*

Definition 2.9 (Auxiliary-Input One-Way Functions, ai-OWF)**.** For m_1, m_2 being polynomials, a function family $\mathcal{F} = \{f_a : \{0,1\}^{m_1(|a|)} \to \{0,1\}^{m_2(|a|)}\}_{a \in \{0,1\}^*}$ is said to be an *Auxiliary-Input One-Way Function (ai-OWF)* if $\mathcal{F}$ is efficiently computable and for every non-uniform PPT machine $\mathcal{A}$ there is a negligible function $\nu(\cdot)$ such that for all large enough $n \in \mathbb{N}$, there exists some $a \in \{0,1\}^n$ such that we have

$$\Pr_{x \leftarrow \mathcal{U}_{m_1(n)}}\big[f_a(A(a, f_a(x))) = f_a(x)\big] \leq \nu(n).$$

Remark 2.3. *If in the above definition the string a (for a given n) is always fixed to be 0^n, then the definition collapses to that of a standard one-way function.*

Remark 2.4. *Similar to the above variant of (standard) OWFs, we can also extend the definitions of weak and distributional one-way functions to the auxiliary input setting in the natural manner.*

Definition 2.10 (Infinitely-Often One-Way Functions, ioOWF)**.** For m_1, m_2 being polynomials, a function family $\mathcal{F} = \{f_n : \{0,1\}^{m_1(n)} \to \{0,1\}^{m_2(n)}\}$ is an *Infinitely Often One-Way Function (ioOWF)* if $\mathcal{F}$ is efficiently computable and if for every non-uniform PPT algorithm A there is a negligible function $\nu(\cdot)$ and an infinite set $S_A \subseteq \mathbb{N}$ such that we have

$$\Pr_{r \leftarrow \{0,1\}^{m_1(n)}}\big[f_n(A(f_n(r))) = f_n(r)\big] \leq \nu(n)$$

for all $n \in S_A$.

Remark 2.5. *When the properties of any of the primitives above hold only for infinitely many $n \in \mathbb{N}$, we refer to them as infinitely-often versions. Similarly, infinitely-often versions of complexity classes also be defined – e.g., ioP is the set of languages that have deterministic polynomial-time algorithms that are correct on some infinite set of input lengths.*

Remark 2.6. *Similar to above, we can also define weak and distributional infinitely often one-way functions with straightforward modifications.*

Lemma 2.6 ([IL89])**.** *There is an explicit, efficient transformation from any distributional one-way function to a standard one-way function.*

Lemma 2.7 ([Yao82])**.** *There is an explicit, efficient transformation from any weak one-way function to a standard one-way function.*

Remark 2.7. *Both results cited above work as stated for converting (infinitely-often or auxiliary-input) weak or distributional one-way functions to (infinitely-often or auxiliary-input) one-way functions.*

2.4 Zero-Knowledge Protocols

In this section, we formally define zero-knowledge protocols, specifically *non-interactive zero-knowledge* (NIZK) and *public-coin zero-knowledge* proofs and arguments.

2.4.1 Non-Interactive Zero-Knowledge

We describe the non-interactive zero-knowledge proofs or arguments in the *common reference string* model. In particular, there is no interaction between the prover and the verifier. Both parties refer to a common reference string r, which is randomly sampled, to run the protocol. For an NP language $\mathcal{L}$, on input x, the honest prover holds the NP witness w and generates a message $\pi \leftarrow \mathsf{P}(x, w; r)$; then the verifier computes $0/1 \leftarrow \mathsf{V}(x; r, \pi)$, where by convention, 1 denotes the acceptance of the proof.

Definition 2.11 (Non-Interactive Zero-Knowledge, NIZK)**.** For $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0, 1]$ and a language $\mathcal{L} \in \mathsf{NP}$, an $(\epsilon_c, \epsilon_s, \epsilon_z)$-*Non-Interactive Zero-Knowledge (NIZK) proof* for $\mathcal{L}$ consists of algorithms $(\mathsf{Gen}, \mathsf{P}, \mathsf{V})$, where Gen samples the *common reference string* in polynomial time, V is a deterministic polynomial-time verifier and P is a computationally unbounded prover. Let n be the length of the input x and $\mathcal{R}_{\mathcal{L}}$ denote the corresponding NP relation. The protocol should satisfy the following properties.

1. *Completeness.* For any $x \in \mathcal{L}_n$ and any witness w such that $(x, w) \in \mathcal{R}_{\mathcal{L}}$

$$\Pr_{\substack{r \leftarrow \mathsf{Gen}(1^n) \\ \pi \leftarrow \mathsf{P}(x,w;r)}} [\mathsf{V}(x; r, \pi) = 1] \geq 1 - \epsilon_c(n).$$

2. *Soundness.* For any $x \in \{0, 1\}^n \setminus \mathcal{L}_n$ and any prover algorithm P^*, the following holds

$$\Pr_{\substack{r \leftarrow \mathsf{Gen}(1^n) \\ \pi^* \leftarrow \mathsf{P}^*(x;r)}} [\mathsf{V}(x; r, \pi^*) = 1] \leq \epsilon_s(n),$$

3. *Computational Zero-Knowledge.* There exists a probabilistic polynomial-time simulator Sim such that for any $x \in \mathcal{L}_n$ and $(x, w) \in \mathcal{R}_{\mathcal{L}}$

$$\Delta_c(\mathsf{Sim}(x); \mathsf{View}\langle \mathsf{P}, \mathsf{V} \rangle(x, w)) \leq \epsilon_z(n),$$

where the transcript $\mathsf{View}\langle \mathsf{P}, \mathsf{V}\rangle(x, w)$ represents the view of the verifier in the protocol with input x and witness w given to the prover, consisting of the common reference string r and the prover's message π.

If the honest prover P is constrained to be computationally efficient, and the soundness condition is required to hold only against computationally efficient provers P^*, the protocol is called an *NIZK argument*.

Remark 2.8. *The simulator $\mathsf{Sim}(x)$ is randomized on input x. When we need a deterministic description, we explicitly expose the random coins and include them as part of the input, which is written as $\mathsf{Sim}(x; \rho)$. Equivalently, $\mathsf{Sim}(x)$ represents the distribution of $\mathsf{Sim}(x; \rho)$ when ρ is drawn uniformly randomly. For convenience, we denote by $\mathsf{Sim}_i(x)$ the simulator $\mathsf{Sim}(x)$ restricted to outputting only the i-th message. For example, in the NIZK protocols, $\mathsf{Sim}_1(x)$ only samples the marginal distribution on the common reference string r while $\mathsf{Sim}_2(x)$ simulates the distribution of π.*

Remark 2.9. *We have defined the verifier as being a deterministic algorithm in its decision of whether to accept a given execution. While this is not the most general possible notion, by and large known protocols all have a final deterministic verifier step and typically this is the notion considered in most definitions. Nevertheless, we are able to show our results even for the more general notion of randomized verification. The proof of this version of our results is more involved, and the crucial lemma is presented in Section A.*

2.4.2 Public-Coin Zero-Knowledge

Beyond non-interactive protocols, we study the broader class of public-coin zero-knowledge arguments or proofs.

Definition 2.12 (Public-Coin Zero-Knowledge)**.** For $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0, 1]$ and a language $\mathcal{L}$, an $(\epsilon_c, \epsilon_s, \epsilon_z)$-*public-coin Zero-Knowledge (ZK) proof* for $\mathcal{L}$ consists of algorithms (P, V), where V is a deterministic polynomial-time verifier and P is a computationally unbounded prover. In an execution of the protocol V is given as input the instance x and P the instance x and a witness w. The protocol should satisfy the following.

1. *Syntax and notation.* In each round, first a uniformly random string r_i of pre-specified length is sampled and sent to the prover, and the prover responds with a message π_i computed as $\pi_i \leftarrow \mathsf{P}_i(x, w; r_1, \pi_1, \ldots, r_i)$. At the end, the verifier decides whether to accept the transcript $(r_1, \pi_1, \ldots, r_k, \pi_k)$ by computing $0/1 \leftarrow \mathsf{V}(x; r_1, \pi_1, \ldots, r_k, \pi_k)$. Denote by $\langle \mathsf{P}, \mathsf{V}\rangle(x, w)$ the output of the protocol on a common input x and a witness w (held only by the prover), and $\mathsf{View}\langle \mathsf{P}, \mathsf{V}\rangle(x, w)$ represents the transcript of the execution, consisting of $(r_1, \pi_1, \ldots, r_k, \pi_k)$. Here, k is the number of rounds of the protocol.

2. *Completeness.* For any $x \in \mathcal{L}_n$, for any witness w that $(x, w) \in \mathcal{R}_{\mathcal{L}}$,

$$\Pr\left[\langle \mathsf{P}, \mathsf{V}\rangle(x, w) = 1\right] \geq 1 - \epsilon_c(n).$$

3. *Soundness.* For any $x \in \{0,1\}^n \setminus \mathcal{L}_n$, for any prover P^*

$$\Pr\left[\langle \mathsf{P}^*, \mathsf{V}\rangle(x) = 1\right] \leq \epsilon_s(n).$$

4. *Computational Zero-knowledge*: There exists a probabilistic polynomial-time simulator Sim, such that for any $x \in \mathcal{L}_n$ and $(x, w) \in \mathcal{R}_{\mathcal{L}}$

$$\Delta_c(\mathsf{Sim}(x); \mathsf{View}\langle \mathsf{P}, \mathsf{V}\rangle(x, w)) \leq \epsilon_z(n).$$

   If the honest prover P is constrained to be computationally efficient, and the soundness condition is required to hold only against computationally efficient provers P^*, the protocol is called a *ZK argument*.

Remark 2.10. *We often think of the public coins r_i's as being sent by the verifier to the prover. We consider the process of the verifier sending a randomness and the prover replying with a message as one round in the protocol, where each round contains two messages. For convenience in notation, when the first message in the protocol is from the prover, we sometimes pretend that there is an empty message from the verifier before that.*

3 Auxiliary-Input One-Way Functions

In this section, we show that if a language has a Zero-Knowledge proof or argument with errors satisfying certain conditions, then for any instance, we can define a function such that any inverter for that function can be used to decide the membership of that instance in the language. Worst-case hardness of the language then gives us an auxiliary-input one-way function, which will be used in later sections to construct one-way functions from hard languages that have such proof systems.

We do this for Non-Interactive ZK protocols in Section 3.1 and for general public-coin ZK protocols in Section 3.2. In Section 3.3, we use additional ideas to improve the range of errors that can be used, at the cost of the proof being non-black-box, and yielding only infinitely often secure OWFs.

Remark 3.1. *In this section, we state the main lemmas for both ZK proofs and ZK arguments. To avoid excessive repetition, in the proofs of these lemmas, we only deal with the case of arguments. It may be verified that these proofs do not rely on the efficiency of the honest prover, and work nearly as is for proof systems as well.*

3.1 Reductions from NIZK

Lemma 3.1. *For some $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$, suppose a language $\mathcal{L} \in \mathsf{NP}$ has an $(\epsilon_c, \epsilon_s, \epsilon_z)$-NIZK proof or argument (with deterministic verification). Then there exists a reduction R, which is a polynomial-time oracle-aided algorithm, and a polynomial-time computable function family $\mathcal{F} = \{f_x\}_{x \in \{0,1\}^*}$ such that, for any probabilistic polynomial-time algorithm $\mathcal{A}$, any polynomial p, and all large enough $n \in \mathbb{N}$,*

1. *For any $x \in \mathcal{L}_n$, if $\mathcal{A}$ inverts f_x with probability at least $(1 - 1/p(n))$, then*

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \geq 1 - \epsilon_c(n) - \epsilon_z(n) - \frac{1}{p(n)}.$$

2. *For any $x \in \{0,1\}^n \setminus \mathcal{L}_n$,*

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \leq \epsilon_s(n).$$

Proof of Lemma 3.1. Let Sim be the simulator that satisfies the zero-knowledge requirement, and suppose it uses ℓ bits of randomness, where $\ell = \ell(n)$. Let $\mathcal{F} = \{f_x\}_{x \in \{0,1\}^*}$ be a function family where f_x is computed as:

$$\begin{aligned} f_x(\rho): \quad & (r, \pi) \leftarrow \mathsf{Sim}(x; \rho) \\ & a \leftarrow \mathsf{V}(x; r, \pi) \\ & \text{output } (r, a) \end{aligned}$$

where ρ serves as the randomness of Sim. Assume that $\mathcal{A}$ is a PPT algorithm for potentially inverting f_x's. Construct a reduction R with oracle access to $\mathcal{A}$, which works as follows:

$$\begin{aligned} R^{\mathcal{A}}(x): \quad & r \leftarrow \mathsf{Gen}(1^n) \\ & \rho \leftarrow \mathcal{A}(x; (r, 1)) \\ & \text{if } f_x(\rho) = (r, 1) \text{ output } 1 \\ & \text{else output } 0 \end{aligned}$$

We first prove that R and $\mathcal{F}$ satisfy the condition 1. For an asymptotic $n \in \mathbb{N}$ and $x \in \mathcal{L}_n$, suppose that $\mathcal{A}$ inverts f_x with probability at least $(1 - 1/p(n))$, that is

$$\Pr_{(r,a) \leftarrow f_x(\mathcal{U}_\ell)}\left[f_x(\mathcal{A}(x; (r, a))) = (r, a)\right] \geq 1 - \frac{1}{p(n)}. \tag{8}$$

Denote by D_S, D_P and D_I the following distributions, with w being any valid witness for x that satisfies $(x, w) \in \mathcal{R}_{\mathcal{L}}$.

- D_S: sample $\rho \leftarrow \mathcal{U}_\ell$, $(r, \pi) \leftarrow \mathsf{Sim}(x; \rho)$, $a \leftarrow \mathsf{V}(x; r, \pi)$, output (r, a)

- D_P: sample $r \leftarrow \mathsf{Gen}(1^n)$, $\pi \leftarrow \mathsf{P}(x, w; r)$, $a \leftarrow \mathsf{V}(x; r, \pi)$, output (r, a)
- D_I: sample $r \leftarrow \mathsf{Gen}(1^n)$, output $(r, 1)$

By the data processing inequality and zero-knowledge, we have:

$$\Delta_c(D_S; D_P) \leq \Delta_c(\mathsf{Sim}(x); \mathsf{View}\langle \mathsf{P}, \mathsf{V}\rangle(x, w)) \leq \epsilon_z(n). \tag{9}$$

The completeness ensures that:

$$\Pr_{(r,a)\leftarrow D_P}[a = 1] \geq 1 - \epsilon_c(n).$$

Since the marginal distributions of r induced by D_P and D_I are the same, we have:

$$\begin{aligned}
\Delta_s(D_P; D_I) &= \mathop{\mathbb{E}}_{r\leftarrow\mathsf{Gen}(1^n)}[\Delta_s(D_I|r; D_P|r)] \\
&= \mathop{\mathbb{E}}_{r\leftarrow\mathsf{Gen}(1^n)}\left[\Pr_{a\leftarrow D_I|r}[a = 1] - \Pr_{a\leftarrow D_P|r}[a = 1]\right] \\
&= 1 - \Pr_{(r,a)\leftarrow D_P}[a = 1] \\
&\leq \epsilon_c(n)
\end{aligned} \tag{10}$$

By the triangle inequality, the distance between D_S and D_I can be upper bounded by:

$$\Delta_c(D_S; D_I) \leq \Delta_c(D_P; D_I) + \Delta_c(D_S; D_P) \leq \epsilon_c(n) + \epsilon_z(n). \tag{11}$$

Combining (8) and (11), since both f_x and $\mathcal{A}$ are efficient algorithms, we obtain that:

$$\begin{aligned}
\Pr\left[R^{\mathcal{A}}(x) = 1\right] &= \Pr_{r\leftarrow\mathsf{Gen}(1^n)}[f_x(\mathcal{A}(x; (r, 1))) = (r, 1)] \\
&= \Pr_{(r,a)\leftarrow D_I}[f_x(\mathcal{A}(x; (r, a))) = (r, a)] \\
&\geq \Pr_{(r,a)\leftarrow D_S}[f_x(\mathcal{A}(x; (r, a))) = (r, a)] - \Delta_c(D_S, D_I) \\
&= \Pr_{(r,a)\leftarrow f_x(\mathcal{U}_\ell)}[f_x(\mathcal{A}(x; (r, a))) = (r, a)] - \Delta_c(D_S, D_I) \\
&\geq 1 - \frac{1}{p(n)} - (\epsilon_c(n) + \epsilon_z(n)),
\end{aligned}$$

which shows condition 1.

Next, we show that condition 2 holds. When $x \in \{0, 1\}^n \setminus \mathcal{L}$, for any polynomial-time algorithm $\mathcal{A}$, we will show that the soundness guarantees that:

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \leq \epsilon_s(n). \tag{12}$$

We prove (12) by contradiction. Suppose $\Pr\left[R^{\mathcal{A}}(x)=1\right] > \epsilon_s(n)$, which is equivalent to

$$\Pr_{\substack{r\leftarrow\mathsf{Gen}(1^n)\\ \rho^*\leftarrow\mathcal{A}(r,1)}}\left[f_x(\rho^*)=(r,1)\right] > \epsilon_s(n).$$

Then, there is a construction of an efficient malicious prover P^* as follows: when receiving $r\leftarrow\mathsf{Gen}(1^n)$, it computes $\rho^*\leftarrow\mathcal{A}(r,1)$, $\pi^*\leftarrow\mathsf{Sim}(x;\rho^*)$ and outputs π^*. Note that, when the inverter successfully finds a correct pre-image of $(r,1)$, P^* produces a valid proof π^* on which the verifier accepts. Thus,

$$\Pr_{\substack{r\leftarrow\mathsf{Gen}(1^n)\\ \pi^*\leftarrow\mathsf{P}^*(x;r)}}\left[\mathsf{V}(x;r,\pi)=1\right] \geq \Pr_{\substack{r\leftarrow\mathsf{Gen}(1^n)\\ \rho^*\leftarrow\mathcal{A}(r,1)}}\left[f_x(\rho^*)=(r,1)\right] > \epsilon_s(n),$$

which contradicts the soundness. □

Corollary 3.2. *For $\epsilon_c,\epsilon_s,\epsilon_z:\mathbb{N}\to[0,1]$, suppose there is an* NP *language $\mathcal{L}\notin\mathsf{ioP/poly}$ that has an $(\epsilon_c,\epsilon_s,\epsilon_z)$-NIZK proof or argument with $\epsilon_c+\epsilon_s+\epsilon_z <_n 1$. Then auxiliary-input one-way functions exist.*

Proof of Corollary 3.2. Let R be the oracle-aided algorithm and $\mathcal{F}$ be the function family guaranteed by Lemma 3.1. There exists a polynomial q such that:

$$\epsilon_c(n)+\epsilon_s(n)+\epsilon_z(n)+\frac{1}{q(n)}<1$$

holds for all sufficiently large $n\in\mathbb{N}$. Then, let $p=2q$. Suppose that there is a (non-uniform) PPT algorithm $\mathcal{A}$ that, for infinitely many $n\in\mathbb{N}$, for every $x\in\mathcal{L}_n$, inverts f_x with probability at least $(1-1/p(n))$. By Lemma 3.1, for every $x\in\mathcal{L}_n$,

$$\Pr\left[R^{\mathcal{A}}(x)=1\right]\geq 1-\epsilon_c(n)-\epsilon_z(n)-\frac{1}{p(n)};$$

and for every $x\in\{0,1\}^n\setminus\mathcal{L}_n$,

$$\Pr\left[R^{\mathcal{A}}(x)=1\right]\leq\epsilon_s(n).$$

Since both $\mathcal{A}$ and R are polynomial-time algorithms and

$$1-\epsilon_c(n)-\epsilon_z(n)-\frac{1}{2q(n)} >_n \epsilon_s(n),$$

it contradicts $\mathcal{L}\notin\mathsf{ioP/poly}$. Hence, $\mathcal{F}$ is a weak auxiliary-input one-way function: for any efficient non-uniform adversary, for all sufficiently large n, there exists a function f_x, $x\in\{0,1\}^n$ on which the adversary cannot successfully invert f_x with probability at least $(1-1/p(n))$. By [Yao82], the existence of weak ai-OWFs implies the existence of (standard) ai-OWFs, which concludes the proof. □

3.2 Reductions from Public-Coin ZK

Lemma 3.3. *For some $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$, $t : \mathbb{N} \to \mathbb{N}$, suppose a language $\mathcal{L} \in \mathsf{NP}$ has an $(\epsilon_c, \epsilon_s, \epsilon_z)$-public-coin ZK proof or argument with t messages. Then there exists a reduction R, which is a polynomial-time oracle-aided algorithm, and a polynomial-time computable function family $\mathcal{F} = \{f_x\}_{x \in \{0,1\}^*}$ such that, for any probabilistic polynomial-time algorithm $\mathcal{A}$, any polynomial p, and all large enough $n \in \mathbb{N}$,*

1. *For any $x \in \mathcal{L}_n$, if $\mathcal{A}$ distributionally inverts f_x with deviation at most $1/p(n)$, then*

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \geq 1 - \epsilon_c(n) - (t(n) - 1) \cdot \epsilon_z(n) - \frac{t(n)}{p(n)}.$$

2. *For any $x \in \{0,1\}^n \setminus \mathcal{L}_n$,*

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \leq \epsilon_s(n).$$

Proof of Lemma 3.3. Denote by n the input length. Let Sim be the simulator that satisfies the zero-knowledge requirement, and suppose it uses ℓ randomness bits, where $\ell = \ell(n)$. Let $k = k(n)$ and $t = t(n)$ be the number of rounds and messages, respectively. Let $m_i = m_i(n)$ be the number of bits of public randomness used by the verifier in the i-th round, for $i \in [k]$. When $t = 2k$, the verifier sends a random string first in the protocol; when $t = 2k - 1$, the prover starts first. For convenience, let $m_1 = 0$ when $t = 2k - 1$, simply taking the verifier's first message to be an empty string.

Construct a function family $\mathcal{F} = \{f_x\}_{x \in \{0,1\}^*}$ as follows:

$\underline{f_x(i, \rho)}$:

1. $(r_1, \pi_1, \ldots, r_k, \pi_k) \leftarrow \mathsf{Sim}(x; \rho)$
2. $a \leftarrow \mathsf{V}(x; r_1, \pi_1, \ldots, r_k, \pi_k)$
3. If $i = k$ output $(r_1, \pi_1, \ldots, r_{i-1}, \pi_{i-1}, r_i, a)$
4. Else output $(r_1, \pi_1, \ldots, r_{i-1}, \pi_{i-1}, r_i, 1)$

Note that the output length may vary with different inputs. One can equalize the output lengths by padding with a fixed constant string after those short outputs. We keep the current definition for simplicity. Assume that $\mathcal{A}$ is a PPT algorithm for inverting f_x's, where the output of $\mathcal{A}$ consists of two parts (i, ρ). Let $\mathcal{A}_2$ denote the algorithm obtained from $\mathcal{A}$ by projecting its output onto the second component ρ.

Define an efficient prover $\tilde{\mathsf{P}}$ with oracle access to $\mathcal{A}$, where we denote by $\tilde{\mathsf{P}}_i$ the prover's algorithm in the i-th round. The prover proceeds as follows:

$\underline{\tilde{\mathsf{P}}_i^{\mathcal{A}}(r_1, \pi_1, \ldots, r_i)}$:

1. $\rho \leftarrow \mathcal{A}_2(x; r_1, \pi_1, \ldots, r_i, 1)$
2. $\pi_i \leftarrow \mathsf{Sim}_{2i}(x; \rho)$
3. Output π_i

The reduction $R^{\mathcal{A}}$ runs the following: on input x, it simulates the protocol $\langle \tilde{\mathsf{P}}^{\mathcal{A}}, \mathsf{V} \rangle$ (with the same syntax as $\langle \mathsf{P}, \mathsf{V} \rangle$), and outputs 1 if and only if the protocol $\langle \tilde{\mathsf{P}}^{\mathcal{A}}, \mathsf{V} \rangle(x)$ accepts. Since both prover $\tilde{\mathsf{P}}$ and verifier V are efficient, R runs in polynomial time. We state the following claims.

Claim 3.1. *When $x \in \mathcal{L}_n$, if $\mathcal{A}$ distributionally inverts f_x with deviation at most $1/p(n)$*

$$\Pr\left[\langle \tilde{\mathsf{P}}^{\mathcal{A}}, \mathsf{V} \rangle(x) = 1\right] \geq 1 - \epsilon_c(n) - (t(n) - 1) \cdot \epsilon_z(n) - \frac{k(n)}{p(n)}.$$

Claim 3.2. *When $x \in \{0,1\}^n \setminus \mathcal{L}_n$*

$$\Pr\left[\langle \tilde{\mathsf{P}}^{\mathcal{A}}, \mathsf{V} \rangle(x) = 1\right] \leq \epsilon_s(n).$$

Claim 3.2 follows immediately from the soundness of the ZK protocol. We present the proof of Claim 3.1 in Section 3.2.1. Putting the above claims together completes the proof of the lemma. □

Corollary 3.4. *For $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$ and $t : \mathbb{N} \to \mathbb{N}$, suppose there is an NP language $\mathcal{L} \notin \mathsf{ioP/poly}$ that has a t-message public-coin $(\epsilon_c, \epsilon_s, \epsilon_z)$-ZK proof or argument with $\epsilon_c + \epsilon_s + (t-1)\epsilon_z <_n 1$. Then auxiliary-input one-way functions exist.*

Proof of Corollary 3.4. Let R be the oracle-aided algorithm and $\mathcal{F}$ be the function family defined in the proof of Lemma 3.3. Since $\epsilon_c + \epsilon_s + (t-1)\epsilon_z <_n 1$, there exists a polynomial q such that

$$\epsilon_c(n) + \epsilon_s(n) + (t(n) - 1) \cdot \epsilon_z(n) + \frac{1}{q(n)} < 1$$

holds for all sufficiently large $n \in \mathbb{N}$. Then, let $p = 2kq$. Suppose that there is a (non-uniform) PPT algorithm $\mathcal{A}$ such that, for infinitely many $n \in \mathbb{N}$, for every $x \in \mathcal{L}_n$, distributional inverts f_x with deviation at most $1/p(n)$. By Lemma 3.3, for every $x \in \mathcal{L}_n$,

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \geq 1 - \epsilon_c(n) - (t(n) - 1) \cdot \epsilon_z(n) - \frac{k(n)}{p(n)};$$

and for every $x \in \{0,1\}^n \setminus \mathcal{L}_n$,

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \leq \epsilon_s(n).$$

Both $\mathcal{A}$ and R are polynomial-time algorithms and

$$1 - \epsilon_c(n) - (t(n) - 1) \cdot \epsilon_z(n) - \frac{1}{2q(n)} >_n \epsilon_s(n),$$

which contradicts $\mathcal{L} \notin \mathsf{ioP/poly}$. Hence, $\mathcal{F}$ is an auxiliary-input distributional one-way function. By [IL89], the existence of ai-dOWFs implies the existence of ai-OWFs, which concludes the proof. □

3.2.1 Proof of Claim 3.1

Proof. Let $x \in \mathcal{L}_n$, suppose that $\mathcal{A}$ distributionally inverts f_x with deviation at most $1/p(n)$, which means that

$$\Delta_s\left(\mathcal{U}, f_x(\mathcal{U}); \mathcal{A}(f_x(\mathcal{U})), f_x(\mathcal{U})\right) \leq \frac{1}{p(n)}.$$

We abuse the notation by omitting x from the algorithm's input and the subscript of $\mathcal{U}$, assume that the length of the uniform distribution $\mathcal{U}$'s output always matches the input length of f_x.

First, consider the following distribution D_I.

$$\begin{aligned} D_I: \quad & r_1 \leftarrow \mathcal{U}_{m_1}, \pi_1 \leftarrow \tilde{\mathsf{P}}_1^{\mathcal{A}}(x; r_1) \\ & \cdots \\ & r_{k-1} \leftarrow \mathcal{U}_{m_{k-1}}, \pi_{k-1} \leftarrow \tilde{\mathsf{P}}_{k-1}^{\mathcal{A}}(x; r_1, \pi_1, \ldots, r_{k-1}) \\ & r_k \leftarrow \mathcal{U}_{m_k} \\ & \text{Output } (r_1, \pi_1, \ldots, r_{k-1}, \pi_{k-1}, r_k, 1) \end{aligned}$$

D_I runs the protocol $\langle \tilde{\mathsf{P}}^{\mathcal{A}}, \mathsf{V} \rangle$ to generate the transcript except for the last proof π_k, and pads the transcript with 1. Denote the string by $s = (r_1, \ldots, \pi_{k-1}, r_k, a)$. Observe that

$$\Pr\left[\langle \tilde{\mathsf{P}}^{\mathcal{A}}, \mathsf{V} \rangle(x) = 1\right] \geq \Pr_{s \leftarrow D_I}\left[f_x(k, \mathcal{A}(s)) = s\right], \tag{13}$$

since D_I is the distribution of the inputs on which $\tilde{\mathsf{P}}$ invokes $\mathcal{A}$ before sending their last message π_k and whenever the inverter $\mathcal{A}$ finds a good randomness, $\tilde{\mathsf{P}}_k^{\mathcal{A}}$ outputs a valid proof such that the protocol accepts.

Similarly, define the distributions D_S, which samples a transcript from the simulator and outputs the transcript together with the corresponding verification bit.

$$
\begin{aligned}
D_S: \quad & \rho \leftarrow \mathcal{U}_\ell \\
& (r_1, \pi_1, \ldots, r_k, \pi_k) \leftarrow \mathsf{Sim}(x; \rho) \\
& a \leftarrow \mathsf{V}(x; r_1, \pi_1, \ldots, r_k, \pi_k) \\
& \text{output } (r_1, \pi_1, \ldots, r_k, a)
\end{aligned}
$$

Note that the distribution outputting (k, ρ, s) by sampling $s \leftarrow D_S$ and $(k, \rho) \leftarrow \mathcal{A}(s)$ is exactly the same as the distribution $(\mathcal{A}(f_x(k, \mathcal{U}_\ell)), f_x(k, \mathcal{U}_\ell))$. By the property of indistinguishability, we have

$$
\begin{aligned}
&\left| \Pr_{(k,\rho,s) \leftarrow (\mathcal{A}(f_x(k,\mathcal{U}_\ell)), f_x(k,\mathcal{U}_\ell))} [f_x(k, \rho) = s] - \Pr_{(k,\rho,s) \leftarrow ((k,\mathcal{U}_\ell), f_x(k,\mathcal{U}_\ell))} [f_x(k, \rho) = s] \right| \\
&\quad \leq \Delta_s((k, \mathcal{U}_\ell), f_x(k, \mathcal{U}_\ell); \mathcal{A}(f_x(k, \mathcal{U}_\ell)), f_x(k, \mathcal{U}_\ell)),
\end{aligned}
$$

where the first probability equals $\Pr_{s \leftarrow D_S} [f_x(\mathcal{A}(s)) = s]$ and the second is exactly 1. Thus, we derive

$$
\Pr_{s \leftarrow D_S} [f_x(\mathcal{A}(s)) = s] \geq 1 - \Delta_s((k, \mathcal{U}_\ell), f_x(k, \mathcal{U}_\ell); \mathcal{A}(f_x(k, \mathcal{U}_\ell)), f_x(k, \mathcal{U}_\ell)). \tag{14}
$$

Towards proving an upper-bound on $\Delta(D_S, D_I)$ to connect (13) and (14), we define the following distributions as intermediate hybrids, for $i \in [k]$. Below, w is any valid witness for x that satisfies $(x, w) \in \mathcal{R}_\mathcal{L}$.

$$
\begin{aligned}
D_S^{(i)}: \quad & \rho \leftarrow \mathcal{U}_\ell \\
& (r_1, \pi_1, \ldots, r_{i-1}, \pi_{i-1}) \leftarrow \mathsf{Sim}_{1 \ldots 2(i-1)}(x; \rho) \\
& r_i \leftarrow \mathcal{U}_{m_i}, \pi_i \leftarrow \tilde{\mathsf{P}}_i^{\mathcal{A}}(x; r_1, \ldots, \pi_{i-1}, r_i) \\
& \ldots \\
& r_{k-1} \leftarrow \mathcal{U}_{m_{k-1}}, \pi_{k-1} \leftarrow \tilde{\mathsf{P}}_{k-1}^{\mathcal{A}}(x; r_1, \ldots, \pi_{i-1}, r_i, \ldots, r_{k-1}) \\
& r_k \leftarrow \mathcal{U}_{m_k} \\
& \text{output } (r_1, \ldots, \pi_{i-1}, r_i, \ldots, r_k, 1)
\end{aligned}
$$

$$
\begin{aligned}
D_P^{(i)}: \quad & r_1 \leftarrow \mathcal{U}_{m_1}, \pi_1 \leftarrow \mathsf{P}_1(x, w; r_1) \\
& \cdots \\
& r_{i-1} \leftarrow \mathcal{U}_{m_{i-1}}, \pi_{i-1} \leftarrow \mathsf{P}_{i-1}(x, w; r_1, \ldots, r_{i-1}) \\
& r_i \leftarrow \mathcal{U}_{m_i}, \pi_i \leftarrow \tilde{\mathsf{P}}_i^{\mathcal{A}}(x; r_1, \ldots, \pi_{i-1}, r_i) \\
& \cdots \\
& r_{k-1} \leftarrow \mathcal{U}_{m_{k-1}}, \pi_{k-1} \leftarrow \tilde{\mathsf{P}}_{k-1}^{\mathcal{A}}(x; r_1, \ldots, \pi_{i-1}, r_i, \ldots, r_{k-1}) \\
& r_k \leftarrow \mathcal{U}_{m_k} \\
& \text{output } (r_1, \ldots, \pi_{i-1}, r_i, \ldots, r_k, 1)
\end{aligned}
$$

The messages in the first $(i-1)$ rounds are sampled by the simulator Sim in $D_S^{(i)}$ while those in $D_P^{(i)}$ are sampled by the protocol $\langle \mathsf{P}, \mathsf{V} \rangle$ run by the honest prover with an NP witness w.

In both distributions, starting from the i-th iteration, the remaining transcripts are sampled following the protocol with $\tilde{\mathsf{P}}$. Clearly, $D_S^{(1)} = D_P^{(1)} = D_I$. By data processing inequality, we have

$$
\Delta_c \left(D_S^{(i)}; D_P^{(i)} \right) < \Delta_c(\mathsf{Sim}(x); \mathsf{View}\langle \mathsf{P}, \mathsf{V} \rangle(x, w)) < \epsilon_z(n). \tag{15}
$$

Then, we connect $D_P^{(i)}$ and $D_S^{(i+1)}$ by modifying the sampling strategy in $D_S^{(i)}$ slightly. For $i \in [k]$, define $D_M^{(i)}$,

$$
\begin{aligned}
D_M^{(i)}: \quad & \rho \leftarrow \mathcal{U}_\ell \\
& (r_1, \pi_1, \ldots, \pi_{i-1}, r_i) \leftarrow \mathsf{Sim}_{1\ldots(2i-1)}(x; \rho) \\
& \pi_i \leftarrow \tilde{\mathsf{P}}_i^{\mathcal{A}}(x; r_1, \ldots, \pi_{i-1}, r_i) \\
& r_{i+1} \leftarrow \mathcal{U}_{m_{i+1}}, \pi_{i+1} \leftarrow \tilde{\mathsf{P}}_{i+1}^{\mathcal{A}}(x; r_1, \ldots, \pi_{i-1}, r_i, \pi_i, r_{i+1}) \\
& \cdots \\
& r_{k-1} \leftarrow \mathcal{U}_{m_{k-1}}, \pi_{k-1} \leftarrow \tilde{\mathsf{P}}_{k-1}^{\mathcal{A}}(x; r_1, \ldots, \pi_{i-1}, r_i, \ldots, r_{k-1}) \\
& r_k \leftarrow \mathcal{U}_{m_k} \\
& \text{output } (r_1, \ldots, \pi_{i-1}, r_i, \pi_i \ldots, r_k, 1)
\end{aligned}
$$

Observe that $(r_1, \ldots, \pi_{i-1}, r_i)$ is sampled from the simulator in $D_M^{(i)}$ while it is sampled by the protocol $\langle \mathsf{P}, \mathsf{V} \rangle$ in $D_P^{(i)}$; and the later part of the outputs $(\pi_i, \ldots, r_k)$ are generated in the same way in both $D_M^{(i)}$ and $D_P^{(i)}$. By the data processing inequality

$$\Delta_c \left(D_M^{(i)}, D_P^{(i)} \right) \leq \Delta_c(\mathsf{Sim}(x); \mathsf{View}\langle \mathsf{P}, \mathsf{V} \rangle(x, w)) \leq \epsilon_z(n). \tag{16}$$

Note that when the first message is sent by the prover, that is $t = 2k - 1$, then the distributions $D_M^{(1)}$ and $D_P^{(1)}$ are identical.

The only difference between $D_S^{(i+1)}$ and $D_M^{(i)}$ is how the i-th proof is sampled. For $D_S^{(i+1)}$, equivalently, $(r_1, \pi_1, \ldots, r_i, \pi_i)$ is generated by:

$$\begin{aligned}
&\rho \leftarrow \mathcal{U}_\ell \\
&(r_1, \pi_1, \ldots, r_i, 1) \leftarrow f_x(i, \rho) \\
&\pi_i \leftarrow \mathsf{Sim}_{2i}(x; \rho)
\end{aligned}$$

Recall the prover's strategy $\tilde{\mathsf{P}}$, $(r_1, \pi_1, \ldots, r_i, \pi_i)$ in $D_M^{(i)}$ is sampled by

$$\begin{aligned}
&\rho \leftarrow \mathcal{U}_\ell \\
&(r_1, \pi_1, \ldots, r_i, 1) \leftarrow f_x(i, \rho) \\
&\hat{\rho} \leftarrow \mathcal{A}_2(r_1, \pi_1, \ldots, r_i, 1) \\
&\pi_i \leftarrow \mathsf{Sim}_{2i}(x; \hat{\rho})
\end{aligned}$$

The distance between their marginal distributions on $(r_1, \pi_1, \ldots, r_i, \pi_i)$ is upper bounded by $\Delta_s((i, \mathcal{U}_\ell), f_x(i, \mathcal{U}_\ell); \mathcal{A}(f(i, \mathcal{U}_\ell)), f_x(i, \mathcal{U}_\ell))$. The remaining outputs of $D_S^{(i+1)}$ and $D_M^{(i)}$ follow the same sampling procedures, based on the previous transcript $(r_1, \pi_1, \ldots, r_i, \pi_i)$. Then by the data processing inequality, we obtain

$$\Delta_s \left(D_S^{(i+1)}, D_M^{(i)} \right) \leq \Delta_s \left((i, \mathcal{U}_\ell), f_x(i, \mathcal{U}_\ell); \mathcal{A}(f(i, \mathcal{U}_\ell)), f_x(i, \mathcal{U}_\ell) \right), \tag{17}$$

Consider the distance between $D_P^{(k)}$ and the following distribution D_P:

$$\begin{aligned}
D_P : \quad & r_1 \leftarrow \mathcal{U}_{m_1}, \pi_1 \leftarrow \mathsf{P}_1(x, w; r_1) \\
& \ldots \\
& r_k \leftarrow \mathcal{U}_{m_k}, \pi_k \leftarrow \mathsf{P}_k(x, w; r_1, \ldots, r_k) \\
& a \leftarrow \mathsf{V}(x; r_1, \ldots, \pi_k) \\
& \text{output } (r_1, \ldots, \ldots, r_k, a)
\end{aligned}$$

The only difference between D_P and $D_P^{(k)}$ is that, the last verification bit in D_P is sampled by the verifier V while the counterpart in $D_P^{(k)}$ is assigned to be 1. By completeness and using the same argument as (10)

$$\Delta_s\left(D_P; D_P^{(k)}\right) \leq \epsilon_c(n). \tag{18}$$

By zero-knowledge, we have

$$\Delta_c(D_S; D_P) \leq \Delta_c(\mathsf{Sim}(x); \mathsf{View}\langle \mathsf{P}, \mathsf{V}\rangle(x, w)) \leq \epsilon_z(n). \tag{19}$$

Combining (15), (16), (17), (18) and (19), it follows from the triangle inequality that, when $t = 2k$

$$\begin{aligned}
\Delta_c(D_S; D_I) &\leq \Delta_c(D_S; D_P) + \Delta_c\left(D_P; D_P^{(k)}\right) \\
&\quad + \sum_{i=1}^{k-1}\left(\Delta_c\left(D_P^{(i+1)}; D_S^{(i+1)}\right) + \Delta_c\left(D_S^{(i+1)}; D_M^{(i)}\right) + \Delta_c\left(D_M^{(i)}; D_P^{(i)}\right)\right) \\
&\leq \epsilon_c(n) + (2k-1)\cdot\epsilon_z(n) + \sum_{i\in[k-1]} \Delta_s((i,\mathcal{U}_\ell), f_x(i,\mathcal{U}_\ell); \mathcal{A}(f(i,\mathcal{U}_\ell)), f_x(i,\mathcal{U}_\ell)) \\
&\leq \epsilon_c(n) + (2k-1)\cdot\epsilon_z(n) + \frac{k}{p(n)}.
\end{aligned}$$

When $t = 2k-1$, $\Delta_c(D_M^{(1)}, D_P^{(1)}) = 0$, thus

$$\Delta_c(D_S; D_I) \leq \epsilon_c(n) + (2k-2)\cdot\epsilon_z(n) + \frac{k}{p(n)}.$$

Note that the last inequality holds because, assuming without loss of generality that the first part of the output of $\mathcal{A}$ is always correct,

$$\frac{1}{k}\sum_{i\in[k]} \Delta_s((i,\mathcal{U}_\ell), f_x(i,\mathcal{U}_\ell); \mathcal{A}(f(i,\mathcal{U}_\ell)), f_x(i,\mathcal{U}_\ell)) = \Delta_s\left(\mathcal{U}, f_x(\mathcal{U}); \mathcal{A}(f_x(\mathcal{U})), f_x(\mathcal{U})\right).$$

We conclude that

$$\begin{aligned}
\Pr\left[\langle\tilde{\mathsf{P}}^{\mathcal{A}}, \mathsf{V}\rangle(x) = 1\right] &\geq \Pr_{s\leftarrow D_I}\left[f_x(\mathcal{A}(s)) = s\right] \\
&\geq \Pr_{s\leftarrow D_S}\left[f_x(\mathcal{A}(s)) = s\right] - \Delta_c(D_S, D_I) \\
&\geq 1 - \epsilon_c(n) - (t-1)\cdot\epsilon_z(n) - \frac{k}{p(n)}.
\end{aligned}$$

This proves the claim. □

3.3 Reductions from Constant-Round Public-Coin ZK

For the lemma below, as opposed to the other (similar) ones, we consider the number of rounds in the protocol rather than the number of messages. Recall that a round consists of a message from the verifier, followed by a message from the prover. If the protocol starts with the prover sending a message, we think of it as starting with an empty message from the verifier instead.

Lemma 3.5. *For some $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$ and any constant $k \in \mathbb{N}$, suppose a language $\mathcal{L} \in \mathsf{NP}$ has a k-round public-coin $(\epsilon_c, \epsilon_s, \epsilon_z)$-ZK proof or argument. Then there exists a polynomial-time oracle-aided algorithm R and families of oracle-aided functions $\mathcal{F}_1, \ldots, \mathcal{F}_k$ satisfying the following.*

For $i \in [k]$, the family $\mathcal{F}_i = \{f_{i,x}\}_{x \in \{0,1\}^*}$ consists of functions that require access to $(k-i)$ oracles, and are polynomial-time computable given these oracles. For any sequence of probabilistic polynomial-time algorithms $\mathcal{A}_1, \ldots, \mathcal{A}_k$, any polynomial p, and all large enough n, we have the following.

1. *For any $x \in \mathcal{L}_n$, if for all $i \in [k]$, $\mathcal{A}_i$ distributionally inverts $f_{i,x}^{\mathcal{A}_{i+1},\ldots,\mathcal{A}_k}$ with deviation at most $1/p(n)$, then*

$$\Pr\left[R^{\mathcal{A}_1 \ldots \mathcal{A}_k}(x) = 1\right] \geq 1 - \epsilon_c(n) - k \cdot \epsilon_z(n) - \frac{3k-2}{p(n)} - \mathsf{negl}(n).$$

2. *For any $x \in \{0,1\}^n \setminus \mathcal{L}_n$,*

$$\Pr\left[R^{\mathcal{A}_1 \ldots \mathcal{A}_k}(x) = 1\right] \leq \epsilon_s(n).$$

Proof of Lemma 3.5. Denote by n the input length, and fix any polynomial p. Let Sim be the simulator that satisfies the zero-knowledge requirement, and suppose it uses ℓ randomness bits, where $\ell = \ell(n)$. Let k be the number of rounds which is a constant, and $m_i = m_i(n)$ be the number of bits of public randomness used by the verifier in the i-th round, for $i \in [k]$.

Construct a function family $\mathcal{F}_k = \{f_{k,x} : x \in \{0,1\}^n\}_{n \in \mathbb{N}}$ as follows

$\underline{f_{k,x}(\rho)}$:

1. $(r_1, \pi_1, \ldots, r_k, \pi_k) \leftarrow \mathsf{Sim}(x; \rho)$
2. $a \leftarrow \mathsf{V}(r_1, \pi_1, \ldots, r_k, \pi_k)$
3. Output $(r_1, \pi_1, \ldots, r_k, a)$

Let $\mathcal{A}_k$ be a polynomial-time algorithm for inverting $\mathcal{F}_k$. Note that these $\mathcal{A}_i$'s (defined later for $i \in [k-1]$) are potentially randomized algorithms. To leverage it in the construction of deterministic functions and oracles, we make their internal randomness explicit by treating the randomness as part of the input. Specifically, we write the algorithms as $\mathcal{A}_i(x; r_1, \pi_1, \ldots, r_i, a; \mathsf{rd})$, where x is the auxiliary input indicating that $\mathcal{A}_i$ is supposed to invert the function $f_{i,x}^{\mathcal{A}_{i+1},\ldots,\mathcal{A}_k}$ on output $(r_1, \ldots, r_i, a)$, and rd is the randomness that $\mathcal{A}_i$ uses. The same applies to $\mathcal{B}_i$'s (also defined later).

We next define an oracle-aided algorithm $\mathcal{B}_k$ that, given access to $\mathcal{A}_k$, essentially measures the acceptance likelihood of a given transcript and outputs a proof π_k as follows.

$\underline{\mathcal{B}_k^{\mathcal{A}_k}(x; r_1, \pi_1, \ldots, r_k)}$:

1. $\rho \leftarrow \mathcal{A}_k(x; r_1, \pi_1, \ldots, r_k, 1)$
2. If $f_{k,x}(\rho) = (r_1, \pi_1, \ldots, r_k, 1)$
   - $\pi_k \leftarrow \mathsf{Sim}_{2k}(x; \rho)$
   - Output $(1, \pi_k)$
3. Else output $(0, \bot)$

When algorithm $\mathcal{A}_k$ successfully finds a pre-image ρ, such that the corresponding transcript is accepted by the verifier, then $\mathcal{B}_k$ outputs 1 and a valid proof π_k such that the verification accepts the entire transcript $(r_1, \pi_1, \ldots, r_k, \pi_k)$. Otherwise, $\mathcal{B}_k$ outputs $(0, \bot)$ which represents failure of inversion.

For $i \in [k]$, we write $\mathcal{B}_{i,1}$ and $\mathcal{B}_{i,2}$ to denote the algorithm that outputs only the first component and the second of $\mathcal{B}_i$'s output respectively, and let $\mathcal{B}_i = \mathcal{B}_i^{\mathcal{A}_i \ldots \mathcal{A}_k}$ for simplicity. Later, $\mathcal{B}_i$'s are defined with respect to $\mathcal{A}_i$'s for $i \in [k-1]$.

For $i \in [k-1]$, define the family $\mathcal{F}_i = \{f_{i,x}\}_{x \in \{0,1\}^*}$ with oracle access to $\mathcal{A}_{i+1}, \ldots, \mathcal{A}_k$ as follows. Here, we set $q = n \cdot p(n)^2$. The input of function $f_{i,x}$ consists of a string ρ of length suitable for the randomness of Sim, strings $\sigma_1, \ldots, \sigma_q$ each of length m_{i+1} corresponding to the $(i+1)$-th verifier's message (public coins), and strings $\mathsf{rd}_1, \ldots, \mathsf{rd}_q$ each of the length suitable for the randomness of $\mathcal{B}_{i+1}$.

$\underline{f_{i,x}^{\mathcal{A}_{i+1} \ldots \mathcal{A}_k}(\rho, \sigma_1, \ldots, \sigma_q, \mathsf{rd}_1, \ldots, \mathsf{rd}_q)}$:

1. $(r_1, \pi_1, \ldots, r_k, \pi_k) \leftarrow \mathsf{Sim}(x; \rho)$
2. $\mathsf{est} \leftarrow \frac{1}{q} \sum_{j \in [q]} \mathcal{B}_{i+1,1}^{\mathcal{A}_{i+1} \ldots \mathcal{A}_k}(x; r_1, \pi_1, \ldots, r_i, \pi_i, \sigma_j; \mathsf{rd}_j)$

3. Output $(r_1, \pi_1, \ldots, r_i, \mathsf{est})$

With an overwhelming probability, the value est reflects the value of its expectation

$$\mathbb{E}\left[\mathsf{est}\right] = \mathop{\mathbb{E}}_{r_{i+1} \leftarrow \mathcal{U}_{m_{i+1}}} \left[\mathcal{B}_{i+1,1}(r_1, \pi_1, \ldots, r_i, \pi_i, r_{i+1})\right],$$

which in turn measures the ability of the inversion by $\mathcal{A}_{i+1}$ to find a completion of the simulated partial transcript for the first i rounds specified by ρ to an accepting one.

Similarly, let $\mathcal{A}_i$ be a polynomial-time algorithm to potentially invert $\mathcal{F}_i$. Let $\tau = 1/p(n)$. An efficient oracle-aided algorithm $\mathcal{B}_i$ is defined as follows.

$\underline{\mathcal{B}_i^{\mathcal{A}_i \ldots \mathcal{A}_k}(x; r_1, \pi_1, \ldots, r_i)}$:

1. For $a = q^{k-i}, q^{k-i} - 1, \ldots, 2, 1$ in decreasing order
   - $(\rho, \sigma_1, \ldots, \sigma_q, \mathsf{rd}_1, \ldots, \mathsf{rd}_q) \leftarrow \mathcal{A}_i(x; r_1, \pi_1, \ldots, r_i, a/q^{k-i})$
   - If $f_{i,x}^{\mathcal{A}_{i+1}, \ldots, \mathcal{A}_k}(\rho, \sigma_1, \ldots, \sigma_q, \mathsf{rd}_1, \ldots, \mathsf{rd}_q) = (r_1, \pi_1, \ldots, r_i, a/q^{k-i})$
     - $\pi_i \leftarrow \mathsf{Sim}_{2i}(x; \rho)$
     - $\hat{\sigma}_1, \ldots, \hat{\sigma}_q \leftarrow \mathcal{U}_{m_{i+1}}$
     - $\hat{\mathsf{est}} \leftarrow \frac{1}{q} \sum_{j \in [q]} \mathcal{B}_{i+1,1}^{\mathcal{A}_{i+1} \ldots \mathcal{A}_k}(x; r_1, \pi_1, \ldots, r_i, \pi_i, \hat{\sigma}_j)$
     - If $\left|\hat{\mathsf{est}} - a/q^{k-i}\right| < \tau$
       - Output $(a/q^{k-i}, \pi_i)$
2. Output $(0, \perp)$

The output consists of two parts: a value a/q^{k-1} in $[0, 1]$ and a proof π_i. With an overwhelming probability, the value a/q^{k-1} estimates well the acceptance probability when the prover's response to $(r_1, \pi_1, \ldots, r_i)$ is π_i. This is formally defined and proved later. The algorithm iterates over a from q^{k-i} to 1 to find an almost optimal proof that maximizes the acceptance rate, which is utilized to construct the distinguisher later.

Equipped with the algorithms $\mathcal{B}_i$'s, we are ready to construct a polynomial-time prover $\tilde{\mathsf{P}}$ with oracle access to $\mathcal{A}_i$'s. Denote by $\tilde{\mathsf{P}}_i$ the prover's algorithm in the i-th round.

$$\tilde{\mathsf{P}}_i^{\mathcal{A}_i \ldots \mathcal{A}_k}(x; r_1, \pi_1, \ldots, r_i) = \mathcal{B}_{i,2}^{\mathcal{A}_i \ldots \mathcal{A}_k}(x; r_1, \pi_1, \ldots, r_i)$$

The reduction R runs the protocol $\langle \tilde{\mathsf{P}}, \mathsf{V} \rangle(x)$ itself and outputs 1 if and only if the protocol accepts. The efficiency of $\tilde{\mathsf{P}}$ ensures the polynomial runtime of R. We state the following claims.

Claim 3.3. *When $x \in \mathcal{L}_n$, if for all $i \in [k]$, $\mathcal{A}_i$ distributionally inverts $f_{i,x}^{\mathcal{A}_{i+1},\ldots,\mathcal{A}_k}$ with deviation at most $1/p(n)$, then*

$$\Pr\left[\langle \tilde{\mathsf{P}}^{\mathcal{A}_1\ldots\mathcal{A}_k}, \mathsf{V}\rangle(x) = 1\right] \geq 1 - \epsilon_c(n) - k \cdot \epsilon_z(n) - \frac{3k-2}{p(n)} - \mathsf{negl}(n).$$

Claim 3.4. *When $x \in \{0,1\}^n \setminus \mathcal{L}_n$,*

$$\Pr\left[\langle \tilde{\mathsf{P}}^{\mathcal{A}_1\ldots\mathcal{A}_k}, \mathsf{V}\rangle(x) = 1\right] \leq \epsilon_s(n).$$

Claim 3.4 follows immediately from the soundness of the ZK protocol. We present the proof of Claim 3.3 in Section 3.3.1. Putting the above claims together completes the proof of the lemma.

□

Remark 3.2. *In this approach, the functions are defined recursively and the number of function families is closely related to the round number k, so we can only achieve the implication result for any constant k. We do not know how to make it work beyond a finite number of rounds. This issue has also been observed in other work that employ similar recursive constructions [BT24, MV24].*

Corollary 3.6. *For $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$ and a constant $k \in \mathbb{N}$, suppose there is an NP language $\mathcal{L} \notin \mathsf{P/poly}$ that has a k-round public-coin $(\epsilon_c, \epsilon_s, \epsilon_z)$-ZK proof or argument with $\epsilon_c + \epsilon_s + k \cdot \epsilon_z <_n 1$. Then infinitely-often auxiliary-input one-way functions exist.*

Proof of Corollary 3.6. Let R be the oracle-aided algorithm and $\mathcal{F}_1, \ldots, \mathcal{F}_k$ be the oracle-aided function family defined in the proof of Lemma 3.5. Since $\epsilon_c + \epsilon_s + k \cdot \epsilon_z <_n 1$, there exists a polynomial q such that

$$\epsilon_c(n) + \epsilon_s(n) + k \cdot \epsilon_z(n) + \frac{1}{q(n)} < 1$$

holds for all sufficiently large $n \in \mathbb{N}$. Then, let $p = 3kq$. Suppose that there is no infinitely-often ai-dOWF, which means that there are (non-uniform) PPT algorithms $\mathcal{A}_1, \ldots, \mathcal{A}_k$ such that, for sufficiently large $n \in \mathbb{N}$, for every $x \in \mathcal{L}_n$ and all $i \in [k]$, $\mathcal{A}_i$ distributionally inverts $f_{i,x}$ with deviation at most $1/p(n)$. By Lemma 3.3

$$\Pr\left[R^{\mathcal{A}_1\ldots\mathcal{A}_k}(x) = 1\right] \geq 1 - \epsilon_c(n) - k \cdot \epsilon_z(n) - \frac{3k-2}{p(n)} - \mathsf{negl}(n);$$

and for every $x \in \{0,1\}^n \setminus \mathcal{L}_n$,

$$\Pr\left[R^{\mathcal{A}_1\ldots\mathcal{A}_k}(x) = 1\right] \leq \epsilon_s(n).$$

But the $\mathcal{A}_i$'s and R are polynomial-time algorithms and

$$1 - \epsilon_c(n) - k \cdot \epsilon_z(n) - \frac{3k-2}{3k} \cdot \frac{1}{q(n)} >_n \epsilon_s(n),$$

which contradicts $\mathcal{L} \notin \mathsf{P/poly}$. Hence, $\mathcal{F}$ is an auxiliary-input distributional one-way function. By [IL89], the existence of ai-dOWFs implies the existence of ai-OWFs, which concludes the proof. □

Remark 3.3. *Our approach to obtaining a contradiction relies on the overlap of input lengths on which all of the algorithms $\mathcal{A}_1, \ldots, \mathcal{A}_k$ succeed in their respective inversions. The negation of this assumption only implies the existence of infinitely-often one-way functions. Given that we only get infinitely-often security anyway, we are able to weaken our assumption to $\mathcal{L} \notin \mathsf{P/poly}$ instead of needing $\mathcal{L} \notin \mathsf{ioP/poly}$ as in our other results.*

3.3.1 Proof of Claim 3.3

Before proceeding with Claim 3.3, first consider the value

$$\mathcal{B}(x) = \mathop{\mathbb{E}}_{r_1 \leftarrow \mathcal{U}_{m_1}} \left[\mathcal{B}_{1,1}(x; r_1)\right]. \tag{20}$$

For simplicity, we omit the superscript that denotes the oracle access throughout this section. We note that $\mathcal{B}$ approximates the acceptance probability of $\langle \tilde{\mathsf{P}}, \mathsf{V} \rangle$ well, that is

$$\mathcal{B}(x) \approx \Pr\left[\langle \tilde{\mathsf{P}}, \mathsf{V} \rangle(x) = 1\right],$$

which is formally stated as follows.

Claim 3.5. *For k, p as defined in Lemma 3.5, where k is a constant representing the number of rounds in the protocol and p is a polynomial, we have*

$$\left|\Pr\left[\langle \tilde{\mathsf{P}}, \mathsf{V} \rangle(x) = 1\right] - \mathcal{B}(x)\right| \leq \frac{2(k-1)}{p(n)} + \mathsf{negl}(n).$$

Claim 3.5 states that $\mathcal{B}$ provides a good approximation of the probability that $\langle \tilde{\mathsf{P}}, \mathsf{V} \rangle$ accepts on x. To derive the desired lower bound on this probability as stated in Claim 3.3, for $x \in \mathcal{L}_n$, we therefore establish a lower bound on $\mathcal{B}$ assuming that all $\mathcal{A}_i$'s distributionally inverts $f_{i,x}$ well.

Claim 3.6. *For $x \in \mathcal{L}_n$ and $\epsilon_c, \epsilon_z, k, p$ as defined in Lemma 3.5, where k is a constant representing the number of rounds in protocol and p is a polynomial, we assume that for all $i \in [k]$, the efficient algorithm $\mathcal{A}_i$ satisfies that*

$$\Delta_s(\mathcal{U}, f_{i,x}(\mathcal{U}); \mathcal{A}_i(f_{i,x}(\mathcal{U})), f_{i,x}(\mathcal{U})) < \frac{1}{p(n)} \tag{21}$$

then we have

$$\mathcal{B}(x) \geq 1 - \epsilon_c(n) - k \cdot \epsilon_z(n) - \frac{k}{p(n)} - \mathsf{negl}(n).$$

With Claim 3.5 and Claim 3.6, both of whose proofs are provided immediately afterwards, we are ready to prove Claim 3.3.

Proof of Claim 3.3. From Claim 3.5, we have

$$\left|\Pr\left[\langle\tilde{\mathsf{P}}, \mathsf{V}\rangle(x) = 1\right] - \mathcal{B}(x)\right| \leq \frac{2(k-1)}{p(n)} + \mathsf{negl}(n).$$

For $x \in \mathcal{L}_n$, by Claim 3.6, $\mathcal{B}(x)$ can be lower bounded by

$$\mathcal{B}(x) \geq 1 - \epsilon_c(n) - k \cdot \epsilon_z(n) - \frac{k}{p(n)} - \mathsf{negl}(n).$$

Therefore,

$$\Pr\left[\langle\tilde{\mathsf{P}}, \mathsf{V}\rangle(x) = 1\right] \geq 1 - \epsilon_c(n) - k \cdot \epsilon_z(n) - \frac{3k-2}{p(n)} - \mathsf{negl}(n).$$

□

Proof of Claim 3.5. For $i \in [k]$, denote by $\mathcal{T}_i(x; r_1, \pi_1, \ldots, r_i)$ the following value, which represents the true probability that $\langle\tilde{\mathsf{P}}, \mathsf{V}\rangle(x)$ accepts for the first $(2i-1)$ messages being $(r_1, \pi_1, \ldots, r_i)$ in the protocol. In particular,

$$\mathcal{T}_i(x; r_1, \pi_1, \ldots, r_i) = \Pr_{\substack{\pi_i \leftarrow \tilde{\mathsf{P}}_i(x; r_1, \pi_1, \ldots, r_i) \\ \cdots \\ r_k \leftarrow \mathcal{U}_{m_k}, \pi_k \leftarrow \tilde{\mathsf{P}}_k(x; r_1, \pi_1, \ldots, r_k)}} \left[\mathsf{V}(x; r_1, \ldots, \pi_k) = 1\right].$$

Observe that $\mathcal{T}_i(x; r_1, \pi_1, \ldots, r_i)$ can be defined recursively from $\mathcal{T}_{i+1}(x; r_1, \pi_1, \ldots, r_{i+1})$

$$\begin{aligned}
\mathcal{T}_i(x; r_1, \pi_1, \ldots, r_i) &= \Pr_{\substack{\pi_i \leftarrow \tilde{\mathsf{P}}_i(x; r_1, \pi_1, \ldots, r_i) \\ \cdots \\ r_k \leftarrow \mathcal{U}_{m_k}, \pi_k \leftarrow \tilde{\mathsf{P}}_k(x; r_1, \pi_1, \ldots, r_k)}} \left[\mathsf{V}(x; r_1, \ldots, \pi_k) = 1\right] \\
&= \Pr_{\substack{\pi_i \leftarrow \tilde{\mathsf{P}}_i(x; r_1, \pi_1, \ldots, r_i) \\ r_{i+1} \leftarrow \mathcal{U}_{m_{i+1}}, \pi_{i+1} \leftarrow \tilde{\mathsf{P}}_i(x; r_1, \pi_1, \ldots, r_{i+1}) \\ \cdots \\ r_k \leftarrow \mathcal{U}_{m_k}, \pi_k \leftarrow \tilde{\mathsf{P}}_k(x; r_1, \pi_1, \ldots, r_k)}} \left[\mathsf{V}(x; r_1, \ldots, \pi_k) = 1\right] \\
&= \mathop{\mathbb{E}}_{\substack{\pi_i \leftarrow \mathcal{B}_{i,2}(x; r_1, \pi_1, \ldots, r_i) \\ r_{i+1} \leftarrow \mathcal{U}_{m_{i+1}}}} \left[\mathcal{T}_{i+1}(x; r_1, \pi_1, \ldots, r_i, \pi_i, r_{i+1})\right]. \qquad (22)
\end{aligned}$$

For the last equality, we remind the reader that

$$\tilde{\mathsf{P}}_i(x; r_1, \pi_1, \ldots, r_i) = \mathcal{B}_{i,2}(x; r_1, \pi_1, \ldots, r_i).$$

We also note that the probability that $\langle \tilde{\mathsf{P}}, \mathsf{V} \rangle(x)$ accepts can be written as

$$\Pr\left[\langle \tilde{\mathsf{P}}, \mathsf{V} \rangle(x) = 1\right] = \mathop{\mathbb{E}}_{r_1 \leftarrow \mathcal{U}_{m_1}} \left[\mathcal{T}_1(x; r_1)\right]. \tag{23}$$

In the following, we show a more general statement: for any $i \in [k]$, for any input $(x; r_1, \pi_1, \ldots, r_i)$, the expected value of the output $\mathcal{B}_{i,1}$ is close to $\mathcal{T}_i$

$$\left|\mathcal{T}_i(x; r_1, \pi_1, \ldots, r_i) - \mathbb{E}\left[\mathcal{B}_{i,1}(x; r_1, \pi_1, \ldots, r_i)\right]\right| \leq 2(k-i)(\tau + e^{-2\tau^2 q}) \tag{24}$$

where τ and q are the parameters defined in $\mathcal{B}_i$'s to be $1/p(n)$ and $n \cdot p(n)^2$, respectively. This inequality implies that for any $i \in [k]$, the expected value of $\mathcal{B}_{i,1}(x; r_1, \pi_1, \ldots, r_i)$ always reflects the probability that the protocol $\langle \tilde{\mathsf{P}}, \mathsf{V} \rangle(x)$ accepts conditioned on the first $(2i-1)$ messages being $(r_1, \pi_1, \ldots, r_i)$.

We prove (24) by induction. When $i = k$, on input $(x; r_1, \pi_1, \ldots, r_k)$, recall that the algorithm $\mathcal{B}_k$ proceeds as follows.

$\underline{\mathcal{B}_k(x; r_1, \pi_1, \ldots, r_k)}$:

1. $\rho \leftarrow \mathcal{A}_k(r_1, \pi_1, \ldots, r_k, 1)$
2. If $f_{k,x}(\rho) = (r_1, \pi_1, \ldots, r_k, 1)$
   - $\pi_k \leftarrow \mathsf{Sim}_{2k}(x; \rho)$
   - Output $(1, \pi_k)$
3. Else output $(0, \perp)$

The probability that V accepts $(x; r_1, \pi_1, \ldots, r_k)$ when the last proof π_k is generated by $\tilde{\mathsf{P}}_k = \mathcal{B}_{k,2}$ is exactly the probability that $\mathcal{B}_{k,1}$ outputs 1, since once $f_{k,x}(\rho) = (r_1, \pi_1, \ldots, r_k, 1)$ holds, the π_k corresponding to ρ satisfies $\mathsf{V}(x; r_1, \pi_1, \ldots, r_k, \pi_k) = 1$.

$$\begin{aligned}
\mathcal{T}_k(x; r_1, \pi_1, \ldots, r_k) &= \Pr_{\pi_k \leftarrow \mathcal{B}_{k,2}(x; r_1, \pi_1, \ldots, r_k)} \left[\mathsf{V}(x; r_1, \pi_1, \ldots, r_k, \pi_k) = 1\right] \\
&= \Pr\left[\mathcal{B}_{k,1}(x; r_1, \pi_1, \ldots, r_k) = 1\right] \\
&= \mathbb{E}\left[\mathcal{B}_{k,1}(x; r_1, \pi_1, \ldots, r_k)\right].
\end{aligned}$$

Thus, (24) is satisfied when $i = k$.

Suppose that $i = l + 1$ meets the condition (24) for $l \in [k-1]$. Recall the adaptive construction of $\mathcal{B}_l$'s,

$\underline{\mathcal{B}_l(x; r_1, \pi_1, \ldots, r_l)}$:

1. For $a = q^{k-l}, q^{k-l} - 1, \ldots, 2, 1$ in decreasing order
   - $(\rho, \sigma_1, \ldots, \sigma_q, \mathsf{rd}_1, \ldots, \mathsf{rd}_q) \leftarrow \mathcal{A}_l(r_1, \pi_1, \ldots, r_l, a/q^{k-l})$
   - If $f_{l,x}(\rho, \sigma_1, \ldots, \sigma_q, \mathsf{rd}_1, \ldots, \mathsf{rd}_q) = (r_1, \pi_1, \ldots, r_l, a/q^{k-l})$
     - $\pi_l \leftarrow \mathsf{Sim}_{2l}(x; \rho)$
     - $\hat{\sigma}_1, \ldots, \hat{\sigma}_q \leftarrow \mathcal{U}_{m_{l+1}}$
     - $\hat{\mathsf{est}} \leftarrow \frac{1}{q} \sum_{j \in [q]} \mathcal{B}_{l+1,1}(x; r_1, \pi_1, \ldots, r_l, \pi_l, \hat{\sigma}_j)$
     - If $\left|\hat{\mathsf{est}} - a/q^{k-l}\right| < \tau$
       - Output $(a/q^{k-l}, \pi_l)$
2. Output $(0, \perp)$

Denote the output of $\mathcal{B}_l$ by (est_o, π_l). Denote by $\hat{\mathsf{est}}$ the last estimate computed by $\mathcal{B}_l$ prior to returning, setting it to 0 if $\mathcal{B}_l$ outputs $(0, \perp)$. It is ensured by the definition of $\mathcal{B}_l$ that

$$\left|\mathsf{est}_o - \hat{\mathsf{est}}\right| < \tau.$$

If $\pi_l \neq \perp$, since the estimation $\hat{\mathsf{est}}$ is computed by averaging q samples of $\mathcal{B}_{l+1,1}$, then by a Hoeffding bound

$$\Pr\left[\left|\hat{\mathsf{est}} - \mathop{\mathbb{E}}_{r_{l+1} \leftarrow \mathcal{U}_{m_{l+1}}} [\mathcal{B}_{l+1,1}(x; r_1, \pi_1, \ldots, r_l, \pi_l, r_{l+1})]\right| > \tau\right] < 2e^{-2\tau^2 q}.$$

Further, the above inequality remains true in the case $\pi_l = \perp$, as in this case both quantities $\hat{\mathsf{est}}$ and the expectation of $\mathcal{B}_{l+1,1}$ above are 0.

Thus, the output of $\mathcal{B}_l$ is guaranteed to satisfy

$$\Pr_{(\mathsf{est}_o, \pi_l) \leftarrow \mathcal{B}_l(x; r_1, \pi_1, \ldots, r_l)}\left[\left|\mathsf{est}_o - \mathop{\mathbb{E}}_{r_{l+1} \leftarrow \mathcal{U}_{m_{l+1}}} [\mathcal{B}_{l+1,1}(x; r_1, \ldots, \pi_l, r_{l+1})]\right| > 2\tau\right] < 2e^{-2\tau^2 q}.$$

As the outputs of $\mathcal{B}_{l,1}$ and $\mathcal{B}_{l+1,1}$ always lie in the range $[0, 1]$, the above implies

$$\left|\mathbb{E}\left[\mathcal{B}_{l,1}(x; r_1, \pi_1, \ldots, r_l)\right] - \mathop{\mathbb{E}}_{\substack{\pi_l \leftarrow \mathcal{B}_{l,2}(x; r_1, \pi_1, \ldots, r_l) \\ r_{l+1} \leftarrow \mathcal{U}_{m_{l+1}}}} [\mathcal{B}_{l+1,1}(x; r_1, \ldots, \pi_l, r_{l+1})]\right| < 2(\tau + e^{-2t^2 q}). \tag{25}$$

Therefore, we derive that

$$\begin{aligned}
&\left|\mathcal{T}_l(x;r_1,\pi_1,\ldots,r_l)-\mathbb{E}\left[\mathcal{B}_{l,1}(x;r_1,\pi_1,\ldots,r_l)\right]\right| \\
&< \left|\mathop{\mathbb{E}}_{\substack{\pi_l\leftarrow\mathcal{B}_{l,2}(x;r_1,\pi_1,\ldots,r_l)\\ r_{l+1}\leftarrow\mathcal{U}_{m_{l+1}}}}\left[\mathcal{T}_{l+1}(x;r_1,\ldots,\pi_l,r_{l+1})-\mathbb{E}\left[\mathcal{B}_{l+1,1}(x;r_1,\ldots,\pi_l,r_{l+1})\right]\right]\right| \\
&\quad + 2(\tau+e^{-2t^2q}) \\
&\leq \mathop{\mathbb{E}}_{\substack{\pi_l\leftarrow\mathcal{B}_{l,2}(x;r_1,\pi_1,\ldots,r_l)\\ r_{l+1}\leftarrow\mathcal{U}_{m_{l+1}}}}\left[\left|\mathcal{T}_{l+1}(x;r_1,\ldots,\pi_l,r_{l+1})-\mathbb{E}\left[\mathcal{B}_{l+1,1}(x;r_1,\ldots,\pi_l,r_{l+1})\right]\right|\right] \\
&\quad + 2(\tau+e^{-2t^2q}) \\
&\leq 2(k-l)(\tau+e^{-2t^2q}).
\end{aligned}$$

The first inequality follows from the triangle inequality and (22) and (25). The last inequality holds since we assume that (24) is true for $i=l+1$. Therefore, we find that (24) holds for $i=l$, which completes the induction and further concludes that (24) is satisfied for any $i\in[k]$.

When $q=n\cdot p(n)^2$ and $\tau=1/p(n)$, by (24), we obtain that, for any x and r_1

$$\left|\mathcal{T}_1(x;r_1)-\mathbb{E}\left[\mathcal{B}_{1,1}(x;r_1)\right]\right|\leq 2(k-1)(\tau+e^{-2t^2q})\leq 2(k-1)\left(\frac{1}{p(n)}+e^{-2n}\right).$$

By combing the above with (20) and (23), we conclude

$$\begin{aligned}
\left|\Pr\left[\langle\tilde{\mathsf{P}},\mathsf{V}\rangle(x)=1\right]-\mathcal{B}(x)\right| &= \left|\mathop{\mathbb{E}}_{r_1\leftarrow\mathcal{U}_{m_1}}\left[\mathcal{T}_1(x;r_1)\right]-\mathop{\mathbb{E}}_{r_1\leftarrow\mathcal{U}_{m_1}}\left[\mathcal{B}_{1,1}(x;r_1)\right]\right| \\
&\leq \frac{2(k-1)}{p(n)}+\mathsf{negl}(n).
\end{aligned}$$

□

Proof of Claim 3.6. Let w be a valid NP witness for x satisfying $(x,w)\in\mathcal{R}_{\mathcal{L}}$. Consider the following distributions.

1. D_S: sample $\rho\leftarrow\mathcal{U}_\ell$, $(r_1,\pi_1,\ldots,r_k,\pi_k)\leftarrow\mathsf{Sim}(x;\rho)$, $a\leftarrow\mathsf{V}(x;r_1,\ldots,\pi_k)$, output $(r_1,\pi_1,\ldots,r_k,a)$
2. D_I: sample $(r_1,\pi_1,\ldots,r_k)\leftarrow\mathsf{View}_{2k-1}\langle\mathsf{P},\mathsf{V}\rangle(x,w)$, output $(r_1,\pi_1,\ldots,r_k,1)$

For simplicity, denote the transcript $(r_1, \pi_1, \dots)$ sampled by $\mathsf{View}_i\langle \mathsf{P}, \mathsf{V}\rangle(x, w)$, which only contains the first i messages in the protocol $\langle \mathsf{P}_w, \mathsf{V}\rangle$ on x. Analogous to Eq. (11), by completeness, zero-knowledge, and data processing inequality,

$$\Delta_c(D_S; D_I) \leq \epsilon_c(n) + \epsilon_z(n).$$

Recall the algorithm $\mathcal{B}_k$.

$\underline{\mathcal{B}_k(x; r_1, \pi_1, \dots, r_k)}$:

1. $\rho \leftarrow \mathcal{A}_k(r_1, \pi_1, \dots, r_k, 1)$
2. If $f_{k,x}(\rho) = (r_1, \pi_1, \dots, r_k, 1)$
   - $\pi_k \leftarrow \mathsf{Sim}_{2k}(x; \rho)$
   - Output $(1, \pi_k)$
3. Else output $(0, \perp)$

Since the inverter $\mathcal{A}_k$ satisfies (21), and the transcript on which $\mathcal{B}_k$ queries $\mathcal{A}_k$ is the same as D_I when $(r_1, \pi_1, \dots, r_k)$ is generated by the protocol, by the data processing inequality

$$\begin{aligned}\mathop{\mathbb{E}}_{(r_1,\pi_1,\dots,r_k)\leftarrow \mathsf{View}_{2k-1}\langle \mathsf{P},\mathsf{V}\rangle(x)} [\mathcal{B}_{k,1}(x; r_1, \pi_1, \dots, r_k)] &\geq 1 - \frac{1}{p(n)} - \Delta_c(D_S; D_I) \\ &\geq 1 - \frac{1}{p(n)} - \epsilon_c(n) - \epsilon_z(n). \end{aligned} \tag{26}$$

For each $i \in [k]$, we consider the following values.

$$\mathop{\mathbb{E}}_{(r_1,\pi_1,\dots,r_i)\leftarrow \mathsf{View}_{2i-1}\langle \mathsf{P},\mathsf{V}\rangle(x)} [\mathcal{B}_{i,1}(x; r_1, \pi_1, \dots, r_i)]$$

With the observation from (24), the above value is somewhat related to the acceptance probability when a prover runs the ZK protocol with the honest prover's strategy P for the first $(i-1)$ rounds and our inverter-based prover $\tilde{\mathsf{P}}$ for the rest of the rounds.

Next, we show the connection between cases i and $i+1$. In particular, we provide a lower bound $\mathbb{E}[\mathcal{B}_{i,1}]$ using $\mathbb{E}[\mathcal{B}_{i+1,1}]$. Let Est_i denote the estimation procedure of the expectation of $\mathcal{B}_{i+1,1}$. More specifically,

$\underline{\mathsf{Est}_i(x; r_1, \pi_1, \dots, r_i, \pi_i; \sigma_1, \dots, \sigma_q)}$:

1. $\mathsf{est} \leftarrow \frac{1}{q}\sum_{i\in[q]} \mathcal{B}_{i+1,1}(x; r_1, \pi_1, \ldots, r_i, \pi_i, \sigma_j)$
2. Output est

The input $(\sigma_1, \ldots, \sigma_q)$ can be viewed as randomness. For simplicity, we sometimes omit $(\sigma_1, \ldots, \sigma_q)$ in the input to describe a randomized algorithm $\mathsf{Est}_i(x; r_1, \pi_1, \ldots, r_i, \pi_i)$ where $\sigma_1, \ldots, \sigma_q$ are drawn uniformly randomly. Recall that $\mathcal{B}_i$ works as follows.

$\underline{\mathcal{B}_i(x; r_1, \pi_1, \ldots, r_i)}$:

1. For $a = q^{k-i}, q^{k-i} - 1, \ldots, 2, 1$ in decreasing order
   - $(\rho, \sigma_1, \ldots, \sigma_q, \mathsf{rd}_1, \ldots, \mathsf{rd}_q) \leftarrow \mathcal{A}_i(r_1, \pi_1, \ldots, r_i, a/q^{k-i})$
   - If $f_{i,x}(\rho, \sigma_1, \ldots, \sigma_q, \mathsf{rd}_1, \ldots, \mathsf{rd}_q) = (r_1, \pi_1, \ldots, r_i, a/q^{k-i})$
     - $\hat{\pi}_i \leftarrow \mathsf{Sim}_{2i}(x; \rho)$
     - $\hat{\mathsf{est}} \leftarrow \mathsf{Est}_i(x; r_1, \ldots, r_1, \pi_1, \ldots, r_i, \hat{\pi}_i)$
     - If $\left|\hat{\mathsf{est}} - a/q^{k-i}\right| < \tau$
       - Output $(a/q^{k-i}, \hat{\pi}_i)$
2. Output $(0, \perp)$

Additionally, define an algorithm $\hat{\mathcal{B}}_i$ as follows that almost simulates the procedure of $\mathcal{B}_i$ in a single iteration except for the final check.

$\underline{\hat{\mathcal{B}}_i(x; r_1, \pi_1, \ldots, r_i, \mathsf{est})}$:

1. $(\rho, \sigma_1, \ldots, \sigma_q, \mathsf{rd}_1, \ldots, \mathsf{rd}_q) \leftarrow \mathcal{A}_i(r_1, \pi_1, \ldots, r_i, \mathsf{est})$
2. If $f_{i,x}(\rho, \sigma_1, \ldots, \sigma_q, \mathsf{rd}_1, \ldots, \mathsf{rd}_q) = (r_1, \pi_1, \ldots, r_i, \mathsf{est})$
   - $\hat{\pi}_i \leftarrow \mathsf{Sim}_{2i}(x; \rho)$
   - $\hat{\mathsf{est}} \leftarrow \mathsf{Est}_i(x; r_1, \pi_1, \ldots, r_i, \hat{\pi}_i)$
   - Output $\hat{\mathsf{est}}$
3. Output $\perp$

Denote by $E(\mathsf{est}, \hat{\mathsf{est}})$ the event that the distance between two inputs are close. Let $E(\mathsf{est}, \hat{\mathsf{est}}) = 0$ if the input contains $\perp$. Otherwise,

$$E(\mathsf{est}, \hat{\mathsf{est}}) = \mathbf{1}\left[\left|\mathsf{est} - \hat{\mathsf{est}}\right| < \tau\right].$$

$\mathcal{B}_{i,1}(x; r_1, \pi_1, \ldots, r_i)$ basically iterates $\hat{\mathcal{B}}_i(x; r_1, \pi_1, \ldots, r_i, \mathsf{est})$ to obtain $\hat{\mathsf{est}}$, from $\mathsf{est} = 1, 1 - 1/q^{k-i}, \ldots$ to $1/q^{k-i}$ in decreasing order, and outputs the largest est when it finds that $E(\mathsf{est}, \hat{\mathsf{est}}) = 1$. Then, for any $(x; r_1, \pi_1, \ldots, r_i)$ and any est, the following is true since, with probability at least $\Pr_{\hat{\mathsf{est}}}\left[\left|\mathsf{est} - \hat{\mathsf{est}}\right| < \tau\right]$, $\mathcal{B}_{i,1}$ outputs a value at least est.

$$\mathbb{E}\left[\mathcal{B}_{i,1}(x; r_1, \pi_1, \ldots, r_i)\right] \geq \mathop{\mathbb{E}}_{\hat{\mathsf{est}} \leftarrow \hat{\mathcal{B}}_i(x; r_1, \pi_1, \ldots, r_i, \mathsf{est})} \left[\mathsf{est} \cdot E(\mathsf{est}, \hat{\mathsf{est}})\right]. \tag{27}$$

In the following, we show a lower bound for the right-hand side considering $(r_1, \pi_1, \ldots, r_i)$ is generated by the protocol.

The probability that est' fails the event $E(\mathsf{est}, \mathsf{est}')$ is only influenced by two parts: the performance of the inverter $\mathcal{A}_i$ and the estimation of $\mathcal{B}_{i+1}$. Next, we lower-bound the probability that $E(\mathsf{est}, \mathsf{est}') = 1$ on some input distribution that is of interest to us. Consider the following procedure $\mathcal{P}$:

$\underline{\mathcal{P}_{i,x}((\rho, \sigma_1, \ldots, \sigma_q, \mathsf{rd}_1, \ldots, \mathsf{rd}_q), (r_1, \pi_1, \ldots, r_i, \mathsf{est}))}$:

1. If $f_{i,x}(\rho, \sigma_1, \ldots, \sigma_q, \mathsf{rd}_1, \ldots, \mathsf{rd}_q) = (r_1, \pi_1, \ldots, r_i, \mathsf{est})$
   - $\hat{\pi}_i \leftarrow \mathsf{Sim}_{2i}(x; \rho)$
   - $\hat{\mathsf{est}} \leftarrow \mathsf{Est}_i(x; r_1, \pi_1, \ldots, r_i, \hat{\pi}_i)$
   - Output $(\mathsf{est}, \hat{\mathsf{est}})$
2. Else output $(\mathsf{est}, \perp)$

Let est and $\hat{\mathsf{est}}$ be sampled from Est_i independently, by Chernoff bounds, both estimation values are concentrated around the expectation, then with only small probability, these two values differ significantly

$$\Pr_{\substack{(r_1, \pi_1, \ldots, r_i, \pi_i) \leftarrow \mathsf{Sim}_{1\ldots 2i}(x) \\ \mathsf{est}, \hat{\mathsf{est}} \leftarrow \mathsf{Est}_i(x; r_1, \pi_1, \ldots, r_i, \pi_i)}} \left[\left|\mathsf{est} - \hat{\mathsf{est}}\right| > \tau\right] < 4e^{-\tau^2 q/2}, \tag{28}$$

where the distribution $(\mathsf{est}, \hat{\mathsf{est}})$ sampled from is the same as $\mathcal{P}_{i,x}(\mathcal{U}, f_{i,x}(\mathcal{U}))$. When $q = n \cdot p(n)^2$ and $\tau = 1/p(n)$, the above value is at most $4e^{-n/2}$, negligible in n.

Notice that the following two distributions are equivalent.

$$\mathcal{P}_{i,x}(\mathcal{A}_i(f_{i,x}(\mathcal{U})), f_{i,x}(\mathcal{U})) = \left\{ \begin{array}{c} (r_1, \pi_1, \ldots, r_i, \pi_i) \leftarrow \mathsf{Sim}_{1\ldots 2i}(x) \\ \mathsf{est} \leftarrow \mathsf{Est}_i(x; r_i, \pi_1, \ldots, r_i, \pi_i) \\ \hat{\mathsf{est}} \leftarrow \hat{\mathcal{B}}_i(x; r_1, \pi_1, \ldots, r_i, \mathsf{est}) \\ \text{output } (\mathsf{est}, \hat{\mathsf{est}}) \end{array} \right\}$$

By our assumption (21), the performance of $\mathcal{A}_i$ ensures that

$$\Delta_s(\mathcal{U}, f_{i,x}(\mathcal{U}); \mathcal{A}_i(f_{i,x}(\mathcal{U})), f_{i,x}(\mathcal{U})) < \frac{1}{p(n)},$$

which implies that

$$\begin{aligned} & \mathop{\mathbb{E}}_{\substack{(r_1,\ldots,\pi_i)\leftarrow\mathsf{Sim}_{1\ldots 2i}(x) \\ \mathsf{est}\leftarrow\mathsf{Est}_i(x;r_1,\pi_1,\ldots,r_i,\pi_i) \\ \hat{\mathsf{est}}\leftarrow\hat{\mathcal{B}}_i(x;r_1,\pi_1,\ldots,r_i,\mathsf{est})}} \left[E(\mathsf{est}, \hat{\mathsf{est}})\right] \\ > & \mathop{\mathbb{E}}_{\substack{(r_1,\pi_1,\ldots,r_i,\pi_i)\leftarrow\mathsf{Sim}_{1\ldots 2i}(x) \\ \mathsf{est},\hat{\mathsf{est}}\leftarrow\mathsf{Est}_i(x;r_1,\pi_1,\ldots,r_i,\pi_i)}} \left[E(\mathsf{est}, \hat{\mathsf{est}})\right] - \frac{1}{p(n)} \\ > & 1 - \frac{1}{p(n)} - \mathsf{negl}(n), \end{aligned}$$

where the first inequality is obtained by the data processing inequality and (21) and the second follows (28). Since $\Delta_c(\mathsf{Sim}(x); \mathsf{View}\langle \mathsf{P}, \mathsf{V}\rangle(x)) \leq \epsilon_z(n)$, when $(r_1, \ldots, \pi_i)$ is sampled by the protocol

$$\mathop{\mathbb{E}}_{\substack{(r_1,\pi_1,\ldots,r_i,\pi_i)\leftarrow\mathsf{View}_{2i}\langle\mathsf{P},\mathsf{V}\rangle(x) \\ \mathsf{est}\leftarrow\mathsf{Est}_i(x;r_1,\pi_1,\ldots,r_i,\pi_i) \\ \hat{\mathsf{est}}\leftarrow\hat{\mathcal{B}}_i(x;r_1,\pi_1,\ldots,r_i,\mathsf{est})}} \left[E(\mathsf{est}, \hat{\mathsf{est}})\right] > 1 - \epsilon_z(n) - \frac{1}{p(n)} - \mathsf{negl}(n).$$

Then, by (27), we obtain

$$\begin{aligned} & \mathop{\mathbb{E}}_{(r_1,\pi_1,\ldots,r_i)\leftarrow\mathsf{View}_{2i-1}\langle\mathsf{P},\mathsf{V}\rangle(x)} \left[\mathcal{B}_{i,1}(x; r_1, \pi_1, \ldots, r_i)\right] \\ & \geq \mathop{\mathbb{E}}_{\substack{(r_1,\pi_1,\ldots,r_i,\pi_i)\leftarrow\mathsf{View}_{2i}\langle\mathsf{P},\mathsf{V}\rangle(x) \\ \mathsf{est}\leftarrow\mathsf{Est}_i(x;r_1,\pi_1,\ldots,r_i,\pi_i) \\ \hat{\mathsf{est}}\leftarrow\hat{\mathcal{B}}_i(x;r_1,\pi_1,\ldots,r_i,\mathsf{est})}} \left[\mathsf{est} \cdot E(\mathsf{est}, \hat{\mathsf{est}})\right] \\ & \geq \mathop{\mathbb{E}}_{\substack{(r_1,\ldots,r_i,\pi_i)\leftarrow\mathsf{View}_{2i}\langle\mathsf{P},\mathsf{V}\rangle(x) \\ \mathsf{est}\leftarrow\mathsf{Est}_i(x;r_i,\ldots,r_i,\pi_i)}} \left[\mathsf{est}\right] - \epsilon_z(n) - \frac{1}{p(n)} - \mathsf{negl}(n), \end{aligned} \tag{29}$$

where the last inequality is obtained because est is valued in $[0, 1]$, $\mathbb{E}[\mathsf{est} \cdot E(\mathsf{est}, \hat{\mathsf{est}})] \geq \mathbb{E}[\mathsf{est}] - \Pr[E(\mathsf{est}, \hat{\mathsf{est}}) = 0]$. By the definition of Est_i, for any input $(x; r_1, \pi_1, \ldots, r_i, \pi_i)$,

$$\mathop{\mathbb{E}}_{\mathsf{est}\leftarrow\mathsf{Est}_i(x;r_1,\pi_1,\ldots,r_i,\pi_i)} \left[\mathsf{est}\right] = \mathop{\mathbb{E}}_{r_{i+1}\leftarrow\mathcal{U}_{m_{i+1}}} \left[\mathcal{B}_{i+1,1}(x; r_1, \pi_1, \ldots, r_i, \pi_i, r_{i+1})\right].$$

Therefore, we relate the expected values of $\mathcal{B}_i$ and $\mathcal{B}_{i+1}$ by the following

$$(29) \quad \geq \mathop{\mathbb{E}}_{(r_1,\ldots,\pi_i,r_{i+1})\leftarrow \mathsf{View}_{2i+1}\langle \mathsf{P},\mathsf{V}\rangle(x)} [\mathcal{B}_{i+1,1}(x; r_1, \ldots, \pi_i, r_{i+1})] - \epsilon_z(n) - \frac{1}{p(n)} - \mathsf{negl}(n).$$

Combing (26) and above, we have

$$\mathop{\mathbb{E}}_{(r_1,\ldots,r_i)\leftarrow \mathsf{View}_{2i+1}\langle \mathsf{P},\mathsf{V}\rangle(x)} [\mathcal{B}_{i,1}(x; r_1, \ldots, r_i)]$$
$$\geq 1 - \epsilon_c(n) - (k - i + 1)\left(\epsilon_z(n) + \frac{1}{p(n)}\right) - \mathsf{negl}(n).$$

Therefore, let $i = 1$

$$\mathcal{B}(x) = \mathop{\mathbb{E}}_{r_1\leftarrow \mathcal{U}_{m_1}} [\mathcal{B}_{1,1}(x; r_1)] \geq 1 - \epsilon_c(n) - k \cdot \epsilon_z(n) - \frac{k}{p(n)} - \mathsf{negl}(n),$$

which concludes the proof. □

4 One-Way Functions

We complete our transformation by extending the result of [HN24] that shows that assuming there are zero-knowledge arguments for NP, auxiliary-input one-way functions imply standard one-way functions. We follow the approach taken by [CHK25] in the context of weak zero-knowledge arguments, showing that the auxiliary-input OWFs obtained in the previous sections also imply standard one-way functions. The resulting theorems are as follows.

Theorem 4.1. *If* $\mathsf{NP} \not\subseteq \mathsf{ioP/poly}$ *and, for some* $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0, 1]$ *such that* $\epsilon_c + \epsilon_s + \epsilon_z <_n 1$*, every language in* NP *has an* $(\epsilon_c, \epsilon_s, \epsilon_z)$*-NIZK proof, then one-way functions exist. The same holds for NIZK arguments.*

Theorem 4.2. *If* $\mathsf{NP} \not\subseteq \mathsf{ioP/poly}$ *and, for some* $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0, 1]$ *and* $t : \mathbb{N} \to \mathbb{N}$ *such that* $\epsilon_c + \epsilon_s + (t-1)\cdot\epsilon_z <_n 1$*, every language in* NP *has a* t*-message public-coin* $(\epsilon_c, \epsilon_s, \epsilon_z)$*-ZK proof, then one-way functions exist. The same holds for ZK arguments.*

Theorem 4.3. *If* $\mathsf{NP} \not\subseteq \mathsf{P/poly}$ *and, for some* $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0, 1]$ *and constant* $k \in \mathbb{N}$ *such that* $\epsilon_c + \epsilon_s + k \cdot \epsilon_z <_n 1$*, every language in* NP *has a* k*-round public-coin* $(\epsilon_c, \epsilon_s, \epsilon_z)$*-ZK proof, then infinitely often one-way functions exist. The same holds for ZK arguments.*

The proofs of these theorems follow from combining the respective lemmas in Section 3 with lemmas stated later in this section. Theorems 4.1 and 4.2 are proven in Section 4.2.1, and Theorem 4.3 in Section 4.2.2.

As observed in [CHK25, Section 5] (which in turn draws on the approach in [LMP24]), the overall implication from weak ZK to OWFs can be broken up into three parts. The first

part consists of showing that having such a weak ZK protocol for worst-case hard language implies an auxiliary-input OWF, and this we have shown in Section 3 (for various flavors of ZK protocols). The two remaining steps are the following:

1. Show that auxiliary-input OWFs imply that there exist what are called in [CHK25] as *one-sided average-case hard* languages.

2. Show that (the corresponding flavor of) weak ZK protocols for such one-sided average-case hard languages imply standard OWFs.

We outline these transformations below for each flavor of weak ZK that we have considered so far. Some of the proofs below are along the lines of those of similar lemmas from prior work, especially from [CHK25].

4.1 One-Sided Average-Case Hardness from ai-OWF

This lemma appears in [CHK25, Lemma 4]. This result is somewhat independent, and does not have anything to do with weak zero-knowledge. We can thus use it as is.

Definition 4.1 (One-Sided Average-Case BPP)**.** The class $\mathsf{1AvgBPP/poly}$ consists of average-case problems $(\mathcal{L}, \mathcal{D})$ for which there is a non-uniform probabilistic polynomial-time algorithm $\mathcal{A}$ and functions $a, b : \mathbb{N} \to [0, 1]$ with $a >_n b$, such that for all sufficiently large n

1. For every $x \in \mathcal{L}_n$, $\Pr[\mathcal{A}(x) = 1] \geq a(n)$.

2. $\Pr_{x \leftarrow \mathcal{D}_n}[\mathcal{A}(x) = 1 | x \notin \mathcal{L}_n] \leq b(n)$.

Lemma 4.4 ([CHK25, Lemma 4])**.** *Suppose that there exists an auxiliary-input one-way function. Then there exists an $\mathcal{L} \in \mathsf{NP}$ such that $(\mathcal{L}, \mathcal{U}) \notin \mathsf{io1AvgBPP/poly}$ and $\Pr_{x \leftarrow \mathcal{U}_n}[x \notin \mathcal{L}_n] \geq 1/2$ for all $n \in \mathbb{N}$.*

The following variant also follows from the proof of the above lemma.

Lemma 4.5. *Suppose that there exists an infinitely-often ai-OWF, then there exists a language $\mathcal{L} \in \mathsf{NP}$ such that $(\mathcal{L}, \mathcal{U}) \notin \mathsf{1AvgBPP/poly}$ and $\Pr_{x \leftarrow \mathcal{U}_n}[x \notin \mathcal{L}_n] \geq 1/2$ for all $n \in \mathbb{N}$.*

4.2 OWF from One-Sided Average-Case Hardness

4.2.1 NIZK and Public-Coin ZK

Combining Lemma 3.1 and Lemma 3.3 with the amplification for converting weak and distributional one-way functions into standard form [Yao82, IL89], we obtain the following lemma.

Lemma 4.6. *For some $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$ and $t : \mathbb{N} \to \mathbb{N}$, consider a language $\mathcal{L} \in \mathsf{NP}$ with an $(\epsilon_c, \epsilon_s, \epsilon_z)$-NIZK protocol (in which case $t = 2$) or a t-message $(\epsilon_c, \epsilon_s, \epsilon_z)$-public-coin ZK protocol. For any polynomials p_1, p_2, there is a reduction R, which is a polynomial-time oracle-aided algorithm, and a polynomial-time computable function family $\mathcal{F} = \{f_x\}_{x \in \{0,1\}^*}$ such that for any probabilistic polynomial-time algorithm $\mathcal{A}$ and all large enough n,*

1. *When $x \in \mathcal{L}_n$, if $\mathcal{A}$ inverts f_x with probability at least $1/p_1(n)$, then*

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \geq 1 - \epsilon_c(n) - (t(n) - 1) \cdot \epsilon_z(n) - 1/p_2(n).$$

2. *When $x \in \{0,1\}^n \setminus \mathcal{L}_n$,*

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \leq \epsilon_s(n).$$

Note that our constructions of R and $\mathcal{F}$ in the proofs of Lemma 3.1 and Lemma 3.3 do not satisfy the above conditions. However, reductions and functions satisfying the above lemma can be obtained from the previous construction by applying the techniques in [Yao82, IL89]. Next, we use the following analogue of [CHK25, Lemma 5].

Lemma 4.7. *For $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$ and $t : \mathbb{N} \to \mathbb{N}$, consider any language $\mathcal{L}$ such that $(\mathcal{L}, \mathcal{U}) \notin \mathsf{io1AvgBPP/poly}$, and $\Pr_{x \leftarrow \mathcal{U}_n}[x \notin \mathcal{L}_n] \geq 1/2$ for all $n \in \mathbb{N}$. If there is an $(\epsilon_c, \epsilon_s, \epsilon_z)$-NIZK protocol (in which case $t = 2$) or a t-message $(\epsilon_c, \epsilon_s, \epsilon_z)$-public-coin ZK protocol for $\mathcal{L}$ with $\epsilon_c + \epsilon_s + (t-1) \cdot \epsilon_z <_n 1$, then one-way functions exist.*

Proof of Lemma 4.7. Since $\epsilon_c + \epsilon_s + (t-1) \cdot \epsilon_z <_n 1$, there exists a polynomial q such that

$$\epsilon_c(n) + \epsilon_s(n) + (t(n) - 1) \cdot \epsilon_z(n) + \frac{1}{q(n)} < 1 \tag{30}$$

holds for all sufficiently large $n \in \mathbb{N}$. Then, let $p = 4q$. Let $\mathcal{F} = \{f_x\}_{x \in \{0,1\}^*}$ be the construction in Lemma 4.6, and let $f(x, r) = (x, f_x(r))$.

Suppose that there is a (non-uniform) PPT algorithm $\mathcal{A}$ that inverts f for infinitely many $n \in \mathbb{N}$ with probability $(1 - 1/p(n))$. In particular, we assume that

$$\Pr_{(x,y) \leftarrow f(\mathcal{U})}[f(\mathcal{A}(x, y)) = (x, y)] \geq 1 - \frac{1}{p(n)}, \tag{31}$$

where we denote by $f(\mathcal{U})$ the distribution of the output of f when the input is sampled uniformly. Denote by $\mathcal{A}_x(y)$ the algorithm: on input y, it runs $(\hat{x}, \hat{r}) \leftarrow \mathcal{A}(x, y)$; if $\hat{x}$ matches x, it outputs $\hat{r}$; else outputs $\perp$. Then, we construct an algorithm $\mathcal{C}$ as follows.

Algorithm $\mathcal{C}(x)$:

1. Sample $y \leftarrow f_x(\mathcal{U})$

2. Let $\hat{r} \leftarrow \mathcal{A}_x(y)$
3. If $f_x(\hat{r}) = y$, output $R^{\mathcal{A}}(x)$
4. Else output 1

Consider $x \in \mathcal{L}_n$, if $\mathcal{A}_x$ correctly inverts f_x on random input r with probability at least $1/p(n)$, then by Lemma 4.6, we have that

$$\Pr\left[\mathcal{C}(x) = 1\right] \geq 1 - \Pr\left[R^{\mathcal{A}}(x) \neq 1\right] \geq 1 - \epsilon_c(n) - (t(n) - 1) \cdot \epsilon_z(n) - \frac{1}{p(n)}.$$

If the probability that $\mathcal{A}_x$ finds the correct pre-image of $f_x(r)$ is upper-bounded by $1/p(n)$,

$$\Pr\left[\mathcal{C}(x) = 1\right] \geq 1 - \Pr_{y \leftarrow f_x(\mathcal{U})}\left[f_x(\mathcal{A}_x(y)) = y\right] \geq 1 - \frac{1}{p(n)}.$$

Then, the probability that $\mathcal{C}(x)$ outputs 1 is at least

$$\Pr\left[\mathcal{C}(x) = 1\right] \geq 1 - \epsilon_c(n) - (t(n) - 1) \cdot \epsilon_z(n) - \frac{1}{p(n)}.$$

Consider the case $x \leftarrow \{0,1\}^n \setminus \mathcal{L}_n$,

$$\begin{aligned}
&\Pr_{\substack{x \leftarrow \mathcal{U}_n \\ y \leftarrow f_x(\mathcal{U})}}\left[f(\mathcal{A}(x,y)) \neq (x,y) | x \notin \mathcal{L}_n\right] \cdot \Pr_{x \leftarrow \mathcal{U}_n}\left[x \notin \mathcal{L}_n\right] \\
&\leq \Pr_{\substack{x \leftarrow \mathcal{U}_n \\ y \leftarrow f_x(\mathcal{U})}}\left[f(\mathcal{A}(x,y)) \neq (x,y)\right] \\
&\leq \frac{1}{p(n)},
\end{aligned}$$

where the last inequality holds because (31), and we make an assumption on the language

$$\Pr_{x \leftarrow \mathcal{U}_n}\left[x \notin \mathcal{L}_n\right] \geq \frac{1}{2}$$

then we obtain that

$$\Pr_{\substack{x \leftarrow \mathcal{U}_n \\ y \leftarrow f_x(\mathcal{U})}}\left[f_x(\mathcal{A}_x(y)) \neq y | x \notin \mathcal{L}_n\right] = \Pr_{\substack{x \leftarrow \mathcal{U}_n \\ y \leftarrow f_x(\mathcal{U})}}\left[f(\mathcal{A}(x,y)) \neq (x,y) | x \notin \mathcal{L}_n\right] \leq \frac{2}{p(n)}.$$

Therefore, by a union bound

$$\begin{aligned}
&\Pr_{x\leftarrow\mathcal{U}_n}[\mathcal{C}(x)=1|x\notin\mathcal{L}_n]\\
&\leq \Pr_{\substack{x\leftarrow\mathcal{U}_n\\ y\leftarrow f_x(\mathcal{U})}}[f_x(\mathcal{A}_x(y))\neq y|x\notin\mathcal{L}_n]+\Pr_{x\leftarrow\mathcal{U}_n}\left[R^{\mathcal{A}}(x)=1|x\notin\mathcal{L}_n\right]\\
&\leq \frac{2}{p(n)}+\epsilon_s(n).
\end{aligned}$$

$\mathcal{C}$ is an algorithm that decides the language $\mathcal{L}$ in the one-sided average case, since by (30)

$$\frac{2}{p(n)}+\epsilon_s(n)<_n 1-\epsilon_c(n)-(t(n)-1)\cdot\epsilon_z(n)-\frac{1}{p(n)}$$

which contradicts $\mathcal{L}\notin\mathsf{io1AvgBPP/poly}$. Therefore, f is a weak OWF and from [Yao82], it implies the existence of one-way functions. □

Proving Theorems 4.1 and 4.2 Combining Corollary 3.2, Lemma 4.4 and Lemma 4.7 yields Theorem 4.1. Analogously, for any round public-coin ZK protocol, by putting Corollary 3.4, Lemma 4.4 and Lemma 4.7 together, we obtain Theorem 4.2.

4.2.2 Constant-Round Public-Coin ZK

Lemma 3.5 combined with [IL89] yields the following lemma.

Lemma 4.8. *For some $\epsilon_c,\epsilon_s,\epsilon_z:\mathbb{N}\to[0,1]$ and any constant $k\in\mathbb{N}$, suppose a language $\mathcal{L}\in\mathsf{NP}$ has a k-round public-coin $(\epsilon_c,\epsilon_s,\epsilon_z)$-ZK proof or argument. For any polynomials p_1,p_2, there exists a polynomial-time oracle-aided algorithm R and families of oracle-aided functions $\mathcal{F}_1,\ldots,\mathcal{F}_k$ satisfying the following.*

For $i\in[k]$, the family $\mathcal{F}_i=\{f_{i,x}\}_{x\in\{0,1\}^*}$ consists of functions that require access to $(k-i)$ oracles, and are polynomial-time computable given these oracles. For any sequence of probabilistic polynomial-time algorithms $\mathcal{A}_1,\ldots,\mathcal{A}_k$ and all large enough n, we have the following.

1. *For every $x\in\mathcal{L}_n$, if for all $i\in[k]$, $\mathcal{A}_i$ inverts $f_{i,x}$ with probability at least $1/p_1(n)$, then*
$$\Pr\left[R^{\mathcal{A}_1\cdots\mathcal{A}_k}(x)=1\right]\geq 1-\epsilon_c(n)-k\cdot\epsilon_z(n)-1/p_2(n)$$
2. *For every $x\in\{0,1\}^n\setminus\mathcal{L}_n$,*
$$\Pr\left[R^{\mathcal{A}_1\cdots\mathcal{A}_k}(x)=1\right]\leq\epsilon_s(n).$$

Lemma 4.9. *For $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$ and a constant $k \in \mathbb{N}$, consider any language $\mathcal{L}$ such that $(\mathcal{L}, \mathcal{U}) \notin \mathsf{1AvgBPP/poly}$, and $\Pr_{x \leftarrow \mathcal{U}_n}[x \notin \mathcal{L}_n] \geq 1/2$ for all $n \in \mathbb{N}$. If there is a k-round $(\epsilon_c, \epsilon_s, \epsilon_z)$-public-coin ZK protocol for $\mathcal{L}$ with $\epsilon_c + \epsilon_s + k \cdot \epsilon_z <_n 1$, then infinitely-often one-way functions exist.*

Proof of Lemma 4.9. Since $\epsilon_c + \epsilon_s + k \cdot \epsilon_z <_n 1$, there exists a polynomial q such that

$$\epsilon_c(n) + \epsilon_s(n) + k \cdot \epsilon_z(n) + \frac{1}{q(n)} < 1 \tag{32}$$

holds for all sufficiently large $n \in \mathbb{N}$. Then, let $p = 2(k+1)q$. Let $\mathcal{F}_1, \ldots, \mathcal{F}_k$ be the constructions from Lemma 4.6. Suppose that there are (non-uniform) PPT algorithms $\mathcal{A}_1, \ldots, \mathcal{A}_k$ that invert the corresponding function for all sufficiently large $n \in \mathbb{N}$. For simplicity, we omit the superscript that describes the oracle access in the function constructions, denote $\mathcal{F}_i = \{f_{i,x}\}$ for $i \in [k]$ and let $f_i(x, r) = (x, f_{i,x}(r))$. The efficiency conditions for the algorithms $\mathcal{A}_i$'s implies that all functions f_i's are polynomial-time computable. In particular, we assume that for each $i \in [k]$,

$$\Pr_{(x,y) \leftarrow f_i(\mathcal{U})}[f_i(\mathcal{A}_i(x,y)) = (x,y)] \geq 1 - \frac{1}{p(n)}.$$

Denote by $\mathcal{A}_{i,x}(y)$ the algorithm: on input y, it runs $(\hat{x}, \hat{r}) \leftarrow \mathcal{A}_i(x,y)$; if $\hat{x}$ matches x, it outputs $\hat{r}$; else outputs $\perp$. In the following, we construct an algorithm $\mathcal{C}$ towards solving the language $\mathcal{L}$ in the one-sided average case.

Algorithm $\mathcal{C}(x)$:

1. For $i \in [k]$
   - Sample $y \leftarrow f_{i,x}(\mathcal{U})$
   - Set $\hat{r}_i \leftarrow \mathcal{A}_{i,x}(y)$
   - If $f_{i,x}(\hat{r}_i) \neq y$, output 1
2. Output $R^{\mathcal{A}_1 \cdots \mathcal{A}_k}(x)$

Consider $x \in \mathcal{L}_n$, if for all $i \in [k]$, $\mathcal{A}_{i,x}$ correctly inverts $f_{i,x}$ on random input r with probability at least $1/p(n)$, then by Lemma 4.8, we have that

$$\Pr[\mathcal{C}(x) = 1] \geq \Pr\left[R^{\mathcal{A}_1 \cdots \mathcal{A}_k}(x) = 1\right] \geq 1 - \epsilon_c(n) - k \cdot \epsilon_z(n) - \frac{1}{p(n)}.$$

If there is an $i \in [k]$ such that the probability that $\mathcal{A}_{i,x}$ finds the correct pre-image of $f_{i,x}(r)$ is upper-bounded by $1/p(n)$,

$$\Pr[\mathcal{C}(x) = 1] \geq 1 - \Pr_{y \leftarrow f_{i,x}(\mathcal{U})}[f_{i,x}(\mathcal{A}_{i,x}(y)) = y] \geq 1 - \frac{1}{p(n)}.$$

Therefore, the probability that $\mathcal{C}(x)$ outputs 1 is at least

$$\Pr\left[\mathcal{C}(x)=1\right] \geq 1-\epsilon_c(n)-k\cdot\epsilon_z(n)-\frac{1}{p(n)}.$$

When $x \in \{0,1\}^n \setminus \mathcal{L}_n$, by our assumption on the algorithm, for each $i \in [k]$

$$\begin{aligned}
&\Pr_{(x,y)\leftarrow f_i(\mathcal{U})}\left[f_i(\mathcal{A}(x,y)) \neq (x,y) | x \notin \mathcal{L}_n\right] \cdot \Pr_{x\leftarrow\mathcal{U}_n}\left[x \notin \mathcal{L}_n\right] \\
&\leq \Pr_{(x,y)\leftarrow f_i(\mathcal{U})}\left[f_i(\mathcal{A}(x,y)) \neq (x,y)\right] \\
&\leq \frac{1}{p(n)}
\end{aligned}$$

and the language satisfying

$$\Pr_{x\leftarrow\mathcal{U}_n}\left[x \notin \mathcal{L}_n\right] \geq \frac{1}{2}$$

we obtain

$$\Pr_{\substack{x\leftarrow\mathcal{U}_n \\ y\leftarrow f_{i,x}(\mathcal{U})}}\left[f_{i,x}(\mathcal{A}_{i,x}(y)) \neq y | x \notin \mathcal{L}_n\right] \leq \frac{2}{p(n)}.$$

Therefore, by a union bound

$$\begin{aligned}
&\Pr_{x\leftarrow\mathcal{U}_n}\left[\mathcal{C}(x)=1 | x \notin \mathcal{L}_n\right] \\
&\leq \sum_{i\in[k]} \Pr_{\substack{x\leftarrow\mathcal{U}_n \\ y\leftarrow f_{i,x}(\mathcal{U})}}\left[f_{i,x}(\mathcal{A}_{i,x}(y)) \neq y | x \notin \mathcal{L}_n\right] + \Pr_{x\leftarrow\mathcal{U}_n}\left[R^{\mathcal{A}_1\cdots\mathcal{A}_k}(x)=1 | x \notin \mathcal{L}_n\right] \\
&\leq \frac{2k}{p(n)} + \epsilon_s(n).
\end{aligned}$$

$\mathcal{C}$ is an algorithm that decides the language $\mathcal{L}$ in the one-sided average case as by (32),

$$\frac{2k}{p(n)} + \epsilon_s(n) <_n 1-\epsilon_c(n)-k\cdot\epsilon_z(n)-\frac{1}{p(n)}$$

which contradicts $\mathcal{L} \notin \mathsf{io1AvgBPP/poly}$. Thus, at least one of the recursively defined f_i's is a weak OWF and by [Yao82], one-way functions exist. □

Proving Theorem 4.3 Theorem 4.3 follows directly from Corollary 3.6, Lemma 4.5, and Lemma 4.9.

Acknowledgements

This work was supported by the National Research Foundation, Singapore, under its NRF Fellowship programme, award no. NRF-NRFF14-2022-0010.

Commercial AI tools (ChatGPT, Claude) were used as typing assistants for grammar and basic editing, and for mild assistance with LaTeX formatting.

References

[AK25] Benny Applebaum and Eliran Kachlon. NIZK amplification via leakage-resilient secure computation. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology - CRYPTO 2025 - 45th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2025, Proceedings, Part VII*, volume 16006 of *Lecture Notes in Computer Science*, pages 462–479. Springer, 2025.

[BG24] Nir Bitansky and Nathan Geier. Amplification of non-interactive zero knowledge, revisited. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part IX*, volume 14928 of *Lecture Notes in Computer Science*, pages 361–390. Springer, 2024.

[BT24] Jan Buzek and Stefano Tessaro. Collision resistance from multi-collision resistance for all constant parameters. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part V*, volume 14924 of *Lecture Notes in Computer Science*, pages 429–458. Springer, 2024.

[CHK25] Suvradip Chakraborty, James Hulett, and Dakshita Khurana. On weak nizks, one-way functions and amplification. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology - CRYPTO 2025 - 45th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2025, Proceedings, Part VII*, volume 16006 of *Lecture Notes in Computer Science*, pages 580–610. Springer, 2025.

[GJS19] Vipul Goyal, Aayush Jain, and Amit Sahai. Simultaneous amplification: The case of non-interactive zero-knowledge. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II*, volume 11693 of *Lecture Notes in Computer Science*, pages 608–637. Springer, 2019.

[HN24] Shuichi Hirahara and Mikito Nanashima. One-way functions and zero knowledge. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 1731–1738. ACM, 2024.

[IL89] Russell Impagliazzo and Michael Luby. One-way functions are essential for complexity based cryptography (extended abstract). In *30th Annual Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, USA, 30 October - 1 November 1989*, pages 230–235. IEEE Computer Society, 1989.

[LMP24] Yanyi Liu, Noam Mazor, and Rafael Pass. A note on zero-knowledge for NP and one-way functions. *IACR Cryptol. ePrint Arch.*, page 800, 2024.

[MV24] Changrui Mu and Prashant Nalini Vasudevan. Instance-hiding interactive proofs - (extended abstract). In Elette Boyle and Mohammad Mahmoody, editors, *Theory of Cryptography - 22nd International Conference, TCC 2024, Milan, Italy, December 2-6, 2024, Proceedings, Part I*, volume 15364 of *Lecture Notes in Computer Science*, pages 3–34. Springer, 2024.

[Ost91] Rafail Ostrovsky. One-way functions, hard on average problems, and statistical zero-knowledge proofs. In *Proceedings of the Sixth Annual Structure in Complexity Theory Conference, Chicago, Illinois, USA, June 30 - July 3, 1991*, pages 133–138. IEEE Computer Society, 1991.

[OW93] Rafail Ostrovsky and Avi Wigderson. One-way fuctions are essential for non-trivial zero-knowledge. In *Second Israel Symposium on Theory of Computing Systems, ISTCS 1993, Natanya, Israel, June 7-9, 1993, Proceedings*, pages 3–17. IEEE Computer Society, 1993.

[Yao82] Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 80–91. IEEE Computer Society, 1982.

A On Randomized Verification

Lemma A.1. *For $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$, suppose a language $\mathcal{L} \in \mathsf{NP}$ has an $(\epsilon_c, \epsilon_s, \epsilon_z)$-NIZK proof or argument (with randomized verification). Then there exists a reduction R, which is a polynomial-time oracle-aided algorithm, and a function family $\mathcal{F} = \{f_x\}_{x \in \{0,1\}^*}$, such that for any probabilistic polynomial-time algorithm $\mathcal{A}$, any polynomial p, and all large enough $n \in \mathbb{N}$*

1. *For every $x \in \mathcal{L}_n$, if $\mathcal{A}$ distributionally inverts f_x with deviation at most $1/p(n)$, then*

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \geq 1 - \epsilon_c(n) - \epsilon_z(n) - \frac{3}{p(n)} - \mathsf{negl}(n).$$

2. *For every $x \in \{0,1\}^n \setminus \mathcal{L}_n$,*

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \leq \epsilon_s(n).$$

Proof Sketch of Lemma A.1. Let Sim be the simulator that satisfies the zero-knowledge requirement. Let $q = n \cdot p(n)^2$. Let $\mathcal{F} = \{f_x\}_{x \in \{0,1\}^*}$ be a function family where f_x is defined as

$\underline{f_x(\rho; \sigma_1, \ldots, \sigma_q)}$:

1. $(r, \pi) \leftarrow \mathsf{Sim}(x; \rho)$
2. $\mathsf{est} \leftarrow \frac{1}{q} \sum_{i \in [q]} \mathsf{V}(x; r, \pi, \sigma_i)$
3. Output (r, est)

Note that ρ is the randomness of Sim and $\sigma_1, \ldots, \sigma_q$ serve as the randomness of V. Assume that $\mathcal{A}$ is a PPT algorithm for inverting f_x's. Let $\tau = 1/p(n)$. Construct an efficient algorithm $\mathcal{B}$ with oracle access to $\mathcal{A}$ as follows:

$\underline{\mathcal{B}^{\mathcal{A}}(x; r)}$:

1. For $a = q, q-1, \ldots, 1$ in decreasing order
   - $(\rho; \sigma_1, \ldots, \sigma_q) \leftarrow \mathcal{A}(r, a/q)$
   - If $f_x(\rho; \sigma_1, \ldots, \sigma_q) = (r, a/q)$
     - $\pi \leftarrow \mathsf{Sim}_2(x; \rho)$
     - $\hat{\sigma}_1, \ldots, \hat{\sigma}_q \leftarrow \mathcal{U}$
     - $\hat{\mathsf{est}} \leftarrow \frac{1}{q} \sum_{i \in [q]} \mathsf{V}(x; r, \pi, \hat{\sigma}_i)$
     - If $\left|\hat{\mathsf{est}} - a/q\right| < t$, output $(a/q, \pi)$
2. Output $(0, \perp)$ otherwise

Let $\mathcal{B}_1$ and $\mathcal{B}_2$ be the algorithm that outputs the first and the second part of $\mathcal{B}$'s output respectively and let $\tilde{\mathsf{P}}^{\mathcal{A}} = \mathcal{B}_2^{\mathcal{A}}$ be an efficient prover's strategy on input $(x; r)$. Then, the reduction $R^{\mathcal{A}}$ runs the protocol $\langle \tilde{\mathsf{P}}^{\mathcal{A}}, \mathsf{V} \rangle$ itself and accepts if the protocol accepts. For convenience, denote $\tilde{\mathsf{P}} = \tilde{\mathsf{P}}^{\mathcal{A}}$.

We prove that R and $\mathcal{F}$ satisfy condition 1. For $x \in \mathcal{L}_n$, suppose that $\mathcal{A}$ distributionally inverts f_x with deviation at most $1/p(n)$, that is

$$\Delta_s(\mathcal{A}(f_x(\mathcal{U})), f_x(\mathcal{U}); \mathcal{U}, f_x(\mathcal{U})) \leq 1/p(n).$$

Suppose that (est^*, π^*) is the value on which $\tilde{\mathsf{P}}(x; r)$ returns, by a Chernoff bound, with an overwhelming probability $(1 - \mathsf{negl}(n))$ we have

$$|\mathsf{est}^* - \mathbb{E}\left[\mathsf{V}(x; r, \pi^*)\right]| < 2\tau,$$

which implies

$$\left| \mathop{\mathbb{E}}_{\substack{r \leftarrow \mathsf{Gen}(1^n), \pi \leftarrow \tilde{\mathsf{P}}(x;r) \\ \sigma \leftarrow \mathcal{U}}} \left[\mathsf{V}(x; r, \pi, \sigma)\right] - \mathop{\mathbb{E}}_{r \leftarrow \mathsf{Gen}(1^n)} \left[\mathcal{B}_1(x; r)\right] \right| \leq \frac{2}{p(n)} + \mathsf{negl}(n).$$

Analogous to the argument shown before, by setting $\mathcal{B}_1$ and $\mathcal{B}_{2,1}$ in the proof of Claim 3.6 to be $\mathcal{B}$ and V, we have that

$$\begin{aligned} \mathop{\mathbb{E}}_{r \leftarrow \mathsf{Gen}(1^n)} \left[\mathcal{B}_1(x; r)\right] &\geq \mathop{\mathbb{E}}_{\substack{r \leftarrow \mathsf{Gen}(1^n), \pi \leftarrow \mathsf{P}(x,w;r) \\ \sigma \leftarrow \mathcal{U}}} \left[\mathsf{V}(x; r, \pi, \sigma)\right] - \epsilon_z(n) - \frac{1}{p(n)} - \mathsf{negl}(n) \\ &\geq 1 - \epsilon_c(n) - \epsilon_z(n) - \frac{1}{p(n)} - \mathsf{negl}(n). \end{aligned}$$

Therefore,

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] = \mathop{\mathbb{E}}_{\substack{r \leftarrow \mathsf{Gen}(1^n) \\ \pi \leftarrow \tilde{\mathsf{P}}(x;r) \\ \sigma \leftarrow \mathcal{U}}} \left[\mathsf{V}(x; r, \pi, \sigma)\right] \geq 1 - \epsilon_c(n) - \epsilon_z(n) - \frac{3}{p(n)} - \mathsf{negl}(n).$$

Condition 2 holds, since when $x \notin \mathcal{L}$, for any polynomial-time algorithm $\mathcal{A}$, the soundness guarantees

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] = \mathop{\mathbb{E}}_{\substack{r \leftarrow \mathsf{Gen}(1^n) \\ \pi \leftarrow \tilde{\mathsf{P}}(x;r) \\ \sigma \leftarrow \mathcal{U}}} \left[\mathsf{V}(x; r, \pi, \sigma)\right] \leq \epsilon_s(n).$$

□

ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il