SNARGs for NP from Unprovability of Mathematical Theorems

(Or: How to use the simplicity of cryptographic reasoning)

Yao-Ching Hsieh* Abhishek Jain† Jiatu Li‡ Surya Mathialagan§

Abstract

Modern cryptography relies on the intractability of computational problems. We present an approach to build cryptography from a new source of hardness: *proving mathematical theorems*.

Our main result is a construction of succinct non-interactive arguments (SNARGs) for NP under a new, but natural assumption on the hardness of proving lower bounds in the area of proof complexity. Specifically, our assumption states that it is impossible to prove, within a weak bounded arithmetic theory, the correctness of certifying hard tautologies against Extended Frege. This assumption is inspired by an informal mathematical challenge proposed by Razborov [Ann. Math. '15], and can be viewed as a generalization of an unconditional unprovability result due to Krajíček and Pudlák [J. Symb. Log. '89].

Our construction is, in fact, a simple variant of the SNARG constructed by Jin, Kalai, Lombardi, and Vaikuntanathan [STOC '24]. While the soundness of their construction was only proven for a subclass of NP, we prove its soundness for all NP under our assumption. At the heart of our result is the key observation that cryptographic reasoning is *simple* in a formal sense: the security proof of most cryptographic primitives can be formalized in a weak theory. In particular, we show how to formalize the scheme of Jin et al. in Jeřábek's theory APC_1 [J. Symb. Log. '07] – a weak theory in bounded arithmetic.

*University of Washington, ychsieh@cs.washington.edu.
†NTT Research and Johns Hopkins, abhishek.jain@ntt-research.com.
‡MIT, jiatuli@mit.edu.
§NTT Research, surya.mathialagan@ntt-research.com.

Contents

I Overview of Results **1**

1 Introduction **1**
- 1.1 Our Results . . . 1
- 1.2 Discussions on Assumption 1 . . . 4
- 1.3 Future Directions . . . 7
- 1.4 Related Work and Comparison . . . 8
- 1.5 Organization . . . 10

2 Technical Overview **11**
- 2.1 Main Idea: Lifting EF-SNARG to SNARG . . . 11
- 2.2 Our Choice: Theory APC_1 . . . 12
- 2.3 Formalizing Cryptography in APC_1 . . . 13
- 2.4 Formalizing [JKLV24] SNARG in APC_1 . . . 17

II Assumption and Security Lifting Lemma **23**

3 Preliminaries **23**
- 3.1 Recap of Mathematical Logic . . . 23
- 3.2 Jeřábek's Theory APC_1 . . . 23
- 3.3 Succinct Non-Interactive Arguments (SNARGs) . . . 25
- 3.4 Learning With Error . . . 26
- 3.5 SXDH . . . 26

4 The Bounded Theory and Our Unprovability Assumption **27**
- 4.1 Nonlogical Axiom 1: $\mathsf{prBPP} = \mathsf{prP}$ . . . 27
- 4.2 Distributions, Indistinguishability, and Hybrid Arguments . . . 29
- 4.3 Nonlogical Axiom 2: Hardness of Learning with Error . . . 30
- 4.4 Nonlogical Axiom 3: Hardness of SXDH . . . 32
- 4.5 The Unprovability Assumption . . . 35

5 Formalization of SNARGs and Security Lifting Lemma **35**
- 5.1 Search Security Game . . . 35
- 5.2 SNARG for Languages with Short Proof of Non-membership . . . 36
- 5.3 Security Lifting Lemma: From Provably Secure EF-SNARG to SNARG . . . 37

6 Discussions on the Unprovability Assumption **40**
- 6.1 Possible Variants and Attacks . . . 40
- 6.2 Razborov's Challenge: Proof Complexity from Computational Complexity . . . 41
- 6.3 Verifiable Proofs . . . 42
- 6.4 Choice of the Theory $\mathcal{T}$ . . . 44

III Formalization of SNARG in Bounded Arithmetic **47**

7 Toolkit for Cryptography in Bounded Arithmetic **47**

7.1 Basic Probability Principles . . . 47
7.2 A Useful Lemma for Sampling . . . 49
7.3 Averaging Arguments . . . 49
7.4 Tools for Indistinguishability . . . 51
7.5 Tools for Search Security Games . . . 54
7.6 Tools for Almost Identical Distributions . . . 58

8 Fully Homomorphic Encryption 59
8.1 Definition of Secret Key Gate-by-gate FHE . . . 60
8.2 Formalization of FHE in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ . . . 61
8.3 GSW FHE Construction . . . 63
8.4 Security Analysis for the Base Case . . . 63
8.5 Security Analysis for Leveled GSW . . . 66
8.6 Efficiency and Correctness . . . 68

9 Somewhere Extractable Hashing 68
9.1 Formalization of SEH in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ . . . 69
9.2 HW SEH Construction . . . 70
9.3 Security Analysis of SEH . . . 71

10 Batch Arguments for NP 74
10.1 Formalization of BARG in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ . . . 75
10.2 WW BARG Construction . . . 76
10.3 Preparations: Number-Theoretic Lemmas . . . 79
10.4 Security Analysis of BARG . . . 81
10.5 BARG with Set Extraction . . . 87

11 SNARG Construction and Analysis 88
11.1 Definitions and Tools . . . 88
11.2 Local assignment generators . . . 89
11.3 Encrypt-Hash-and-BARG and Construction . . . 95
11.4 Proof of JKLV Security Theorem (Theorem 11.21) . . . 98
11.5 Proof of Theorem 11.23 . . . 110

A List of Internal Theorems in Bounded Arithmetic 127
A.1 Infrastructure: Consistency of CAPP . . . 127
A.2 Probability Theory . . . 127
A.3 Cryptographic Tools . . . 128
A.4 Cryptographic Theorems . . . 129

B Extended Frege with Constant Width 130

Part I
Overview of Results

1 Introduction

Succinct non-interactive arguments (SNARGs) [Mic94] allow a prover to convince a verifier about the validity of an NP statement by sending a single message. The key property of SNARGs is *succinctness*: The size of the proof and the running time of the verifier is much smaller than the size of the NP witness. SNARGs are defined in the common reference string (CRS) model, where the prover and the verifier receive a common string honestly sampled from some distribution. SNARGs guarantee computational soundness; that is, no efficient cheating prover can convince a verifier of a false statement.

SNARGs are a highly sought-after cryptographic primitive due to their many real-world applications, most significantly in the area of blockchains. An extensive line of research spanning more than two decades has studied the feasibility and efficiency of SNARGs. Currently, SNARGs for all NP are known from two different classes of assumptions: Non-standard assumptions such as Random Oracles or knowledge assumptions [Mic94, Gro10], and Obfustopia assumptions [SW14, WW24, JKLM25], namely, indistinguishability obfuscation [BGI+01] and witness encryption [GGSW13]. This is in sharp contrast to interactive succinct arguments that are known from a fairly mild assumption of collision-resistant hash functions [Kil92].

The most pressing question in the study of SNARGs is whether their existence can be established from standard cryptographic assumptions such as learning with errors (LWE) or group-based assumptions that are not known to imply obfuscation. A recent line of work has made progress on this front for prominent subclasses of NP, including batch-NP (i.e., conjunctions of NP statements) and all deterministic computations [KPY19, JKKZ21, CJJ21, CJJ22, KVZ21, HJKS22, WW22, CGJ+23, BBK+23, NWW24, JJ25]. The state-of-the-art result in this line of work is due to Jin et al. [JKLV24] who constructed SNARGs from LWE that achieve (non-adaptive[1]) soundness for all NP languages that admit a proof of non-membership in the Extended-Frege (EF) propositional proof system[2] [CR79, Coo75]. A language $\mathcal{L}$ with a relation R admits a proof of non-membership if for every $x \notin \mathcal{L}$, there exists a polynomial-sized EF proof of the fact that $\forall w, R(x, w) = 0$.

We refer to such a scheme as an EF-SNARG. Since proofs of non-membership are unlikely to exist for languages outside NP $\cap$ coNP, it would seem unlikely that EF-SNARGs can achieve soundness for all NP.

1.1 Our Results

Somewhat surprisingly, we show that under a new, but natural complexity assumption, EF-SNARG achieves soundness for *all* NP. In the following, we state our assumption and our results in detail.

First, let us mention the kind of assumptions that we do *not* make in this work. It is not difficult to see that EF-SNARG is a SNARG for all NP under the very strong assumption that Extended Frege is p-bounded, i.e., every tautology ϕ admits a $\mathsf{poly}(|\phi|)$ size Extended Frege proof. To see this, note that for any language $\mathcal{L}$ with relation R, the claim $\forall w, R(x, w) = 0$ is a tautology if $x \notin \mathcal{L}$. Thus, the above assumption implies that every language $\mathcal{L}$ has a polynomial-sized EF proof of non-membership. However, this would contradict

[1] A SNARG is said to satisfy non-adaptive soundness if the adversary chooses the challenge statement independent of the CRS. In this work, unless mentioned otherwise, we always consider non-adaptive soundness.

[2] Extended Frege (see [CR79]) is a propositional proof system where one can introduce *extension variables* to abbreviate formulae, which allows manipulations of *circuits*.

the widely held belief that $\mathsf{NP} \neq \mathsf{coNP}$. In fact, Cook and Reckhow [CR79] proved that $\mathsf{NP} \neq \mathsf{coNP}$ if and only if all propositional proof systems are not p-bounded. In other words, we expect that there are tautologies that are hard against *any* propositional proof system fixed in advance.

Nevertheless, despite decades of efforts, such hard tautologies appear to be hard to construct even for Extended Frege. We refer the reader to Section 6.2 for a comprehensive survey. Indeed, Razborov [Raz15] explicitly proposed the following problem:

> "*Proving lower bounds for strong proof systems P like Frege or Extended Frege modulo any hardness assumption in the purely computational world, however strong but still natural and believable.*"

Here, hardness assumptions "in the purely computational world" exclude $\mathsf{NP} \neq \mathsf{coNP}$ but include *any* falsifiable cryptographic assumption. So far, Razborov's challenge remains open.[3]

Unprovability as a source of hardness. We propose a plausible assumption that formalizes the intractability of resolving Razborov's challenge. We then show that under our assumption, a variant of EF-SNARG in [JKLV24] is a SNARG for all NP. To realize this goal, we seek hardness from *unprovability* in formal mathematical theories.

Unprovability results are abundant. Gödel's incompleteness theorem [Göd31] constructs explicit unprovability statements for any reasonably strong mathematical theory. Another famous example is the unprovability of Continuum Hypothesis in ZFC [Coh63, Coh64]. In theoretical computer science, there is a rich literature on the unprovability of complexity upper and lower bounds in weak mathematical theories (see Section 1.4 and [Oli25] for a comprehensive survey).

How could unprovability results be useful for cryptography? In general, it could be the case that a complexity assumption (say $\mathsf{NP} \neq \mathsf{coNP}$) is *true*, but it is *unprovable* in some formal mathematical theory.[4] For cryptographic applications that would benefit from a strong (and potentially, unbelievable) assumption (such as $\mathsf{NP} = \mathsf{coNP}$), the unprovability of its negation might be sufficient. While this intuition is simple and promising, so far, it has not yielded any results in cryptography.

Our results. We show that the EF-SNARG construction of [JKLV24] is indeed a SNARG for NP under a new assumption that, intuitively, asserts the non-existence of *simple* solutions to Razborov's challenge. We start by stating our assumption.

Assumption 1 (Hardness Certification Assumption, Informal; see Assumption 3)**.** There is a constant $k \in \mathbb{N}$ such that for any language $\Phi \in \mathsf{NP}$, one of the following conditions does not hold:

- (*Completeness*). There are infinitely many tautologies ϕ such that $\phi \in \Phi$.
- (*Provable Soundness*). The following sentence is *provable* in a mathematical theory $\mathcal{T}$.[5]

  Every $\phi \in \Phi$ requires at least $|\phi|^k$ size Extended Frege proof under assumptions in Γ.

The language Φ is said to be a *certification* of n^k-size Extended Frege lower bounds.

[3] The general question of whether proof complexity lower bounds follow from "purely computational" lower bounds have a longer history; see, e.g., [Kra05, Section 4] and references therein.

[4] A very strong form of NP-vs-coNP style separation is indeed unprovable in Cook's theory PV_1; see Section 6.4.

[5] This is equivalent to say that the negation of the sentence, namely there exists $\phi \in \Phi$ with polynomial sized EF proof, is *consistent* with the theory $\mathcal{T}$.

If both properties hold, the NP witness of $\phi \in \Phi$ serves as a certification that ϕ does not have short EF proofs. It is *succinct* as the proof is much shorter than exhaustive search over all short EF proofs, *useful* in that it accepts infinitely many tautologies, and *convincing* provided that the mathematical theory $\mathcal{T}$ is sound. The *Hardness Certification Assumption* asserts that this is impossible.

If this assumption is false, we can obtain super-polynomial Extended Frege lower bound under the assumption set Γ in theory $\mathcal{T}$. In this work, the assumption set Γ consists of a derandomization assumption, i.e., $\mathsf{prBPP} = \mathsf{prP}$, and two standard assumptions in cryptography, namely, LWE and the Symmetric External Diffie-Hellman (SXDH) assumption over bilinear maps; thus falsifying our assumption resolves Razborov's challenge. Furthermore, we instantiate theory $\mathcal{T}$ with Jeřábek's theory APC_1 [Jeř07a] – a weak theory in Bounded Arithmetic. Thus, the proof is *simple* in a formal sense: a certification Φ must be explicitly provided, and its soundness must be provable in the weak mathematical theory APC_1. In general, a mathematical proof that positively answers Razborov's challenge might not necessarily satisfy these restrictions.

Using the Hardness Certification Assumption, we prove the following theorem:

Theorem 1.1 (Main Result, Informal; see Corollary 11.22)**.** *Assuming* $\mathsf{prBPP} = \mathsf{prP}$*, the hardness of LWE and SXDH, and Assumption 1, SNARGs for* NP *exist. Indeed, (a variant) of the EF-SNARG in [JKLV24] is a SNARG for* NP *with non-adaptive soundness. For instances of length* n*, the SNARG has proofs of size* $\mathsf{poly}(\lambda)$ *and a CRS of size* $\mathsf{poly}(\lambda, n^k)$ *for some* $k \in \mathbb{N}$*.*

Theorem 1.1 suggests that either SNARGs for NP exist, or there is a surprisingly simple resolution of Razborov's challenge. This presents a "win-win scenario" for cryptography and proof complexity, but in our view, the latter win might be too good to be true.

On the simplicity of cryptographic reasoning. To obtain Theorem 1.1, we formalize the soundness proof of EF-SNARG in Jeřábek's theory APC_1. Specifically, we consider a particular instantiation of the EF-SNARG construction of [JKLV24] wherein we use non-interactive batch arguments based on SXDH [WW22].[6] Along the way, we show how to formalize and prove security of many cryptographic primitives based on lattices and groups in theory APC_1. Put differently, we show and leverage the fact that cryptographic reasoning is *simple*, in a formal sense. We elaborate on this in Section 2.

Cryptography from unprovability. Modern cryptography typically relies on the hardness of solving computational problems. In this work, we demonstrate how to build cryptography from a new source of hardness: proving theorems from weak mathematical theories.

Our work is inspired by a recent work of Ilango [Ila25] that constructs "effective zero-knowledge proofs" with no interaction and no setup from a proof complexity assumption. The assumption underlying our work is quite different from [Ila25]: we use the unprovability of a *concrete and meaningful* mathematical theorem from *weak* proof systems, while Ilango considers separations between strong proof systems; see Section 1.4 for more discussions.

Although we focus on the application to SNARGs in this work, we believe that the perspective of using *unprovability assumptions* for cryptography merits further investigation. The toolkit we develop in this work might be useful in this endeavor. See Section 1.3 for a discussion on future directions.

[6] Our specific choice of batch arguments is guided by the simplicity of the security proof of the batch argument, and we do not view it as essential to our results. See Section 1.3 for discussion.

1.2 Discussions on Assumption 1

In this subsection, we provide a deeper dive into our assumption and its connections to existing lines of research in proof complexity and cryptography.

Why is Razborov's challenge relevant? To start with, we give a high-level explanation of why the intractability of Razborov's challenge could be useful. Note that if an EF-SNARG scheme does not satisfy the security of SNARG, then

- either it does not even satisfy EF-SNARG security;
- or Extended Frege is not p-bounded.

This immediately implies that if any EF-SNARG scheme (e.g. [JKLV24]) is provably secure as EF-SNARG from falsifiable cryptographic assumptions, and is provably insecure as a SNARG for NP, we can resolve Razborov's challenge: under the falsifiable cryptographic assumptions, Extended Frege is not p-bounded. That is, EF-SNARG is *effectively*[7] a SNARG (i.e. cannot be provably refuted as SNARG), or one can give a positive answer to Razborov's challenge.

In this work, we significantly sharpen this idea. Instead of aiming for effective security, we show that under Assumption 1, we can achieve *standard security* of SNARGs.

Two ideas behind the assumption. Assumption 1 is a combination of two key observations that we elaborate on in the following. First, certifying hard tautologies against Extended Frege is *nontrivial*. To our knowledge, the only known method to certify that a formula is indeed hard against Extended Frege is to exhaustively search over all possible proofs, which takes exponential time. In a sense, a language Φ violating Assumption 1 will provide a substantial speedup to certifying hardness of certain tautologies, using its NP witnesses.[8] Note that this relies on the widely believed conjecture that $\mathsf{NP} \neq \mathsf{coNP}$, as otherwise any language in coNP (including the language of hard tautologies) will be in NP.

We emphasize, however, that the first perspective alone does not lead to a meaningful result. To see this, note that we could obtain a seemingly similar result with a much simpler proof if we replace the *provable soundness* property in Assumption 1 by vanilla *soundness*:

- (*Soundness*). Every $\phi \in \Phi$ requires at least $|\phi|^k$ size Extended Frege proof.

However, this assumption contradicts a standard proof complexity conjecture that Extended Frege is not *optimal*; see Section 6.1 for more details.

To address this issue, we introduce our second perspective of *unprovability*: even if it is possible to certify hard tautologies, the correctness of the certification may not be provable in *a mathematical theory* $\mathcal{T}$. Note that, to our knowledge, our assumption and the conjecture that Extended Frege is not *optimal* are consistent with each other after introducing the unprovability perspective.

In cryptography, it is desirable to base security on as weak assumptions as possible. We would therefore like to choose the *weakest possible* mathematical theory $\mathcal{T}$ and the *weakest possible* assumption set Γ in Assumption 1 to support Theorem 1.1. This will sharpen Theorem 1.1 as a win-win result: if the assumption is false, it would lead to stronger and more surprising *provability result* in proof complexity. In the following, we discuss in detail both of these aspects, the theory $\mathcal{T}$ and the assumption set Γ.

[7] The notion of effective security is proposed by Ilango [Ila25] in the context of zero-knowledge proofs; see Section 1.4 for discussion.

[8] A minor issue of this speedup view is that a hardness certification Φ may also accept formulas that are *not* tautologies; this will be discussed later in Remark 6.5.

Theory	Infinite Sets as Symbols	Defines Functions[9]	Example of Unprovable Results (provable in the theory above it, if any)
ZFC	Yes	Highly Uncomputable	Continuum Hypothesis [Coh63, Coh64]
Peano	No	Primitive Recursive	Infinitary Ramsey Theorem [PH77]
T_2	No	$\mathsf{FP}^{\mathsf{PH}}$	$\forall x\, \exists y\, x = \log_2 y$ [Par71]
APC_1	No	$\mathsf{FP}^{\mathsf{AVOID}}(\subseteq \mathsf{FP}^{\Sigma_2^p})$	Weak Pigeonhole Principle (conditional) [Jeř07b]
PV_1	No	FP	dual Weak Pigeonhole Principle (conditional) [ILW23]

Table 1: Comparison between mathematical theories from a *Constructivism* point of view: ZFC set theory [Sup72], Peano Arithmetic [Kra95, Chapter 2], Primitive Recursive Arithmetic (PRA) [Cel22, Chapter 4], Buss's T_2 hierarchy [Bus85], Jeřábek's APC_1 [Jeř07a, BKT14], and Cook's PV_1 [Coo75].
The last three theories are considered bounded theories. The definition of the range avoidance problem AVOID and related results can be found in [Kor25].

Bounded Arithmetic and APC_1. As mentioned above, our choice for theory $\mathcal{T}$ is APC_1 introduced by Jeřábek [Jeř07a]. The name APC stands for **AP**proximate **C**ounting. It is called APC_1 to emphasize that its correspondence to the first-level of the polynomial hierarchy ($\Delta_1\mathsf{P} = \mathsf{P}$), and there is a theory APC_2 with stronger counting functionality (for $\Delta_2\mathsf{P} = \mathsf{P}^{\mathsf{NP}}$); see, e.g., [BKT14].[10]

Theory APC_1 is among theories collectively known as *Bounded Arithmetic*, weak fragments of Peano Arithmetic corresponding to *complexity classes*. In a nutshell, bounded theories have limited reasoning capability as they can define functions only from their corresponding complexity classes. For instance, Cook's theory PV_1 [Coo75] only defines polynomial-time functions, and Buss's hierarchy $T_2^0, T_2^1, T_2^2, \ldots$ [Bus85] only defines functions from polynomial-time hierarchy. Theories are defined for classes ranging from AC^0 to PSPACE and beyond; see [Kra95, CN10]. In addition, bounded theories can be viewed as uniform variants of propositional proof systems via propositional translations (see, e.g., [Coo75, Kra19]).

For our purpose, the theory $\mathcal{T}$ should be as weak as possible because we assume the unprovability of sentences in $\mathcal{T}$. Cook's theory PV_1 [Coo75] corresponding to FP does not suffice as the sentences in Assumption 1 (i.e. $\mathsf{prBPP} = \mathsf{prP}$ and cryptographic assumptions) cannot be naturally formalized in PV_1 — they involve probability and randomized algorithms, whereas PV_1 only defines polynomial-time algorithms. Therefore, we choose the theory APC_1 [Jeř07a], which is the **weakest possible theory** known to even *state* the sentences. This theory is stronger than PV_1 but weaker than T_2^2, the second level of Buss's T_2 hierarchy [BKT14]. Table 1 provides a brief comparison between different theories; see also Section 3.2 for more discussions of APC_1.

The assumption set Γ. Similar to the choice of the base theory APC_1, the assumption set Γ — the *non-logical axioms* for proving the soundness of the hardness certification Φ, or the assumption to address Razborov's challenge — should also be as weak as possible. In this work, the assumption set Γ consists of a

[9] The meaning of "defining a function $f(x)$" in a theory $\mathcal{T}$, intuitively, is that $\mathcal{T} \vdash \forall x\, \exists! y\, f(x) = y$. More formally, we mean functions definable using Σ_1^b-formulas in the language of $\mathcal{T}$, see, e.g., [Bus85] and [Kra95, Chapter 6].

[10] This theory is built upon first-order logic (FOL), the most well-studied logical foundation for mathematics. Nevertheless, FOL is not the only option; the theory PV [Coo75] is built on a much simpler equational logic (also known as "logic-free" system), and the theory iS_2^1 (see, e.g., [CU93]) is built on intuitionistic first-order logic.

widely-believed derandomization assumption $\mathsf{prBPP} = \mathsf{prP}$, and two standard falsifiable cryptographic assumptions: LWE and SXDH.

Our use of these assumptions is guided by a particular variant of EF-SNARG [JKLV24] that we analyze in APC_1. LWE is used for (leveled) fully homomorphic encryption (FHE) [GSW13] and SXDH is used for batch arguments (BARG) for NP [WW22] — both key ingredients in EF-SNARG. We emphasize that we avoid using *generic* assumptions such as FHE and BARGs as axioms; indeed, we show in Section 6.4.2 that there exist FHE and BARG schemes that are secure under plausible assumptions, yet adding their correctness to Γ would unconditionally *invalidate* Assumption 1.

Remark 1.2 (Robustness of our results)**.** Both the soundness property and assumptions in Γ need to be formalized in the language of the mathematical theory $\mathcal{T} = \mathsf{APC}_1$, and there could be many possible formalizations. We stress that our results work using standard formalization methodology in bounded arithmetic literature (see, e.g., [Jeř07a, MP20, CN10]) and do not exploit special properties of formalization details; see Section 4. Subsequently, our result should hold under any standard formalizations.

Advantages of our unprovability assumption. A positive consequence of our assumption is that our SNARG construction relies only on the *polynomial* hardness of the underlying cryptographic assumptions. This reveals an intriguing possibility that SNARGs for NP exist even if, say, $\mathsf{NP} \subseteq \mathsf{DTIME}(2^{n^{o(1)}})$. Note that existing SNARG constructions based on iO are not compatible with this possibility, as existing construction of iO requires sub-exponential assumptions to accommodate the standard complexity leveraging argument. These assumptions break down when $\mathsf{NP} \subseteq \mathsf{DTIME}(2^{n^{o(1)}})$.

Our result is compatible with known black-box lower bounds for SNARGs. Recall that Gentry and Wichs [GW11] ruled out adaptively sound SNARGs with a black-box reduction to falsifiable assumptions. More recently, [CJ25] extended this result to non-adaptive SNARGs with a "succinct" CRS (i.e., with size smaller than the witness length) for so-called "one-query" reductions. Going beyond these results, one could perhaps conjecture an even stronger lower bound for SNARGs (that remains to be proven): any non-adaptive SNARG construction with a black-box security reduction must rely on subexponentially-falsifiable assumptions. Our result bypasses such a potential barrier because our reduction to the hardness assumption is *non black-box* – it requires the code of the adversary (see Section 2.1). Moreover, our assumption is a proof complexity assumption which does not seem to fit into the formulation of polynomial-time falsifiable assumption in [GW11].

Cryptanalysis and connections to existing research. In standard complexity-based cryptography, the confidence of standard assumptions is gradually built from decades of research in algorithms (for cryptanalysis) and complexity theory. While our assumption on the unprovability of Extended Frege lower bounds is new, we argue that it is, and likely will be in the future, supported by active lines of research in proof complexity and bounded arithmetic.

(*Cryptanalysis*). To attack Assumption 1, one needs to (1) propose a language $\Phi \in \mathsf{NP}$, (2) prove or practically demonstrate its completeness, and (3) formalize its soundness property in the bounded arithmetic theory. The first two steps are standard cryptanalysis of complexity-theoretic assumptions, while the third step aligns with an active line of research known as *bounded reverse mathematics* [CN10, CLO24, AT25].

In more detail, a major goal of bounded reverse mathematics is to formalize TCS (including computational complexity and proof complexity) and mathematics in general in weak fragments of bounded arithmetic. In particular, the theory APC_1 is known to formalize the exponential PCP theorem [Pic15b], circuit lower bounds [MP20], and many other probabilistic proofs [Le19]. Research in this direction may provide technical

tools and intuition for potential attacks to Assumption 1, as well as other unprovability assumptions that may be found useful in the future.

(*Unprovability results*). Although unconditionally proving Assumption 1 seems unlikely, techniques have been established to prove strong unprovability results in bounded theories PV_1, APC_1, and beyond (see, e.g., [PS21, CKKO21, LO23, ABM23]). Most interestingly, super-polynomial lower bounds for Extended Frege are known to be unprovable *unconditionally* in Cook's theory PV_1 by a classic result of Krajíček and Pudlák [KP89]. Our Assumption 1 generalizes this unprovability results in several dimensions, making it out of reach of current techniques. Nevertheless, it would be surprising if Assumption 1 is false given this unprovability result [KP89]; see Section 6.4 for more discussion. We also refer interested readers to [Oli25] and Section 1.4 for a survey.

As more techniques are being developed for both upper and lower bounds on *provability*, more cryptographic applications may be found, with the opportunity towards a new, fruitful theory in the intersection of proof complexity and cryptography.

1.3 Future Directions

Improving our results. We foresee a few avenues for strengthening Theorem 1.1.

(*Weakening the assumptions*). First, while the original EF-SNARG [JKLV24] was shown secure only assuming LWE, we show the security of our scheme under an additional assumption, namely, SXDH. This is because to show that the EF-SNARG is secure in our theory, we additionally have to consider all of the ingredients used to construct the EF-SNARG in a white-box manner and show their security directly from the underlying assumptions. A key ingredient of the EF-SNARG construction is batch-arguments for NP (BARGs). Current BARG constructions from LWE [CJJ22] are involved and rely on several ingredients (both information theoretic and cryptographic), all of which we would have to formalize in our theory. In this work, we focus on demonstrating how to use Assumption 1, and therefore adopt the simpler BARG construction of [WW22] whose security analysis is more direct.

We expect that with additional work, one could construct an EF-SNARG whose security can be proven in APC_1 assuming only LWE (and $\mathsf{prBPP} = \mathsf{prP}$). We leave this optimization for future work.

(*Weakening the base theory*). Secondly, the main reason to choose APC_1 as the base theory is that it is the weakest known theory to even *state* our cryptographic assumptions. The theory APC_1 extends PV_1 with dual Weak Pigeonhole Principle (dWPHP); as we will explain shortly, dWPHP is both the foundation of formalizing approximate counting, and a powerful combinatorial principle itself. For instance, APC_1 proves a very strong form of Chernoff bound (see [Jeř07a, Proposition 2.18]).

In this work, we do not use the full power of dWPHP as a combinatorial principle (albeit, having it as an axiom makes formalization easier). It could be interesting to introduce a weaker theory that is still able to state our cryptographic assumptions, and use it as the base theory of Assumption 1 for proving Theorem 1.1; see, e.g., [CLOW26] for a concrete proposal.

(*Extending to* iO). Finally, we point out that our assumption can also be applied in the context of indistinguishability obfuscation (iO). The work of Jain and Jin (JJ) [JJ22] shows how to construct iO for circuits where security holds for circuits C_0 and C_1 only if there exists an EF proof of the fact that $\forall x, C_0(x) = C_1(x)$. We call an iO scheme with this security guarantee an $\mathsf{EF\text{-}iO}$ scheme. A key advantage of the JJ construction is that it relies only on $2^{p(\lambda)}$-security of the underlying assumptions, for a fixed polynomial p, independent of the input length of the circuit. In contrast, existing constructions of iO with

standard security [JLS21, JLS22, RVV24] rely upon $2^{p(\lambda,n)}$-security of its underlying assumptions to achieve security for circuits with input length n.

We believe that techniques similar to our work can be used to show the security of the JJ construction of EF-iO in an extension of APC_1 (although possibly from different assumptions than LWE and SXDH). One could then use Assumption 1 to show that JJ EF-iO is in fact an iO for *all* circuits. Such a result would yield standard iO security from only $2^{p(\lambda)}$-security of the underlying assumptions. We leave full exploration of this direction to future work.

Cryptography from unprovability. An exciting future direction is leveraging unprovability as a resource in cryptographic applications beyond the ones considered in this work.

Consider the following motivating example: Cryptographic constructions are often analyzed in idealized models such as the random oracle model. However, in practice, the random oracle is instantiated with a widely used hash function (say) H, which results in only heuristic security. While the random oracle methodology is known to be unsound in general [CGH04], attacks against "natural" constructions following this paradigm are quite hard to come by. An interesting direction is to establish standard model security of such a scheme using an unprovability assumption such as the impossibility of mathematically proving that H does not satisfy a particular property of Random Oracles. Falsifying such an assumption would likely improve our understanding of widely used hash functions.

The above is an example of a broader phenomenon in cryptography where we have a candidate scheme for a task for which we neither have a security proof nor an attack. If our intuition favors the former and the main barrier towards a positive result appears to be a limitation of techniques, could unprovability assumptions be helpful in breaking this impasse?

1.4 Related Work and Comparison

SNARGs for NP. Starting with [Mic94], there is a large body of works (see, e.g., [Gro10, Lip12, BCI+13, BCCT13, BCC+17]) that construct SNARGs for NP in the random oracle model or other idealized models, or based on non-standard knowledge assumptions.

A parallel line of work [KR09, KRR14, KP16, BHK17, BKK+18, CCH+19, KPY19, JKKZ21, CJJ21, CJJ22, WW22, DGKV22, PP22, JJ22, KLVW23, KLV23, CGJ+23, BBK+23, JKLV24, JKLM25, JJ25] has constructed SNARGs for various subclasses of NP (including batch-NP and all deterministic computations) from standard assumptions such as LWE, SXDH or sub-exponential DDH. In the standard model, SNARGs for all NP are only known from Obfustopia assumptions, which in turn can be based on a combination of sub-exponentially falsifiable assumptions [JLS21, JLS22, RVV24]. This includes constructions based on iO [SW14, WW24, WZ24, WW25], and witness encryption (WE) with a proof of correctness in Extended Frege [JKLM25]. The latter work shows a more general result, namely, (a variant of) the EF-SNARG of [JKLV24] is secure for all NP assuming LWE and the existence of any two-message laconic argument system with a proof of correctness in Extended Frege. WE (or iO) with proof of correctness implies such an argument system. The assumption used in their work is incomparable to ours; in particular, we do not rely on Obfustopia assumptions. Concurrent to our work, the work of Devadas et al. [DHK+26] construct SNARGs for NP assuming LWE and the existence of a non-signaling PCP with a proof of correctness. This assumption, as far as we are aware, is incomparable to ours. The authors of [DHK+26] also present a sufficient (but not necessary) proof complexity assumption on the norm of nullstellensatz refutations. We refer the reader to their work for details.

Cryptography from Proof Complexity. Starting from [JJ22], a recent line of work has leveraged results from proof complexity to build cryptography. We provide a brief summary and comparison.

(*Application of proof complexity upper bounds*). Jain and Jin [JJ22] construct an input-succinct iO scheme for Turing machines where the obfuscation size does not grow with input length, and security holds for all machines M_0 and M_1 whose equivalence can be proven in Cook's Theory PV. Recently, [JJMP25] proved the security of this construction is for all TMs assuming the existence of witness-succinct witness-encryption (i.e., where ciphertext size does not grow with the witness size) with a proof of correctness in PV. The works of [MDS25, JJMP25] achieve further efficiency improvements to the scheme of [JJ22], either in runtime or obfuscation size.

The work of [JJ22] also constructed SNARGs with a small common reference string (CRS) based on iO for NP langauges with EF proofs of non-membership. Subsequently, [JKLV24] constructed EF-SNARGs with large CRS based on LWE. More recently, [JJ25] replaced LWE with sub-exponential DDH for NP languages with proofs of non-membership in TC^0-Frege. Most recently, [JKLM25] bootstrapped [JKLV24] to all NP under Obfustopia assumptions, as discussed above.

(*Application of proof complexity lower bounds*). While the above works leverage *upper bounds* in proof complexity, a recent work of Ilango [Ila25] uses *lower bounds* in proof complexity to build cryptography. By relying on Krajíček and Pudlák's conjecture [KP89] that there is no optimal proof system, [Ila25] constructs a one-message proof system, without any setup, that is statistically sound and *effectively* zero-knowledge. Informally, the latter property says that it is hard to disprove the existence of a simulator for their one-message proof system in a fixed theory such as ZFC. This new "effectively sound" primitive can be used to construct cryptographic primitives with standard security definition, e.g., Non-Interactive Witness Hiding Proofs with uniform provers and verifiers.

(*Comparison with our work*). Compared to the above works, one can view our work as demonstrating how to use *both* upper and lower bounds in proof complexity in tandem. Our work relies on the existence of constructions (i.e. EF-SNARGs) based on upper bounds (i.e. existence of an polynomial-sized EF proof). Although the upper bound is unlikely to be true for all tautologies, we show that the unprovability of lower bounds suffices. As bounded theories can be viewed as uniform counterparts of propositional proof system via propositional translation [Coo75, Kra95, Kra19], the unprovability assumption can be viewed as a (uniform) proof complexity lower bound.[11]

We note that the connection between a "statement" and "unprovability of its negation" is also a key idea in Ilango's result [Ila25]. The distinction is that Ilango considers cryptographic primitives with *effective* security[12], which is most meaningful when the mathematical theory is *strong*. Indeed, effective zero-knowledge against *generic* systems such as ZFC is achieved using the no optimal proof system assumption. In this work, the result is most meaningful when the theory is weak, and thus we work with Jeřábek's theory APC_1, the weakest possible theory available for our purpose.

[11] In this sense, our unprovability assumption is a meta-complexity assumption. It considers the proof complexity of proof complexity, where the former stands for unprovability in bounded arithmetic, and the latter is to say that the sentence conjectured to be unprovable is about Extended Frege lower bound. It is similar to the hardness of MCSP or $\mathrm{MK}^t\mathrm{P}$ (see, e.g., [LP21]) that asserts the computational complexity of a problem that itself is about computational complexity.

[12] Our result is closer to the aforementioned construction of Non-Interactive Witness Hiding Proofs in [Ila25] – it also achieves standard security using proof complexity assumptions. Intuitively, Ilango only needs to work with a strong system such as ZFC thanks to the generality of the no optimal proof system conjecture. For our purposes, it might be unsafe to assume Assumption 1 for theories such as ZFC.

(Un)provability of complexity theory. There is a rich literature on the (un)provability of complexity theory in bounded arithmetic.

(*Formalization*). PV_1 is known to formalize Cook-Levin theorem and PCP theorem [Pic15a], proof complexity lower bounds for Resolution [CP90], weak formulation of circuit lower bounds [Raz95a] and many results regarding error-correcting codes and hardness amplification [Jeř05]. APC_1 (and its conservative extension HARD^A) is known to formalize results in derandomization [Jeř07a, Jeř05], Goldreich-Levin theorem [Le19], Schwartz-Zippel lemma [AT25], and strong formulation of circuit lower bounds [MP20]. There has also been interests in formalizing broader mathematics, see, e.g., [Woo81, PWW88, Oja04]. Similar to completeness results in computational complexity, certain results in complexity theory are known to be *equivalent* to combinatorial principles with respect to weak theories [CLO24, LLR24, AT25].

(*Unprovablity*). Cook [Coo75] proved (by adapting Gödel's incompleteness theorem) that the consistency of PV is unprovable in PV. Unconditional unprovability results has been established for complexity upper bounds [CK07, KO17, BM20, BKO20, CKKO21, ABM23] and complexity lower bounds [Raz95a, Raz95b, Kra97, Kra11b, Pic15a, PS21, LO23, CLO25]. There are also interesting unprovability results based on computational or proof-theoretic assumptions, see, e.g., [KP98, ILW23, CLO24, Kra24, CRT25, CKK+25, GC25].

Standard textbooks [Kra95, CN10, Kra19, Kra11a, Kra25] are also good references for classical results, and the survey [Oli25] covers more recent results.

1.5 Organization

The paper consists of three parts. In the first part of the paper (i.e. this section and Section 2), we provide an overview of our results, including the intuition of the theory APC_1, our assumption, and a proof sketch of Theorem 1.1.

In the second part of the paper, we present our results in more details. Necessary background on logic, bounded arithmetic, and cryptography is provided in Section 3. In Section 4, we formally define our bounded theory and the assumption. In Section 5, we formally state and prove a *security lifting lemma* that turns an EF-SNARG with simple (i.e. APC_1) security proof into a SNARG under our hardness assumption. We then discuss possible variants, attacks, and the connection of our assumption to Razborov's challenge in Section 6.

With the security lifting lemma in place, it remains to construct EF-SNARG (from standard assumptions) and prove its security in the theory APC_1. This is done in the third part of the paper.

- We first setup in Section 7 a comprehensive toolkit for formalizing cryptography in APC_1. This includes basic probability principles, useful tools for statistically close distributions, indistinguishability distributions, search security games.
- In Section 8, we formalize the correctness and security of FHE, and show that the standard construction from LWE [GSW13] can be carried out in APC_1.
- In Section 9, we formalize somewhere extractable hash (SEH), and shows that the construction due to Hubáček and Wichs [HW15] can be formalized in APC_1.
- In Section 10, we formalize BARGs and show that the Waters-Wu construction [WW22] from SXDH can be formalized in the theory APC_1.

- Finally, we combine the components above to implement a variant of EF-SNARG in [JKLV24] that is provably secure in APC_1 from LWE and SXDH, in Section 11. This completes the proof of Theorem 1.1 (see Corollary 11.22) by combining the EF-SNARG construction and the security lifting lemma.

A condensed list of lemmas and theorems formalized in bounded arithmetic is provided in Appendix A, for the convenience of future research. We also provide a definition of Extended Frege propositional proof systems and a structural lemma that is useful in formalizing [JKLV24] in Appendix B.

2 Technical Overview

We begin this overview by first discussing how to prove Theorem 1.1 from Assumption 1.

2.1 Main Idea: Lifting EF-SNARG to SNARG

In this work, we show how to 'lift" a SNARG construction with soundness only against EF-provable false statements (EF-SNARG for short) to a full-fledged SNARG for any language in NP, assuming the hardness certification assumption in Assumption 1. The key observation underlying our results is that cryptographic reasoning is *simple*. Specifically, we show in this overview that the security analysis of (a variant of) EF-SNARG construction in [JKLV24] can be formalized in a weak theory, specifically, APC_1. At a high-level, to accomplish this, we show that:

(*i*) Basic concepts in cryptographic proofs, such as indistinguishability, search security games, and hybrid arguments, can be naturally formalized in the theory APC_1.

(*ii*) A careful inspection of the security analysis in [JKLV24] shows that most of the proof can be carried out using the basic concepts developed in (*i*). The remaining, relatively minor, combinatorial or number-theoretic components can be formalized in APC_1 on a case-by-case basis.

We will discuss this in more detail in the subsequent sections.

Assuming an EF-SNARG construction with a *simple* security analysis, we prove our main theorem (Theorem 1.1) under the hardness certification assumption (Assumption 1) for some appropriate theory $\mathcal{T}$ and assumption set Γ. Indeed, this is a consequence of a generic *security lifting lemma* showing that under Assumption 1 instantiated with appropriate $\mathcal{T}$ and Γ, an EF-SNARG construction that is provably secure in $\mathcal{T}$ from assumptions in Γ is indeed secure as a SNARG.

The lemma is proved by contradiction: Suppose there exists some language $L \in \mathsf{NP}$ for which the EF-SNARG is not a SNARG, we can construct a certification $\Phi \in \mathsf{NP}$ of Extended Frege lower bounds that violates the assumption as follows.

- (*Construction of* Φ). We define $\phi \in \Phi$ if and only if ϕ is of the form "$x \notin L$", and there exists a polynomial-sized cheating prover $\mathcal{A}_x$ of the EF-SNARG that fools the verifier. [13]
- (*Completeness*). If this EF-SNARG is not a SNARG for NP, there are infinitely many strings x such that $x \notin L$, but a polynomial-sized adversary $\mathcal{A}_x$ that fools the verifier. In particular, $\phi_x :=$ "$x \notin L$" is a tautology such that $\phi_x \in \Phi$ for any such x.

[13] The certification Φ admits a probabilistic polynomial-time verifier (i.e. it is in prMA). By putting $\mathsf{prBPP} = \mathsf{prP}$ as an assumption in Γ, the verifier can be provably derandomized, i.e., $\Phi \in \mathsf{NP}$; see also Remark 2.2.

- (*Provable Soundness*). For every string x and $\phi_x := \text{“}x \notin L\text{”}$, the security of EF-SNARG asserts that if ϕ_x admits a short Extended Frege proof, then no efficient adversary $\mathcal{A}_x$ can convince the verifier that $x \in L$. Therefore, such $\phi_x \notin \Phi$. Crucially, if the security analysis of EF-SNARG can be formalized in $\mathcal{T}$ from assumptions in Γ, then the soundness argument of Φ can also be formalized in the theory $\mathcal{T}$ from Γ.

Note that the code of $\mathcal{A}_x$ is *explicitly* used to certify the hardness of ϕ_x in EF, i.e. the above argument makes *non black-box* use of the adversary.

Therefore, it suffices to formalize the soundness of the EF-SNARG in the theory $\mathcal{T}$ from assumptions in Γ. In Section 2.4 (and later in Section 11), we show that there exists an EF-SNARG construction whose security analysis can be formalized in $\mathcal{T} := \mathsf{APC}_1$ from assumptions $\Gamma := \{\text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}, \mathsf{LWE}, \mathsf{SXDH}\}$, which, combined with the *security lifting lemma*, yields our main SNARG construction. In the rest of the paper, we use $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ to denote the theory $\mathcal{T}$ augmented with assumptions Γ.

Overview of this section. In the remainder of this section, we provide a more detailed overview on how the EF-SNARG construction in [JKLV24] can be formalized and proven in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$. We first give an overview of our base theory APC_1 in Section 2.2. In Section 2.3, we discuss how to formalize cryptographic reasoning in APC_1. Finally, we give a brief sketch on how we show the security of the JKLV EF-SNARG in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ in Section 2.4.

2.2 Our Choice: Theory APC_1

A natural starting point for formalization is Cook's theory PV (see Section 3.2 for details on this theory). Multiple prior works have demonstrated how to formalize the *functionality* of cryptographic primitives in the theory PV_1 [JJ22, JKLM25, JJMP25], e.g., proving that encryption of a message decrypts to the same message.

However, formalizing *security* of cryptographic primitives is more challenging. In particular, security often involves probabilistic claims such as $\Pr_{x \leftarrow \{0,1\}^\lambda}[\mathcal{A}(x) = 1] \leq \varepsilon$. We cannot formalize the inequality in PV_1, as the probability on the left hand side is not known to be computable in deterministic polynomial time, thus cannot be written as a PV_1 function symbol. In other words, we cannot even state the security properties in PV_1.

The theory APC_1. To formalize probabilistic reasoning, we take the theory APC_1 as our starting point. APC_1 is an extension of PV_1 with the dual Weak Pigeonhole Principle (dWPHP), and was introduced to overcome the limitations of PV_1 in reasoning about probability. Building on earlier results of Wilkie [Kra95, Theorem 7.3.7] and Thapen [Tha05], Jeřábek [Jeř07a] further observed that dWPHP informally stated as follows, is closely related to probabilistic reasoning:[14]

> *For every circuit $C : \{0,1\}^n \to \{0,1\}^m$, where $m > n$, there exists a string $y \in \{0,1\}^m$ such that for every $x \in \{0,1\}^n$, $C(x) \neq y$.*[15]

By formalizing a form of Nisan-Wigderson pseudorandom generator [NW94], Jeřábek proved the following theorem that enables approximate counting on circuit-definable sets:

[14] The actual definition of dWPHP is slightly stronger than the version we stated here; see Section 3.2 and [Jeř07a] for more details.

[15] In the language of complexity theory, it states that the range avoidance problem is total (see [Kor25]).

Theorem 2.1 ([Jeř07a, Theorem 2.7], informal)**.** *The following theorem is provable in* $\mathsf{PV}_1 + \mathsf{dWPHP}$*: For every circuit* $C : \{0,1\}^n \to \{0,1\}$*, there exists a number* $s \leq 2^n$ *such that there is an explicit pair of "almost" bijection computable by circuits between* $X = \{x \in \{0,1\}^n \mid C(x) = 1\}$ *and* $s = \{0, 1, \ldots, s-1\}$*.*

Here, "almost" bijection means that a small portion of strings are not mapped back to themselves, which means that there is always an additive error term in approximate counting. This theorem shows that by representing approximate cardinalities via explicit "almost" bijections, the notion of set size for circuit-definable sets can be formalized in this theory. This provides a natural formalization of approximate counting and probability with additive error.

Beyond formalizing probabilities, Jeřábek [Jeř07a] also showed that we can perform meaningful mathematics with respect to this formalization by proving *counting and probability principles* in $\mathsf{PV}_1 + \mathsf{dWPHP}$: inclusion-exclusion principle, union bound, Chernoff bound, etc.

For technical reasons, we introduce an additional axiom formalizing $\mathsf{prBPP} = \mathsf{prP}$. For a PV function $\mathsf{capp}(\cdot,\cdot)$, which is intended to be the algorithm that solves Circuit Acceptance Probability Problem, we introduce an axiom showing its correctness:

> "$\mathsf{prBPP} = \mathsf{prP}$" : For any circuit C and $X := \{x \in \{0,1\}^n \mid C(x) = 1\}$, $\mathsf{capp}(C, 1^\kappa)$ outputs the size of $|X|$ (as defined by Theorem 2.1) up to an additive error of $2^n/\kappa$.

Therefore, we can formalize *approximate probability* in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$" as

$$\Pr_\delta[C(x) = y] := \mathsf{capp}(C_y, 1^{\delta^{-1}})/2^n, \quad \text{where } C_y(x) := (C(x) \stackrel{?}{=} y) \tag{2.1}$$

with an approximation error δ whose inverse is encoded in unary (denoted as $\delta^{-1} \in \mathsf{Log}$).

Remark 2.2 (an additional axiom: $\mathsf{prBPP} = \mathsf{prP}$)**.** The introduction of the axiom "$\mathsf{prBPP} = \mathsf{prP}$" is for two reasons. First, it simplifies the formalization of approximate counting (see Section 4.1 for related discussions). Second, it is also used to derandomize the hardness certification Φ in Assumption 1 from prMA to NP (see Section 5.3).

Note that for the former purpose (i.e. formalizing approximate counting), we can also use the theory HARD^A in [Jeř07a]. It is a *conservative extension* of APC_1 and thus does not increase the proof complexity strength of the theory. Nevertheless, as we will need $\mathsf{prBPP} = \mathsf{prP}$ for derandomization anyway, we can simplify our work using $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$" as the base theory.

2.3 Formalizing Cryptography in APC_1

We next explain how we formalize cryptographic reasoning in APC_1.

Asymptotic security. A key challenge in formalizing security notions and assumptions in APC_1 is handling *asymptotic* security, i.e., statements of the form

> For every adversary $\mathcal{A}$ with polynomial runtime $t(\lambda)$ and every inverse-polynomial advantage $\varepsilon(\lambda)$, there exists a large enough security parameter λ_0 such that for all $\lambda > \lambda_0$, $\mathcal{A}(1^\lambda)$ "wins" with probability at most $\varepsilon(\lambda)$.

To formalize such statements, we need to capture the "large enough" λ_0 in APC_1. In much of cryptography literature, when we state the security of an assumption, the value of λ_0 is merely assumed to *exist*. To capture

the asymptotic security without imposing additional restrictions on λ_0, we treat λ_0 as an arbitrary function of t and ε, whose existence is assumed *externally* to the theory, i.e., it does not need to be computable or even definable in APC_1. We then define the security as a *set* of sentences in APC_1.[16]

Example 2.1 (Asymptotic security $\mathcal{D}$)**.** Fix an arbitrary function λ_0 (not necessarily definable within the theory). An asymptotic security notion $\mathcal{D}_{\lambda_0}$ can be formalized as the following infinite set of sentences in APC_1:

$$\phi_{t,\mathsf{param}}: \quad \forall \lambda \geq \lambda_0(t,\mathsf{param}),\ \forall x \in \mathcal{S}_{\mathsf{param}(\lambda)},\ \mathcal{D}^{(\lambda,\mathsf{param},x)} \text{ is } (t(\lambda), 1/t(\lambda))\text{-secure.}$$

The set contains $\phi_{t,\mathsf{param}}$ for all polynomials t and all (appropriate) param represented as PV_1 function symbols, and $\mathcal{S}_{\mathsf{param}(\lambda)}$ denotes a definable set given $\mathsf{param}(\lambda)$. For example, param may be the input length of an encryption scheme, and $\mathcal{S}_{\mathsf{param}(\lambda)}$ could be the message space. Each value of $\lambda_0(t,\mathsf{param})$ is treated as a constant number and is hard-coded to the sentence in the language of the theory.

Note that the above assumption is equivalent to assuming $(t(\lambda), \varepsilon(\lambda))$-security for all polynomials t and all inverse-polynomial ε.

The final step towards fully formalizing asymptotic security is formalizing the notion of *parameterized* security, i.e., the sentence "$\mathcal{D}^{(\lambda,\mathsf{param},x)}$ is $(t, 1/t)$-secure" for fixed parameters $(\lambda, t, \mathsf{param}, x)$. In this work, we focus on the two most common forms of security notions: *indistinguishability-type* security (as in the case of secret-key encryption) and *search-type* security (as in the case of the setting of SNARGs). In both indistinguishability- and search-type security notions, the parameterized security asserts the acceptance probabilities of circuits, i.e., every adversary $\mathcal{A}$ running in time t wins with probability at most ε. We can then formalize these notions in APC_1 using the approximate probability formalization in Equation (2.1). In the following, we provide concrete examples of formalizing these two types of security notions in APC_1.

Example 2.2 (Semantic security of secret-key encryption)**.** Consider a secret-key (fully-homomorphic) bit encryption scheme with algorithms $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{Gen}(1^\lambda)$, $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{sk}, b \in \{0,1\})$. Let $\mathsf{param} = k(\cdot)$ be a polynomial describing the message length, and $\mathcal{S}_{\mathsf{param}(\lambda)} = \{0,1\}^{k(\lambda)} \times \{0,1\}^{k(\lambda)}$ be the space of message pairs. Fix any λ, param, and message pair $(m_0, m_1) \in \mathcal{S}_{\mathsf{param}(\lambda)}$. We define the following two circuits:

- $C_0^{\lambda,\mathsf{param},(m_0,m_1)}$: On input random seed $(\mathsf{sd}_{\mathsf{Gen}}, \mathsf{sd}_{\mathsf{Enc}})$ compute $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{Gen}(1^\lambda; \mathsf{sd}_{\mathsf{Gen}})$, $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{sk}, m_0; \mathsf{sd}_{\mathsf{Enc}})$, and output $(\mathsf{ek}, \mathsf{ct})$, where Enc for a k-bit message is defined as encrypting each bit of the message separately.
- $C_1^{\lambda,\mathsf{param},(m_0,m_1)}$: Same as $C_0^{\lambda,\mathsf{param},m_0,m_1}$ but encrypting m_1 instead of m_0.

The parameterized semantic security states that every adversary $\mathcal{A}$ with runtime t cannot distinguish the output distribution of C_0 and C_1 with advantage greater than ε. In APC_1, we formalize the indistinguishability by the sentence:

$$\forall \delta^{-1} \in \mathsf{Log}, \forall \mathcal{A} \text{ s.t. } |\mathcal{A}| \leq t,\ |\Pr_\delta[\mathcal{A}(C_0(\mathsf{sd})) = 1] - \Pr_\delta[\mathcal{A}(C_1(\mathsf{sd})) = 1]| \leq \varepsilon + O(\delta), \tag{2.2}$$

where $O(\delta)$ hides unspecified constant factors. This sentence states that the advantage of any adversary $\mathcal{A}$ can be bounded by an upper bound arbitrarily close to ε. Let D_0, D_1 be the distributions of $C_0(\mathsf{sd})$ and $C_1(\mathsf{sd})$ respectively. We denote the formulation of Equation (2.2) as $D_0 \approx_{t(\lambda),\varepsilon} D_1$.

[16] Formalization by *sets* of sentences is a common approach in bounded arithmetic to deal with nonconstructive parts of mathematical statements, see also [CLO24, Section 1.2], [CKKO21, Section 5.1], [LO23, Theorem 2.1], [MP20, Section 1.5].

Finally, the asymptotic semantic security for the encryption scheme is formalized as the following set of sentences in APC_1:

$$\phi_{t,k}: \quad \forall \lambda \geq \lambda_0(t,k),\ \forall (m_0,m_1) \in \{0,1\}^{2k(\lambda)},\ D_0^{\lambda,k,(m_0,m_1)} \approx_{t(\lambda),1/t(\lambda)} D_1^{\lambda,k,(m_0,m_1)}.$$

Example 2.3 (Soundness of EF-SNARG)**.** Consider an EF-SNARG for language $\mathcal{L}$ with algorithms $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^n, 1^\ell)$, $\pi \leftarrow \mathcal{P}(\mathsf{crs}, x, w)$, and $b \leftarrow \mathcal{V}(\mathsf{crs}, x, \pi)$, where $x \in \mathcal{L}$ is a statement of length $n(\lambda)$ with witness w of length $m(\lambda)$, and ℓ is the upper bound of the Extended Frege proof length of statements $x \notin \mathcal{L}$. Let $\mathsf{param} = (n(\cdot), \ell(\cdot))$ and $\mathcal{S}_{\mathsf{param}(\lambda)} = \{0,1\}^{n(\lambda)} \times \{0,1\}^{\ell(\lambda)}$. Fix any λ, param, and $(x,\tau) \in \mathcal{S}_{\mathsf{param}(\lambda)}$, we define the following *game* $\mathcal{G} = (C_1, C_2)$ consist of two circuits:

- $C_1^{\lambda,\mathsf{param},(x,\tau)}$: On input random seed $\mathsf{sd}_{\mathsf{Gen}}$, compute $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^{\mathsf{param}(\lambda)}; \mathsf{sd}_{\mathsf{Gen}})$, and output the challenge $\mathsf{chall} = \mathsf{crs}$ and a state $\mathsf{state} = \mathsf{crs}$.
- $C_2^{\lambda,\mathsf{param},(x,\tau)}$: On input an answer ans along with the state state, compute $\mathcal{V}(\mathsf{state}, x, \pi)$, and verify that checks whether τ is a valid Extended Frege proof of "$x \notin \mathcal{L}$". Output 1 if both verification passes, and output 0 otherwise.

For any adversary $\mathcal{A}$ which takes as input the challenge chall and outputs an answer ans, we define the circuit $T_{\mathcal{G},\mathcal{A}}$ that on input $\mathsf{sd}, \mathsf{sd}_{\mathcal{A}}$, runs $(\mathsf{chall}, \mathsf{state}) \leftarrow \mathcal{C}_1^\lambda(\mathsf{sd})$, $\mathsf{ans} \leftarrow \mathcal{A}(\mathsf{chall}; \mathsf{sd}_{\mathcal{A}})$, and forwards the output of $\mathcal{C}_2^\lambda(\mathsf{state}, \mathsf{ans})$.

The parameterized soundness of EF-SNARG states that for every $(x,\tau) \in \mathcal{S}_{\mathsf{param}(\lambda)}$ where τ is a valid Extended Frege proof of "$x \notin \mathcal{L}$", every adversary $\mathcal{A}$ with runtime t should not be able to find a proof π such that the verifier accepts with probability larger than ε. In APC_1, we formalize this using the notation of advantages. We say that the advantage of $\mathcal{A}$ satisfies $\mathsf{Adv}_{\mathcal{C}_1,\mathcal{C}_2}[\mathcal{A}] \leq \varepsilon$ if

$$\forall \delta \in \mathsf{Log}, \Pr_\delta[T_{\mathcal{G},\mathcal{A}}(\mathsf{sd}, \mathsf{sd}_{\mathcal{A}}) = 1] \leq \varepsilon + O(\delta).$$

Similarly, we write $\mathsf{Adv}_{\mathcal{C}_1,\mathcal{C}_2}[\mathcal{A}] \geq \varepsilon$ if $\Pr_\delta[T_{\mathcal{G},\mathcal{A}}(\mathsf{sd}, \mathsf{sd}_{\mathcal{A}}) = 1] \geq \varepsilon - \Omega(\delta)$.

Finally, the asymptotic soundness of EF-SNARG is formalized as the following set of sentences in APC_1:

$$\phi_{t,n,\ell}: \quad \begin{aligned} &\forall \lambda \geq \lambda_0(t,n,\ell), \forall (x,\tau) \in \{0,1\}^{n(\lambda)+\ell(\lambda)}, \\ &\qquad \forall \mathcal{A} \text{ s.t. } |\mathcal{A}| \leq t(\lambda),\ \mathsf{Adv}_{\mathcal{C}_1^{\lambda,n,\ell,(x,\tau)},\mathcal{C}_2^{\lambda,n,\ell,(x,\tau)}}[\mathcal{A}] \leq 1/t(\lambda). \end{aligned}$$

Toolkit for security proofs in APC_1. Having formalized asymptotic security in APC_1, we can now formalize *security proofs* in APC_1, i.e., prove one asymptotic security claim $\mathcal{D}$ assuming another claim $\mathcal{D}'$. At a high level, such a proof would require showing that every parameterized sentence $\phi_{t,\mathsf{param}}$ in $\mathcal{D}$ can be derived from some sentence $\phi'_{t',\mathsf{param}'}$ in $\mathcal{D}'$ in APC_1, where the mapping $(t, \mathsf{param}) \mapsto (t', \mathsf{param}')$ corresponds to an efficiently computable security loss. One could, in principle, write such proofs from first principles. However, this would involve explicitly handling approximate probabilities of Equation (2.1), which can quickly become infeasible to write and verify.[17]

To modularize the task, we develop a *toolkit* of lemmas in APC_1 that can be used as building blocks in security proofs. The toolkit includes standard tools in cryptographic proofs, such as hybrid arguments,

[17] It is instructive to draw an analogy to mature mathematical areas such as calculus. While one could compute derivatives of a function via first principles, we instead use a "toolkit" of known derivatives along with other modular tools such as chain rule to simplify the computation of a derivative.

reductions, and advantage amplification. All lemmas in this toolkit hide the details of handling approximate probabilities, and allow us to write security proofs that are more readable and closer in style to standard cryptographic proofs.

Below we list several standard lemmas relevant to the later section that we can prove in APC_1, each providing a clean interface. We refer the reader to Section 7 for the full list of lemmas in the toolkit.

- **Identical distributions.** For two circuits C_0 and C_1 sampling identical distributions D_0, D_1, if they differ up to a permutation over the randomness space, i.e., $C_0 \circ f \equiv C_1$ for some bijection f over the randomness space, then $D_0 \approx_{t,1/t} D_1$ for any unary variable t.[18]
- **Reduction Lemma.** Let C_0', C_1' be the circuits computing D_0', D_1', and let $R \circ C_b'(\mathsf{sd}, \mathsf{sd}') = R(C_b'(\mathsf{sd}'), \mathsf{sd})$ circuits computing distribution D_b by first sampling from D_b' via computing $C_b'(\mathsf{sd}')$, then compute its output based on the sample and some independent randomness sd. If $D_0' \approx_{t,\varepsilon} D_1'$ (for unary t), then $D_0 \approx_{t-|R|,\varepsilon} D_1$.
- **Hybrid Argument.** For a sequence of efficiently sampleable distributions $H_0, H_1, \ldots, H_L$, if $H_{i-1} \approx_{t,\varepsilon} H_i$ for every $i \in [L]$, then $H_0 \approx_{t,L\cdot\varepsilon} H_L$. Combining with the Reduction Lemma, we can also show that, for efficiently sampleable distributions $H_{0,1}, \ldots, H_{0,L}$ and $H_{1,1}, \ldots, H_{1,L}$, if $H_{0,i} \approx_{t,\varepsilon} H_{1,i}$ for every $i \in [L]$, then

  $$H_{0,1} \times H_{0,2} \times \cdots \times H_{0,L} \approx_{t-Ls,L\varepsilon} H_{1,1} \times H_{1,2} \times \cdots \times H_{1,L},$$

  where the product distributions are sampled by sampling each component independently, and s is the maximum size of the circuits sampling each $H_{b,i}$; see Product Hybrid Lemma.
- **Properties of game advantages.** The following lemmas are proved within $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$"; see Section 7.5.
  - (*Game Complement Lemma*). Let (C_1, C_2) be a game, and $\overline{C}_2$ be the circuit outputing the complement of C_2. Then, for any adversary $\mathcal{A}$,[19]

    $$\mathsf{Adv}_{(C_1,\overline{C}_2)}[\mathcal{A}] \geq 1 - \mathsf{Adv}_{(C_1,C_2)}[\mathcal{A}]$$

  - (*Game Composition Lemma*). Let (C_1, C_2) be a game, and suppose that C_1 and C_1' are (t,ε)-indistinguishable. Then, for any adversary $\mathcal{A}$ of size at most $t - |C_2|$, we have

    $$\mathsf{Adv}_{(C_1',C_2)}[\mathcal{A}] \leq \mathsf{Adv}_{(C_1,C_2)}[\mathcal{A}] + \varepsilon.$$

  - (*Adversary Indistinguishability Lemma*). Let (C_1, C_2) be a game, and $\mathcal{A}_0, \mathcal{A}_1$ be two adversaries. If for every fixed challenge chall, the two distributions D_b^{chall} generated by the circuit $\mathcal{A}_b(\mathsf{chall}, \cdot)$ are (t,ε)-indistinguishable, and that $t \geq |C_1| + |C_2|$, then

    $$\mathsf{Adv}_{(C_1,C_2)}[\mathcal{A}_0] - \varepsilon \leq \mathsf{Adv}_{(C_1,C_2)}[\mathcal{A}_1] \leq \mathsf{Adv}_{(C_1,C_2)}[\mathcal{A}_0] + \varepsilon$$

[18] We use slightly different notation in technical sections. Two distributions are said to be *isomorphic* if they differ up to a permutation over the randomness space. In particular, they are said to be *identical* if the circuits are functionally equivalent; see Section 4.2 for more details.

[19] This is a shorthand notation for $\mathsf{Adv}_{(C_1,C_2)}[\mathcal{A}] \leq \varepsilon \implies \mathsf{Adv}_{(C_1,\overline{C}_2)}[\mathcal{A}] \geq 1 - \varepsilon$; similar shorthand is also used below.

– (*Game Union Bound Lemma*). For any games $(C_1, C_2^{(1)})$, $(C_1, C_2^{(2)})$ and any adversary $\mathcal{A}$,

$$\mathsf{Adv}_{(C_1, C_2^1 \vee C_2^{(2)})}[\mathcal{A}] \leq \mathsf{Adv}_{(C_1, C_2^{(1)})}[\mathcal{A}] + \mathsf{Adv}_{(C_1, C_2^{(2)})}[\mathcal{A}],$$

where $C_2^{(1)} \vee C_2^{(2)}$ is the circuit that outputs 1 if either $C_2^{(1)}$ or $C_2^{(2)}$ output 1.

– (*Game Reduction Lemma*). Let (C_1, C_2) be a game, and R_1, R_2 be efficient circuits of appropriate input/output length. Then, for any adversary $\mathcal{A}$ and any advantage ε, we have

$$\mathsf{Adv}_{(R_1 \circ C_1, C_2 \circ R_2)}[\mathcal{A}] \geq \varepsilon \implies \mathsf{Adv}_{(C_1, C_2)}[R_2 \circ \mathcal{A} \circ R_1] \geq \varepsilon.$$

- **Error Reduction Lemma.** For any game $\mathcal{G} = (C_1, C_2)$ and an adversary $\mathcal{A}$ with inverse polynomial advantage p, Let k be a unary variable. Then, the k-fold adversary $\mathcal{A}^{\times k}$ has advantage at least $1 - (1 - cp)^k$ for some constant $c < 1$ against the k-fold game $\mathcal{G}^{\vee k}$, where the k-fold adversary runs $\mathcal{A}$ in k parallel, and wins the k-fold game if any of the k instances wins.

2.4 Formalizing [JKLV24] SNARG in APC_1

We now show how to formalize the security proof of the EF-SNARG construction from [JKLV24] in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$: the theory $\mathsf{APC}_1 + \mathsf{prBPP} = \mathsf{prP} + \mathsf{LWE}_{\lambda_0^{\mathsf{LWE}}} + \mathsf{SXDH}_{\lambda_0^{\mathsf{SXDH}}}$, where the axioms $\mathsf{LWE}_{\lambda_0^{\mathsf{LWE}}}$ and $\mathsf{SXDH}_{\lambda_0^{\mathsf{SXDH}}}$ are the Learning With Errors and Symmetric External Diffie-Hellman assumptions, formulated as asymptotic security notions as in Example 2.1 for some arbitrary functions λ_0^{LWE}, $\lambda_0^{\mathsf{SXDH}}$. In this exposition, we assume some basic familiarity with the cryptographic components, including FHE, SEH, and BARG, used in the JKLV SNARG construction.

We break the proof into two parts: first, we show at a high level that the building blocks of the EF-SNARG construction, i.e., fully-homomorphic encryption (FHE), somewhere extractable hash functions (SEH), and batch arguments (BARG) can each be proven secure in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$. Then, we show how to combine the security guarantees of these building blocks to prove the security of the overall EF-SNARG construction in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$.

2.4.1 Ingredients

The construction of our EF-SNARG is near identical to the encrypt-hash-and-BARG construction of JKLV [JKLV24]. As the name suggests, the construction relies on three main tools:

- A leveled fully homomorphic encryption scheme (FHE) (Definition 8.1). We will instantiate this via the (secret-key) Gentry-Sahai-Waters [GSW13] construction of FHE from LWE.
- A somewhere extractable hash family with local openings (SEH) (Definition 9.1). We will instantiate this via the Hubáček-Wichs [HW15] construction of somewhere extractable hashing from FHE.
- A somewhere extractable batch-argument scheme (BARG) (Definition 10.1), which we instantiate via the Waters-Wu [WW22] construction from SXDH.[20]

We now argue that the security of each of the above instantiations can be proved in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$. We give a more detailed argument for FHE, and brief sketches for the analyses of SEH and BARG.

[20] As discussed in Section 1.3, we conjecture that one can show the security of an LWE-based BARG scheme in bounded arithmetic, but we choose to instead analyze the Waters-Wu construction since it has a more elementary proof of security.

FHE and SEH in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}]$. The secret-key variant of GSW fully-homomorphic encryption scheme [GSW13] has secret key being a random vector $\mathsf{sk} = \mathbf{s} \leftarrow \mathbb{Z}_q^n$, an empty evaluation key ek, and the encryption of a bit $b \in \{0,1\}$ is given by

$$\mathsf{ct} := \begin{pmatrix} \mathbf{A} \\ \mathbf{s}^\intercal \mathbf{A} + \mathbf{e}^\intercal \end{pmatrix} + b\mathbf{G}., \text{ where } \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{e} \leftarrow \chi^m,$$

To argue the semantic security (See Example 2.2) of this scheme in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}]$, we need to show that for every two messages $m_0, m_1 \in \{0,1\}^k$, the distributions of encryptions of m_0 and m_1 are indistinguishable. The proof is essentially combining statements straight from our toolkit. Consider the following hybrids:

- $\mathcal{H}_0^{m_b}$: For $i \in [k]$, sample ciphertext $\mathsf{ct}_i = \begin{pmatrix} \mathbf{A}_i \\ \mathbf{s}^\intercal \mathbf{A}_i + \mathbf{e}_i^\intercal \end{pmatrix} + m_b[i] \cdot \mathbf{G}$.
- $\mathcal{H}_1^{m_b}$: For $i \in [k]$, sample ciphertext $\mathsf{ct}_i = \begin{pmatrix} \mathbf{A}_i \\ \mathbf{r}_i^\intercal \end{pmatrix} + m_b[i] \cdot \mathbf{G}$ for random $\mathbf{r} \leftarrow \mathbb{Z}_q^n$.

The indistinguishability between $\mathcal{H}_0^{m_b}$ and $\mathcal{H}_1^{m_b}$ follows from the Reduction Lemma (stated in Section 2.3) and the LWE assumption. If, from the LWE assumption, $\{\mathbf{s}^\intercal \mathbf{A}_i + \mathbf{e}_i^\intercal\} \approx_{t',1/t'} \{\mathbf{r}_i^\intercal\}$, then by our toolkit, we have $\mathcal{H}_0^{m_b} \approx_{t'-s,1/t'} \mathcal{H}_1^{m_b}$, where s is the size of the reduction circuit, upper bounded by the size of the circuit computing $\mathcal{H}_0^{m_b}$. Furthermore, the indistinguishability between $\mathcal{H}_1^{m_0}$ and $\mathcal{H}_1^{m_1}$ follows from the fact that they are *identical distributions*[21], as they differ up to a randomness shift by $(m_0 - m_1) \cdot \mathbf{G}$. Finally, by the Hybrid Argument in our toolkit, we conclude that $\mathcal{H}_0^{m_0} \approx_{t'-s,3/t'} \mathcal{H}_0^{m_1}$. Therefore, by setting $t' = \max(3t, t+s)$, we conclude that (omitting LWE parameters for simplicity)

$$\text{LWE is } (t', 1/t')\text{-secure} \implies \text{FHE is } (t, 1/t)\text{-semantically secure.}$$

In other words, every sentence in the set $\mathsf{FHESecure}_{\lambda_0^{\mathsf{FHE}}}$ defining the semantic security of the above FHE scheme can be derived from APC_1+"$\mathsf{prBPP} = \mathsf{prP}$"+$\mathsf{LWE}_{\lambda_0^{\mathsf{LWE}}}$ for $\lambda_0^{\mathsf{FHE}}(t,k) = \lambda_0^{\mathsf{LWE}}(t' = \max(3t, t+s))$. Note that we omit the choice of LWE parameters here for simplicity, see Section 8 for the full argument.

For the somewhere extractable hash function SEH from [HW15], the corresponding security property, i.e., the key indistinguishability property, is a direct application of the FHE semantic security[22]. In particular, the key indistinguishability expands to

$$\{\mathsf{FHE.Enc}(\mathsf{sk}_{i-1}, \mathsf{sk}_i)\}_{i \in [L]} \approx_{t,1/t} \{\mathsf{FHE.Enc}(\mathsf{sk}_{i-1}, 0^k)\}_{i \in [L]},$$

which again follows from reduction plus hybrid argument as in the FHE semantic security proof.

Batch arguments in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{SXDH}]$. The group-based BARG construction in [WW22] relies on elementary tools. This makes it easy to formalize its security analysis in APC_1. We slightly modify the construction so that it enjoys *perfect* correctness and somewhere extractability; this allows us to formalize properties and prove these properties in theory PV_1. Moreover, the CRS indistinguishability property

[21] Indeed, the two circuits sampling the distributions differ by a permutation; these two distributions are formally defined to be *isomorphic*; see Section 4.2.

[22] For technical reasons, we need to rely on an FHE scheme which additionally satisfies "malicious gate correctness". We gloss over this detail here.

can be almost directly reduced (via the Reduction Lemma) to the following indistinguishability of group elements:

$$(g, g^a, g^{b_1}, g^{b_2}, g^{ab_1}, g^{ab_2}) \approx_{t,1/t} (g, g^a, g^{b_1}, g^{b_2}, g^{c_1}, g^{c_2}),$$

where g is a generator in either source groups of a prime order bilinear group, and a, b_1, b_2, c_1, c_2 are random exponents in $\mathbb{Z}_p$. The above indistinguishability can be easily proven from the SXDH assumption via a Hybrid Argument. We refer the reader to Section 10 for details.

2.4.2 EF-SNARG in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{FHE}, \mathsf{SEH}, \mathsf{BARG}]$

We now give an informal description of the encrypt-hash and BARG construction of JKLV [JKLV24]. We make several subtle changes to the scheme, and refer the reader to Remark 11.20 for details. Informally, the CRS of the scheme comprises of the following components:

- a FHE ciphertext ct encrypting some dummy circuit E, and an FHE evaluation key ek,
- a BARG CRS crs,
- SEH hash keys $\mathsf{hk}_1, \ldots, \mathsf{hk}_{\mathsf{loc}}$ for some $\mathsf{loc} = \mathsf{poly}(\lambda)$. For simplicity, we write $\mathsf{hk} = (\mathsf{hk}_1, \ldots, \mathsf{hk}_{\mathsf{loc}})$, and we define the hashing as well as giving local openings with respect to hk as simply hashing and giving local openings to each hk_i in parallel.

At a high level, an EF-SNARG prover generates a proof as follows: for an NP circuit C, an instance x, and a witness w, it does the following:

- Compute $C(x, w)$. Let τ denote the wire values of this computation.
- Evaluate the dummy circuit E *homomorphically* on τ to obtain wire values τ' of this computation.
- Hash the wire values τ and τ' to obtain hash value h.
- Compute a batch-proof π that for every gate g in the computation of C and E, the prover has local openings from h to wire values w_1, w_2, w_3 such that the gate computation is satisfied.
- Output $(h, \pi, \rho_{\mathsf{out}})$, where ρ_{out} is an opening from h to the output wire of C.

The verifier accepts if the BARG proof π is accepted, and ρ_{out} is an opening to $\mathsf{out} = 1$.

Local assignment generators. We first recap the proof strategy from JKLV. The security proof proceeds by constructing *local assignment generators* [PR17]. A local assignment generator $\mathsf{LocalGen}$ for a circuit C with locality ℓ is an algorithm which takes as input ℓ wires of C, outputs an assignment to the wires, while satisfying the following guarantees:

- **Local consistency:** Any assignment output by $\mathsf{LocalGen}$ is *locally consistent*, i.e. respects any gates contained in the queried set.
- **Computationally non-signaling:** Let T_0 and T_1 be subsets of wires of C of size at most ℓ. Then, the marginal distributions of $\mathsf{LocalGen}(T_0)$ and $\mathsf{LocalGen}(T_1)$ on $T_0 \cap T_1$ are computationally indistinguishable.

We additionally say $\mathsf{LocalGen}$ is "accepting" if $\mathsf{LocalGen}$ assigns the value 1 to the output gate of C with probability $1 - \mathsf{negl}(\lambda)$. Our first step is to formalize these guarantees of a local assignment generator in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ (for more details, see Section 11.1). We say that $\mathsf{LocalGen}$ is a (ℓ, ε, t)-local assignment generator if the following hold.

- **ε-Local Consistency.** Consider a set $T = \{w_1, w_2, w_3\}$ where w_1 and w_2 are inputs and w_3 is the output of some gate g in the circuit. Construct the game $\mathcal{G}_g = (C_1, C_2)$ where C_1 is the circuit which outputs $T = \{w_1, w_2, w_3\}$, and C_2 is the circuit which takes as input $\{\sigma_1, \sigma_2, \sigma_3\}$ and outputs 1 if $\sigma_3 \neq g(\sigma_1, \sigma_2)$ (i.e. the wire values are inconsistent), and 0 if it is consistent. We say $\mathsf{LocalGen}$ is ε-consistent if
$$\mathsf{Adv}_{\mathcal{G}_g}[\mathsf{LocalGen}] \leq \varepsilon$$
for every gate g in the circuit.

- **(t, ε)-Non-Signaling:** Consider the following distribution $\mathcal{D}_{T_0,T_1,b}$ does the following:
  - Sample $(\sigma_w)_{w \in T_b} \leftarrow \mathsf{LocalGen}(T_b)$.
  - Output $(\sigma_w)_{w \in T_0 \cap T_1}$.

  We say that $\mathsf{LocalGen}$ satisfies (t, ε)-non-signaling if the distributions $\mathcal{D}_{T_0,T_1,0}$ and $\mathcal{D}_{T_0,T_1,1}$ are (t, ε)-indistinguishable.

The security proof of [JKLV24] has two steps.

(1) **Local assignment generator for any extension circuit.** Consider any prover $\mathcal{P}$ which creates accepting proofs for a statement x^* with probability ε. First, note that by FHE security, this probability does not diminish if the circuit E is replaced by any circuit E_{x^*} when the CRS is generated. We can use such a $\mathcal{P}$ to construct an *accepting* local assignment generator for the circuit $C(x^*, \cdot)$ extended with E_{x^*}.

(2) **Construct an extension circuit which does not have an accepting local assignment generator.** As a second step, we show that if $x^* \notin \mathcal{L}$ and there is a size ℓ proof that $\forall w, C_{x^*}(w) = 0$, then there exists some circuit E_{x^*} of size $\mathsf{poly}(\ell)$ for which there is no *accepting* local assignment generator.

By combining the above two steps, we reach a contradiction.

While the security proof of JKLV is involved, we show how to break it down into steps which can all be formalized using the toolkit from Section 2.3. Unlike the proofs of security for GSW FHE or WW BARG, this analysis heavily relies on the formulation of *search* games.

In the following, we highlight some examples of the types of arguments that appear and illustrate how we use the various tools developed in Section 2.3; full details are given in Section 11. We focus on step (1) of the proof. We will assume some familiarity with the analysis of the encrypt-hash-and-BARG SNARG. Consider a cheating prover $\mathcal{P}$ of size t^* such that $\mathsf{Adv}_{(C_1,C_2)}[\mathcal{P}] \geq \frac{1}{t^*(\lambda)}$, where the search game (C_1, C_2) is as defined in Example 2.3 for instance $x^* \notin \mathcal{L}$. Suppose that the underlying FHE, SEH, and BARG constructions are $(t_{\mathsf{FHE}}, 1/t_{\mathsf{FHE}})$, $(t_{\mathsf{SEH}}, 1/t_{\mathsf{SEH}})$, and $(t_{\mathsf{BARG}}, 1/t_{\mathsf{BARG}})$ secure respectively.

Amplifying the success probability. As a first step, we need to *amplify* the success probability of $\mathcal{P}$ via *parallel repetition* from some inverse polynomial $\frac{1}{t^*(\lambda)}$ to at least $1 - \frac{1}{p(\lambda)}$ for some polynomial p. Using

the Error Reduction Lemma discussed in Section 2.3 with $R := O(t^* \log p)$, we can show that $\mathcal{P}^* = \mathcal{P}^{\times R}$ satisfies

$$\mathsf{Adv}_{(C_1^{\times R}, C_2^{\vee R})}[\mathcal{P}^*] \geq 1 - \left(1 - \frac{1}{2t^*(\lambda)}\right)^R \geq 1 - \frac{1}{p(\lambda)}.$$

This parallel repetition will allow us to obtain at least one CRS-proof pair (crs, π) such that the SNARG verifier accepts with probability $1 - \frac{1}{p(\lambda)}$. For simplicity of exposition, suppose that $R = 1$ for the remainder of this exposition (i.e. there is no parallel repetition).

Using FHE security. Next, we need to show that if one changes the CRS algorithm to generate FHE ciphertext ct' encrypting a different circuit E_{x^*}, the prover success probability does not diminish by too much. We formalize this as follows. One can view the EF-SNARG search game for x^* as (C_1, C_2) as in Example 2.3, and (C_1', C_2) as the modified game where the CRS is generated with ct' instead. By alluding to the $(t_{\mathsf{FHE}}, 1/_{\mathsf{FHE}})$-FHE indistinguishability, one can argue via reduction that the outputs of C_1, C_1' are $(t_{\mathsf{FHE}} - s, 1/t_{\mathsf{FHE}})$ indistinguishable, where s is an upper bound on the size of C_1 and C_1'. Now, if $\mathcal{P}$ has size at most $t_{\mathsf{FHE}} - s - |C_2|$, we can use Game Composition Lemma to argue that:

$$\mathsf{Adv}_{(C_1', C_2)}[\mathcal{P}] \geq \mathsf{Adv}_{(C_1, C_2)}[\mathcal{P}] - 1/t_{\mathsf{FHE}}.$$

Using SEH key indistinguishability. We now use this $\mathcal{P}$ to construct a local assignment generator for the circuit $\widehat{C}$, which is the circuit C augmented with the homomorphic evaluation of E on C. At a high level, $\mathsf{LocalGen}^{\mathcal{P}}(T)$ is constructed as follows: on input a set T of size at most loc,

- Generate a trapdoored CRS, where all of the hash keys are sampled such that the wire values in T are extractable. (If $|T| < \mathsf{loc}$, simply generate the key to be extractable on a dummy index 0.)
- Receive an accepting proof π^* from the prover $\mathcal{P}$.
- Extract the wire assignment to T using the trapdoor of the SEH hash.

We now sketch why $\mathsf{LocalGen}$ satisfies (t', ε')-non-signaling for some parameters t', ε' to be determined. For simplicity, we will only sketch the argument for sets $T_0 = \{t_1, \ldots, t_u\}$ and $T_1 = \{t_1, \ldots, t_{u+1}\}$ (which differ only by one element). Then, $\mathsf{LocalGen}(T_0)$ samples the CRS with $\mathsf{hk}_1, \ldots, \mathsf{hk}_u$ extractable on $t_1, \ldots, t_u$, and $\mathsf{LocalGen}(T_1)$ samples the CRS to be extractable on $\mathsf{hk}_1, \ldots \mathsf{hk}_u, \mathsf{hk}_{u+1}$ are extractable on $t_1, \ldots, t_u$. Let $\mathcal{H}_{\mathsf{SEH},i}$ denote the distribution that outputs an SEH key which is extractable on index i.

Consider the distributions $\mathcal{D}_b$ defined as follows:

- Sample all keys $\mathsf{hk}_1, \ldots, \mathsf{hk}_u, \mathsf{hk}_{u+2}, \ldots \mathsf{hk}_{\mathsf{loc}}$ with trapdoors as in $\mathsf{LocalGen}(T_0)$.
- If $b = 0$, output $\mathsf{hk}_{u+1} \leftarrow \mathcal{H}_{\mathsf{SEH},0}$. Else, sample $\mathsf{hk}_{u+1} \leftarrow \mathcal{H}_{\mathsf{SEH},t_{u+1}}$.
- Sample the rest of the CRS and query $\mathcal{P}$ on this CRS to obtain a proof π. If π is not an accepting proof, output $\perp$.
- Extract the wire assignment to T_0 using the trapdoors and output them.

It can be verified that $\mathcal{D}_0$ is identical to the distribution of $\mathcal{D}_{T_0,T_1,0}$, and $\mathcal{D}_1$ is identical to distribution of $\mathcal{D}_{T_0,T_1,1}$.[23] Moreover, one can write $\mathcal{D}_0 = R \circ \mathcal{H}_{\mathsf{SEH},0}$ and $\mathcal{D}_1 = R \circ \mathcal{H}_{\mathsf{SEH},t_{u+1}}$. By SEH key indistinguishability, we have that $\mathcal{H}_{\mathsf{SEH},0}$ and $\mathcal{H}_{\mathsf{SEH},t_{u+1}}$ are $(t_{\mathsf{SEH}}, 1/t_{\mathsf{SEH}})$-indistinguishable. Therefore, by applying the

[23] Again, these distributions are formally proved to be isomorphic (i.e. identical up to a permutation of the seed).

Reduction Lemma, one can argue that $\mathcal{D}_0$ and $\mathcal{D}_1$ are $(t_{\mathsf{SEH}} - s, 1/t_{\mathsf{SEH}})$-indistinguishable, where s is a bound on the size of $\mathcal{D}_b$. Therefore, $\mathcal{D}_{T_0,T_1,0} \approx_{t',\varepsilon'} \mathcal{D}_{T_0,T_1,1}$ for $t' \leq t_{\mathsf{SEH}} - s$ and $\varepsilon' \geq 1/t_{\mathsf{SEH}}$.

We can then extend the argument to demonstrate non-signaling of arbitrary pairs of sets T_0 and T_1 via a Hybrid Argument.

Using BARG indistinguishability and extractability. We now rely on BARG security to argue that $\mathsf{LocalGen}$ satisfies ε'-consistency for some parameter ε'. Fix a gate g in the circuit $\widehat{C}$, and consider the consistency game $\mathcal{G}_g$, and let $T = \{w_1, w_2, w_3\}$. We first consider $\mathsf{LocalGen}'$ which is identical to $\mathsf{LocalGen}$, except that it instead samples the trapdoored CRS to ensure that the BARG is extractable on index 'g' (recall that the batch-statement is proving a claim over the gates g of the circuit). By a simple invocation of the Reduction Lemma and the BARG CRS indistinguishability, it is easy to see that the outputs of $\mathsf{LocalGen}(T)$ and $\mathsf{LocalGen}'(T)$ are $(t_{\mathsf{BARG}} - s, 1/t_{\mathsf{BARG}})$ indistinguishable, where s is the size of $\mathsf{LocalGen}$. As shown in Section 2.3, one can invoke Adversary Indistinguishability Lemma of $\mathsf{LocalGen}$ to argue that

$$\mathsf{Adv}_{\mathcal{G}_g}[\mathsf{LocalGen}] \leq \mathsf{Adv}_{\mathcal{G}_g}[\mathsf{LocalGen}'] + 1/t_{\mathsf{BARG}}. \tag{2.3}$$

Let (C_1, C_2) be the EF-SNARG security game, and let C_1^* be the modified circuit which samples the CRS where the hash keys are extractable on $\{w_1, w_2, w_3\}$, and the BARG CRS to be extractable on index g.

We now claim that

$$\mathsf{Adv}_{\overline{\mathcal{G}}_g}[\mathsf{LocalGen}'] \geq \mathsf{Adv}_{(C_1^*,C_2)}[\mathcal{P}]. \tag{2.4}$$

Note that in the left-hand side, we are taking the *complement* of the game $\mathcal{G}_g$, where the output of $\mathcal{C}_2$ is flipped. Similar to the case of FHE indistinguishability, one can show via a hybrid argument and game indistinguishability that the RHS is lower bounded by $1 - \frac{1}{p(\lambda)} - \mathsf{poly}(1/t_{\mathsf{BARG}}, 1/t_{\mathsf{SEH}})$.

To show the above, we rely on reduction between search games, and the fact that if $\mathcal{P}$ outputs accepting proofs in $\mathsf{LocalGen}'$, we can prove in PV that the corresponding wire assignment from $\mathsf{LocalGen}'(T)$ must be consistent. We sketch the proof below.

- Suppose $\mathcal{P}$ in $\mathsf{LocalGen}'$ outputs an accepting SNARG proof $(h, \pi, \rho_{\mathsf{out}})$. In particular, π is an accepting BARG proof.
- By somewhere extractability of the BARG on gate g, one can extract $(\sigma_1, \sigma_2, \sigma_3, \rho_1, \rho_2, \rho_3)$ from π which is a valid witness for the instance g in the batch-statement. This means that $g(\sigma_1, \sigma_2) = \sigma_3$, and ρ_i are valid openings to the hash h for $i \in [3]$.
- Recall that $\mathsf{LocalGen}'(T)$ sampled $\mathsf{hk}_1, \mathsf{hk}_2, \mathsf{hk}_3$ to be extractable on wires w_1, w_2, w_3. By the somewhere extractability of the SEH, for $i \in [3]$, if there exists openings ρ_i certifying that the hash h_i opens to σ_i, then the extracted value from h_i must be equal to σ_i.
- Therefore, the extracted values satisfy $\sigma_3 = g(\sigma_1, \sigma_2)$.
- Hence, $\mathsf{LocalGen}'$ outputs $\{\sigma_1, \sigma_2, \sigma_3\}$ which satisfies gate g.

By proving the correctness of BARG and SEH extraction in PV, we complete the proof. Specifically, by combining Equation (2.3), Equation (2.4), and invoking the Game Complement Lemma, we have

$$\mathsf{Adv}_{\mathcal{G}_g}[\mathsf{LocalGen}] \leq (1 - \mathsf{Adv}_{(C_1^*,C_2)}[\mathcal{P}]) + 1/t_{\mathsf{BARG}} \leq \frac{1}{p(\lambda)} + \mathsf{poly}(1/t_{\mathsf{BARG}}, 1/t_{\mathsf{SEH}}).$$

In other words, $\mathsf{LocalGen}$ satisfies ε' consistency if we choose $\varepsilon' \geq \frac{1}{p(\lambda)} + \mathsf{poly}(1/t_{\mathsf{BARG}}, 1/t_{\mathsf{SEH}})$.

Part II
Assumption and Security Lifting Lemma

3 Preliminaries

3.1 Recap of Mathematical Logic

We assume basic familiarity with mathematical logic and bounded arithmetic; readers are referred to standard textbooks [Bus85, Kra95, Kra19] as well as the survey [Oli25] that covers more recent results. We provide a quick recap of basic concepts in logic.

- (*Logic*). Throughout the paper, we focus on standard classical first-order logic. A *language* (or *vocabulary*) is a set of names of constants, functions, and predicates. A *formula* in a language are formed from constants, functions symbols, and predicates from the language, variables, logical connectives (e.g., $\wedge, \vee, \neg$), and quantifiers (e.g., $\forall, \exists$). A *variable* in a formula is said to be *free* if it is not captured by any quantifier. A *sentence* is a formula with no free variable.

- (*Models*). Models of first-order logic is a tuple $\mathcal{M} = (\mathcal{U}, \mathcal{I}_c, \mathcal{I}_f, \mathcal{I}_P)$, where $\mathcal{U}$ is a set called *universe*, $\mathcal{I}$ is a mapping from constant symbols to elements in $\mathcal{U}$, $\mathcal{I}_f$ is a mapping from function symbols to functions over $\mathcal{U}$, and $\mathcal{I}_P$ is a mapping from predicates to relations over $\mathcal{U}$. With these mappings, we can decide whether a sentence is true or false over a model $\mathcal{M}$; $\mathcal{M} \models \varphi$ means that the sentence φ is true over $\mathcal{M}$. For a set of sentences Γ, we say $\Gamma \models \varphi$ if for every model $\mathcal{M}$ such that $\mathcal{M} \models \psi$ for every $\psi \in \Gamma$, it must satisfy $\mathcal{M} \models \varphi$.

- (*Proofs*). We say $\Gamma \vdash \varphi$ if there is a proof of φ from Γ; we may fix any sound and complete proof system such that $\Gamma \vdash \varphi$ if and only if $\Gamma \models \varphi$.

- (*Theory*). A *theory* $\mathcal{T}$ in a language $\mathcal{L}$ is a set of first-order sentences in the language. Each sentence $\varphi \in \mathcal{T}$ is said to be an *axiom*. We say that ψ is a *theorem* of $\mathcal{T}$ if $\mathcal{T} \vdash \psi$. The *standard model* of a theory is the model that it is intended to capture; in particular, a theory should be *sound*, namely it satisfies its standard model.

- (*Extensions*). Let $\mathcal{T}_1, \mathcal{T}_2$ be theories over languages $\mathcal{L}_1, \mathcal{L}_2$, respectively. The theory $\mathcal{T}_2$ is said to be an *extension* of $\mathcal{T}_1$ if $\mathcal{L}_1 \subseteq \mathcal{L}_2$, and every $\mathcal{T}_1$-provable sentence in the language of $\mathcal{L}_1$ is also provable in $\mathcal{T}_2$. It is said to be a *conservative extension* if every $\mathcal{T}_2$-provable sentence in the language of $\mathcal{L}_1$ is also provable in $\mathcal{T}_1$. In other words, a conservative extension $\mathcal{T}_2$ of $\mathcal{T}_1$ does not prove any *new* theorem that can be stated in the original language $\mathcal{L}_1$.

3.2 Jeřábek's Theory APC_1

We will work with the first-order theory APC_1 defined by Jeřábek [Jeř07a] that supports approximate counting as the base theory for formalization.

Notation. Following standard set-theoretic notation, a number a is considered identical to the set $\{0, 1, \ldots, a-1\}$. We define $[a] := \{1, 2, \ldots, a\}$. We use $x \in \mathsf{Log}$ to denote that x is an abbreviation of $|X|$, i.e., the length of another variable. In other words, algorithms running in time $\mathsf{poly}(x)$ (or equivalently,

$\mathsf{poly}(|X|)$) are considered as feasible algorithms. Similarly, we use $x \in \mathsf{LogLog}$ to denote that x is an abbreviation of $||X||$ for another variable X. We use $f : \mathsf{Log} \to \mathsf{Log}$ to denote that f is a function where for every $x \in \mathsf{Log}$, $f(x) \in \mathsf{Log}$.

The Theory PV and PV_1. PV is a bounded theory defined by Cook [Coo75] that aims to characterize *polynomial-time reasoning*. Loosely speaking, it (only) contains polynomial-time algorithms as function symbols (defined via a variant of Cobham's recursion-theoretic characterization [Cob65]), and allows (only) induction over polynomial-time decidable predicates. The definition of PV is tedious and we refer interested readers to, e.g., [Kra19, Chapter 12] and [Li25].

The theory PV is an *equational* theory (also known as a "logic-free" theory) that uses equality as the only predicate symbol. The theory PV_1 is the extension of PV to the classical first-order logic that allows propositional connectives (e.g. $\wedge, \vee, \neg$) and quantifiers ($\forall$ and $\exists$). This extension allows formalization of more complicated mathematical statements. On the other hand, it is a *conservative extension* (see, e.g., [Bus85]), which means that any equation that is provable in PV_1 is also provable in PV – the extension does not make the theory any stronger in terms of proving equations.

The Theory APC_1. APC_1 is the first-order theory that extends the theory PV_1 with the following axiom $\mathsf{dWPHP}(\mathsf{PV})$ (stands for dual Weak Pigeonhole Principle):

> For every $c \in \mathsf{Log}$, a, and circuit C that computes a function $C : ac \to a(c+1)$, then there exists $y \in a(c+1)$ such that for every $x \in ac$, $C(x) \neq y$.

In other words, every polynomial-size function whose co-domain is slightly larger than its domain must not be surjective. Note that this is said to be *weak* as a uniformly random $y \in [a(c+1)]$ will be outside of the range of C with non-negligible (more precisely, $1/c$) probability.

The main motivation to introduce $\mathsf{dWPHP}(\mathsf{PV})$ as an axiom is to formalize approximate counting, which is not supported by PV_1. To explain Jeřábek's approximate counting mechanism, we need to introduce a few abbreviations.

A set X is said to be a *bounded definable set* if $X = \{x < a \mid C(x) = 1\}$, where C is a Boolean circuit that defines X. We use $x \in X$ to denote $x < a \wedge C(x) = 1$, and $X \subseteq b$ to denote $\forall x \in X\ x < b$. Note that bounded definable sets are *not* objects in the theory APC_1, but an abbreviation in the meta-theory for the simplicity of presentation. For two bounded definable sets $X \subseteq a$ and $Y \subseteq b$, we define

$$X \times Y := \{ay + x \mid x \in X, y \in Y\} \subseteq ab,$$
$$X \uplus Y := X \cup \{y + a \mid y \in Y\} \subseteq a + b.$$

We say that $C : X \to Y$ if C is a circuit from X to Y, i.e., for every $x \in X$, $C(x) \in Y$. We use $C : X \hookrightarrow Y$ to denote a circuit $C : X \to Y$ that is injective, i.e., for distinct $x_1, x_2 \in X$, $C(x_1) \neq C(x_2)$. We use $C : X \twoheadrightarrow Y$ to denote that C is onto, i.e., for every $y \in Y$, there is an $x \in X$ such that $C(x) = y$. Note that it does not necessarily imply $C : X \to Y$; there can be some $x \in X$ such that $C(x) \notin Y$.

Definition 3.1 (in APC_1)**.** Let $X, Y \subseteq 2^n$ be definable sets, $\varepsilon \leq 1$, we say that X is ε-approximately smaller than Y, denoted by $X \lesssim_\varepsilon Y$, if there exists a circuit G and $v \neq 0$ such that

$$G : v \times (Y \uplus \varepsilon 2^n) \twoheadrightarrow v \times X.$$

In plain words, we say that a set X is ε-approximately smaller than Y if there is an onto mapping from Y to X up to an small padding (i.e. $\varepsilon 2^n$). The variable $v \neq 0$ is introduced for technical reasons, and we refer readers to [Jeř07a] for more details.

Definition 3.2 (in APC_1)**.** We say that X and Y are ε-approximately of equal size, denoted by $X \approx_\varepsilon Y$, if $X \lesssim_\varepsilon Y$ and $Y \lesssim_\varepsilon X$. In particular, we say that X is ε-approximately of size s if $X \approx_\varepsilon s$.

Proposition 3.3 ([Jeř07a, Lemma 2.10])**.** *Let $X, Y, X', Y', Z \subseteq 2^n$ and $W, W' \subseteq 2^n$ be bounded definable sets, and $\varepsilon, \delta \leq 1$. The following statements are provable in PV_1.*

(1) If $X \lesssim_\varepsilon Y, \varepsilon \leq \delta$, then $X \lesssim_\delta Y$.
(2) If $X \lesssim_0 Y$, then $X \lesssim_\varepsilon Y$.
(3) If $X \lesssim_\varepsilon Y$, $Y \lesssim_\delta Z$, then $X \lesssim_{\delta+\varepsilon} Z$.
(4) If $X \lesssim_\varepsilon X'$, $Y \lesssim_\delta Y'$, and X', Y' are separable by the set W (i.e., $X' \subseteq W$ and $Y' \subseteq 2^n \setminus W$), then $X \cup Y \lesssim_{\varepsilon+\delta} X' \cup Y'$.
(5) If $X \lesssim_\varepsilon X', W \lesssim_\delta W'$, then $X \times W \lesssim_{\varepsilon+\delta+\varepsilon\delta} X' \times W'$.

Lemma 3.4 ([Jeř07a, Lemma 2.11])**.** *Let $X, Y \subseteq 2^n$ be bounded definable sets, $s, t, u \leq 2^n$, $\varepsilon, \delta, \eta, \xi \leq 1$, $\xi^{-1} \in \mathsf{Log}$. The following statements are provable in APC_1.*

(1) There exists $s \leq 2^n$ such that $X \approx_\xi s$.
(2) $s \lesssim_\varepsilon X \lesssim_\delta t$ implies $s \leq t + (\varepsilon + \delta + \xi) \cdot 2^n$.
(3) $X \lesssim_\xi Y$ or $Y \lesssim_\xi X$.
(4) $X \lesssim_\varepsilon Y$ implies $2^n \setminus Y \lesssim_{\varepsilon+\xi} 2^n \setminus X$.
(5) $X \approx_\varepsilon s, Y \approx_\delta t, X \cap Y \approx_\eta u$ imply $X \cup Y \approx_{\varepsilon+\delta+\eta+\xi} s + t - u$.

Item (1) of Lemma 3.4, in plain words, means that we can approximately count every bounded definable set in the theory APC_1. Item (2) asserts the consistency of size comparison. Item (3) asserts that the size of any two bounded definable sets can be compared approximately. Item (4) shows that size comparison is consistent under complementation. Finally, item (5) is a form of inclusion-exclusion principle for two bounded definable sets, which, in particular, implies the union bound (as $X \cap Y \approx_\eta u > 0$).

We defer the discussion of other approximate counting principles to subsequent sections.

3.3 Succinct Non-Interactive Arguments (SNARGs)

In this section, we will recap the definition of a succinct non-interactive argument, or SNARG.

A SNARG system for an NP relation $\mathcal{M}$ consists of polynomial-time algorithms $(\mathsf{Gen}, \mathcal{P}, \mathcal{V})$ with the following syntax:

- The randomized setup algorithm Gen takes as input a security parameter $\lambda \in \mathbb{N}$ and an input length n, both in unary, and outputs a pair of common reference string crs.
- The prover algorithm $\mathcal{P}$ takes as input the common reference string crs an input $x \in \{0,1\}^n$ and its associated witness $w \in \{0,1\}^m$, and outputs a proof π.
- The verifier algorithm $\mathcal{V}$ takes as input the common reference string crs, an input $x \in \{0,1\}^n$ and a proof π. It outputs a bit indicating if it accepts or rejects.

Definition 3.5 (SNARG)**.** A triple of algorithms $(\mathsf{Gen}, \mathcal{P}, \mathcal{V})$ is a SNARG system for a Turing machine $\mathcal{M}$ if the following hold:

Completeness. For every $\lambda, n \in \mathbb{N}$ and every $x \in \{0,1\}^n$ and $w \in \{0,1\}^m$ such that $\mathcal{M}(x,w) = 1$,

$$\Pr\left[\mathcal{V}(\mathsf{crs}, x, \pi) = 1 \ : \ \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^n), \\ \pi \leftarrow \mathcal{P}(\mathsf{crs}, x, w) \end{array} \right] = 1.$$

Efficiency. The length of crs is $\mathsf{poly}(\lambda, n, m)$. The length of a proof π is $\mathsf{poly}(\lambda, \log n, \log m)$. The runtime of $\mathcal{V}$ is $\mathsf{poly}(|\mathsf{crs}|, |\pi|, n, m)$.

Non-adaptive Soundness. For every $x \in \{0,1\}^n$ where $x \notin \mathcal{L}_\mathcal{M}$ and every poly-size adversary Adv, there is a negligible function μ such that

$$\Pr\left[\mathcal{V}(\mathsf{crs}, x, \pi^*) = 1 \ : \ \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^n), \\ \pi^* \leftarrow \mathsf{Adv}(\mathsf{crs}) \end{array} \right] \leq \mu(\lambda).$$

Remark 3.6. Recall that given a SNARG, one can generically improve the runtime of the verifier to $\mathsf{poly}(|\pi|, \lambda, \log n, \log m) \cdot n$ by applying a RAM delegation protocol or SNARG for P. In more detail, one can compute a hash of the crs, $\mathsf{crs}_\mathcal{V}$, and the new verifier $\mathcal{V}'(\mathsf{crs}_\mathcal{V}, x, \pi)$ can then use a RAM delegation protocol to verify that $\mathcal{V}(\mathsf{crs}, x, \pi)$ would have indeed accepted in quasilinear time (see for example [WW24, Remark 2.7] for more details). Note that we do not have to prove that this transformation is secure in our theory since under our assumption, the resulting SNARG for NP will be sound in standard (CRS) model. Therefore, for simplicity, we will focus on constructing a SNARG with this weaker verifier efficiency.

3.4 Learning With Error

We now recall the learning with errors assumption (LWE) [Reg05]. Let $\lambda \in \mathbb{N}$ be the security parameter. Given a finite set W, $\mathcal{U}(W)$ denotes the uniform distribution over W. Let $\mathcal{D}_{\mathbb{Z},\sigma}$ be the discrete Gaussian distribution with standard deviation σ. The LWE assumption is defined as follows.

Definition 3.7 (Learning With Errors (LWE))**.** Given $n, m, q \in \mathbb{N}$ and $\sigma > 0$ with $n, m \in \mathsf{poly}(\lambda)$, $q \leq 2^{\sqrt{n}}$, and $\sigma \geq 2\sqrt{n}$, the LWE assumption $\mathsf{LWE}_{n,m,q,\sigma}$ asserts that

$$(\mathbf{A}, \mathbf{s}^\mathsf{T}\mathbf{A} + \mathbf{e}) \approx_c (\mathbf{A}, \mathbf{b}),$$

where $\mathbf{s} \leftarrow \mathcal{U}(\mathbb{Z}_q^n)$, $\mathbf{A} \leftarrow \mathcal{U}(\mathbb{Z}_q^{n \times m})$, $\mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^m$, and $\mathbf{b} \leftarrow \mathcal{U}(\mathbb{Z}_q^m)$.

Remark 3.8. It was shown in [Reg05, Pei09] that when $\sigma > 2\sqrt{n}$, the $\mathsf{LWE}_{n,m,q,\sigma}$ assumption is at least as hard GapSVP_γ and SIVP_γ for $\gamma = \tilde{O}(nq/\sigma)$. On the other hand, the current best known polynomial-time attacks ([LLL82] and its descendants) against GapSVP_γ and SIVP_γ works for $\gamma = 2^{\Theta(n \log\log n / \log n)}$. Picking modulus $q \leq 2^{n^{1-\delta}}$ for some constant $\delta > 0$ ensures that the LWE assumption does not fall in the above efficiently breakable regime.

3.5 SXDH

The Symmetric eXternal Diffie-Hellman (SXDH) assumption states that the Decisional Diffie-Hellman (DDH) problem is hard in both source groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of a bilinear group.

Definition 3.9 (SXDH)**.** Let $\lambda \in \mathbb{N}$ be the security parameter. Let $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, g_T, e) \leftarrow \mathsf{GroupGen}(1^\lambda)$ be a bilinear group generator that on input 1^λ outputs the description of groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ of prime order p along with generators g_1, g_2, g_T and an efficient bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. The SXDH assumption $\mathsf{SXDH}_{\mathsf{GroupGen}}$ with respect to $\mathsf{GroupGen}$ states that

$$(g_1^{u_1}, g_1^{v_1}, g_1^{u_1 v_1}, g_2^{u_2}, g_2^{v_2}, g_2^{u_2 v_2}) \approx_c (g_1^{u_1}, g_1^{v_1}, g_1^{w_1}, g_2^{u_2}, g_2^{v_2}, g_2^{w_2})$$

where $u_1, v_1, w_1, u_2, v_2, w_2 \leftarrow \mathcal{U}(\mathbb{Z}_p)$.

4 The Bounded Theory and Our Unprovability Assumption

In this section, we define our bounded theory $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$. It is built from the first-order theory APC_1 defined by Jeřábek [Jeř07a] and additional non-logical axioms: $\mathsf{prBPP} = \mathsf{prP}$, hardness of the standard Learning With Error, and several other cryptographic assumptions. We note that, as the non-logical axioms are not known to be true, our theory is sound only if all these assumptions are true.

4.1 Nonlogical Axiom 1: $\mathsf{prBPP} = \mathsf{prP}$

The first non-logical axiom is a standard derandomization assumption: every probabilistic polynomial-time algorithm can be derandomized with at most a polynomial time overhead. For convenience, we formalize the assumption in the equivalent form that the search version of Circuit Acceptance Probability Problem (CAPP) is approximately computable in polynomial-time.

Definition 4.1. Search CAPP is defined as the following search problem: Given a circuit $C : \{0,1\}^n \to \{0,1\}$ and 1^t, output any number within $|\{x \in \{0,1\}^n \mid C(x) = 1\}| \pm (1/t) \cdot 2^n$.

Proposition 4.2 (folklore; see, e.g., [Gol11])**.** *Search CAPP admits a deterministic polynomial-time algorithm if and only if* $\mathsf{prBPP} = \mathsf{prP}$.

Assume that $\mathsf{prBPP} = \mathsf{prP}$ and $\mathsf{capp}(\cdot,\cdot)$ be a PV-function symbol corresponding to the polynomial-time algorithm for Search CAPP. The axiom $\mathsf{prBPP} = \mathsf{prP}$ is defined as follows:

Definition 4.3 (in APC_1)**.** We use "$\mathsf{prBPP} = \mathsf{prP}$" as the abbreviation of the following statement in the language of APC_1: For every circuit $C : \{0,1\}^n \to \{0,1\}$, $t \in \mathsf{Log}$, let $X := \{x \in \{0,1\}^n \mid C(x) = 1\} \subseteq \{0,1\}^n$, if $X \approx_\varepsilon s$ for some $\varepsilon < 1$ and $s \leq 2^n$, then

$$|\mathsf{capp}(C, 1^t) - s| \leq (\varepsilon + t^{-1}) \cdot 2^n.$$

In plain words, "$\mathsf{prBPP} = \mathsf{prP}$" states that the algorithm $\mathsf{capp}(\cdot,\cdot)$ is approximately consistent with the size of a set, which is formally defined in Definition 3.2. Effectively, the function symbol $\mathsf{capp}(\cdot,\cdot)$ can be used to perform approximate counting. We may abuse the notation to write $\mathsf{capp}_\xi(C)$ as an abbreviation of $\mathsf{capp}(C, 1^{\xi^{-1}})$, and write $\mathsf{size}_\xi(X)$ as an abbreviation of $\mathsf{capp}_\xi(C)$ for a bounded definable set $X := \{x \in \{0,1\}^n \mid C(x) = 1\}$.

In addition, we use $\Pr_\xi[X]$ (resp. $\Pr_\xi[C]$) as an abbreviation of $\mathsf{size}_\xi(X)/2^n$ (resp. $\mathsf{capp}_\xi(C)/2^n$). As $\mathsf{capp}(\cdot,\cdot)$ is a PV function symbol, the induction principle over PV terms that involve $\Pr_\xi[\cdot]$ or $\mathsf{size}_\xi(\cdot)$ is admissible in PV_1.

Applications of the algorithm $\mathsf{capp}(\cdot,\cdot)$**.** The following lemma is an example to use $\mathsf{capp}(\cdot,\cdot)$ for approximate counting. It formalizes the *hybrid argument*, a basic technique in cryptographic security analysis.

Lemma 4.4 (hybrid argument)**.** *The following statements are provable in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$*: Let* $n, m \in \mathsf{Log}$ *and* $X_0, X_1, \ldots, X_m \subseteq \{0,1\}^n$ *be a list of bounded definable sets, and* $\delta^{-1} \in \mathsf{Log}$*. Let* $\varepsilon_0, \ldots, \varepsilon_{m-1} < 1$*,* $\varepsilon := \sum_{i=0}^{m-1} \varepsilon_i$*.*

Suppose that for every $i < m$*,* $|\Pr_\delta[X_i] - \Pr_\delta[X_{i+1}]| \le \varepsilon_i$*. Then* $|\Pr_\delta[X_m] - \Pr_\delta[X_0]| \le \varepsilon$*.*

Proof. We argue in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. We prove by induction on $i \le m$ that

$$|\Pr_\delta[X_i] - \Pr_\delta[X_0]| \le \varepsilon_0 + \cdots + \varepsilon_{i-1}.$$

This induction principle is admissible in PV_1, as $\Pr_\delta$ is implemented by the PV-term $\mathsf{capp}(\cdot,\cdot)$.

The base case is trivial. Suppose that the inequality holds for $i < m$, then

$$\begin{aligned} &|\Pr_\delta[X_{i+1}] - \Pr_\delta[X_0]| \\ \le &|\Pr_\delta[X_{i+1}] - \Pr_\delta[X_i]| + |\Pr_\delta[X_i] - \Pr_\delta[X_0]| \\ \le &\varepsilon_i + (\varepsilon_0 + \cdots + \varepsilon_{i-1}) = \varepsilon_0 + \cdots + \varepsilon_i, \end{aligned}$$

where the third line follows from the assumption and the induction hypothesis. This completes the proof, as for $i = m$, we have $|\Pr_\delta[X_m] - \Pr_\delta[X_0]| \le \varepsilon$. □

Remark 4.5. The universal quantification over $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$ in above propositions is necessary as $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$: It formalizes the fact that whatever high inverse-polynomial precision we use for approximate counting, the advantage of the adversary is always at most roughly ε.

Note that in the standard model, $\forall \delta^{-1}, \beta^{-1} \in \mathsf{Log}\ \Pr_\delta[C] \le p + \delta + \beta$ is equivalent to say that $\Pr[C] \le p$, as $\delta, \beta > 0$ can be arbitrarily small. Here, the error term δ accounts for the error of the function $\Pr_\delta[\cdot]$ itself (as formalized by the axiom “$\mathsf{prBPP} = \mathsf{prP}$”), while an addition error term β is added for the technicality that APC_1 only supports approximate counting. In more details, quantifying over $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$ allows us to use algorithms with runtime $\mathsf{poly}(|\delta^{-1}|, |\beta^{-1}|)$ in the proof, which is sometimes necessary. This is a standard approach in formalizing approximate counting in bounded arithmetic, see, e.g., [Jeř07a] for more details.

Probabilistic Circuits. We follow the convention to model adversaries of cryptographic primitives as *probabilistic circuits*. Let $n, m, r \in \mathsf{Log}$, a probabilistic circuit of input length n, output length m, and seed length r is a deterministic circuit $C : \{0,1\}^n \times \{0,1\}^r \to \{0,1\}^m$, where the first second part of its input is to take the random seed. In particular, a deterministic circuit can be viewed as a probabilistic circuit with seed length 0. We use $C(x; \mathsf{sd})$ to denote the evaluation of the circuit C on input x and seed sd.

We introduce the following notation for simplicity of presentation:

- (*Composition*). Let C_1, C_2 be probabilistic circuits such that the output length of C_2 is equal to the input length of C_1. We use $C_1 \circ C_2$ to denote the probabilistic circuit that, given input x, samples seed $\mathsf{sd}_1, \mathsf{sd}_2$ for C_1 and C_2 independently, and outputs $C_1(C_2(x; \mathsf{sd}_2); \mathsf{sd}_1)$.

4.2 Distributions, Indistinguishability, and Hybrid Arguments

Before diving into our other non-logical axioms that correspond to cryptographic assumptions, we first clarify the formalization of a few basic concepts — distributions, indistinguishability, and hybrid arguments — in the theory $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. These notions could “wrap up” the technicality in dealing with probability defined via approximate counting, which will greatly simplify our formalizations.

Distributions. A distribution $\mathcal{D}$ is generally formalized by the circuit $C : \{0,1\}^\ell \to \{0,1\}^n$ that samples the distribution, namely, $\mathcal{D} = C(\mathcal{U}_\ell)$. Distributions that are not sampled by explicit circuits are not allowed. We provide a couple of definitions of identical and (statistically) close distributions.

Definition 4.6 (identical distributions, in PV_1)**.** Two distributions $\mathcal{D}_1, \mathcal{D}_2$ defined by circuits C_1, C_2 are said to be *identical*, denoted by $\mathcal{D}_1 \equiv \mathcal{D}_2$, if the circuits are functionally equivalent.

Definition 4.7 (isomorphic distributions, in PV_1)**.** Two distributions $\mathcal{D}_1, \mathcal{D}_2$ defined by circuits $C_1, C_2 : \{0,1\}^\ell \to \{0,1\}^n$ are said to be *isomorphic*, denoted by $\mathcal{D}_1 \cong \mathcal{D}_2$, if there are circuits $f, g : \{0,1\}^\ell \to \{0,1\}^\ell$ such that:

- $f(g(x)) = g(f(x)) = x$ for every $x \in \{0,1\}^\ell$.
- $C_1 \circ f$ and C_2 are functionally equivalently, i.e., $C_1(f(x)) = C_2(x)$ for every $x \in \{0,1\}^\ell$.

Note that the second bullet in the definition of isomorphic distributions is equivalent to that C_1 and $C_2 \circ g$ are functionally equivalent. Moreover, by defining $f(x) = g(x) = x$, it immediately follows that:

Proposition 4.8. PV_1 *proves that two identical distributions are also isomorphic.*

Definition 4.9 (almost identical distributions, in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$)**.** Let $\mathcal{D}_1, \mathcal{D}_2$ be distributions defined by circuits $C_1, C_2 : \{0,1\}^\ell \to \{0,1\}^n$, and $\varepsilon \in (0,1)$. Let $T_{C_1,C_2} : \{0,1\}^\ell \to \{0,1\}$ be the circuit that given $x \in \{0,1\}^\ell$, outputs 1 if and only if $C_1(x) \neq C_2(x)$. Then $\mathcal{D}_1$ is said to be *ε-almost identical* to $\mathcal{D}_2$, denoted by $\mathcal{D}_1 \approx_\varepsilon \mathcal{D}_2$, if for any $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$, $\Pr_\delta[T_{C_1,C_2}] \leq \delta + \beta + \varepsilon$.

Definition 4.10 (almost isomorphic distributions, in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$)**.** Let $\mathcal{D}_1, \mathcal{D}_2$ be distributions defined by circuits C_1, C_2, and $\varepsilon \in [0,1)$. We say that $\mathcal{D}_1$ is *ε-almost isomorphic* to $\mathcal{D}_2$, denoted by $\mathcal{D}_1 \simeq_\varepsilon \mathcal{D}_2$, if there are distributions $\mathcal{D}'_1, \mathcal{D}'_2$ defined by circuits C'_1, C'_2 such that

- $\mathcal{D}_1 \cong \mathcal{D}'_1$ and $\mathcal{D}_2 \cong \mathcal{D}'_2$;
- $\mathcal{D}'_1 \approx_\varepsilon \mathcal{D}'_2$.

Proposition 4.11. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves that two identical (resp. isomorphic) distributions are* 0*-almost identical (resp. isomorphic).*

Proposition 4.12. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves that for any $\varepsilon \in [0,1]$, two ε-almost identical distributions are ε-almost isomorphic.*

Indistinguishability. We will then define the *computational indistinguishability* of the distributions. We focus on non-uniform adversary model, i.e., the adversary is a deterministic circuit.

Definition 4.13 (indistinguishablity, in $\mathsf{APC} + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$)**.** Let $\ell, t, n \in \mathsf{Log}$, $\varepsilon \in [0,1)$, and $\mathcal{D}_1, \mathcal{D}_2$ be distributions defined by circuits $C_1, C_2 : \{0,1\}^\ell \to \{0,1\}^n$. We say that $\mathcal{D}_1$ is (t,ε)-indistinguishable to $\mathcal{D}_2$, denoted by $\mathcal{D}_1 \approx_{t,\varepsilon} \mathcal{D}_2$, if the following holds: For every $r \leq t$ and probabilistic circuit $\mathcal{A}$:

$\{0,1\}^n \times \{0,1\}^r \to \{0,1\}$ of size at most t, let $T_{\mathcal{A},i} : \{0,1\}^{\ell+r} \to \{0,1\}$ be the circuit $\mathcal{A} \circ C_i$, i.e., given $x \in \{0,1\}^\ell$ and $\mathsf{sd} \in \{0,1\}^r$, outputs $\mathcal{A}(C_i(x); \mathsf{sd})$. Then for every $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$,

$$\left|\Pr_\delta[T_{\mathcal{A},1}] - \Pr_\delta[T_{\mathcal{A},2}]\right| \leq 2\delta + \beta + \varepsilon.$$

The following lemma shows that if two distributions are almost isomorphic, they are also indistinguishable. The proof of the lemma is standard but tedious and is deferred to Section 7.4.

Lemma 4.14 (Isomorphism Lemma). *The following is provable in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. *Let* $\ell, n, t \in \mathsf{Log}$, $\varepsilon \in [0,1)$, *and* $\mathcal{D}_1, \mathcal{D}_2$ *be distributions defined by circuits* $C_1, C_2 : \{0,1\}^\ell \to \{0,1\}^n$. *If* $\mathcal{D}_1 \simeq_\varepsilon \mathcal{D}_2$, *then* $\mathcal{D}_1 \approx_{t,\varepsilon} \mathcal{D}_2$.

Hybrid Argument. We will need the standard hybrid argument: For a sequence of distributions $\mathcal{H}_0, \mathcal{H}_1, \ldots, \mathcal{H}_n$ such that $\mathcal{H}_i \approx_{t,\varepsilon} \mathcal{H}_{i+1}$, $H_0 \approx_{t,\varepsilon n} \mathcal{H}_n$. Formally:

Lemma 4.15 (Hybrid Argument). *The following is provable in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. *Let* $n, \ell, t \in \mathsf{Log}$, $\varepsilon_0, \ldots, \varepsilon_{n-1} > 0$, $\varepsilon := \sum_{i=0}^{n-1} \varepsilon_i$, $C_0, C_1, \ldots, C_n : \{0,1\}^\ell \to \{0,1\}^n$ *be a sequence of circuits, and* $\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_n$ *be the distributions defined by the circuits. Suppose that for every* $i < n$, $\mathcal{D}_i \approx_{t,\varepsilon_i} \mathcal{D}_{i+1}$, *then* $\mathcal{D}_0 \approx_{t,\varepsilon} \mathcal{D}_n$.

Proof. We argue in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. Fix $n, \ell, t \in \mathsf{Log}$, $\varepsilon > 0$, and circuits $C_0, \ldots, C_n$. Let $\mathcal{D}_i$ be the distribution defined by C_i. Suppose that for every $i \in [n]$, $\mathcal{D}_{i-1} \approx_{t,\varepsilon} \mathcal{D}_i$. We will prove that $\mathcal{D}_0 \approx_{t,\varepsilon} \mathcal{D}_n$.

Suppose, towards a contradiction, that there is a probabilistic circuit $\mathcal{A}$ of size at most t and $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$ such that

$$\left|\Pr_\delta[T_{\mathcal{A},0}] - \Pr_\delta[T_{\mathcal{A},n}]\right| > 2\delta + \beta + \varepsilon,$$

where $T_{\mathcal{A},i}(x, \mathsf{sd}) := \mathcal{A}(C_i(x); \mathsf{sd})$. Let $\eta^{-1} \in \mathsf{Log}$ be determined later. By Precision Consistency of CAPP, we have

$$\left|\Pr_\eta[T_{\mathcal{A},0}] - \Pr_\eta[T_{\mathcal{A},n}]\right| > (\beta - 2\eta) + \varepsilon.$$

By Lemma 4.4, there exists an $i < n$ such that

$$\left|\Pr_\eta[T_{\mathcal{A},i}] - \Pr_\eta[T_{\mathcal{A},i+1}]\right| > \frac{\beta - 2\eta}{n} + \varepsilon_i \geq 3\eta + \varepsilon_i,$$

where the last inequality holds if we set $\eta := \beta/(20n)$. This leads to a contradiction to $\mathcal{D}_i \approx_{t,\varepsilon_i} \mathcal{D}_{i+1}$ and concludes the lemma. □

4.3 Nonlogical Axiom 2: Hardness of Learning with Error

The second axiom we will use is the standard hardness assumption for LWE, as defined in Definition 3.7. To start with, we first define (in English) a parameterized version of LWE, where the size and advantage of the adversary is fixed.

Definition 4.16 ((t,ε)-Learning With Errors (LWE)). Given $n, m, q, t \in \mathbb{N}$ and $\sigma, \varepsilon > 0$, the LWE assumption $\mathsf{LWE}^{t,\varepsilon}_{n,m,q,\sigma}$ asserts that for every t-size adversary $\mathcal{A}$,

$$|\Pr[\mathcal{A}(\mathbf{A}, \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal) = 1] - \Pr[\mathcal{A}(\mathbf{A}, \mathbf{b}^\intercal) = 1]| \leq \varepsilon,$$

where $\mathbf{s} \leftarrow \mathcal{U}(\mathbb{Z}_q^n)$, $\mathbf{A} \leftarrow \mathcal{U}(\mathbb{Z}_q^{n\times m})$, $\mathbf{e} \leftarrow \mathcal{D}^m_{\mathbb{Z},\sigma}$, and $\mathbf{b} \leftarrow \mathcal{U}(\mathbb{Z}_q^m)$.

In other words, the distributions $(\mathbf{A}, \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal)$ and $(\mathbf{A}, \mathbf{b}^\intercal)$ are (t,ε)-indistinguishable.

Parameterized version of LWE. As described in Section 4.1, we will use the approximate counting function provided by the axiom "$\mathsf{prBPP} = \mathsf{prP}$" to formalize approximate counting. We first formalize the assumption for fixed parameters $n, m, q, \sigma, t, \varepsilon$.

Definition 4.17 (Parameterized LWE, in $\mathsf{APC}_1 + $ "$\mathsf{prBPP} = \mathsf{prP}$")**.** Let $n, m, t, \sigma \in \mathsf{Log}$, $q \geq 1$, and $\varepsilon > 0$. Let $\mathsf{Gen}_0, \mathsf{Gen}_1$ be the circuits that take an ℓ-bit input and sample $(\mathbf{A}, \mathbf{s}^\intercal \mathbf{A} + \mathbf{e}^\intercal)$ and $(\mathbf{A}, \mathbf{b}^\intercal)$, respectively, where $\mathbf{s} \leftarrow \mathcal{U}(\mathbb{Z}_q^n)$, $\mathbf{A} \leftarrow \mathcal{U}(\mathbb{Z}_q^{n \times m})$, $\mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^m$, and $\mathbf{b} \leftarrow \mathcal{U}(\mathbb{Z}_q^m)$.

Let $\mathcal{D}_0, \mathcal{D}_1$ be the distribution defined by $\mathsf{Gen}_0, \mathsf{Gen}_1$, respectively. Then $\mathsf{LWE}_{n,m,q,\sigma}^{t,\varepsilon}$ is defined as the formula $\mathcal{D}_0 \approx_{t,\varepsilon} \mathcal{D}_1$.

Remark 4.18 (Binary vs unary encoding of parameters)**.** Note that here, as we have $n, m, t, \sigma \in \mathsf{Log}$ encoded in unary, numbers such as 2^n and 2^σ exist. In contrast, $q \geq 1$ is encoded in binary, so we cannot assume that 2^q exists. This distinction is deliberate, as we will eventually set q to be exponential in n in the FHE construction.

Remark 4.19 (How to sample the distributions)**.** For simplicity, we always assume that q is a power of two, so that it is easy to sample uniformly random vectors and matrices in $\mathbb{Z}_q$ from random seeds in binary.[24] The discrete Gaussian distribution $\mathcal{D}_{\mathbb{Z},\sigma}$ over $\mathbb{Z}$ *cannot* be directly formalized in the theory. Nevertheless, we formalize it as the distribution defined by a circuit (see, e.g., [GPV08]) that samples a distribution that is statistically close to $\mathcal{D}_{\mathbb{Z},\sigma}^m$.

In addition, we assume that the circuit sampling $\mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^m$ will never output $\mathbf{e}$ such that $\|\mathbf{e}\|_\infty \geq \sigma\sqrt{(n+1)w}$. This is without loss of generality (in the standard model), as the probability that $\|\mathbf{e}\|_\infty \geq \sigma\sqrt{(n+1)w}$ is negligible.

Asymptomatic version of LWE. We move on to describe the asymptomatic version of the LWE assumption. Following the standard approach in bounded arithmetic, we formalize the asymptomatic assumption as an infinite *set of sentences*, instead of a single sentence.

The asymptotic version of LWE that we will use it the following: For every PV functions taking unary-encoded $\lambda \in \mathsf{Log}$ such that $n(\lambda), m(\lambda), t(\lambda) \in \mathsf{poly}(\lambda)$, $q(\lambda) \leq 2^{\sqrt{n(\lambda)}}$,[25] and $\sigma(\lambda) \geq 2\sqrt{n(\lambda)}$, there is a negligible function $\varepsilon(\lambda)$ such that the LWE assumption holds with these parameters. We formalize it as the following set of sentences.

Definition 4.20 (LWE, in $\mathsf{APC}_1 + $ "$\mathsf{prBPP} = \mathsf{prP}$")**.** Let $\lambda_0[p_1, p_2, p_3, p_4, p_5]$ be a mapping from five functions $p_1, \ldots, p_5$ to $\mathbb{N}$, not necessarily PV definable.

We define LWE_{λ_0} as the following set of sentences: For every PV functions $n(\lambda), m(\lambda), \sigma(\lambda), t(\lambda) : \mathsf{Log} \to \mathsf{Log}$ a PV function $q(\lambda)$ taking $\lambda \in \mathsf{Log}$, such that $n(\lambda), m(\lambda), t(\lambda) \in \mathsf{poly}(\lambda)$, $q(\lambda) \leq 2^{\sqrt{n(\lambda)}}$, and $\sigma(\lambda) \geq 2\sqrt{n(\lambda)}$, the set includes

$$\forall \lambda > \lambda_0[n, m, q, \sigma, t], \mathsf{LWE}_{n(\lambda),m(\lambda),q(\lambda),\sigma(\lambda)}^{t(\lambda),1/t(\lambda)}.$$

Proposition 4.21. *Suppose that the LWE assumption is true, then there is a mapping λ_0 such that every sentence in LWE_{λ_0} is true (in the standard model).*

[24] Alternatively, one can use the approximate sampler given by the $\mathbb{Z}_p$ Sampling Lemma that we will explain shortly. We stick to q be a power of two here for simplicity.

[25] Here, $q(\lambda)$ is definable as $\lambda \in \mathsf{Log}$. In more detail, $q(\lambda)$ is the abbreviation of $q(\Lambda) \leq 2^{\sqrt{n}}$, where $\lambda := |\Lambda| \in \mathsf{Log}$. We write $q(\lambda)$ for simplicity, but readers should keep in mind that $q(\lambda)$ is meaningful only when $\lambda \in \mathsf{Log}$.

We stress that the mapping λ_0 is used to specify the smallest input length that the security property holds. It captures the non-constructive part of the LWE assumption, and is not intended to be a function symbol in PV. In particular, Proposition 4.21 may not necessarily yield a mapping λ_0 that is definable in the theory.

4.4 Nonlogical Axiom 3: Hardness of SXDH

The third non-logical axiom we will use is the standard hardness assumption for the *Symmetric External Diffie-Hellman* (SXDH) problem over prime-order bilinear groups, as defined in Definition 3.9.

In this work, instead of describing bilinear groups as the source and target groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$, we directly require that all elements within the three groups are represented as bit strings of length κ, and all group operations are defined as PV functions. The correctness of the group operations will be included as axioms.

Definition 4.22 (Prime-Order Bilinear Group)**.** A prime-order asymmetric bilinear group

$$\mathcal{G} = (1^\kappa, p, \mathsf{Map}, \mathsf{Val}, \mathsf{Add}, \mathsf{Mul}, \mathsf{Pair})$$

is described by a size parameter κ, the order $p \leq 2^\kappa$, and functions $\mathsf{Map}, \mathsf{Val}, \mathsf{Add}, \mathsf{Mul}, \mathsf{Pair}$ computing the following group operations.

- $\mathsf{Map} : \{0,1\}^\kappa \times \{1,2,T\} \to \{0,1\}^\kappa$, where $\mathsf{Map}(a, b) \to g_b^a$ maps $\mathbb{Z}_p$ elements a to (encodings of) group elements $g_b^a \in \mathbb{G}_b$ for $b \in \{1,2,T\}$.
- $\mathsf{Val} : \{0,1\}^\kappa \times \{1,2,T\} \to \{0,1\}$, where $\mathsf{Val}(X, b)$ outputs 1 if X is in the image of $\mathsf{Map}(\mathbb{Z}_p, b)$ (i.e., $X \in \mathbb{G}_b$), and outputs 0 otherwise.
- $\mathsf{Add} : \{0,1\}^\kappa \times \{0,1\}^\kappa \to \{0,1\}^\kappa$, where $\mathsf{Add}(g_b^a, g_b^{a'}) \to g_b^{a+a'}$ maps two group elements $g_b^a, g_b^{a'} \in \mathbb{G}_b$ to their group addition $g_b^{a+a'} \in \mathbb{G}_b$ for $b \in \{1,2,T\}$.
- $\mathsf{Mul} : \{0,1\}^\kappa \times \{0,1\}^\kappa \to \{0,1\}^\kappa$, where $\mathsf{Mul}(c, g_b^a) \to g_b^{ac}$ maps a scalar $c \in \mathbb{Z}_p$ and a group element $g_b^a \in \mathbb{G}_b$ to their scalar multiplication $g_b^{ac} \in \mathbb{G}_b$ for $b \in \{1,2,T\}$.
- $\mathsf{Pair} : \{0,1\}^\kappa \times \{0,1\}^\kappa \to \{0,1\}^\kappa$, where $\mathsf{Pair}(g_1^a, g_2^b) \to e(g_1^a, g_2^b) = g_T^{ab}$ maps two group elements $g_1^a \in \mathbb{G}_1$ and $g_2^b \in \mathbb{G}_2$ on the source group to their bilinear pairing $g_T^{ab} \in \mathbb{G}_T$.

We define $\mathsf{Cor}_\mathcal{G}$ to be the following set of sentences describing the primality of the order and the correctness of the group operations:

- For all integer $a \in (1, p)$, $a \nmid p$.
- For all $a, a' \in \mathbb{Z}_p, b \in \{1,2,T\}$, $a \neq a' \implies \mathsf{Map}(a, b) \neq \mathsf{Map}(a', b)$.
- For all $x \in \{0,1\}^\kappa, b \in \{1,2,T\}$, $\mathsf{Val}(x, b) = 1 \iff \exists a \in \mathbb{Z}_p, x = \mathsf{Map}(a, b)$.
- For all $a, a' \in \mathbb{Z}_p, b \in \{1,2,T\}$, $\mathsf{Add}(\mathsf{Map}(a, b), \mathsf{Map}(a', b)) = \mathsf{Map}(a + a', b)$.
- For all $a, c \in \mathbb{Z}_p, b \in \{1,2,T\}$, $\mathsf{Mul}(c, \mathsf{Map}(a, b)) = \mathsf{Map}(ca, b)$.
- For all $a, a' \in \mathbb{Z}_p$, $\mathsf{Pair}(\mathsf{Map}(a, 1), \mathsf{Map}(a', 2)) = \mathsf{Map}(aa', T)$.

The formalization of the sentences in PV_1 are straightforward; in particular, elements in $\mathbb{Z}_p$ are (again) represented as bit strings of length κ.

For a group generator $\mathsf{GroupGen}(1^\lambda; \mathsf{sd}) \to \mathcal{G}$, we define $\mathsf{Cor}_{\mathsf{GroupGen}}$ as sentence in the language of PV_1: For every $\lambda \in \mathsf{Log}$ and sd, all sentences in $\mathsf{Cor}_{\mathcal{G}}$ hold for every $\mathcal{G}$ output by $\mathsf{GroupGen}(1^\lambda; \mathsf{sd})$.

To sample a random element in $\mathbb{Z}_p$, we simply sample a sufficiently large integer r and output $r \bmod p$. This can be implemented straightforwardly by a PV function Samp_p. The following lemma formalizes the correctness of the sampling algorithm.

Lemma 4.23 ($\mathbb{Z}_p$ Sampling Lemma)**.** *For all $p > 0$, let $\gamma := 10p$ and $d = \lceil \log \gamma^{-1} \rceil + \lceil \log p \rceil$, we define*

$$\mathsf{Samp}_p : \{0,1\}^d \to \{0,1\}^{\lceil \log p \rceil}$$

be the circuit parsing its input as an integer $r \in [0, 2^d)$ and outputting $r \bmod p$. We can prove in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$" that for all $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$,

$$\forall y \in \mathbb{Z}_p, \Pr_\delta[\mathsf{Samp}_p(r) = y] \leq \delta + \beta + p^{-1}.$$

Proof. We argue in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". For each $y \in \mathbb{Z}_p$, let circuit $G_y(r) : \{0,1\}^d \to \{0,1\}$ be the circuit outputting 1 if $\mathsf{Samp}_p(r) = y$ and 0 otherwise. Let $X_y \subseteq \{0,1\}^d$ be the bounded set defined by circuit G_y, and $\eta^{-1} \in \mathsf{Log}$ be a parameter to be determined later. By Lemma 3.4 (1), there exists an $s \leq 2^d$ such that $X_y \approx_\eta s$.

Note that there is a provably surjective mapping F from $\{0,1\}^{\lceil \log \gamma^{-1} \rceil + 1}$ to X_y: $F(u) := pu + y$, where u is parsed as an integer in $[0, 2^{\lceil \log \gamma^{-1} \rceil + 1})$. By definition, $X_y \lesssim_0 2^{\lceil \log \gamma^{-1} \rceil + 1}$, and thus $s \lesssim_\eta X_y \lesssim_0 2^{\lceil \log \gamma^{-1} \rceil + 1}$. By Lemma 3.4 (2), we have that

$$s \leq 2^{\lceil \log \gamma^{-1} \rceil + 1} + 2\eta \cdot 2^d \leq (p^{-1} + 2\eta) \cdot 2^d.$$

Moreover, by "$\mathsf{prBPP} = \mathsf{prP}$", we know that

$$\Pr_\delta[G_y] \leq s \cdot 2^{-d} + \eta + \delta \leq p^{-1} + \delta + 3\eta.$$

It completes the proof by setting $\eta := \beta/3$. □

Note that the above bound that we prove is much weaker than the actual bound, which can be made arbitrarily close to p^{-1} by setting γ large enough, matching the uniform distribution required by the standard SXDH assumption. Nevertheless, the above bound is sufficient for our purpose, and removes the need to work with the extra parameter γ in the rest of the proof in APC_1.

Furthermore, we can prove that the distribution sampled by the algorithm Samp_p is (almost) shift-invariant. Recall that two distributions $\mathcal{H}_1, \mathcal{H}_2$ are said to be ε-almost isomorphic, denoted by $\mathcal{H}_1 \simeq_\varepsilon \mathcal{H}_2$, if we can shift both permutations such that they are almost identical (see Definition 4.10).

Lemma 4.24 ($\mathbb{Z}_p$ Shift-Invariance Lemma)**.** *The following sentence is provable in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". Let $\mathsf{Samp}_p^x(r) = (\mathsf{Samp}_p(r) - x) \bmod p$ be the circuit that shifts the sampled $\mathbb{Z}_p$ element by x. For all $p > 0$ and $x, y \in [0, p)$, $\mathcal{D}_p^x \simeq_{p^{-1}} \mathcal{D}_p^y$, where $\mathcal{D}_p^x$ and $\mathcal{D}_p^y$ are the distributions defined by circuits $\mathsf{Samp}_p^x, \mathsf{Samp}_p^y$ respectively.*

Proof. We argue in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. Recall that for $\gamma := 10p$ and $d = \lceil \log \gamma^{-1} \rceil + \lceil \log p \rceil$, the algorithm $\mathsf{Samp}_p(r)$ takes $r \in [0, 2^d)$ and outputs $r \bmod p$, and $\mathsf{Samp}_p^x(r)$ outputs $r - x \bmod p$ instead. Let $f : \{0,1\}^d \to \{0,1\}^d$ be the function as

$$f_x(r) := \begin{cases} r + x & \text{when } 0 \le r < 2^d - x; \\ r - (2^d - x) & \text{otherwise.} \end{cases}$$

It is clear that f is a bijection from $\{0,1\}^d$ to $\{0,1\}^d$. Let $\widetilde{\mathcal{D}}_p^x$ be the distribution defined by $\mathsf{Samp}_p^x \circ f_x$ and $\widetilde{\mathcal{D}}_p^y$ be defined by $\mathsf{Samp}_p^y \circ f_y$. To prove that $\mathcal{D}_p^x \simeq_{p^{-1}} \mathcal{D}_p^y$, it suffices to prove that $\widetilde{\mathcal{D}}_p^x \approx_{p^{-1}} \widetilde{\mathcal{D}}_p^y$. Let T_{C_1,C_2} be the circuit in Definition 4.9, and we take $C_1 := \mathsf{Samp}_p^x \circ f_x$ and $C_2 := \mathsf{Samp}_p^y \circ f_y$, our goal is to prove that for every $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$,

$$\Pr_\delta[T_{C_1,C_2}] \le p^{-1} + \delta + \beta.$$

Let X be the set defined by T_{C_1,C_2}. Note that $T_{C_1,C_2}(u) = 1$ only if $u > 2^d - p$, and thus there is a provably surjective mapping from p to X. The rest of the proof is similar to that of the $\mathbb{Z}_p$ Sampling Lemma and is left as an exercise. □

With the above definition, we can now define the parameterized SXDH assumption.

Definition 4.25 (Parameterized SXDH assumption)**.** Let $\mathcal{G} = (1^\kappa, p, \mathsf{Map}, \mathsf{Val}, \mathsf{Add}, \mathsf{Mul}, \mathsf{Pair})$ be a prime-order asymmetric bilinear group. Let $t, \lambda \in \mathsf{Log}$, $\gamma = 2^{-\kappa}$, and $\varepsilon > 0$. Let Gen_0, Gen_1 be two circuits defined as follows:

$$\mathsf{Gen}_0 : u_1, u_2, v_1, v_2 \leftarrow \mathbb{Z}_p;\ \mathsf{Output}(\mathcal{G}, \{\mathsf{Map}(u_b, b), \mathsf{Map}(v_b, b), \mathsf{Map}(u_b v_b, b)\}_{b\in\{1,2\}})$$
$$\mathsf{Gen}_1 : u_1, u_2, v_1, v_2, w_1, w_2 \leftarrow \mathbb{Z}_p;\ \mathsf{Output}(\mathcal{G}, \{\mathsf{Map}(u_b, b), \mathsf{Map}(v_b, b), \mathsf{Map}(w_b, b)\}_{b\in\{1,2\}})$$

The parameterized SXDH assumption $\mathsf{SXDH}_{\lambda,\mathcal{G}}^{t,\varepsilon}$ is defined as the conjunction of the sentences:

- Group well-formedness: $\mathsf{Cor}_\mathcal{G}$ (Definition 4.22).
- Group largeness: $p \ge 2^\lambda$.
- SXDH hardness: $\mathsf{Gen}_0 \approx_{t,\varepsilon} \mathsf{Gen}_1$.

For $k \in \mathsf{Log}$, we define the parameterized k-parallel SXDH assumption $k\text{-}\mathsf{SXDH}_{\lambda,\mathcal{G}}^{t,\varepsilon}$ by defining circuits Gen_0^k and Gen_1^k as

$$\mathsf{Gen}_0^k : u_1, u_2, \{v_{1,i}, v_{2,i}\}_{i\in[k]} \leftarrow \mathbb{Z}_p;\ \mathsf{Output}(\mathcal{G}, \{\mathsf{Map}(u_b, b), \mathsf{Map}(v_{b,i}, b), \mathsf{Map}(u_b v_{b,i}, b)\}_{b\in\{1,2\},i\in[k]})$$
$$\mathsf{Gen}_1^k : u_1, u_2, \{v_{1,i}, v_{2,i}, w_{1,i}, w_{2,i}\}_{i\in[k]} \leftarrow \mathbb{Z}_p;\ \mathsf{Output}(\mathcal{G}, \{\mathsf{Map}(u_b, b), \mathsf{Map}(v_{b,i}, b), \mathsf{Map}(w_{b,i}, b)\}_{b\in\{1,2\},i\in[k]})$$

We note that SXDH and k-SXDH is equivalent up to a $O(k)$ blow-up in parameters t and ε, and it is provable in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$; see the Parallel SXDH Lemma.

Definition 4.26 (SXDH assumption)**.** We define $\mathsf{SXDH}_{\lambda_0,\mathsf{GroupGen}}$ as the following set of sentences. For every PV function $t(\lambda)$ which is polynomial, the set includes

$$\forall \lambda \ge \lambda_0[t], \mathsf{SXDH}_{\lambda,\mathsf{GroupGen}(1^\lambda)}^{t(\lambda),1/t(\lambda)}$$

We say the SXDH assumption holds with respect to GroupGen if there exists some λ_0 such that $\mathsf{SXDH}_{\lambda_0,\mathsf{GroupGen}}$ is true. For $k \in \mathsf{Log}$, we define the k-parallel SXDH assumption $k\text{-}\mathsf{SXDH}_{\lambda_0,\mathsf{GroupGen}}$ similarly to the parameterized case.

4.5 The Unprovability Assumption

We now describe our main assumption: The unprovability of Extended Frege lower bound in the theory $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ that we defined in Section 4.

Let ϕ be a propositional formula and $k \in \mathbb{N}$. We use $\mathsf{EF\text{-}LB}_k(\phi)$ to denote the following formula: For every π of length at most $|\phi|^k$, π is not a valid Extended Frege proof concluding ϕ. Note that for every $k \in \mathbb{N}$, $\mathsf{EF\text{-}LB}_k(\phi)$ can be formalized as a formula in the language of PV_1, as there is a straightforward polynomial-time algorithm that verifies an Extended Frege proof.

Our assumption suggests that there is no $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$-*provably sound certification* of fixed polynomial Extended Frege lower bound. Formally:

Definition 4.27. Let $q(n)$ be a polynomial, $k \in \mathbb{N}$, and $\mathcal{T}$ be a theory that contains PV_1. An NP verifier $V(\phi, \tau)$ with proof length $|\tau| = q(|\phi|)$ is said to be a $\mathcal{T}$-provably sound certification of n^k-size Extended Frege lower bounds if it satisfies the following two conditions.

- (*Completeness*). There are infinitely many *tautologies* $\{\phi_i\}_{i\in\mathbb{N}}$ such that for every $i \in \mathbb{N}$, $\mathsf{EF\text{-}LB}_k(\phi_i)$ is true and $V(\phi_i, \tau) = 1$ for some $\tau \in \{0,1\}^{q(|\phi_i|)}$.
- (*Provable Soundness*). The theory $\mathcal{T}$ proves the following sentence:

$$\forall \phi \, \forall \tau \in \{0,1\}^{q(|\phi|)} \left(V(\phi, \tau) = 1 \to \mathsf{EF\text{-}LB}_k(\phi)\right). \tag{4.1}$$

In plain words, Equation (4.1) states that if ϕ has a V-proof, the Extended Frege lower bound is true for ϕ, or equivalently, there is no $|\phi|^k$-size Extended Frege proof concluding ϕ.

We are now ready to describe our main unprovability assumption. Let $\mathcal{T}$ be an extension of $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$", we introduce the following assumption:

Assumption 2 (No provable certification of EF lower bound)**.** There exists a $k \in \mathbb{N}$ such that for every polynomial q and NP verifier V with proof length q, V is not a $\mathcal{T}$-provably sound certification of n^k-size Extended Frege lower bounds.

In this paper, we will instantiate with the theory $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ that extends $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$" with non-logical axioms that are falsifiable cryptographic assumptions:

Assumption 3 (Instantiation)**.** Assumption 2 holds for $\mathcal{T} := \mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$.

5 Formalization of SNARGs and Security Lifting Lemma

5.1 Search Security Game

Definition 5.1 (Search game, based on [GK15])**.** Let $r := r(\lambda), n := n(\lambda), m := m(\lambda), \ell := \ell(\lambda)$ be polynomials. A search game is defined by a triple $(\mathcal{C}_1, \mathcal{C}_2, c)$, where $\mathcal{C}_1 : \{0,1\}^r \to \{0,1\}^{n+m}$, and $\mathcal{C}_2 : \{0,1\}^{\ell+m} \to \{0,1\}$ are polynomial sized circuits, and $c \in [0,1]$ is a constant. We say the game is secure if for all polynomial-sized adversaries $\mathcal{A}$,

$$\Pr\left[\, b = 1 \;:\; \begin{array}{c} (\mathsf{chall}, \mathsf{state}) \leftarrow \mathcal{C}_1(\mathsf{sd}) \\ \mathsf{ans} \leftarrow \mathcal{A}(\mathsf{chall}) \\ b \leftarrow \mathcal{C}_2(\mathsf{ans}, \mathsf{state}) \end{array} \right] \leq c + \mathsf{negl}(\lambda).$$

We say that the game is publicly verifiable if $\mathsf{state} = \varnothing$, and privately verifiable otherwise.

Definition 5.2 (Advantage in a security game, in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$). Let $r, n, m, \ell \in \mathsf{Log}$, $\mathcal{G} = (C_1, C_2, c)$ be a search security game, where $C_1 : \{0,1\}^r \to \{0,1\}^n \times \{0,1\}^m$, and $C_2 : \{0,1\}^\ell \times \{0,1\}^m \to \{0,1\}$ be circuits, and $c, \varepsilon \in [0,1]$. Let $\mathcal{A}$ be a probabilistic circuit of input length n, seed length r', and output length ℓ, and $T_{\mathcal{G},\mathcal{A}} : \{0,1\}^r \times \{0,1\}^{r'} \to \{0,1\}$ be the circuit that, given $(\mathsf{sd}, \mathsf{sd}')$, computes $(\mathsf{chall}, \mathsf{state}) \leftarrow C_1(\mathsf{sd})$, and outputs $C_2(\mathcal{A}(\mathsf{chall}; \mathsf{sd}'), \mathsf{state})$.

We say that $\mathcal{A}$ has advantage at most ε in a game $\mathcal{G} = (C_1, C_2, c)$, if for any $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$,

$$\Pr_\delta[T_{\mathcal{G},\mathcal{A}}] \leq c + \varepsilon + \delta + \beta.$$

We denote this by $\mathsf{Adv}_{\mathcal{G}}(\mathcal{A}) \leq \varepsilon$. Similarly, one say that $\mathsf{Adv}_{\mathcal{G}}(\mathcal{A}) \geq \varepsilon$ if for any $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$,

$$\Pr_\delta[T_{\mathcal{G},\mathcal{A}}] \geq c + \varepsilon - \delta - \beta.$$

Definition 5.3 (Secure Search Game, in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$). Let $r, n, m, \ell, t \in \mathsf{Log}$, $C_1 : \{0,1\}^r \to \{0,1\}^{n+m}$, and $C_2 : \{0,1\}^{\ell+m} \to \{0,1\}$ be circuits, and $c, \varepsilon \in [0,1]$. We say that a game $\mathcal{G} = (C_1, C_2, c)$ is (t, ε)-*secure* if for any adversaries of size t, $\mathsf{Adv}_{\mathcal{G}}(\mathcal{A}) \leq \varepsilon$.

5.2 SNARG for Languages with Short Proof of Non-membership

Our key cryptographic construction is a SNARG for any language $L \in \mathsf{NP}$ that is sound assuming *short proof of non-membership* in Extended Frege system (called EF-SNARG for simplicity). We start with its definition in English, and then explain how to formalize it in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$.

Syntax. Let $L \in \mathsf{NP}$ be a language and $M(x, w)$ be its NP verifier with witness length $|w| = m(|x|)$ for some $m = m(n) \in \mathsf{poly}(n)$. The syntax of EF-SNARG for L (with respect to the verifier M) consists of polynomial time algorithms $\mathsf{Gen}, \mathcal{P}, \mathcal{V}$ as follows:

- $\mathsf{Gen}(1^\lambda, 1^n, 1^\ell)$ is a randomized algorithm that takes in unary the security parameter λ, the input length n, and the length of the EF proof ℓ, which we will explain later. It outputs a pair of common reference strings crs.
- $\mathcal{P}(\mathsf{crs}, x, w)$ is a randomized algorithm that takes as input the CRS, an input $x \in \{0,1\}^n$, and an NP witness $w \in \{0,1\}^m$. It outputs a SNARG proof π.
- $\mathcal{V}(\mathsf{crs}, x, \pi)$ is a randomized algorithm that takes as input the CRS, an input $x \in \{0,1\}^n$, and a SNARG proof π. It decides whether to accept the proof.

Semantics. As a proof system, EF-SNARG should satisfy completeness and soundness properties for deciding the language L.

- (*Completeness*). For every $x \in L$ with witness w (i.e. $M(x, w) = 1$), with probability 1, $\mathcal{V}(\mathsf{crs}, x, \pi)$ accepts, where $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^n, 1^\ell)$ and $\pi \leftarrow \mathcal{P}(\mathsf{crs}, x, w)$.
- (*Soundness*). Let ϕ_x be the propositional formula that is a tautology iff $\forall w \in \{0,1\}^m\ M(x, w) = 0$. Then for every $x \notin L$ such that ϕ_x admits an EF proof τ of size at most ℓ and any adversary $\mathcal{A}$ of size $\mathsf{poly}(\lambda)$, there is a negligible function $\mathsf{negl}(\lambda)$ such that

$$\Pr\left[\ \mathcal{V}(\mathsf{crs}, x, \pi) = 1 \ \ : \ \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^n, 1^\ell) \\ \pi \leftarrow \mathcal{A}(\mathsf{crs}, x) \end{array} \right] \leq \mathsf{negl}(\lambda).$$

- (*Efficiency*). Both SNARG and EF-SNARG should also satisfy the efficiency guarantee: the length of crs should be $\mathsf{poly}(\lambda, n, \ell)$, the length of a proof π should be $\mathsf{poly}(\lambda, \log n, \log \ell)$, and the runtime of $\mathcal{V}$ should be $\mathsf{poly}(|\mathsf{crs}|, |\pi|, n, m)$. The efficiency analysis does not need to be formalized in bounded arithmetic; we postpone related discussion to Section 11.

Note that in contrast to a standard SNARG, where the soundness holds for every $x \notin L$, EF-SNARG is only guaranteed to be sound when $x \notin L$ has a short propositional proof.

Formalization of soundness. A crucial observation for our results is the soundness of EF-SNARG can be naturally formalized in the language of $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$, and the security proof of a construction (due to [JKLV24]) can be carried out within $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. We first explain the formalization of soundness property.

Recall that PV defines deterministic polynomial-time functions. The probabilistic polynomial-time algorithm Gen, $\mathcal{P}$, $\mathcal{V}$ are formalized as PV functions additionally takes a random seed, i.e., $\mathsf{Gen}(1^\lambda, 1^n, 1^\ell; \mathsf{sd})$. For simplicity, we assume that the seed lengths of Gen, $\mathcal{P}$, $\mathcal{V}$ are all $r(\lambda, n, \ell) = \mathsf{poly}(\lambda, n, \ell)$.

Similar to Section 4.3, we will first formalize parametrized soundness of EF-SNARG, when λ, n, ℓ, the adversary size t, and the advantage ε are given. We will use the framework of search games defined in Section 5.1.

Definition 5.4 (Parameterized EF-SNARG soundness, in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$)**.** Let $\lambda, n, \ell, t \in \mathsf{Log}$, $\varepsilon \in [0, 1]$, and $(\mathsf{Gen}, \mathcal{P}, \mathcal{V})$ be an EF-SNARG scheme. For every string $x \in \{0,1\}^n$, we define circuits C_1^x, C_2^x:

- C_1^x takes as input $(\mathsf{sd}_1, \mathsf{sd}_2) \in \{0,1\}^{2r}$, runs $\mathsf{Gen}(1^\lambda, 1^n, 1^\ell; \mathsf{sd}_1)$ to obtain crs, and outputs a pair $(\mathsf{chall}, \mathsf{state})$ where $\mathsf{chall} := \mathsf{crs}$ and $\mathsf{state} := (\mathsf{crs}, \mathsf{sd}_2)$.
- C_2^x takes as input a pair $(\mathsf{ans}, \mathsf{state})$, it parses $\mathsf{state} := (\mathsf{crs}, \mathsf{sd}_2)$ and outputs 1 if and only if $\mathcal{V}(\mathsf{crs}, x, \mathsf{ans}; \mathsf{sd}_2) = 1$.

Let $\mathcal{G}_x := (C_1^x, C_2^x, 0)$ be a search game. Then $\mathsf{EF\text{-}SNARG}^{t,\varepsilon}_{\lambda,n,\ell}$ is defined as the following formula: For every $x \in \{0,1\}^n$ and $\tau \in \{0,1\}^\ell$ such that τ is an EF proof of ϕ_x, the game $\mathcal{G}_x$ is (t, ε)-secure.

Let $\lambda_0[p_1, p_2, p_3]$ be a function from PV functions p_1, p_2, p_3 to $\mathbb{N}$. Note that λ_0 is not necessarily a PV function. We can then define the soundness of EF-SNARG using a set of sentences as follows.

Definition 5.5 (EF-SNARG soundness, in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$)**.** $\mathsf{EF\text{-}SNARG}_{\lambda_0}$ is defined as the following set of sentences in the language of $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. For every PV functions $n(\lambda), \ell(\lambda), t(\lambda) : \mathsf{Log} \to \mathsf{Log}$ that are polynomials in λ, the sentence

$$\forall \lambda > \lambda_0[n, \ell, t]\ \mathsf{EF\text{-}SNARG}^{t(\lambda), 1/t(\lambda)}_{\lambda, n(\lambda), \ell(\lambda)}$$

is included in the set $\mathsf{EF\text{-}SNARG}_{\lambda_0}$.

5.3 Security Lifting Lemma: From Provably Secure EF-SNARG to SNARG

Now we are ready to state and prove our main lemma: For every sound extension $\mathcal{T}$ of $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$, any $\mathcal{T}$-provably sound EF-SNARG scheme is also a SNARG scheme if Assumption 2 holds for the theory $\mathcal{T}$. Formally:

Lemma 5.6 (Security Lifting Lemma). *The following holds for every sound extension $\mathcal{T}$ of $\mathsf{APC}_1 + $ "$\mathsf{prBPP} = \mathsf{prP}$". Let $L \in \mathsf{NP}$ and $M(x,w)$ is its verifier, $(\mathsf{Gen}, \mathcal{P}, \mathcal{V})$ be PV functions and $\lambda_0[p_1, p_2, p_3]$ be a function from PV functions to $\mathbb{N}$. Suppose that $(\mathsf{Gen}, \mathcal{P}, \mathcal{V})$ is a sound and complete EF-SNARG for L, and every sentence in $\mathsf{EF\text{-}SNARG}_{\lambda_0}$ is provable in $\mathcal{T}$. Then under Assumption 2 for the theory $\mathcal{T}$, there exists a constant $k \in \mathbb{N}$ such that $(\mathsf{Gen}, \mathcal{P}, \mathcal{V})$ is a sound and complete SNARG for the language L when we fix $\ell := n^k$.*

Proof. Assume that Assumption 2 holds for the theory $\mathcal{T}$. Let $k_{\mathcal{T}} \in \mathbb{N}$ be the constant in Assumption 2. For $x \in \{0,1\}^n$, let ϕ_x be propositional formula that is a tautology if and only if $\forall w \in \{0,1\}^m \; M(x,w) = 0$, where $m = m(n)$ is the witness length. As $L \in \mathsf{NP}$, there is a constant $c \in \mathbb{N}$ such that $|\phi_x| = O(n^c)$. We will set $k := k_{\mathcal{T}} \cdot c + 1$.

As $(\mathsf{Gen}, \mathcal{P}, \mathcal{V})$ is a sound and complete EF-SNARG for L, we know that it must also be a complete SNARG for L. Suppose, towards a contradiction, that $(\mathsf{Gen}, \mathcal{P}, \mathcal{V})$ is not a sound SNARG for L. Then for some polynomial $n = n(\lambda), t = t(\lambda) \in \mathsf{poly}(\lambda)$, there are infinitely many security parameters $\lambda_1, \lambda_2, \ldots$, instances $x_1, x_2, \ldots$ and adversaries $\mathcal{A}_1, \mathcal{A}_2, \ldots$ such that

$$\Pr\left[\mathcal{V}(\mathsf{crs}, x_i, \pi) = 1 \;:\; \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Gen}(1^{\lambda_i}, 1^{n(\lambda_i)}, 1^{\ell(\lambda_i)}) \\ \pi \leftarrow \mathcal{A}_i(\mathsf{crs}, x_i) \end{array} \right] \geq \frac{1}{t(\lambda_i)}. \tag{5.1}$$

where $\ell(\lambda) := n(\lambda)^k$, $|\mathcal{A}_i| \leq t(\lambda_i)$, $x_i \in \{0,1\}^{n(\lambda_i)} \setminus L$.

Consider the following promise problem $\Pi = (\Pi_{\mathsf{YES}}, \Pi_{\mathsf{NO}})$. Given any input that is parsed as a pair $(\phi, \mathcal{A})$, where ϕ is a propositional formula and $\mathcal{A}$ is a probabilistic circuit.

- (YES-instance): $(\phi, \mathcal{A}) \in \Pi_{\mathsf{YES}}$ if $\phi = \phi_x$ for some $x \in \{0,1\}^{n(\lambda)}$, and

$$\Pr\left[\mathcal{V}(\mathsf{crs}, x, \pi) = 1 \;:\; \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Gen}(1^{\lambda}, 1^{n(\lambda)}, 1^{\ell(\lambda)}) \\ \pi \leftarrow \mathcal{A}(\mathsf{crs}, x) \end{array} \right] \geq \frac{1}{t(\lambda)}. \tag{5.2}$$

- (NO-instance): $(\phi, \mathcal{A}) \in \Pi_{\mathsf{NO}}$ if $\phi = \phi_x$ for some $x \in \{0,1\}^{n(\lambda)}$, and

$$\Pr\left[\mathcal{V}(\mathsf{crs}, x, \pi) = 1 \;:\; \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Gen}(1^{\lambda}, 1^{n(\lambda)}, 1^{\ell(\lambda)}) \\ \pi \leftarrow \mathcal{A}(\mathsf{crs}, x) \end{array} \right] \leq \frac{1}{2 \cdot t(\lambda)}. \tag{5.3}$$

It is clear that $\Pi \in \mathsf{prBPP}$. As $\mathcal{T}$ is a sound extension of $\mathsf{APC}_1 + $ "$\mathsf{prBPP} = \mathsf{prP}$", we know that $\mathsf{prBPP} = \mathsf{prP}$, and subsequently $\Pi \in \mathsf{prP}$. Let $V(\phi, \tau)$ be a polynomial-time algorithm that solves Π, i.e., it accepts every string in Π_{YES} and rejects every string in Π_{NO}, on sufficiently large input lengths. Note that such algorithm exists by $\mathsf{prBPP} = \mathsf{prP}$, and we will choose a specific algorithm later for provable soundness in $\mathcal{T}$.

We will then prove that V is a $\mathcal{T}$-provably sound certification of $n^{k_{\mathcal{T}}}$-size Extended Frege lower bounds, which violating Assumption 2 and thus completes the proof.

Completeness. Recall that V is said to be complete if there are infinitely many tautologies ϕ such that $\mathsf{EF\text{-}LB}_{k_{\mathcal{T}}}(\phi)$ is true (i.e. ϕ does not have $|\phi|^{k_{\mathcal{T}}}$-size Extended Frege proofs and $V(\phi, \tau) = 1$ for some witness τ.

Consider the sequence of formulas $\phi_{x_1}, \phi_{x_2}, \ldots$ and witnesses $\mathcal{A}_1, \mathcal{A}_2, \ldots$ It is clear that $V(\phi_{x_i}, \mathcal{A}_i) = 1$ as $(\phi_{x_i}, \mathcal{A}_i) \in \Pi_{\mathsf{YES}}$ by Equation (5.1). Moreover, as $x_i \notin L$, the formulas $\phi_{x_1}, \phi_{x_2}, \ldots$ are tautologies. Thus it suffices to prove that there are infinitely many formulas $\phi \in \{\phi_{x_i}\}_{i \in \mathbb{N}}$ that require at least $|\phi|^{k_{\mathcal{T}}}$-size Extended Frege proofs.

Assume for contradiction that all but finitely many formulas $\phi \in \{\phi_{x_i}\}_{i\in\mathbb{N}}$ have at most $|\phi|^{k_{\mathcal{T}}}$-size Extended Frege proofs. Then for all but finitely many i, ϕ_{x_i} admits Extended Frege proofs of size at most

$$|\phi_{x_i}|^{k_{\mathcal{T}}} = O(n(\lambda_i)^{ck_{\mathcal{T}}}) \le n(\lambda_i)^k = \ell(\lambda_i).$$

Recall that every sentence in $\mathsf{EF\text{-}SNARG}_{\lambda_0}$ is provable in $\mathcal{T}$. Since $\mathcal{T}$ is a sound theory, all sentences in $\mathsf{EF\text{-}SNARG}_{\lambda_0}$ are true (in the standard model), which further implies that $(\mathsf{Gen}, \mathcal{P}, \mathcal{V})$ is a sound and complete EF-SNARG scheme. Subsequently, by the soundness of $(\mathsf{Gen}, \mathcal{P}, \mathcal{V})$ as an EF-SNARG scheme, there is a negligible function $\mathsf{negl}(\lambda)$ such that

$$\Pr\left[\ \mathcal{V}(\mathsf{crs}, x_i, \pi) = 1 \ \ : \ \begin{array}{c}\mathsf{crs} \leftarrow \mathsf{Gen}(1^{\lambda_i}, 1^{n(\lambda_i)}, 1^{\ell(\lambda_i)}) \\ \pi \leftarrow \mathcal{A}_i(\mathsf{crs}, x_i)\end{array}\ \right] \le \mathsf{negl}(\lambda),$$

By Equation (5.3), we have $(\phi_{x_i}, \mathcal{A}_i) \in \Pi_{\mathsf{NO}}$ for large i, which leads to a contradiction.

Provable Soundness. It remains to show that the theory $\mathcal{T}$ proves the sentence:

$$\forall \phi\, \forall \mathcal{A} \in \{0,1\}^{t(\lambda)}\ (V(\phi, \mathcal{A}) = 1 \to \mathsf{EF\text{-}LB}_{k_{\mathcal{T}}}(\phi)), \tag{5.4}$$

where ϕ is parsed as $\phi = \phi_x$ for some $x \in \{0,1\}^{n(\lambda)}$. Now we start to argue in the theory $\mathcal{T}$. Note that we have not chosen the specific algorithm V; it will be chosen as a specific function symbol in the language of $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ below.

Fix any $n, \lambda \in \mathsf{Log}$, any sentence $\phi = \phi_x$, $x \in \{0,1\}^{n(\lambda)}$, and adversary $\mathcal{A} \in \{0,1\}^{t(\lambda)}$. Let $\lambda_0 = \lambda_0[n, \ell, t]$. We can define V such that it rejects every string of length at most $n(\lambda)$ for $\lambda \le \lambda_0$, so in the rest of the proof, we assume that $\lambda > \lambda_0$. Note that to prove Equation (5.4), it suffices to prove that if ϕ admits an Extended Frege proof of size $|\phi|^{k_{\mathcal{T}}} \le \ell(\lambda)$, $V(\phi, \mathcal{A}) = 0$.

Assume that ϕ admits an Extended Frege proof of size $\ell(\lambda)$. Since $\mathcal{T}$ proves every sentence in $\mathsf{EF\text{-}SNARG}_{\lambda_0}$ and $\lambda > \lambda_0$, it proves

$$\mathsf{EF\text{-}SNARG}^{10\cdot t(\lambda), 1/(10\cdot t(\lambda))}_{\lambda, n(\lambda), \ell(\lambda)}.$$

Let $\mathcal{G}_x$ be the search game defined in Definition 5.4. As ϕ admits an Extended Frege proof of size $\ell(\lambda)$, we can conclude (within theory $\mathcal{T}$) that the game $\mathcal{G}_x$ is $(10 \cdot t(\lambda), 1/(10 \cdot t(\lambda)))$-secure.

Now we choose the function symbol in the language of $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ to implement V. By the definition of security of search games, see Definitions 5.2 and 5.3, we know that for every $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$,

$$\Pr_\delta[T_{\mathcal{G}_x, \mathcal{A}}] \le \frac{1}{10 \cdot t(\lambda)} + \delta + \beta, \tag{5.5}$$

where $T_{\mathcal{G}, \mathcal{A}}$ is defined in Definition 5.2. We implement V as the following function symbol: Let $\eta^{-1} := 10 \cdot t(\lambda) \in \mathsf{Log}$, then

$$V(\phi, \mathcal{A}) := \begin{cases} 1 & \Pr_\eta[T_{\mathcal{G}_x, \mathcal{A}}] \ge 8\eta; \\ 0 & \text{otherwise.} \end{cases}$$

Note that the condition $\Pr_\eta[T_{\mathcal{G}_x, \mathcal{A}}] \ge 8\eta$ is verified by the algorithm $\mathsf{capp}(\cdot, \cdot)$ available in the language of $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$.

We need to verify two properties:

- Following previous proofs in $\mathcal{T}$, $V(\phi, \mathcal{A}) = 0$. To see this, notice that by Equation (5.5), we have

$$\Pr_\eta[T_{\mathcal{G}_x,\mathcal{A}}] \leq \frac{1}{10 \cdot t(\lambda)} + 2\eta \leq 3\eta \leq 8\eta.$$

  By definition, we have $V(\phi, \mathcal{A}) = 0$.
- $V(\cdot, \cdot)$ decides the promise problem Π in the standard model (on large input lengths). This part is not necessarily provable in $\mathcal{T}$. As $\mathsf{capp}(\cdot, \cdot)$ is polynomial-time algorithm solving SearchCAPP in the standard model, see Section 4.1, we have that:
  - If $(\phi, \mathcal{A}) \in \Pi_{\mathsf{YES}}$, Equation (5.2) is true, which implies that in the standard model,

$$\Pr_\eta[T_{\mathcal{G}_x,\mathcal{A}}] \geq \frac{1}{t(\lambda)} - \eta \geq 8\eta.$$

    This further implies that $V(\phi, \mathcal{A}) = 1$.
  - If $(\phi, \mathcal{A}) \in \Pi_{\mathsf{NO}}$, Equation (5.3) is true, which implies that in the standard model,

$$\Pr_\eta[T_{\mathcal{G}_x,\mathcal{A}}] \leq \frac{1}{2 \cdot t(\lambda)} + \eta \leq 7\eta.$$

    This further implies that $V(\phi, \mathcal{A}) = 0$.

This completes the proof of the main lemma. □

6 Discussions on the Unprovability Assumption

In this section, we discuss the possible variants of Assumption 2 and its connection to the informal challenge of Razborov [Raz15].

6.1 Possible Variants and Attacks

We first discuss two natural ways to further strengthen Assumption 2, and explain why they will make the assumption insecure.

The completeness condition and nonfasifiablity. In Definition 4.27, we require the certification V to accept infinitely many *tautologies*. We stress that this requirement is necessary for Assumption 2 to be plausible.

Non-tautologies, formulas ϕ that are not true under all assignments, cannot be proven true in Extended Frege (by soundness), making them trivial elements satisfying $\mathsf{EF\text{-}LB}_k$. Moreover, the set of non-tautologies is easy to certify, simply by providing a falsifying assignment as the witness τ. If we remove the requirement that ϕ must be a tautology in the completeness condition, there is a trivial certification PV_1-provably sound certification V which accepts formulas ϕ with a falsifying assignment τ. The soundness of V, which is essentially the soundness of Extended Frege, is provable in PV_1 by a classical result of Cook:

Theorem 6.1 ([Coo75]). *For every* $k \in \mathbb{N}$, $\mathsf{PV}_1 \vdash \forall \phi \, \forall x \ (\phi(x) = 0 \to \mathsf{EF\text{-}LB}_k(\phi))$.

The tautology requirement in the completeness condition also forces the *nonfasifiablity* nature of our assumption (Assumption 2). Given a candidate certification V, a falsifying challenger can always verify its provable soundness simply by checking the proof. However, the challenger cannot efficiently verify V accepts infinitely many tautologies, simply because the challenger cannot efficiently certify tautologies; we postpone related discussions to Section 6.3.

On "provable soundness" vs. "soundness". In Assumption 2, we assume that there is no *provably sound* certification of EF lower bounds in theory $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$. One may wonder whether it is possible to replace the provable soundness requirement by standard soundness (i.e., Equation (4.1) is true in the standard model). We argue that the stronger assumption ruling out all sound certifications is unlikely to be true under a standard proof complexity conjecture that *Extended Frege is not a optimal proof system.*

A propositional proof system P is said to be *optimal* if any other proof system Q can be polynomially simulated by P. A standard conjecture in proof complexity is that there is no optimal proof system [KP89], and in particular, Extended Frege is not an optimal proof system. It immediately follows from a classical idea of Cook [Coo75] that:

Proposition 6.2. *If Extended Frege is not an optimal proof system, then for every $k \in \mathbb{N}$, there is an efficient algorithm A such that for infinitely many n, $A(1^n)$ outputs a tautology ϕ_n of length n such that $\mathsf{EF\text{-}LB}_k(\phi_n)$ is true. We call such an algorithm an n^k-*hard tautology generator *against Extended Frege.*

Proof Sketch. Let P be a proof system that cannot be simulated by EF, it follows from [Coo75] that the explicit sequence of tautologies $\{\sigma_n\}_{n\in\mathbb{N}}$ that formalize the *soundness of P* does not have polynomial-size Extended Frege proofs. The generator A simply outputs the explicit sequence of tautologies $\{\sigma_n\}_{n\in\mathbb{N}}$. □

With the efficient algorithm $A(1^n)$, one can construct a sound certification V of n^k-size Extended Frege lower bounds: $V(\phi, \tau)$ accepts if and only if $\phi = A(1^{|\phi|})$.

Remark 6.3 (Concrete candidates for the non-optimality of EF)**.** There has been concrete candidates for the non-optimality of Extended Frege. For instance, the soundness (or reflection principle) of Extended Frege is conjectured to require super-polynomial size in Extended Frege; see, e.g., [Kra19, Section 19.2]. In particular, since PV proves the soundness of Extended Frege, PV as a propositional proof system is conjectured to be stronger than Extended Frege (see [Coo75, Section 7]).

Pushing it to an extreme, it might be believable that strong mathematical theories such as ZFC or Peano Arithmetic, when used as a propositional proof system, are stronger than Extended Frege.

6.2 Razborov's Challenge: Proof Complexity from Computational Complexity

In a seminal work, Cook and Reckhow [CR79] introduced a formal definition of propositional proof systems, and proved that $\mathsf{NP} = \mathsf{coNP}$ if and only if there is a polynomially-bounded propositional proof system. Therefore, proving super-polynomial lower bounds for natural proof systems, such as Resolution, Frege, and Extended Frege, is necessary for proving that $\mathsf{NP} \neq \mathsf{coNP}$.

Since then, it has been a central focus of proof complexity to proving lower bounds for explicit propositional proof systems, and we briefly review some known results. Haken [Hak85] first proved that the Pigeonhole Principle (PHP) requires exponential size to prove in Resolution; soon after, many other formulas [Urq87, CS88] are proved hard for Resolution. Ajtai [Ajt94] (and subsequently [PBI93, KPW95]) proved that PHP is hard for AC^0-Frege using the *random restriction method* used to prove $\oplus \notin \mathsf{AC}^0$ [Hås86].[26] Krajíček [Kra94] (also see [Kra19, Chapter 17]) introduced a framework called *feasible interpolation*, which was later used to prove, e.g., strong lower bound against Resolution and Cutting Planes [Pud97].

Despite fruitful results for weak proof systems, lower bounds for strong proof systems such as Frege or Extended Frege have been open for about half a century. To our knowledge, only quadratic lower bounds are known for Frege [Kra95, Chapter 13], and the technique is unlikely to prove super polynomial lower

[26] We refer readers to [Raz01] and the references therein for more results related to PHP.

bounds. Also, it is worth noting that we have to search for *new hard tautologies*, as many standard hard tautologies such as PHP are known to admit polynomial-size Extended Frege [Coo75] or Frege proofs [Bus87]. Very few examples have been studied so far, including finite consistency sentences [Kra19, Section 19.2] and proof complexity generators (see [Kra25]).

Razborov's challenge. Indeed, even if we are allowed to use *any* plausible computational assumptions, proving strong lower bounds for Frege or Extended Frege remains an open problem. This has been explicitly asked by Razborov; we quote from [Raz15]:

> "A more accessible task that (in the author's opinion) is almost as interesting would be to show at least that proof complexity lower bounds are at most as hard as comparable problems in the computational world. Let us (informally) identify this task as: *Proving lower bounds for strong proof systems P like Frege or Extended Frege modulo any hardness assumption in the purely computational world, however strong but still natural and believable.*"

Here, "hardness assumption in the purely computational world" includes all plausible falsifiable cryptographic assumptions, but rules out separations such as $\mathsf{NP} \neq \mathsf{coNP}$, which immediately implies superpolynomial lower bounds for *all* propositional proof systems.

Given our limited progress in proving strong proof complexity lower bounds for decades, it may be reasonable to suspect that there are intrinsic limitations of known techniques that prevents us from establishing proof complexity lower bounds, or even resolving Razborov's challenge.

Hardness of computational complexity conjectures. Before diving into this hypothesis, we take a short detour to its counterpart in the "computational world". In computational complexity, there has been many *informal barriers* that capture limitations of certain lower bound techniques: relativization barrier [BGS75], algebraization barrier [AW09], natural proofs barrier [RR97], and many others. The main advantage of informal barriers is conceptual simplicity, which makes them easy to understand and insightful for further research.

On the other hand, informal barriers are limited to certain techniques (rather than mathematical proofs in a broader sense), which cannot capture indirect techniques (see, e.g., [CHO+22]). To address this limitation, an alternative and more systematic approach has been proposed and investigated recently: Instead of considering certain types of techniques, we may prove the *unprovability* of complexity upper and lower bounds in *formal mathematical theories*. Most results consider theories of *Bounded Arithmetic*, weak fragments of the Peano Arithmetic that formalize a large fraction of TCS, including the celebrated PCP theorem [Pic15b]. For several bounded theories, unconditional unprovability results have been established for circuit lower bounds [Kra11b, Pic15a, PS21, LO23, ABM23, CLO25] and upper bounds [CK07, KO17, BM20, BKO20, CKKO21]; we refer readers to the survey [Oli25] for a comprehensive introduction to this line of work.

6.3 Verifiable Proofs

In light of the progress in computational world, we view Assumption 2 and its instantiation Assumption 3 as formal hypotheses that it may be *impossible* to resolve Razborov's challenge within weak bounded arithmetic. To explain this point, we first compare it with the celebrated natural proof barrier [RR97, Rud97].

Natural proofs. We will briefly compare Assumption 2 with a classical conjecture — the non-existence of "natural proofs" — in the context of circuit complexity. To capture the limitation of existing circuit lower bound techniques, Razborov and Rudich [RR97] introduced the concept of *natural proofs* for circuit lower bounds. Let $\mathscr{C}$ be a complexity class, $N = 2^n$, and $s(n)$ be a function. A family of subsets $S_n \subseteq \{0,1\}^N$ is said to be a *$\mathscr{C}$-natural property* against $s(n)$-size circuits if it satisfies the following conditions.

- (*Constructivity*). The problem of deciding whether $x \in S_n$ is in $\mathscr{C}$.
- (*Usefulness*). For every circuit $C : \{0,1\}^n \to \{0,1\}$ of size $s(n)$, the truth-table of C is *not* in S_n.
- (*Largeness*). $|S_n| \geq 2^n/\mathsf{poly}(n)$ for infinitely many $n \in \mathbb{N}$.

It turns out that many existing lower bound techniques rely on the existence of P-natural property. Razborov and Rudich [RR97] proved that assuming sub-exponentially secure OWFs, there is no P-natural property against $\mathsf{poly}(n)$-size circuits. In a subsequent work, Rudich [Rud97] further conjectured that there is no NP-natural property against $\mathsf{poly}(n)$-size circuits.

Verifiable proofs. Our assumption can be interpreted as a counterpart of natural proof barrier in *proof complexity*. Let $\mathcal{T}$ be a theory that extends PV_1. We say that a property of formulas $\Phi = \{\Phi_n\}_{n\in\mathbb{N}}$ is a $\mathcal{T}$-verifiable proof (of Extended Frege lower bounds) if it satisfies the following properties.

- (*Constructivity*). The problem of deciding whether $\phi \in \Phi_n$ given $\phi \in \{0,1\}^n$ is in NP.
- (*Provable Usefulness*). For every formula $\phi \in \{0,1\}^n$ that admits an n^k-size Extended Frege proof, $\phi \notin \Phi_n$. Moreover, this sentence (formalized as a universal sentence in the language of PV_1) is provable in the theory $\mathcal{T}$.
- (*Non-emptiness*). There are infinitely many tautologies ϕ_n such that $\phi_n \in \Phi_n$.

That is, we do not require the *largeness* property as in the natural proof barrier [RR97, Rud97], while additionally require the usefulness property to be provable in $\mathcal{T}$. It is straightforward to see that the non-existence of $\mathcal{T}$-verifiable proofs is equivalent to Assumption 2; in particular, Assumption 3 is equivalent to say that there is no $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$-verifiable proof.

Remark 6.4. The concept of verifiable proof is a fusion of two established paradigms in meta-mathematics of computational complexity mentioned in Section 6.2, namely *informal barriers* and *unprovability in bounded arithmetic*. It remains an intriguing open problem to build our results on a pure informal barrier or a pure unprovability conjecture that is plausible. Note that directly removing the provability condition is not ideal as it implies Extended Frege is optimal (see Proposition 6.2).

Verifiable Proofs of Extended Frege lower bounds are verifiable in the sense that for every formula $\phi \in \{0,1\}^n$, an NP witness w for $\phi \in \Phi_n$ serves as a certification of $\mathsf{EF\text{-}LB}_k(\phi)$ in the theory $\mathcal{T}$, which, in our setting, is a weak fragment of bounded arithmetic. Any one who believes the soundness of $\mathcal{T}$ could be convinced that $\mathsf{EF\text{-}LB}_k(\phi)$ is true, i.e., ϕ requires $|\phi|^k$-size Extended Frege proof, by looking at the proof of usefulness and verifying that w is a correct NP witness for $\phi \in \Phi_n$. One may view the role of verifiable proofs as an exponential time speedup for verifying Extended Frege lower bounds: If no such NP witness is given, one may have to look through all possible Extended Frege proofs of size $|\phi|^k$ to be convinced that $\mathsf{EF\text{-}LB}_k(\phi)$ holds, which takes $2^{|\phi|^k}$ time.

Remark 6.5 (Trivial lower bounds)**.** A main disadvantage of Verifiable Proofs as a tool to understand Extended Frege lower bounds is that, given formula ϕ and a NP witness w for $w \in \Phi_n$, one cannot easily distinguish between the following two cases:

1. (*Real Lower Bounds*). ϕ is a tautology that requires $|\phi|^k$ size Extended Frege proof;

2. (*Trivial Lower Bounds*). ϕ is *not* a tautology, so it is unprovable in Extended Frege.

We argue that verifiable proofs also allow an exponential speedup for verifying "real" lower bounds. Let $n := |\phi|$ and m be the number of variables in ϕ. When $m \ll n$, say $m = n^{0.1}$, one can verify that ϕ is a tautology in time $2^m \ll 2^n$. Even when $m = \mathsf{poly}(\log n)$, there is no known algorithm for verifying $\mathsf{EF\text{-}LB}_k(\phi)$ in $2^{o(n^k)}$ time, while with a verifiable proof, it takes only quasi-polynomial time.

Our assumption and Razborov's challenge. We argue that refuting our instantiation (i.e. Assumption 3) provides a surprising resolution of Razborov's challenge, which, from our view, may be too good to be true.

Suppose that Assumption 3 is false, or equivalently, there is a $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$-verifiable proof. Let $\Phi = \{\Phi_n\}_{n\in\mathbb{N}}$ be the property of the $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$-variable proof. By the non-emptiness property, there are infinitely many tautologies $\phi_n \in \Phi_n$. Suppose in addition that

> $(\diamond)$: the non-logical axioms of $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$, including $\mathsf{prBPP} = \mathsf{prP}$ and several standard and falsifiable cryptographic assumptions, are true.

Then the theory $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ is sound, and by the provable usefulness property, every such formula $\phi_n \notin \Phi_n$ of size n does not have n^k-size Extended Frege proofs — a strong lower bound that is far beyond the reach of current techniques. This resolves Razborov's challenge as $(\diamond)$ is a plausible assumption in the "purely computational world".

In addition, although the existence of tautologies ϕ_n is a semantic guarantee, by the *provable usefulness* property, the *hardness* of these tautologies is provable in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ — a weak bounded theory with help of plausible computational assumptions. As mentioned above, this in a sense allows super polynomial speedup in verifying the hardness of the tautologies. At a high level, it means that refuting Assumption 3 gives a *highly constructive* resolution of Razborov's challenge.

6.4 Choice of the Theory $\mathcal{T}$

We stress that the concept of Verifiable Proofs is more interesting when $\mathcal{T}$ is weaker, especially when we want to make assumptions about their *non-existence*.

6.4.1 Insights from Known Unprovability Results

Regarding the choice of the base theory, we find it less comfortable to assume, for instance, the non-existence of ZFC-verifiable proofs of Extended Frege lower bounds. This is because we barely know anything in theoretical computer science that are unprovable in ZFC, and there are very few tools to analyze ZFC proofs of, say, Extended Frege lower bounds.

Unprovability of Extended Frege lower bounds. In contrast, an early result of Krajíček and Pudlák [KP89] (see also [Oli25, Section 5.2.2]) shows the potential of existing techniques on the unprovability of Extended Frege lower bounds — slightly super polynomial Extended Frege lower bounds are *unconditionally* unprovable in PV_1.

Theorem 6.6 ([KP89]). *PV_1 cannot prove the following sentence: For every 1^n, there is a propositional formula of size $m \geq n$ that is a tautology and requires at least $m^{\log m}$ size to prove in Extended Frege.*

Nevertheless, we note that this result does not formally justify our instantiation (see Assumption 3) for several reasons.

- We need the unprovability of fixed-polynomial lower bounds rather than super-polynomial lower bounds. To our knowledge, the technique of [KP89], which relies on propositional translation of PV [Coo75], is unlikely to extend to the fixed-polynomial regime.
- Our assumption is a fusion of an "informal barrier" and an unprovability conjecture.
- We consider a theory $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ that is stronger than PV_1.

It remains an interesting open problem to bridge or reduce the gap between Assumption 3 and provable hardness results.

Unprovability for APC_1. It is worth mentioning that unprovability results are known against very strong bounded theories, including $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$. Pich and Santhanam [PS21] proved that a strong average-case separation between NP and coNP is unprovable in a theory slightly weaker than APC_1. Following their techniques, Li and Oliveira [LO23] further proved that APC_1 does not prove an average-case separation between Σ_3^p and Π_3^p, i.e., the third levels of the polynomial-time hierarchy. In particular, by looking into [LO23, Theorem 2.1], the unprovability result holds against $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ provided that $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ is consistent.[27] Also, there are several conditional unprovability results for APC_1 and slightly weaker theories, see, e.g., [ILW23, CLO24, CRT25].

We stress that the sentences used in these unprovability results are presumably much stronger than Extended Frege lower bounds. Nevertheless, these results demonstrate the potential of existing techniques in analyzing APC_1.

Remark 6.7 (a glance on techniques)**.** We also note that most of these unprovability results are built on a generic technique called *witnessing theorems* (see, e.g., [Bus85] and [Kra95, Chapter 7]). The unprovability in [KP89], however, relies on more fine-grained tools in logic, including the propositional translation (see [Coo75] and [Kra19, Chapter 12]).

Witnessing theorems are used to analyze provable statements that involve *existential quantifiers*. Therefore, it is unlikely to resolve Assumption 1 using only witnessing theorems, as the sentence in Assumption 1 does not involve existential quantifiers.

6.4.2 Choice of axioms: Assumptions v.s. Primitives.

As argued above, we would like to work with a theory $\mathcal{T}$ that is as weak as possible. One way to achieve this is to weaken the non-logical axioms. Given that our current strategy towards constructing SNARGs goes through assuming crypto axioms and then proving the existence of many intermediate primitives (e.g. FHE, BARG) with care, it might be tempting to directly assume these intermediate primitives as axioms, which "weakens" the theory, and saves effort on writing security proofs in APC_1. We argue that this is not the correct approach for the following reasons.

Assuming existence of primitives as axioms. If we try to replace the axioms to just the *existence* of BARG/FHE, one immediate consequence is that the resulting SNARGs construction is non-constructive — we only know that there are some secure SNARG implemented by some unknown BARG/FHE scheme, and no concrete cryptanalysis can be done with respect to the scheme. Similarly, on the proof complexity side, refuting the unprovability assumption with respect to such a theory requires providing provably hard tautologies with respect to some unknown BARG/FHE scheme. It is not ideal to "strengthen" the assumption simply by making it harder to analyze. Since we are proposing new assumptions, providing a

[27] Note that it is necessary to assume the consistency of $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$, as otherwise all sentences are provable.

concrete attack interface is an important step toward deepening our understanding and gaining confidence in the assumptions.

For the same reason, it is also not ideal to assume the security of a *specific* cryptographic construction of BARG/FHE, as studying the proof complexity strength of basic complexity assumptions is much easier than studying that of specific cryptographic primitives.

Assuming proof lower bounds with respect to arbitrary primitive instantiation. On the other hand, it is also not a good idea to assume no provable certification for $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ for *arbitrary* instantiations of the primitives, as it may lead to a theory where the unprovability assumption is false. Indeed, again under the conjecture that Extended Frege is not an optimal proof system (see Proposition 6.2), we can construct an FHE scheme with an absurd homomorphic evaluation procedure, such that the correctness of the evaluation procedure directly implies the existence of an n^k-hard tautology generator against Extended Frege. Therefore, when this FHE scheme is added as an axiom, the n^k-hard tautology generator is provable in the theory $\mathsf{APC}_1[\mathsf{FHE}]$, and the unprovability assumption is false.

The construction of such an evaluation procedure is straightforward: Let A be an n^k-hard tautology generator against Extended Frege, and let FHE be any fully homomorphic encryption scheme. We construct FHE' which differs with FHE only at the evaluation procedure, when evaluating a circuit C with input size n^k. In this case, the evaluation procedure instead evaluates circuit C' defined as follows:

- Run $A(1^n)$ to get a tautology ϕ_n of size n.
- If the input x is a valid Extended Frege proof of ϕ_n, output $\neg C(x)$; otherwise, output $C(x)$.

It is easy to see that the correctness of the evaluation procedure of FHE' implies that for every n, ϕ_n does not have n^k-size Extended Frege proofs. Therefore, the hardness certification $V(\phi, \tau)$ that accepts if and only if $\phi = A(1^{|\phi|})$ is provably sound in $\mathsf{APC}_1[\mathsf{FHE}']$, thereby refuting the unprovability assumption.

6.4.3 Non-uniform Versions of Assumption 2

We also note that, due to the connection between bounded arithmetic and propositional proof system (see, e.g., [Kra19, Part II]), one can also state a non-uniform variant of Assumption 3 by replacing all provability requirements in bounded theories with that in corresponding propositional proof systems. In natural formalizations, non-uniform versions of the assumptions should imply the uniform versions by propositional translations.

As this requires a reformulation of non-logical axioms in propositional proof systems, which is not the main focus of this paper, we will not dive into this direction.

Part III
Formalization of SNARG in Bounded Arithmetic

7 Toolkit for Cryptography in Bounded Arithmetic

In this section, we develop basic tools for cryptography in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$", including basic probability principles, as well as tools for indistinguishable distributions and search games.

7.1 Basic Probability Principles

The following propositions show that $\Pr_\delta[C]$ is consistent for (1) different approximation parameters δ (2) different circuits that are functionally equivalent (3) complementation.

Proposition 7.1 (Precision Consistency of CAPP)**.** *The following statement is provable in* $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$"*. For every* $n, \delta_1^{-1}, \delta_2^{-1}, \beta^{-1} \in \mathsf{Log}$ *and circuit* $C : \{0,1\}^n \to \{0,1\}$, $|\Pr_{\delta_1}[C] - \Pr_{\delta_2}[C]| \leq \delta_1 + \delta_2 + \beta$.

Proof. We argue in $\mathsf{APC}_1 +$"$\mathsf{prBPP} = \mathsf{prP}$". Fix any $n, \delta_1^{-1}, \delta_2^{-1}, \beta^{-1} \in \mathsf{Log}$ and circuit C. Let $\eta^{-1} \in \mathsf{Log}$ be determined later, $X \subseteq \{0,1\}^n$ be the bounded set defined by C, and s be a number such that $X \approx_\eta s$ given by Lemma 3.4 (1). By "$\mathsf{prBPP} = \mathsf{prP}$", we have that $|\Pr_{\delta_1}[C] - s/2^n| \leq \delta_1 + \eta$ and $|\Pr_{\delta_2}[C] - s/2^n| \leq \delta_2 + \eta$. Subsequently, $|\Pr_{\delta_1}[C] - \Pr_{\delta_2}[C]| \leq \delta_1 + \delta_2 + \eta/2$, and the proposition holds by setting $\eta := \beta/2$. □

Proposition 7.2 (Global Consistency of CAPP)**.** *The following statement is provable in* $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$"*. For every* $n, \delta^{-1}, \beta^{-1} \in \mathsf{Log}$ *and circuits* $C_1, C_2 : \{0,1\}^n \to \{0,1\}$ *that are functionally equivalent,* $|\Pr_\delta[C_1] - \Pr_\delta[C_2]| \leq 2\delta + \beta$.

Proof. We argue in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". Fix $n, \delta^{-1}, \beta^{-1}$ and C_1, C_2. Let $X_1, X_2 \subseteq \{0,1\}^n$ be the bounded sets defined by C_1 and C_2, respectively. As C_1 and C_2 are functionally equivalent, the identity mapping witnesses that $X_1 \approx_0 X_2$. Let $\eta^{-1} \in \mathsf{Log}$ be determined later, and s be a number such that $X_1 \approx_\eta s$ following Lemma 3.4 (1), and thus by Proposition 3.3 (3), we also have $X_2 \approx_\eta s$. By "$\mathsf{prBPP} = \mathsf{prP}$",

$$|\Pr_\delta[C_1] - s/2^n| \leq \delta + \eta, \quad |\Pr_\delta[C_2] - s/2^n| \leq \delta + \eta,$$

and thus $|\Pr_\delta[C_1] - \Pr_\delta[C_2]| \leq 2\delta + 2\eta$. The proposition follows by setting $\eta := \beta/2$. □

Proposition 7.3 (Complementation Consistency of CAPP)**.** *The following statement is provable in* $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$"*. For every* $n, \delta^{-1}, \beta^{-1} \in \mathsf{Log}$ *and circuits* $C_1, C_2 : \{0,1\}^n \to \{0,1\}$ *such that* $C_1(x) = 1 - C_2(x)$*, then* $|\Pr_\delta[C_1] + \Pr_\delta[C_2] - 1| \leq 2\delta + \beta$.

Proof. We argue in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". Fix $n, \delta^{-1}, \beta^{-1} \in \mathsf{Log}$ and circuits C_1, C_2. Let $X_1, X_2 \subseteq \{0,1\}^n$ be the bounded sets defined by C_1, C_2, respectively. Then $X_1 = 2^n \setminus X_2$ and $X_2 = 2^n \setminus X_1$.

Let $\eta^{-1} \in \mathsf{Log}$ be determined later. By Lemma 3.4 (1), there exists $s \leq 2^n$ such that $X_1 \approx_\eta s$. By Lemma 3.4 (4) and Proposition 3.3 (3), we have that

$$X_2 = 2^n \setminus X_1 \lesssim_{2\eta} 2^n \setminus s \approx_0 2^n - s \implies X_2 \lesssim_{2\eta} 2^n - s;$$
$$2^n - s \approx_0 2^n \setminus s \lesssim_{2\eta} 2^n \setminus X_1 = X_2 \implies 2^n - s \lesssim_{2\eta} X_2.$$

Subsequently $X_2 \approx_{2\eta} 2^n - s$. It then follows from "$\mathsf{prBPP} = \mathsf{prP}$" that

$$|\Pr_\delta[C_1] + \Pr_\delta[C_2] - 1| \leq 2\delta + 4\eta \leq 2\delta + \beta,$$

where the last inequality follows by setting $\eta := \beta/4$. □

The following proposition states that for bounded definable sets $X_1 \subseteq X_2$ defined by circuits $C_1, C_2 : \{0,1\}^n \to \{0,1\}$, $\Pr[C_1] \leq \Pr[C_2]$.

Proposition 7.4 (Monotonicity of CAPP). *The following statement is provable in* $\mathsf{APC}_1 + $ "$\mathsf{prBPP} = \mathsf{prP}$"*. For every* $n, \delta^{-1}, \beta^{-1} \in \mathsf{Log}$ *and circuits* $C_1, C_2 : \{0,1\}^n \to \{0,1\}$ *such that* $C_1(x) = 1 \to C_2(x) = 1$ *for every* $x \in \{0,1\}^n$*. Then* $\Pr_\delta[C_1] \leq \Pr_\delta[C_2] + 2\delta + \beta$.

Proof. We argue in $\mathsf{APC}_1 + $ "$\mathsf{prBPP} = \mathsf{prP}$". Fix $n, \delta^{-1}, \beta^{-1}$ and C_1, C_2. Let $X_1, X_2 \subseteq \{0,1\}^n$ be the bounded sets defined by C_1 and C_2, respectively. As $C_1(x) = 1$ implies $C_2(x) = 1$ are functionally equivalent, the identity mapping witnesses that $X_1 \lesssim_0 X_2$. The rest of the proof is similar to that of the Global Consistency of CAPP and is left as an exercise. □

We will also need a form of union bound.

Proposition 7.5 (Union Bound). *The following statement is provable in* $\mathsf{APC}_1 + $ "$\mathsf{prBPP} = \mathsf{prP}$"*. For every* $n, \delta^{-1}, \beta^{-1} \in \mathsf{Log}$ *and circuits* $C, C_1, C_2 : \{0,1\}^n \to \{0,1\}$ *such that* $C(x) = C_1(x) \vee C_2(x)$*, then* $|\Pr_\delta[C] - \Pr_\delta[C_1] - \Pr_\delta[C_2]| \leq 3\delta + \beta$.

Proof. We argue in $\mathsf{APC}_1 + $ "$\mathsf{prBPP} = \mathsf{prP}$". Fix $n, \delta^{-1}, \beta^{-1} \in \mathsf{Log}$ and circuits C, C_1, C_2. Let $X, X_1, X_2 \subseteq \{0,1\}^n$ be the bounded sets defined by C, C_1, C_2, respectively. Then $X = X_1 \cup X_2$.

Let $\eta^{-1} \in \mathsf{Log}$ be determined later. By Lemma 3.4 (1), there exists $s, t, u \leq 2^n$ such that $X_1 \approx_\eta s$, $X_2 \approx_\eta t$, $X_1 \cap X_2 \approx_\eta u$. By Lemma 3.4 (5), we have $X \approx_{4\eta} s + t - u$. By "$\mathsf{prBPP} = \mathsf{prP}$", we have

$$|\Pr_\delta[C] - (s+t-u)/2^n| \leq \delta + 4\eta, \quad |\Pr_\delta[C_1] - s/2^n| \leq \delta + \eta, \quad |\Pr_\delta[C_2] - t/2^n| \leq \delta + \eta.$$

Therefore

$$|\Pr_\delta[C] - \Pr_\delta[C_1] - \Pr_\delta[C_2] + u/2^n| \leq 3\delta + 6\eta$$

which, by $u \geq 0$, immediately implies

$$|\Pr_\delta[C] - \Pr_\delta[C_1] - \Pr_\delta[C_2]| \leq 3\delta + \beta$$

by setting $\eta := \beta/6$. □

Proposition 7.6 (Union Bound (general form)). *The following statement is provable in* $\mathsf{APC}_1 + $ "$\mathsf{prBPP} = \mathsf{prP}$"*. For every* $n, m, \delta^{-1}, \beta^{-1} \in \mathsf{Log}$ *and circuits* $C, C_1, \ldots, C_m : \{0,1\}^n \to \{0,1\}$ *such that* $C(x) := C_1(x) \vee C_2(x) \vee \cdots \vee C_m(x)$*, then*

$$\left|\Pr_\delta[C] - \sum_{i=1}^{m} \Pr_\delta[C_i]\right| \leq \delta \cdot (m+1) + \beta.$$

Proof Sketch. We argue in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. Fix $n, m, \delta^{-1}, \beta^{-1} \in \mathsf{Log}$ and circuits $C, C_1, \ldots, C_m$. We induction on $k \leq m \in \mathsf{Log}$ to prove that

$$\left| \Pr_\eta \left[\bigvee_{i \leq k} C_i \right] - \sum_{i=1}^{k} \Pr_\eta[C_i] \right| \leq 4\eta \cdot (k+1). \tag{7.1}$$

This uses induction principle on a polynomial-time property, and is thus available in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. The base case is trivial, and the induction case can be proved using the Union Bound. Subsequently, by Precision Consistency of CAPP, we have

$$\left| \Pr_\delta[C] - \sum_{i=1}^{m} \Pr_\delta[C_i] \right| \leq (\delta + 2\eta) \cdot (m+1) + \left| \Pr_\eta[C] - \sum_{i=1}^{m} \Pr_\eta[C_i] \right| \leq \delta \cdot (m+1) + \beta,$$

where the last inequality follows from Equation (7.1) by setting $\eta := \beta/(2m+2)$. □

7.2 A Useful Lemma for Sampling

Lemma 7.7 (Constrained Sampling Lemma)**.** *The statement is provable in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$*. Let* $n_1, n_2, m_1, m_2, \ell, \delta^{-1}, \beta^{-1} \in \mathsf{Log}$ *and* $C_i : \{0,1\}^{n_i} \to \{0,1\}^{m_i}$*,* $i \in \{1,2\}$*, and* $D : \{0,1\}^{m_2} \times \{0,1\}^{\ell} \to \{0,1\}^{n_2}$*. Suppose that for every* $x_2 \in \{0,1\}^{n_2}$*, there exists a string* $z_2 \in \{0,1\}^{\ell}$ *satisfying that* $D(C_2(x_2), z_2) = x_2$*.*

Let $F : \{0,1\}^{m_1} \to \{0,1\}^{m_2}$ *and* $T : \{0,1\}^{n_1} \times \{0,1\}^{n_2} \to \{0,1\}$ *be the circuit as follows: Given* $(x_1, x_2) \in \{0,1\}^{n_1} \times \{0,1\}^{n_2}$*, it accepts if and only if* $F(C_1(x_1)) = C_2(x_2)$*. Then* $\Pr_\delta[T] \leq 2^{\ell - n_2} + \delta + \beta$*.*

Proof. We argue in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. Fix $n_1, n_2, m_1, m_2, \ell, \delta^{-1}, \beta^{-1} \in \mathsf{Log}$ and the circuits C_1, C_2, D, F, T. Let $\eta^{-1} \in \mathsf{Log}$ be a parameter to be determined and $X \subseteq \{0,1\}^{n_1} \times \{0,1\}^{n_2}$ be the bounded set defined by T. We can observe that $X \lesssim_0 2^{\ell + n_1}$, as the function

$$G : (z, x)_{\in \{0,1\}^{\ell} \times \{0,1\}^{n_1}} \mapsto (x, D(F(C_1(x)), z))_{\in \{0,1\}^{n_1} \times \{0,1\}^{n_2}}$$

is provably onto X. To see this, notice that for every $x_1, x_2 \in X$, we have $F(C_1(x_1)) = C(x_2)$ by definition. Then there is a string $z \in \{0,1\}^{\ell}$ such that $D(C_2(x_2), z) = x_2$, and subsequently:

$$D(F(C_1(x_1)), z) = D(C_2(x_2), z) = x_2,$$

which means $G(z, x_1) = (x_1, x_2)$. It then follows from “$\mathsf{prBPP} = \mathsf{prP}$” and Proposition 3.3 (3). □

7.3 Averaging Arguments

Lemma 7.8. APC_1 *proves the following sentence. Let* $n, m, \delta^{-1}, \beta^{-1} \in \mathsf{Log}$*,* $X_1, X_2 \subseteq \{0,1\}^n \times \{0,1\}^m$ *be bounded definable sets. Let* $X_i^y := \{x \in \{0,1\}^n \mid xy \in X_i\}$*. If* $X_1^y \approx_\delta X_2^y$ *for every* $y \in \{0,1\}^m$*, we have* $X_1 \approx_{\delta+\beta} X_2$*.*

Proof Sketch. The proof follows closely the proof of [Jeř07a, Proposition 2.16], therefore we will only sketch the proof. We argue in APC_1 that $X_1 \lesssim_{\delta+\beta} X_2$, and the other direction can be derived by symmetricity.

Let $\eta^{-1} \in \mathsf{Log}$ be a parameter to be determined later. By [Jeř07a, Lemma 2.14], there are circuits $S^i(\cdot), V, F^i_\eta(\cdot,\cdot), G^i_\eta(\cdot,\cdot), i \in \{1,2\}$ such that for every $y \in \{0,1\}^m$,

$$F^i_\eta(y,\cdot) : V(S^i(y) + \eta \cdot 2^{n-m}) \twoheadrightarrow V \times X^y_i,$$
$$G^i_\eta(y,\cdot) : V \times (X^y_i \cup \eta \cdot 2^{n-m}) \twoheadrightarrow V \cdot S^i(y).$$

Note that as $X^y_1 \approx_\delta X^y_2$, we know that $|S^1(y) - S^2(y)| \leq \delta + 2\eta$. By composing G^2_η and F^1_η, we can construct $H_\eta(\cdot,\cdot)$ such that for every $y \in \{0,1\}^m$,

$$H(y,\cdot) : V^2 \times (X^y_2 \cup (\delta + 2\eta) \cdot 2^{n-m}) \twoheadrightarrow V^2 \times X^y_1.$$

This further means that we can define $H : V^2 \times (X^2 \cup (\delta + 2\eta) \cdot 2^n) \twoheadrightarrow V^2 \times X_1$, and thus $X_1 \lesssim_{\delta+\beta} X_2$ if we set $\eta := \beta/2$. □

Lemma 7.9 (Averaging Argument on Two Circuits)**.** *The following statement is provable in* $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$"*. Let* $n, \delta^{-1} \in \mathsf{Log}$ *and* $C_1, C_2 : \{0,1\}^n \to \{0,1\}$ *be circuit. Let* $\alpha \subseteq [n]$ *be a subset of indices. Then for every* $\beta^{-1} \in \mathsf{Log}$*, there is an assignment* ρ *to the input variables in* α *such that*

$$|\Pr_\delta[C_1|_\rho] - \Pr_\delta[C_2|_\rho]| \geq |\Pr_\delta[C_1] - \Pr_\delta[C_2]| - (4\delta + \beta).$$

Proof. We argue in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". For simplicity, we assume that $\alpha = \{1, 2, \ldots, m\}$ and will prove that

$$|\Pr_\delta[C_1|_\rho] - \Pr_\delta[C_2|_\rho]| \geq |\Pr_\delta[C_1] - \Pr_\delta[C_2]| - (4\delta + \beta)$$

for some assignment ρ to variables in α.

Let $\eta^{-1} \in \mathsf{Log}$ be a parameter to be determined. For $i \in \{1,2\}$, let $p_i := \Pr_\delta[C_i]$ and S_i be the η-approximate size of $X_i := \{x \mid C_i(x) = 1\}$ by Lemma 3.4 (1). By "$\mathsf{prBPP} = \mathsf{prP}$", we know that

$$|p_i \cdot 2^n - S_i| \leq 2^n \cdot (\delta + \eta). \tag{7.2}$$

Let $\Delta p = |p_1 - p_2|$ and $\Delta S = |S_1 - S_2|$.

Suppose, towards a contradiction, that for every assignment ρ to the input variables in α, $|\Pr_\delta[C_1|_\rho] - \Pr_\delta[C_2|_\rho]| \leq \Delta p - (4\delta + \beta)$. We define

- $X^y_i := \{z \in \{0,1\}^{n-m} \mid C_i(yz) = 1\}$.

It is clear that for every $y \in Y$ (as an assignment to the first m variables), $C_1|_y, C_2|_y$ are the circuits that define X^y_1, X^y_2, respectively.

Let $p^y_i := \Pr_\delta[C_i|_y]$, and s^y_1, s^y_2 be the η-approximate sizes of X^y_1, X^y_2, respectively, given by (1) of Lemma 3.4. By "$\mathsf{prBPP} = \mathsf{prP}$", we know that $|s^y_i - p^y_i \cdot 2^{n-m}| \leq \delta + \eta)$, and subsequently

$$\begin{aligned}\Delta s^y_i := |s^y_1 - s^y_2| &\leq (|p^y_1 - p^y_2| + 2\delta + 2\eta) \cdot 2^{n-m} \\ &\leq (\Delta p - (4\delta + \beta) + 2\delta + 2\eta) \cdot 2^{n-m} \\ &\leq (\Delta p \cdot 2^{n-m} - (2\delta + \beta - 2\eta)) \cdot 2^{n-m}.\end{aligned}$$

This implies that $X^y_1 \approx_{\Delta p - (2\delta+\beta-2\eta)} X^y_2$. Then by Lemma 7.8, we have that $X_1 \approx_{\Delta p - (2\delta+\beta-3\eta)} X_2$. By transitivity of size comparison (see Proposition 3.3 (3)), as

$$S_1 \approx_\eta X_1 \approx_{\Delta p - (2\delta+\beta-3\eta)} X_2 \approx_\eta S_2,$$

we have that $S_1 \approx_{\Delta p - (2\delta+\beta-5\eta)} S_2$. This subsequently implies that $|S_1 - S_2| \leq (\Delta p - (2\delta + \beta - 6\eta)) \cdot 2^n$ by Lemma 3.4 (2). This leads to a contradiction to Equation (7.2) if we set $\eta := \beta/10$. □

The following is a simple corollary of the Averaging Argument on Two Circuits.

Corollary 7.10 (Averaging Argument). *The following statement is provable in* $\mathsf{APC}_1 + \text{“prBPP} = \mathsf{prP}\text{”}$. *Let* $n, \delta^{-1} \in \mathsf{Log}$ *and* $C : \{0,1\}^n \to \{0,1\}$ *be a circuit. Let* $\alpha \subseteq [n]$ *be a subset of indices. Then for every* $\beta^{-1} \in \mathsf{Log}$, *there are assignments* ρ_0, ρ_1 *to the input variables in* α *such that*

$$\Pr_\delta[C|_{\rho_0}] \leq \Pr_\delta[C] + (2\delta + \beta);$$
$$\Pr_\delta[C|_{\rho_1}] \geq \Pr_\delta[C] - (2\delta + \beta).$$

7.4 Tools for Indistinguishability

We first complete the missing proof of the Isomorphism Lemma, which states that two almost isomorphic distributions are also indistinguishable.

Lemma 4.14 (Isomorphism Lemma). *The following is provable in* $\mathsf{APC}_1 + \text{“prBPP} = \mathsf{prP}\text{”}$. *Let* $\ell, n, t \in \mathsf{Log}$, $\varepsilon \in [0,1)$, *and* $\mathcal{D}_1, \mathcal{D}_2$ *be distributions defined by circuits* $C_1, C_2 : \{0,1\}^\ell \to \{0,1\}^n$. *If* $\mathcal{D}_1 \simeq_\varepsilon \mathcal{D}_2$, *then* $\mathcal{D}_1 \approx_{t,\varepsilon} \mathcal{D}_2$.

Proof. We argue in $\mathsf{APC}_1 + \text{“prBPP} = \mathsf{prP}\text{”}$. Fix $\ell, n, t \in \mathsf{Log}$, $\varepsilon \in [0,1)$, $\mathcal{D}_1, \mathcal{D}_2, C_1, C_2$. Suppose that $\mathcal{D}_1 \simeq_\varepsilon \mathcal{D}_2$, i.e., there are distributions $\mathcal{D}'_1, \mathcal{D}'_2$ defined by circuits $C'_1, C'_2 : \{0,1\}^\ell \to \{0,1\}^n$ such that

- $\mathcal{D}_1 \cong \mathcal{D}'_1$ and $\mathcal{D}_2 \cong \mathcal{D}'_2$: namely, there are circuits $f_1, g_1, f_2, g_2 : \{0,1\}^n \to \{0,1\}^n$ such that for $i \in \{1,2\}$, $f_i \circ g_i = g_i \circ f_i = \mathsf{id}_n$ and $C_i \circ f_i \equiv C'_i$.
- $\mathcal{D}'_1 \approx_\varepsilon \mathcal{D}'_2$: namely, let $T_{C'_1,C'_2} : \{0,1\}^\ell \to \{0,1\}$ be the circuit that given $x \in \{0,1\}^\ell$, outputs 1 if and only if $C'_1(x) \neq C'_2(x)$, then $\mathsf{P}_\delta[T_{C'_1,C'_2}] \leq \delta + \beta + \varepsilon$ for every $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$.

Our goal is to prove that $\mathcal{D}_1 \approx_{t,\varepsilon} \mathcal{D}_2$. Fix any circuit $\mathcal{A} : \{0,1\}^n \to \{0,1\}$ of size at most t. Let $T_{\mathcal{A},i} : \{0,1\}^\ell \to \{0,1\}$ be the circuit that, given $x \in \{0,1\}^\ell$, outputs $\mathcal{A}(C_i(x))$. For every $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$, we will prove that

$$\left|\Pr_\delta[T_{\mathcal{A},1}] - \Pr_\delta[T_{\mathcal{A},2}]\right| \leq 2\delta + \beta + \varepsilon. \tag{7.3}$$

Translating Probability to Size Comparison. In the first step, we translate the probability statements to size comparison (see Definition 3.1 and Definition 3.2) in order to apply the tools from Jeřábek's theory APC_1.

Fix $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$ and let $\eta^{-1} \in \mathsf{Log}$ be a parameter to be determined later. Towards a contradiction, we assume that Equation (7.3) does not hold. Let $X_1, X_2 \subseteq \{0,1\}^\ell$ be the bounded definable sets defined by $T_{\mathcal{A},1}, T_{\mathcal{A},2}$, respectively. By Lemma 3.4 (1), there are $s_1, s_2 \leq 2^\ell$ such that $X_1 \approx_\eta s_1$ and $X_2 \approx_\eta s_2$. Moreover, by “prBPP = prP”, we have that

$$\begin{aligned}
&|s_1 - s_2| \\
&\geq \left|\mathsf{capp}(T_{\mathcal{A},1}, 1^{\delta^{-1}}) - \mathsf{capp}(T_{\mathcal{A},2}, 1^{\delta^{-1}})\right| - 2(\delta + \eta) \cdot 2^\ell && \text{(“prBPP} = \mathsf{prP}\text{”)} \\
&\geq (2\delta + \beta + \varepsilon) \cdot 2^\ell - 2(\delta + \eta) \cdot 2^\ell && \text{(Equation (7.3) does not hold)} \\
&\geq (\beta + \varepsilon - 2\eta) \cdot 2^\ell.
\end{aligned}$$

Let $K \subseteq \{0,1\}^\ell$ be the bounded definable set defined by $T_{C'_1,C'_2}$, namely, $x \in K$ if and only if $C'_1(x) \neq C'_2(x)$. By Lemma 3.4 (1), there is $s_K \leq 2^\ell$ such that $K \approx_\eta s_K$. Recall that $\mathsf{P}_\eta[T_{C'_1,C'_2}] \leq 2\eta + \varepsilon$. By “prBPP = prP”, we have that $|s_K - \varepsilon \cdot 2^\ell| \leq 3\eta \cdot 2^\ell$.

Proof via APC_1 Counting Mechanism. Now, let $X_1', X_2' \subseteq \{0,1\}^n$ be the following bounded definable sets: $X_i' := \{x \in \{0,1\}^n \mid f_i(x) \in X_i\}$ for $i \in \{1,2\}$. Note that for $i \in \{1,2\}$, $g_i : X_i \twoheadrightarrow X_i'$ and $f_i : X_i' \twoheadrightarrow X_i$, we know that $X_i \approx_0 X_i'$ (see Definitions 3.1 and 3.2). Therefore, by Proposition 3.3 (3), we have $X_i' \approx_\eta s_i$ for $i \in \{1,2\}$.

Let $Y_i := X_i' \cap K$, and $Z_i := X_i' \setminus K$ for $i \in \{1,2\}$. It turns out that $Z_1 = Z_2$, as $C_1'(x) = C_2'(x)$ for every $x \in \{0,1\}^\ell \setminus K$. Also, we know that $Y_i \lesssim_0 K \lesssim_\eta (\varepsilon + 3\eta) \cdot 2^\ell$. By Lemma 3.4 (1), there is $t_0, t_1, t_2 \leq 2^\ell$ such that

- $Z_1 = Z_2 \approx_\eta t_0$;
- $Y_i \approx_\eta t_i$ for $i \in \{1,2\}$.

By Lemma 3.4 (2), we know that $t_1, t_2 \leq (\varepsilon + 6\eta) \cdot 2^\ell$.

By Lemma 3.4 (5), we have

$$X_1' = Y_1 \cup Z_1 \lesssim_\eta Y_1 \uplus t_1 + t_0 \lesssim_{\varepsilon+7\eta} t_0 \lesssim_\eta Z_2 \lesssim_0 Y_2 \cup Z_2 = X_2';$$
$$X_2' = Y_2 \cup Z_2 \lesssim_\eta Y_2 \uplus t_2 + t_0 \lesssim_{\varepsilon+7\eta} t_0 \lesssim_\eta Z_1 \lesssim_0 Y_1 \cup Z_1 = X_1'.$$

This implies that $X_1' \approx_{\varepsilon+9\eta} X_2'$ and subsequently $s_1 \approx_{\varepsilon+11\eta} s_2$. By Lemma 3.4 (2), we know that

$$(\beta + \varepsilon - 2\eta) \cdot 2^\ell \leq |s_1 - s_2| \leq (\varepsilon + 12\eta) \cdot 2^\ell$$

This leads to a contradiction if we set $\eta := \beta/100$. □

Next, we prove a few convenient lemmas to reason about indistinguishable distributions.

Lemma 7.11 (Averaging Argument for Indistinguishable Distribution)**.** $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves the following sentence. Let $\ell, n \in \mathsf{Log}$, $G_1, G_2 : \{0,1\}^\ell \to \{0,1\}^n$ be circuits, and $\alpha \subseteq [\ell]$ be a set of indices. Let $\mathcal{H}_1, \mathcal{H}_2$ be the distributions defined by G_1 and G_2, and for every assignment ρ to input bits in α, let $\mathcal{H}_1|_\rho, \mathcal{H}_2|_\rho$ be the distributions defined by $G_1|_\rho$ and $G_2|_\rho$.*

Let $t \in \mathsf{Log}$ and $\varepsilon \in [0,1]$. Suppose that $\mathcal{H}_1|_\rho \approx_{t,\varepsilon} \mathcal{H}_2|_\rho$ for every assignment ρ to input bits in α, then $\mathcal{H}_1 \approx_{t,\varepsilon} \mathcal{H}_2$.

Proof. We argue in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. Fix $\ell, n \in \mathsf{Log}$, $\alpha \subseteq [\ell]$, $t \in \mathsf{Log}$ and $\varepsilon \in [0,1]$. We prove by contrapositive. Assume that $\mathcal{H}_1 \approx_{t,\varepsilon} \mathcal{H}_2$ does not hold, there is a circuit $\mathcal{A}$ of size at most t and $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$ such that

$$\left|\Pr_\delta[\mathcal{A} \circ G_1] - \Pr_\delta[\mathcal{A} \circ G_2]\right| > 2\delta + \beta + \varepsilon.$$

Let $\eta^{-1} \in \mathsf{Log}$ be a parameter to be determined later. Fix the circuit $\mathcal{A}$, and by Precision Consistency of CAPP, we have

$$\left|\Pr_\eta[\mathcal{A} \circ G_1] - \Pr_\eta[\mathcal{A} \circ G_2]\right| > \beta - 4\eta + \varepsilon.$$

By the Averaging Argument on Two Circuits, there is an assignment ρ to the input bits in α such that

$$\left|\Pr_\eta[\mathcal{A} \circ G_1|_\rho] - \Pr_\eta[\mathcal{A} \circ G_2|_\rho]\right| > \beta - 9\eta + \varepsilon \geq 2\eta + \eta + \varepsilon,$$

where the last inequality follows by setting $\eta := \beta/20$. This shows that $\mathcal{H}_1|_\rho \approx_{t,\varepsilon} \mathcal{H}_2|_\rho$ does not hold and completes the proof. □

Let $G : \{0,1\}^\ell \to \{0,1\}^n, H : \{0,1\}^{\ell'} \to \{0,1\}^{n'}$ are circuits, we define (G, H) as the circuit that given the concatenation of $x \in \{0,1\}^\ell, x' \in \{0,1\}^{\ell'}$, outputs the concatenation of $G(x)$ and $H(x')$.

Lemma 7.12 (Reduction Lemma). $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves the following sentence. Let* $\ell, n, m, p \in \log$, $G_1, G_2 : \{0,1\}^\ell \to \{0,1\}^n$ *be circuits. Let* $\mathcal{A} : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^p$ *be a circuit of size at most* t', *and define* $\mathcal{A}_b : \{0,1\}^\ell \times \{0,1\}^m \to \{0,1\}^p$ *as the following circuit:*

- *Parse input* $z = (x, \mathsf{sd})$, *where* $x \in \{0,1\}^\ell, \mathsf{sd} \in \{0,1\}^m$.
- *Output* $\mathcal{A}(G_b(x), \mathsf{sd})$.

Let $\mathcal{H}_b$ *be the distributions defined by* G_b *and* $\mathcal{H}'_b$ *be the distribution defined by* $\mathcal{A}_b$.

Let $t, t' \in \mathsf{Log}$ *and* $\varepsilon \in [0,1)$. *Suppose* $\mathcal{H}_1 \approx_{t+t',\varepsilon} \mathcal{H}_2$. *Then,* $\mathcal{H}'_1 \approx_{t,\varepsilon} \mathcal{H}'_2$.

Proof. Let $\alpha = \{n+1, \ldots, m\}$ be the indices of the inputs to $\mathcal{A}_b$. By the Averaging Argument for Indistinguishable Distribution, it suffices to show that for any assignment ρ to inputs in α:

$$\mathcal{H}'_1|_\rho \approx_{t,\varepsilon} \mathcal{H}'_2|_\rho.$$

Suppose that for some ρ, the above equation does not hold. Then, there is a circuit $\mathcal{B}$ of size at most t, and $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$ such that

$$\left|\Pr_\delta[\mathcal{B} \circ (\mathcal{A}_1|_\rho)] - \Pr_\delta[\mathcal{B} \circ (\mathcal{A}_2|_\rho)]\right| > 2\delta + \beta + \varepsilon.$$

By definition of α, note that $\mathcal{A}_b|_\rho$ and $(\mathcal{A}|_\rho) \circ G_b$ are functionally equivalent, and this is provable in PV. Rewrite $\mathcal{B}' = \mathcal{B} \circ (\mathcal{A}|_\rho)$ (by associativity of $\circ$), we have that

$$\left|\Pr_\delta[\mathcal{B}' \circ G_1] - \Pr_\delta[\mathcal{B}' \circ G_2]\right| > 2\delta + \beta + \varepsilon.$$

Since $\mathcal{B}'$ has size at most $t + t'$ and $\mathcal{H}_1 \approx_{t,\varepsilon} \mathcal{H}_2$, we arrive at a contradiction. □

Corollary 7.13 (Independent Side-Information Corollary). $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves the following sentence. Let* $\ell, \ell', n, n' \in \mathsf{Log}$, $G_1, G_2 : \{0,1\}^\ell \to \{0,1\}^n, G' : \{0,1\}^{\ell'} \times \{0,1\}^{n'}$ *be circuits. Suppose* G' *has size* s. *Let* $\mathcal{H}_1, \mathcal{H}_2$ *are distributions defined by* G_1, G_2, *and* $\mathcal{H}'_1, \mathcal{H}'_2$ *be the distributions defined by* $(G_1, G'), (G_2, G')$. *Then for every* $t \in \mathsf{Log}$, $\varepsilon \in [0,1]$, *if* $\mathcal{H}_1 \approx_{t+s+n,\varepsilon} \mathcal{H}_2$, *then* $\mathcal{H}'_1 \approx_{t,\varepsilon} \mathcal{H}'_2$.

Proof Sketch. We argue this by invoking the Reduction Lemma. Let $\mathcal{A} : \{0,1\}^n \times \{0,1\}^{\ell'} \to \{0,1\}^n \times \{0,1\}^{n'}$ be the circuit which takes as input (x, sd) where $x \in \{0,1\}^n, \mathsf{sd} \in \{0,1\}^{\ell'}$, and outputs $x, G'(\mathsf{sd})$. It is easy to see that $\mathcal{A} \circ G_b \equiv (G_b, G')$. Moreover, $\mathcal{A}$ has size at most $s + n$. Therefore, by the Reduction Lemma, we have that $\mathcal{H}'_1 \approx_{t,\varepsilon} \mathcal{H}'_2$. □

Corollary 7.14 (Product Hybrid Lemma). $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves the following sentence. Let* $\ell, \ell', n, n', s \in \mathsf{Log}$, *and let* $G_1, G_2 : \{0,1\}^\ell \to \{0,1\}^n$ *and* $G'_1, G'_2 : \{0,1\}^{\ell'} \to \{0,1\}^{n'}$ *be circuits of size at most* s. *Let* $\mathcal{H}_b$ *be the distribution defined by* G_b, *and* $\mathcal{H}'_b$ *be the distribution defined by* G'_b. *Let* $\mathcal{H}_\Pi$ *be the distribution defined by* (G_1, G_2), *and* $\mathcal{H}'_\Pi$ *be the distribution defined by* (G'_1, G'_2).

Let $t \in \mathsf{Log}$ *and* $\varepsilon_1, \varepsilon_2 \in [0,1)$. *Suppose* $\mathcal{H}_1 \approx_{t+2s,\varepsilon_1} \mathcal{H}'_1$ *and* $\mathcal{H}_2 \approx_{t+2s,\varepsilon_2} \mathcal{H}'_2$. *Then,* $\mathcal{H}_\Pi \approx_{t,\varepsilon_1+\varepsilon_2} \mathcal{H}'_\Pi$.

Proof Sketch. We obtain this by applying the Independent Side-Information Corollary twice, and then using a hybrid argument. □

In the rest of the paper, we will use $\mathcal{H}_1 \times \mathcal{H}_2$ to denote the product distribution of $\mathcal{H}_1$ and $\mathcal{H}_2$, where $\mathcal{H}_1, \mathcal{H}_2$ are distributions defined by explicit circuits.

7.5 Tools for Search Security Games

Complementation of game. Let $\mathcal{G} = (C_1, C_2, 0)$ be a search security game. We define its complement security game as $\overline{\mathcal{G}} := (C_1, \overline{C}_2, 0)$. Intuitively, an adversary wins in $\mathcal{G}$ if and only if it loses in $\overline{\mathcal{G}}$. The following lemma helps to calculate the advantage of the complement game.

Lemma 7.15 (Game Complement Lemma)**.** $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves the following statement. Let* $\mathcal{G} = (C_1, C_2, 0)$ *be a security game, and* $\mathcal{A}$ *be any adversary in* $\mathcal{G}$*. Let* $\overline{\mathcal{G}} = (C_1, \overline{C_2}, 0)$*. If* $\mathsf{Adv}_{\mathcal{G}}(\mathcal{A}) \leq \varepsilon$*, then* $\mathsf{Adv}_{\overline{\mathcal{G}}}(\mathcal{A}) \geq 1 - \varepsilon$*.*

Search game complement. We argue in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. Fix $\mathcal{G} = (C_1, C_2, 0)$, $\overline{\mathcal{G}} = (C_1, \overline{C_2}, 0)$, and the adversary $\mathcal{A}$. Let $T_{\mathcal{G},\mathcal{A}}$ be the circuit defined in Definition 5.2, then assuming $\mathsf{Adv}_{\mathcal{G}}(\mathcal{A}) \leq \varepsilon$, we know that for any $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$,

$$\Pr_\delta[T_{\mathcal{G},\mathcal{A}}] \leq \varepsilon + \delta + \beta. \tag{7.4}$$

Our goal is to prove that $\mathsf{Adv}_{\overline{\mathcal{G}}}(\mathcal{A}) \geq 1 - \varepsilon$. That is, for any $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$,

$$\Pr_\delta\left[T_{\overline{\mathcal{G}},\mathcal{A}}\right] \geq 1 - \varepsilon - \delta - \beta. \tag{7.5}$$

Fix $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$ and let $\eta^{-1} \in \mathsf{Log}$ be a parameter to be determined. One can observe that $T_{\mathcal{G},\mathcal{A}}(x)$ if and only if $T_{\overline{\mathcal{G}},\mathcal{A}}(x) = 0$ for any input x. Therefore, by the Complementation Consistency of CAPP,

$$\Pr_\eta\left[T_{\overline{\mathcal{G}},\mathcal{A}}\right] \geq 1 - \Pr_\eta[T_{\mathcal{G},\mathcal{A}}] - 3\eta \geq 1 - (\varepsilon + 2\eta) - 3\eta \geq 1 - \varepsilon - 5\eta,$$

where the second inequality follows from Equation (7.4). Then by Precision Consistency of CAPP, we have

$$\Pr_\delta\left[T_{\overline{\mathcal{G}},\mathcal{A}}\right] \geq \Pr_\eta\left[T_{\overline{\mathcal{G}},\mathcal{A}}\right] - \delta - 2\eta \geq 1 - \varepsilon - \delta - 6\eta. \tag{7.6}$$

This implies Equation (7.5) by setting $\eta := \beta/6$. □

Composition of games. The following lemma shows that if two search games use the same challenger circuits C_2 and different but indistinguishable samplers C_1, C_1', the advantages of any adversary would be similar in these two games.

Lemma 7.16 (Game Composition Lemma)**.** $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves the following sentence. Let* $r, n, m, t, s \in \mathsf{Log}$*, and* $\varepsilon_1, \varepsilon_2 \in [0, 1]$*. Suppose* $C_1, C_1' : \{0,1\}^r \to \{0,1\}^n \times \{0,1\}^m$ *correspond to* $(t+s, \varepsilon_1)$*-indistinguishable distributions. Let* $\mathcal{G} = (C_1, C_2, c)$ *and* $\mathcal{G}' = (C_1', C_2, c)$*, where* $C_2 : \{0,1\}^\ell \times \{0,1\}^m \to \{0,1\}$ *is a circuit of size* s *and* $c \in [0, 1]$*. Then, we have that for all adversaries* $\mathcal{A}$ *of size* t*,*

$$\mathsf{Adv}_{\mathcal{G}}(\mathcal{A}) \leq \varepsilon_2 \Rightarrow \mathsf{Adv}_{\mathcal{G}'}(\mathcal{A}) \leq \varepsilon_1 + \varepsilon_2, \tag{7.7}$$

$$\mathsf{Adv}_{\mathcal{G}}(\mathcal{A}) \geq \varepsilon_2 \Rightarrow \mathsf{Adv}_{\mathcal{G}'}(\mathcal{A}) \geq \varepsilon_2 - \varepsilon_1. \tag{7.8}$$

Subsequently, if $\mathcal{G}$ *is* (t, ε_2)*-secure,* $\mathcal{G}'$ *is* $(t, \varepsilon_1 + \varepsilon_2)$*-secure.*

Proof. We argue in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. Fix $r, n, m, t, s \in \mathsf{Log}, \varepsilon_1, \varepsilon_2 \in [0, 1], \mathcal{G}, \mathcal{G}'$ and the probabilistic circuit $\mathcal{A}$. We will only prove Equation (7.7), while (7.8) can be proved using the same argument. Suppose that $\mathsf{Adv}_{\mathcal{G}}(\mathcal{A}) \leq \varepsilon_2$, our goal is to prove that $\mathsf{Adv}_{\mathcal{G}'}(\mathcal{A}) \leq \varepsilon_1 + \varepsilon_2$; in more detail, let $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$,

$$\Pr_\delta[T_{\mathcal{G}',\mathcal{A}}] \leq \delta + \beta + \varepsilon_1 + \varepsilon_2, \tag{7.9}$$

where $T_{\mathcal{G}',\mathcal{A}}$ is the circuit defined in Definition 5.2.

Let $\eta^{-1} \in \mathsf{Log}$ be a parameter to be determined later. Let $\mathcal{B}$ be the probabilistic circuit that, given any $(\mathsf{chall}, \mathsf{state}) \in \{0,1\}^n \times \{0,1\}^m$ as input and sd as its seed, outputs $C_2(\mathcal{A}(\mathsf{chall}; \mathsf{sd}), \mathsf{state})$. Notice that $T_{\mathcal{G},\mathcal{A}}$ can be viewed $\mathcal{B} \circ C$ and $T_{\mathcal{G}',\mathcal{A}}$ can be viewed as $\mathcal{B} \circ C'$. In addition, the size of $\mathcal{B}$ is at most $t + s$. As C, C' define $(t+s, \varepsilon_1)$-indistinguishable distributions, we can conclude that

$$\left|\Pr_\eta[\mathcal{B} \circ C] - \Pr_\eta[\mathcal{B} \circ C']\right| \leq 3\eta + \varepsilon_1.$$

Moreover, as $\mathsf{Adv}_{\mathcal{G}}(\mathcal{A}) \leq \varepsilon_2$, we know by definition that $\Pr_\eta[\mathcal{B} \circ C] \leq 2\eta + \varepsilon_2$. By combining these two inequalities we conclude that

$$\Pr_\eta[T_{\mathcal{G},\mathcal{A}}] = \Pr_\eta[\mathcal{B} \circ C'] \leq 5\eta + \varepsilon_1 + \varepsilon_2.$$

By Precision Consistency of CAPP, we can further conclude that

$$\Pr_\delta[T_{\mathcal{G},\mathcal{A}}] \leq \delta + 5\eta + \varepsilon_1 + \varepsilon_2.$$

This implies immediately Equation (7.9) by setting $\eta := \beta/5$. □

Lemma 7.17 (Adversary Indistinguishability Lemma)**.** *The following is provable in* $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$"*. Let $\mathcal{G} = (C_1, C_2, c)$ be a search game, and $\mathcal{A}_1, \mathcal{A}_2 : \{0,1\}^n \times \{0,1\}^{r'} \to \{0,1\}^m$ be adversaries in $\mathcal{G}$. Suppose that the following conditions hold for some $t \in \mathsf{Log}$ and $\varepsilon \in [0,1]$.*

- *For every $x \in \{0,1\}^n$, let $\mathcal{H}_{x,b}$ be the distribution corresponding to $\mathcal{A}_b(x, \cdot)$ for $b \in \{1,2\}$, then $\mathcal{H}_{x,1}$ and $\mathcal{H}_{x,2}$ are (t, ε)-indistinguishable.*
- $|C_1| + |C_2| \leq t$.

Then for every $\tau \in [0,1]$,

$$\mathsf{Adv}_{\mathcal{G}}[\mathcal{A}_1] \geq \tau \implies \mathsf{Adv}_{\mathcal{G}}[\mathcal{A}_2] \geq \tau - \varepsilon$$
$$\mathsf{Adv}_{\mathcal{G}}[\mathcal{A}_1] \leq \tau \implies \mathsf{Adv}_{\mathcal{G}}[\mathcal{A}_2] \leq \tau + \varepsilon$$

Proof Sketch. This follows directly from the definition of indistinguishability (see Definition 4.13) and the advantage of search games (see Definition 5.2), which is left as an exercise. □

Disjunction of games. The following lemma helps to calculate the advantage in games defined as the OR of several other games. Formally:

Lemma 7.18 (Game Union Bound Lemma)**.** $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$" *proves the following sentence. Let $m \in \mathsf{Log}$, $\mathcal{G}_1 = (C_1, C_2^{(1)}, 0)$, $\mathcal{G}_2 = (C_1, C_2^{(2)}, 0), \ldots, \mathcal{G}_m = (C_1, C_2^{(m)}, 0))$ be search games with the same sampler C_1.*

Let $C_2(x) := \bigvee_{i=1}^m C_2^{(i)}(x)$ be the OR of the circuits $C_2^{(1)}, \ldots, C_2^{(m)}$ and $\mathcal{G} := (C_1, C_2, 0)$. Suppose that $\mathcal{G}_i$ is (t, ε_i)-secure for every $i \in [m]$, where $t \in \mathsf{Log}$ and $\varepsilon_1, \ldots, \varepsilon_m \in [0,1]$, $\varepsilon := \varepsilon_1 + \varepsilon_2 + \cdots + \varepsilon_m$, then $\mathcal{G}$ is (t, ε)-secure.

Proof. We argue in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". Fix $m \in \mathsf{Log}$ and circuits $C_1, C_2^{(1)}, \ldots, C_2^{(m)}$, and C_2. Let $t \in \mathsf{Log}$, $\varepsilon_1, \ldots, \varepsilon_m \in [0,1]$, and $\varepsilon := \varepsilon_1 + \cdots + \varepsilon_m$. Fix any adversary $\mathcal{A}$ of size at most t, our goal is to prove that $\mathsf{Adv}_{\mathcal{G}}(\mathcal{A}) \leq \varepsilon$. In more detail, we will prove for every $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$ that

$$\Pr_\delta[T_{\mathcal{G},\mathcal{A}}] \leq \delta + \beta + \varepsilon, \tag{7.10}$$

where $T_{\mathcal{G},\mathcal{A}}$ is the circuit in Definition 5.2.

Fix $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$ and let $\eta^{-1} \in \mathsf{Log}$ be a parameter to be determined later. As $\mathcal{G}_i$ is (t, ε_i)-secure, we know that $\mathsf{Adv}_{\mathcal{G}_i}(\mathcal{A}) \leq \varepsilon_i$, which subsequently implies that

$$\Pr_\eta[T_{\mathcal{G}_i,\mathcal{A}}] \leq 2\eta + \varepsilon_i. \tag{7.11}$$

By the definition of the circuit $T_{\mathcal{G},\mathcal{A}}$ and the game $\mathcal{G}$, we can observe that $T_{\mathcal{G},\mathcal{A}}$ is functionally equivalent to the disjunction of $T_{\mathcal{G}_1,\mathcal{A}}, \ldots, T_{\mathcal{G}_m,\mathcal{A}}$. Therefore by the Union Bound (general form), we have

$$\Pr_\eta[T_{\mathcal{G},\mathcal{A}}] \leq \sum_{i=1}^{m} \Pr_\eta[T_{\mathcal{G}_i,\mathcal{A}}] + \eta \cdot (m+2) \leq \varepsilon + 3\eta \cdot (m+2). \tag{7.12}$$

The theorem then follows by Precision Consistency of CAPP when we set $\eta := \beta/(3m+6)$. □

Reduction between games. The following lemma captures a type of reductions between search games, analogues to the Reduction Lemma.

Lemma 7.19 (Game Reduction Lemma). *The following sentence is provable in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. *Let* $\mathcal{G} = (C_1, C_2, c)$ *and* $\mathcal{G}' = (C_1', C_2', c)$ *be two search games with the same threshold* c, $\mathcal{A}$ *and* $\mathcal{A}'$ *be adversaries in* $\mathcal{G}$ *and* $\mathcal{G}'$, *respectively, where*

- $C_1 : \{0,1\}^r \to \{0,1\}^n \times \{0,1\}^m$, $C_1' : \{0,1\}^{r+r'} \to \{0,1\}^n \times \{0,1\}^m$;
- $\mathcal{A} : \{0,1\}^n \times \{0,1\}^{r_\mathcal{A}} \to \{0,1\}^\ell$, $\mathcal{A}' : \{0,1\}^n \times \{0,1\}^{r_\mathcal{A} + r_{\mathcal{A}'}} \to \{0,1\}^\ell$.

Suppose that for every $\mathsf{sd}_1, \mathsf{sd}_1', \mathsf{sd}_\mathcal{A}, \mathsf{sd}_\mathcal{A}'$,

$$T_{\mathcal{G},\mathcal{A}}(\mathsf{sd}_1, \mathsf{sd}_\mathcal{A}) = 1 \to T_{\mathcal{G}',\mathcal{A}'}(\mathsf{sd}_1 \| \mathsf{sd}_1', \mathsf{sd}_\mathcal{A} \| \mathsf{sd}_{\mathcal{A}'}) = 1, \tag{7.13}$$

where $T_{\mathcal{G},\mathcal{A}}, T_{\mathcal{G}',\mathcal{A}'}$ *are the circuits in Definition 5.2. Then for every* $\varepsilon \in [0,1]$, $\mathsf{Adv}_\mathcal{G}[\mathcal{A}] \geq \varepsilon$ *implies that* $\mathsf{Adv}_{\mathcal{G}'}[\mathcal{A}'] \geq \varepsilon$.

Proof. We argue in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. Fix $\mathcal{G}, \mathcal{G}', \mathcal{A}, \mathcal{A}'$ and $\varepsilon \in [0,1]$ satisfying the conditions above. Since $\mathsf{Adv}_\mathcal{G}[\mathcal{A}] \geq \varepsilon$, we know that for every $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$,

$$\Pr_\delta[T_{\mathcal{G},\mathcal{A}}] \geq c + \varepsilon - (\delta + \beta). \tag{7.14}$$

Our goal is to prove that $\mathsf{Adv}_{\mathcal{G}'}[\mathcal{A}'] \geq \varepsilon$.

Fix any $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$ and let $\eta^{-1} \in \mathsf{Log}$ be a parameter to be determined later. Suppose, towards a contradiction, that

$$\Pr_\delta[T_{\mathcal{G}',\mathcal{A}'}] < c + \varepsilon - (\delta + \beta),$$

by Precision Consistency of CAPP, we have $\Pr_\eta[T_{\mathcal{G}',\mathcal{A}'}] < c + \varepsilon - (\beta - 2\eta)$. By the Averaging Argument, there is an assignment ρ to $\mathsf{sd}_1', \mathsf{sd}_{\mathcal{A}'}$ in the seed of $T_{\mathcal{G}',\mathcal{A}'}$ such that

$$\Pr_\eta[T_{\mathcal{G}',\mathcal{A}'}|_\rho] \leq \Pr_\eta[T_{\mathcal{G}',\mathcal{A}'}] + 3\eta \leq c + \varepsilon - (\beta - 5\eta).$$

Note that both $T_{\mathcal{G},\mathcal{A}}$ and $T_{\mathcal{G}',\mathcal{A}'}|_\rho$ take $(\mathsf{sd}_1, \mathsf{sd}_\mathcal{A})$ as input. Moreover, by Equation (7.13), we have that $T_{\mathcal{G},\mathcal{A}}(x) = 1$ implies $T_{\mathcal{G}',\mathcal{A}'}|_\rho(x) = 1$ for any input x. Thus by the Monotonicity of CAPP, we have

$$\Pr_\eta[T_{\mathcal{G},\mathcal{A}}] \leq \Pr_\eta[T_{\mathcal{G}',\mathcal{A}'}|_\rho] + 3\eta \leq c + \varepsilon - (\beta - 8\eta).$$

This contradicts to Equation (7.14) (instantiated with $\delta = \beta = \eta$) by setting $\eta := \beta/20$. □

Error reduction via repetition. We consider the standard trick of error reduction via repetition. We first prove a general statement that works for any circuit.

Lemma 7.20. *The following statement is provable in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. *Let* $n, k, \delta^{-1}, \beta^{-1} \in \mathsf{Log}$ *and* $C : \{0,1\}^n \to \{0,1\}$, *and* $C^{\wedge k} : \{0,1\}^{nk} \to \{0,1\}$ *be a circuit that satisfies*

$$C^{\wedge k}(x_1, \ldots, x_k) = \bigwedge_{i=1}^{k} C(x_i).$$

Suppose that $\Pr_\delta[C] \leq p$. *Then we have that*

$$\Pr_\delta[C^{\wedge k}] \leq (p + \delta + \beta)^k + \delta + \beta.$$

Proof. We argue in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. Fix $n, k, \delta^{-1}, \beta^{-1} \in \mathsf{Log}$ and circuit C and $C^{\vee k}$. Let $\eta^{-1} \in \mathsf{Log}$ be a parameter to be determined later. Let X and Y be the sets defined by C and $C^{\wedge k}$, respectively. By Lemma 3.4 (1), there are $s \leq 2^n$ and $t \leq 2^{nk}$ such that $X \approx_\eta s$ and $Y \approx_\eta t$. By definition, for some v, there is a surjective function

$$f : v \times (s + \eta \cdot 2^n) \twoheadrightarrow v \times X$$

We first prove that $Y \lesssim_0 (s+\eta\cdot 2^n)^k$ (note that $|s^k| \leq k \log s \in \mathsf{Log}$ is feasible). We identify s^k with the set of tuples $(z_1, \ldots, z_k)$ where $z_1, \ldots, z_k \in s$. Consider the following function $g : v^k \times ((s + \eta \cdot 2^n)^k) \twoheadrightarrow v^k \times Y$. Given any $(v_1, \ldots, v_k) \times (z_1, \ldots, z_k)$, where $z_1, \ldots, z_k \in s + \eta \cdot 2^n$, it outputs

$$(v_1, \ldots, v_k) \times (f(v_1, z_1), f(v_2, z_2), \ldots, f(v_k, z_k)).$$

It is clear that the function is onto Y, and thus $Y \lesssim_0 (s + \eta \cdot 2^n)^k$. Subsequently, $t \lesssim_\eta Y \lesssim_0 (s + \eta \cdot 2^n)^k$, by Lemma 3.4 (2),

$$t \leq (s + \eta \cdot 2^n)^k + 2\eta \cdot 2^{nk}.$$

Finally, as $X \approx_\eta s$, by “$\mathsf{prBPP} = \mathsf{prP}$” we know that $|s - p \cdot 2^n| \leq (\delta + \eta) \cdot 2^n$, and this implies that

$$t \cdot 2^{-nk} \leq (p + \delta + 2\eta)^k + 2\eta.$$

Finally, by $Y \approx_\eta t$ and “$\mathsf{prBPP} = \mathsf{prP}$”, we have

$$\Pr_\delta[C^{\wedge k}] \leq (p + \delta + 2\eta)^k + (\delta + 3\eta).$$

The lemma follows by setting $\eta := \beta/5$. □

The following is a simple corollary by the Complementation Consistency of CAPP.

Corollary 7.21. *The following statement is provable in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. *Let* $n, k, \delta^{-1}, \beta^{-1} \in \mathsf{Log}$ *and* $C : \{0,1\}^n \to \{0,1\}$, *and* $C^{\vee k} : \{0,1\}^{nk} \to \{0,1\}$ *be a circuit that satisfies*

$$C^{\vee k}(x_1, \ldots, x_k) = \bigvee_{i=1}^{k} C(x_i).$$

Suppose that $\Pr_\delta[C] \geq p$. *Then we have that*

$$\Pr_\delta[C^{\vee k}] \geq 1 - (1 - p + \delta + \beta)^k - (\delta + \beta).$$

Now we are ready to state our repetition lemma for search games. Let $\mathcal{G} = (\mathcal{C}_1, \mathcal{C}_2, 0)$ be a game and $k \in \mathsf{Log}$, we define the *k-fold repetition* of $\mathcal{G}$ as the following game $\mathcal{G}^{\vee k} = (\mathcal{C}_1^{\times k}, \mathcal{C}_2^{\vee k}, 0)$, where:

- $\mathcal{C}_1^{\times k}$ is the circuit that, on input $\mathsf{sd}_1, \ldots, \mathsf{sd}_k$, compute $(\mathsf{chall}_i, \mathsf{state}_i) \leftarrow \mathcal{C}_1(\mathsf{sd}_i)$. Set $\mathsf{chall} = (\mathsf{chall}_1, \ldots, \mathsf{chall}_{Ct})$ and $\mathsf{state} = (\mathsf{state}_1, \ldots, \mathsf{state}_k)$.
- $\mathcal{C}^{\vee k}$ is the circuit that, on input $(\mathsf{ans}_1, \ldots, \mathsf{ans}_k, \mathsf{state}_1, \ldots, \mathsf{state}_k)$, outputs 1 if for any $i \in [k]$, $\mathcal{C}_2(\mathsf{ans}_i, \mathsf{state}_i) = 1$, and outputs 0 otherwise.

Similarly, for every adversary $\mathcal{A}$ in $\mathcal{G}$, we define $\mathcal{A}^{\times k}$ as the adversary that simulates $\mathcal{A}$ in each of the k sub-games. Then the following lemma calculates the advantage of an adversary in k-fold repetition of a game.

Lemma 7.22 (Error Reduction Lemma)**.** *The following sentence is provable in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$*. Let $\mathcal{G} = (\mathcal{C}_1, \mathcal{C}_2, 0)$ be a search game and $\mathcal{A}$ be an adversary such that $\mathsf{Adv}_{\mathcal{G}}[\mathcal{A}] \geq p$. Then for $k \in \mathsf{Log}$ and any $\beta^{-1} \in \mathsf{Log}$,*

$$\mathsf{Adv}_{\mathcal{G}^{\vee k}}[\mathcal{A}^{\times k}] \geq 1 - (1 - p + \beta)^k.$$

Proof. We argue in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. Fix any game $\mathcal{G}$, adversary $\mathcal{A}$, and $k \in \mathsf{Log}$. Suppose that $\mathsf{Adv}_{\mathcal{G}}[\mathcal{A}] \geq p$, then by definition, for every $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$,

$$\Pr_\delta[T_{\mathcal{G},\mathcal{A}}] \geq p - \delta - \beta,$$

where $T_{\mathcal{G},\mathcal{A}}$ is the circuit in Definition 5.2.

Now we prove that for every $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$,

$$\Pr_\delta[T_{\mathcal{G}^{\vee k},\mathcal{A}^{\times k}}] \geq 1 - (1 - p + \beta)^k - (\delta + \beta). \tag{7.15}$$

Fix $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$ and let $\eta^{-1} \in \mathsf{Log}$ be a parameter to be determined. Notice that $T_{\mathcal{G}^{\vee k},\mathcal{A}^{\times k}} = T^{\vee k}_{\mathcal{G},\mathcal{A}}$, by Corollary 7.21, we know that

$$\Pr_\eta[T_{\mathcal{G}^{\vee k},\mathcal{A}^{\times k}}] \geq 1 - (1 - (p - 2\eta) + 2\eta)^k - 2\eta \geq 1 - (1 - p + 4\eta)^k - 2\eta.$$

Then Equation (7.15) follows by Precision Consistency of CAPP when we set $\eta := \beta/5$. □

7.6 Tools for Almost Identical Distributions

Let $\mathcal{H}_1, \mathcal{H}_2$ be distributions defined by circuits $H_1, H_2 : \{0,1\}^r \to \{0,1\}^n$, respectively, and $E : \{0,1\}^r \to \{0,1\}$ be a circuit. We use $\mathcal{H}_1|_E + \mathcal{H}_2|_{\neg E}$ to denote the distribution defined by the following circuit: Given $x \in \{0,1\}^r$, it outputs $H_1(x)$ if $E(x) = 1$, and $H_2(x)$ if $E(x) = 0$. The following lemma shows that if E accepts most of its input strings, $\mathcal{H}_1$ is almost identical to $\mathcal{H}_1|_E + \mathcal{H}_2|_{\neg E}$.

Lemma 7.23 (Mixture Lemma)**.** $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves the following statement. Let $n, r, \delta^{-1}, \beta^{-1}, \varepsilon \in \mathsf{Log}$ and $H_1, H_2 : \{0,1\}^r \to \{0,1\}^n$, $E : \{0,1\}^n \to \{0,1\}$. Let $\mathcal{H}_1, \mathcal{H}_2$ be the distributions defined by H_1 and H_2, respectively. Suppose that $\Pr_\delta[\neg E] \leq \varepsilon$, then $\mathcal{H}_1$ and $\mathcal{H}_1|_E + \mathcal{H}_2|_{\neg E}$ are $(\varepsilon + \delta + \beta)$-almost identical.*

Proof. We argue in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. Fix $n, r, \delta^{-1}, \beta^{-1} \in \mathsf{Log}$ and H_1, H_2, E. Let H_E be the circuit that defines the distribution $\mathcal{H}_1|_E + \mathcal{H}_2|_{\neg E}$, our goal is to prove that if $\Pr_\delta[\neg E] \leq \varepsilon$, then for every $\delta'^{-1}, \beta'^{-1} \in \mathsf{Log}$,

$$\Pr_{\delta'}[T_{H_1,H_E}] \leq (\varepsilon + \delta + \beta) + \delta' + \beta', \tag{7.16}$$

where $T_{H_1,H_E} : \{0,1\}^r \to \{0,1\}$ is the circuit that, given x, outputs 1 if and only if $H_1(x) \neq H_E(x)$.

Let $X_{\neg E} \subseteq \{0,1\}^n$ be the bounded set defined by $\neg E$, and X_T be the bounded set defined by T_{H_1,H_E}. Note that by the definition of $\mathcal{H}_1|_E + \mathcal{H}_2|_{\neg E}$, we have $X_T \subseteq X_{\neg E}$, and thus $X_T \lesssim_0 X_{\neg E}$. Let $\eta^{-1} \in \mathsf{Log}$ be determined later, and s be a number such that $X_{\neg E} \approx_\eta s$ by Lemma 3.4 (1). By "$\mathsf{prBPP} = \mathsf{prP}$", we have $|\Pr_\delta[\neg E] - s| \leq \delta + \eta$, and subsequently, $s \leq (\eta + \delta + \varepsilon) \cdot 2^n$.

By Proposition 3.3 (3) that $X_T \lesssim_\eta s$. Subsequently, by "$\mathsf{prBPP} = \mathsf{prP}$", $|\Pr_{\delta'}[T_{H_1,H_E}] - s/2^n| \leq \delta' + \eta$. Then we have

$$\Pr_{\delta'}[T_{H_1,H_E}] \leq \frac{s}{2^n} + \delta' + \eta \leq (\delta + \eta + \varepsilon) + \delta' + \eta.$$

Equation (7.16) then follows by setting $\eta := \min\{\beta, \beta'\}$. □

The following is an immediate corollary of the lemma.

Corollary 7.24. $\mathsf{APC}_1 + $ "$\mathsf{prBPP} = \mathsf{prP}$" *proves the following sentence. Let* $r, r', n, m, t, s \in \mathsf{Log}$ *and* $\varepsilon \in [0,1]$. *Suppose that* $\mathcal{G} = (C_1, C_2, 0)$ *is a* (t,ε)*-secure search game, where* $C_1 : \{0,1\}^r \to \{0,1\}^{n+m}$, $C_2 : \{0,1\}^\ell \times \{0,1\}^m \to \{0,1\}$. *Let* $\mathcal{A} : \{0,1\}^n \times \{0,1\}^{r'} \to \{0,1\}^\ell$ *be a probabilistic circuit.*

Let $H_1, H_2 : \{0,1\}^{r+r'} \to \{0,1\}$ *be circuits, and* $\mathcal{H}_1, \mathcal{H}_2$ *be the distributions defined by* H_1, H_2*, respectively. Suppose* $\mathsf{Adv}_\mathcal{G}(\mathcal{A}) \leq \varepsilon$*, then*

$$\mathcal{H}_1 \approx_\varepsilon \mathcal{H}_1|_{\neg T_{\mathcal{G},\mathcal{A}}} + \mathcal{H}_2|_{T_{\mathcal{G},\mathcal{A}}}, \tag{7.17}$$

where $T_{\mathcal{G},\mathcal{A}}$ *is the circuit defined in Definition 5.2.*

Proof. We argue in $\mathsf{APC}_1 + $ "$\mathsf{prBPP} = \mathsf{prP}$". Fix $r, r', n, m, t, s \in \mathsf{Log}$, $\varepsilon \in [0,1]$, $\mathcal{G}, \mathcal{A}, H_1, H_2$. For simplicity, we use H_R to denote the circuit that defines $\mathcal{H}_1|_{T_{\mathcal{G},\mathcal{A}}} + \mathcal{H}_2|_{\neg T_{\mathcal{G},\mathcal{A}}}$, and $T_{H_1,H_R} : \{0,1\}^{r+r'} \to \{0,1\}$ to denote the circuit that, given x, outputs 1 if and only if $H_1(x) \neq H_R(x)$.

Suppose that Equation (7.17) does not hold, we know that there are $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$ such that $\Pr_\delta[T_{H_1,H_R}] > \delta + \beta + \varepsilon$. By definition, we also know that

$$\mathcal{H}_1 \approx_{\varepsilon+\beta/2} \mathcal{H}_1|_{\neg T_{\mathcal{G},\mathcal{A}}} + \mathcal{H}_2|_{T_{\mathcal{G},\mathcal{A}}} \quad \text{does not hold.}$$

Let $\eta := \beta/20$, it follows that $\mathcal{H}_1$ and $\mathcal{H}_1|_{\neg T_{\mathcal{G},\mathcal{A}}} + \mathcal{H}_2|_{T_{\mathcal{G},\mathcal{A}}}$ are not $((\varepsilon + 5\eta) + \eta + \eta)$-almost identical. By the Mixture Lemma, we know that

$$\Pr_\eta[\neg\neg T_{\mathcal{G},\mathcal{A}}] > \varepsilon + 5\eta.$$

It follows from the Global Consistency of CAPP that

$$\Pr_\eta[T_{\mathcal{G},\mathcal{A}}] > \varepsilon + \eta + \eta,$$

which violates the assumption that $\mathsf{Adv}_\mathcal{G}(\mathcal{A}) \leq \varepsilon$ as $\eta^{-1} \in \mathsf{Log}$ (see Definition 5.2). □

8 Fully Homomorphic Encryption

In this section, we formalize fully homomorphic encryption (FHE) security and prove the security of the *secret-key version* of the Gentry, Sahai and Waters [GSW13] FHE scheme in $\mathsf{APC}_1 + $ "$\mathsf{prBPP} = \mathsf{prP}$" $+ \mathsf{LWE}$.

8.1 Definition of Secret Key Gate-by-gate FHE

Syntax. A leveled, gate-by-gate fully homomorphic encryption scheme with key/ciphertext size $\ell = \ell(\lambda, d) = \mathsf{poly}(\lambda, \log d)$ consists of the following polynomial time algorithms:

$\mathsf{FHE.Gen}(1^\lambda, 1^d) \to (\mathsf{sk}, \mathsf{ek})$. This is a probabilistic algorithm that takes as input a security parameter 1^λ and a circuit depth 1^d. It outputs a secret key $\mathsf{sk} = (\mathsf{sk}_0, \ldots, \mathsf{sk}_d) \in \{0,1\}^{\ell(d+1)}$ consists of the i-th layer secret key sk_i for $0 \le i \le d$, and an evaluation key $\mathsf{ek} = (\mathsf{ek}_0, \ldots, \mathsf{ek}_{d-1}) \in \{0,1\}^{\ell d}$ consists of the i-th layer evaluation key ek_i for $0 \le i < d$.

$\mathsf{FHE.Enc}_{\mathsf{sk}}(b) \to c$. This is a probabilistic algorithm that takes as input a secret key sk and a bit $b \in \{0,1\}$. It outputs a level-0 ciphertext $c \in \{0,1\}^\ell$.

We use the notation $\mathsf{FHE.Enc}_{\mathsf{sk}}(x)$ for $x \in \{0,1\}^k$ to denote the concatenation of encryption of each bit in x.

$\mathsf{FHE.Dec}_{\mathsf{sk}_i}(i, c) \to b$. This is a deterministic algorithm that takes as input a level index $0 \le i \le d$, a secret key sk_i and a ciphertext $c \in \{0,1\}^\ell$. It outputs a bit $b \in \{0,1\}$. When the level i is clear and fixed from the context, we simply write $\mathsf{FHE.Dec}_{\mathsf{sk}}(c)$.

$\mathsf{FHE.GateEval}_{\mathsf{ek}_i}(f, i, c_1, c_2) \to c^*$. This is a deterministic algorithm that takes as input a level index $i < d$, an evaluation key ek_i, the truth table of a *two input bit* function $f\colon \{0,1\} \times \{0,1\} \to \{0,1\}$, and two ciphertexts $c_1, c_2 \in \{0,1\}^\ell$ of level i. It outputs a ciphertext $c^* \in \{0,1\}^\ell$ of level $i+1$. When the level i is clear and fixed from the context, we simply write $\mathsf{FHE.GateEval}_{\mathsf{ek}}(f, c_1, c_2)$.

$\mathsf{FHE.Eval}_{\mathsf{ek}}(f, c_1, \ldots, c_n) \to c^*$. This is a deterministic algorithm defined by $\mathsf{FHE.GateEval}$. The algorithm takes as input the evaluation key ek, a leveled circuit representing a function $f\colon \{0,1\}^n \to \{0,1\}^m$ of depth $d_f \le d$, along with n ciphertexts $c_1, \ldots, c_n \in \{0,1\}^\ell$ of level 0. It runs $\mathsf{FHE.GateEval}$ to evaluate the circuit gate-by-gate and outputs the ciphertexts $c^* \in \{0,1\}^{\ell m}$ of the level d_f output wires.

Definition 8.1 (Fully homomorphic encryption (FHE))**.** A leveled, gate-by-gate fully homomorphic encryption scheme

$$\mathsf{FHE} = \mathsf{FHE.}(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval}, \mathsf{GateEval})$$

is required to satisfy the following properties:

Efficiency. Each ciphertexts c and each level key $\mathsf{sk}_i, \mathsf{ek}_i$ has a fixed polynomial size $\ell(\lambda)$, and the depth of the decryption circuit $\mathsf{FHE.Dec}_{\mathsf{sk}_i}$ for all $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{FHE.Gen}(1^\lambda, 1^d)$ is bounded by a fixed polynomial $p(\lambda, \log d)$. The gate evaluation circuit $\mathsf{FHE.GateEval}_{\mathsf{ek}_i}$ has a size bounded by a fixed polynomial $s(\lambda, \log d)$.

Encryption Correctness. For any choice of $(\mathsf{sk}, \mathsf{ek})$ in the support of $\mathsf{FHE.Gen}(1^\lambda, 1^d)$, any $b \in \{0,1\}$ and any $c \leftarrow \mathsf{FHE.Enc}_{\mathsf{sk}}(b)$ we have $\mathsf{FHE.Dec}_{\mathsf{sk}}(c) = b$.

Honest Evaluation Correctness. For any $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{FHE.Gen}(1^\lambda, 1^d)$, any honestly generated ciphertexts $c_1, \ldots, c_n \in \{0,1\}^\ell$ where $c_i = \mathsf{FHE.Enc}_{\mathsf{sk}}(b_i)$, and any *layered* circuit $f\colon \{0,1\}^n \to \{0,1\}^m$ of depth $d_f \le d$, if we set $c = \mathsf{FHE.Eval}_{\mathsf{ek}}(f, c_1, \ldots, c_n)$ then $\mathsf{FHE.Dec}_{\mathsf{sk}}(d_f, c) = f(b_1, \ldots, b_n)$.

Malicious Gate Correctness. For any choice of $(\mathsf{sk}, \mathsf{ek})$ in the support of $\mathsf{FHE.Gen}(1^\lambda, 1^d)$, any index $0 \le i \le d-1$, any $f : \{0,1\}^2 \to \{0,1\}$, and any ciphertexts $c_1, c_2 \in \{0,1\}^\ell$, if $\mathsf{Dec}_{\mathsf{sk}_i}(i, c_1) = b_1$ and $\mathsf{Dec}_{\mathsf{sk}}(i, c_2) = b_2$, then $\mathsf{FHE.Dec}_{\mathsf{sk}}(i+1, \mathsf{FHE.GateEval}_{\mathsf{ek}_i}(f, i, c_1, c_2)) = f(b_1, b_2)$.

Malicious Evaluation Correctness. For any $(\mathsf{sk}, \mathsf{ek})$ in the support of $\mathsf{FHE.Gen}(1^\lambda, 1^d)$ any ciphertexts $c_1, \ldots, c_n \in \{0,1\}^\ell$ such that $c_i = \mathsf{FHE.Enc}_{\mathsf{sk}_0}(0, b_i)$, and any *layered* circuit $f\colon \{0,1\}^n \to \{0,1\}^m$ of depth $d_f \le d$, if we set $c = \mathsf{FHE.Eval}_{\mathsf{ek}}(f, c_1, \ldots, c_n)$ then $\mathsf{FHE.Dec}_{\mathsf{sk}}(d_f, c) = f(b_1, \ldots, b_n)$.

Semantic Security. The encryption scheme is semantically secure, i.e., for all polynomial sized adversaries $\mathcal{A}$, all polynomial input lengths $k \le |\mathcal{A}|$, and all polynomial depth parameter d, we have that for any $m_0, m_1 \in \{0,1\}^k$,

$$(\mathsf{ek}, \mathsf{FHE.Enc}_{\mathsf{sk}}(m_0)) \approx_c (\mathsf{ek}, \mathsf{FHE.Enc}_{\mathsf{sk}}(m_1)),$$

where $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{FHE.Gen}(1^\lambda, 1^d)$.

8.2 Formalization of FHE in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$

Formalization of functions. As FHE.Dec, FHE.GateEval, and FHE.Eval are deterministic polynomial-time computable functions, these functions are formalized as corresponding PV functions. The probabilistic polynomial-time algorithm FHE.Gen and FHE.Enc are formalized as PV functions that additionally takes a random seed, i.e., $\mathsf{FHE.Gen}(\lambda, 1^d, \mathsf{sd})$ and $\mathsf{FHE.Enc}(\mathsf{sk}, b, \mathsf{sd})$. For simplicity, we assume that the seed lengths of both FHE.Gen and FHE.Enc are $r(\lambda, d) = \mathsf{poly}(\lambda, d)$.

Formalization of encryption correctness. The formalization of correctness is straightforward. In this work, we focus on schemes with *perfect correctness*, and thus the correctness can be formalized by the PV_1 sentences $\mathsf{FHE}_{\mathsf{Cor}}$.

$$\forall \lambda, d \in \mathsf{Log}, \forall \mathsf{sd}_1, \mathsf{sd}_2 \in \{0,1\}^{r(\lambda,d)}, \forall b \in \{0,1\},$$
$$(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{FHE.Gen}(\lambda, 1^d, \mathsf{sd}_1), \mathsf{FHE.Dec}_{\mathsf{sk}}(0, \mathsf{FHE.Enc}_{\mathsf{sk}}(b, \mathsf{sd}_2)) = b.$$

Formalization of evaluation correctness. Similar to the formalization of encryption correctness, the honest evaluation correctness can be formalized by the following sentences:

- **Honest Evaluation Correctness:** The sentence $\mathsf{FHE}_{\mathsf{HonCor}}$ is defined as follows: For any $\lambda, s, d \in \mathsf{Log}$, any layered circuit $f : \{0,1\}^n \to \{0,1\}^m$ of depth $d_f \le d$ and size $s_f \le s$, any random seeds $\mathsf{sd}_0, \mathsf{sd}_1, \mathsf{sd}_2, \ldots, \mathsf{sd}_n \in \{0,1\}^{r(\lambda,d)}$, and any $b_1, \ldots, b_n \in \{0,1\}$

$$(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{FHE.Gen}(\lambda, 1^d; \mathsf{sd}_0), c_i \leftarrow \mathsf{FHE.Enc}_{\mathsf{sk}}(b_i; \mathsf{sd}_i) \text{ for } 1 \le i \le n,$$
$$\mathsf{FHE.Dec}_{\mathsf{sk}}(d_f, \mathsf{FHE.Eval}_{\mathsf{ek}}(f, c_1, \ldots, c_n)) = f(b_1, \ldots, b_n).$$

- **Malicious Gate Correctness:** The sentence $\mathsf{FHE}_{\mathsf{MalGCor}}$ is defined as follows: For any $\lambda, d \in \mathsf{Log}$, any index $0 \le i \le d-1$, any $f : \{0,1\}^2 \to \{0,1\}$, any random seeds $\mathsf{sd} \in \{0,1\}^{r(\lambda,d)}$, and any ciphertexts $c_1, c_2 \in \{0,1\}^\ell$,

$$(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{FHE.Gen}(\lambda, 1^d; \mathsf{sd}), b_i = \mathsf{FHE.Dec}_{\mathsf{sk}}(i, c_i) \text{ for } i = 1, 2,$$
$$\mathsf{FHE.Dec}_{\mathsf{sk}}(i+1, \mathsf{FHE.GateEval}_{\mathsf{ek}_i}(f, i, c_1, c_2)) = f(b_1, b_2).$$

- **Malicious Evaluation Correctness:** The sentence $\mathsf{FHE}_{\mathsf{MalCor}}$ is defined as follows:For any $\lambda, s, d \in \mathsf{Log}$, any layered circuit $f : \{0,1\}^n \to \{0,1\}^m$ of depth $d_f \le d$ and size $s_f \le s$, any random seeds $\mathsf{sd} \in \{0,1\}^{r(\lambda,d)}$, and any $b_1, \ldots, b_n \in \{0,1\}$

$$(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{FHE.Gen}(\lambda, 1^d, \mathsf{sd}), b_i = \mathsf{FHE.Dec}_{\mathsf{sk}}(c_i) \text{ for } 1 \le i \le n,$$
$$\mathsf{FHE.Dec}_{\mathsf{sk}}(d_f, \mathsf{FHE.Eval}_{\mathsf{ek}}(f, c_1, \ldots, c_n)) = f(b_1, \ldots, b_n).$$

Both honest and malicious evaluation correctness is implied by the malicious gate correctness plus the encryption correctness. The proof can be done in PV_1 by induction over gates in the topological ordering, and is left as an easy exercise.

Claim 8.2. *In PV_1, the malicious gate correctness plus the encryption correctness implies the honest evaluation correctness and the malicious evaluation correctness.*

Formalization of security. It remains to formalize the security of secret-key FHE, which states that

$$(\mathsf{ek}, \mathsf{FHE.Enc}_{\mathsf{sk}}(m_0)) \quad \text{and} \quad (\mathsf{ek}, \mathsf{FHE.Enc}_{\mathsf{sk}}(m_1))$$

are indistinguishable to any polynomial sized adversary for any distinct messages $m_0, m_1 \in \{0,1\}^k$ of polynomial length, where $(\mathsf{sk}, \mathsf{ek})$ is sampled by $\mathsf{FHE.Gen}(1^\lambda, 1^d)$.

We first formalize the parameterized version of FHE security. Let $d, t, k, \lambda \in \mathbb{N}$ and $r = r(\lambda, d)$, and $m_0, m_1 \in \{0,1\}^k$ be distinct strings. Let $\ell = \ell(\lambda, \log d) = \mathsf{poly}(\lambda, \log d)$ be the length of the key and ciphertext for one-bit encryption. For $m \in \{0,1\}^k$, we define $\mathsf{Gen}_m : \{0,1\}^{2r} \to \{0,1\}^{d\ell+\ell k}$ to be the following circuits:

- $\mathsf{Gen}_m(\mathsf{sd}_1, \mathsf{sd}_2)$ runs $\mathsf{FHE.Gen}(1^\lambda, 1^d)$ with the seed sd_1 to generate $(\mathsf{sk}, \mathsf{ek})$, then runs $\mathsf{FHE.Enc}_{\mathsf{sk}}(m)$ with the seed sd_2 to obtain a ciphertext $c \in \{0,1\}^{\ell k}$, and then output $(\mathsf{ek}, c) \in \{0,1\}^{d\ell+\ell k}$.

Definition 8.3 (Parametrized FHE Security, in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$)**.** Fix PV functions FHE.Gen, FHE.Enc, and $d, t, k, \lambda \in \mathsf{Log}$, $\varepsilon > 0$. Let $r = r(\lambda, d)$ be the number of random bits required by FHE.Gen and FHE.Enc, and $\ell = \ell(\lambda, \log d)$ is the length of the key and ciphertext, where $\ell, r : \mathsf{Log} \to \mathsf{Log}$, and $\mathsf{Gen}_0, \mathsf{Gen}_1 : \{0,1\}^{2r} \to \{0,1\}^{d\ell+\ell k}$ be the circuits defined as above.

Let $\mathcal{D}_m$ be the distributions defined by Gen_m for $m \in \{0,1\}^k$. The formula $\mathsf{FHESecure}^{t,\varepsilon}_{d,k,\lambda}$ in the language of $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ is defined as

$$\forall m_0, m_1 \in \{0,1\}^k \quad \mathcal{D}_{m_0} \approx_{t,\varepsilon} \mathcal{D}_{m_1}.$$

Similar to the formalization of LWE (see Definition 4.20), the indistinguishability of two distributions are formalized by a set of sentences $\mathsf{FHESecure}_{n,d,t}$, where $n = n(\lambda)$, $d = d(\lambda)$, and $t = t(\lambda)$ are arbitrary polynomials that specify the input length (for circuits in FHE.Eval), the number of evaluation keys, and the running time of the adversary. Also, a function $\lambda_0[n, d, t]$ (not necessarily constructive) specifies the largest λ for which the security condition does not hold.

Definition 8.4 (FHE Security, in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$)**.** Fix PV functions FHE.Gen, FHE.Enc, FHE.Dec, FHE.GateEval, and FHE.Eval and polynomials $d, t, k : \mathsf{Log} \to \mathsf{Log}$. Let $\lambda_0[p_1, p_2, p_3]$ be a mapping from three PV functions to $\mathbb{N}$, which specifies the largest security parameter for which the FHE scheme is insecure, and is not necessarily a PV function.

We define $\mathsf{FHESecure}_{\lambda_0}$ as the following set of sentences: For all PV functions that are polynomials $d(\lambda), t(\lambda), k(\lambda)$, it includes the sentence

$$\forall \lambda > \lambda_0[d, t, k],\ \mathsf{FHESecure}^{t(\lambda),1/t(\lambda)}_{d(\lambda),k(\lambda),\lambda}.$$

8.3 GSW FHE Construction

We first start by giving the "base" GSW scheme $\mathsf{GSW}_{\mathsf{base}}$ in Algorithm 1, which is a secret-key variant of the somewhat homomorphic encryption construction of [GSW13] from LWE. Note that the base scheme does not yet satisfy the leveled FHE definition in Definition 8.1, as the key/ciphertext size grows polynomially with the circuit depth d, and it does not support malicious gate correctness. We will later use this base scheme to construct the leveled GSW scheme in Algorithm 2 that satisfies Definition 8.1.

8.4 Security Analysis for the Base Case

As a stepping stone to proving the security of leveled GSW scheme in Algorithm 2, we first prove the (parameterized) semantic security of the base GSW scheme $\mathsf{GSW}_{\mathsf{base}}$ in Algorithm 1. We formalize the standard proof via a hybrid argument. Let $D, t, k, \lambda \in \mathsf{Log}$ and $m_0, m_1 \in \{0,1\}^k$. Let $r \in \mathsf{Log}$ be the seed length of $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Gen}$ and $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Enc}$. We will define a sequence of distributions corresponding to the immediate hybrids in the security proof.

Hybrid $\mathcal{H}^{\mathsf{base}}_{b,1}$. For $b \in \{0,1\}$, we define $H_{b,1} : \{0,1\}^{(k+1)r} \to \mathbb{Z}_q^{(n+1)\times kw(n+1)}$ as the following circuit. It takes $(\mathsf{sd}_1, \mathsf{sd}_2) \in \{0,1\}^r \times \{0,1\}^{kr}$, generates the secret key $\mathsf{sk} := \mathbf{s}^\intercal \leftarrow \mathbb{Z}_q^n$ using sd_1, generates the encryption randomness $\mathbf{A}_i \leftarrow \mathbb{Z}_q^{n\times(n+1)w}$ and $\mathbf{e}_i \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{(n+1)w}$ for $i = 1, 2, \ldots, k$ using sd_2. Let $m_b = m_{b,1}m_{b,2}\ldots m_{b,k}$ and $\mathbf{A} := (\mathbf{A}_1 \mid \mathbf{A}_2 \mid \cdots \mid \mathbf{A}_k)$ and $\mathbf{e}^\intercal := (\mathbf{e}_1^\intercal \mid \mathbf{e}_2^\intercal \mid \cdots \mid \mathbf{e}_k^\intercal)$, we define

$$(\mathbf{C}_{b,1} \mid \mathbf{C}_{b,2} \mid \cdots \mid \mathbf{C}_{b,k}) := \begin{pmatrix} \mathbf{A} \\ \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal \end{pmatrix} + (m_{b,1}\mathbf{G} \mid m_{b,2}\mathbf{G} \mid \cdots \mid m_{b,k}\mathbf{G}) \tag{8.1}$$

It then outputs $(\mathbf{C}_{b,1} \mid \mathbf{C}_{b,2} \mid \cdots \mid \mathbf{C}_{b,k})$. Let $\mathcal{H}^{\mathsf{base}}_{b,1}$ be the distribution defined by the sampler $H_{b,1}$.

$\mathcal{H}^{\mathsf{base}}_{b,1}$ is essentially the distribution of encryptions of m_b under the base GSW scheme.

Hybrid $\mathcal{H}^{\mathsf{base}}_{b,2}$. We now define the circuit $H_{b,2} : \{0,1\}^{(k+1)r} \to \mathbb{Z}_q^{(n+1)\times kw(n+1)}$ as follows. It samples $\mathsf{sk} := \mathbf{s}^\intercal$ from sd_1, and samples $\mathbf{A}_1, \ldots, \mathbf{A}_k \in \mathbb{Z}_q^{n\times(n+1)w}$ and $\mathbf{u}_1^\intercal, \ldots, \mathbf{u}_k^\intercal \in \{0,1\}^{(n+1)w}$ uniformly at random using sd_2. Instead of Equation (8.1), the circuit outputs

$$(\mathbf{C}_{b,1} \mid \mathbf{C}_{b,2} \mid \cdots \mid \mathbf{C}_{b,k}) := \begin{pmatrix} \mathbf{A} \\ \mathbf{u}^\intercal \end{pmatrix} + (m_{b,1}\mathbf{G} \mid m_{b,2}\mathbf{G} \mid \cdots \mid m_{b,k}\mathbf{G}), \tag{8.2}$$

where $\mathbf{A} := (\mathbf{A}_1 \mid \mathbf{A}_2 \mid \cdots \mid \mathbf{A}_k)$, $\mathbf{u}^\intercal := (\mathbf{u}_1^\intercal \mid \mathbf{u}_2^\intercal \mid \cdots \mid \mathbf{u}_k^\intercal)$, and $\mathbf{e}^\intercal := (\mathbf{e}_1^\intercal \mid \mathbf{e}_2^\intercal \mid \cdots \mid \mathbf{e}_k^\intercal)$. Let $\mathcal{H}^{\mathsf{base}}_{b,2}$ be the distribution defined by the sampler $H_{b,2}$.

Lemma 8.5. *There is a constant $c \geq 1$ such that APC_1 + "$\mathsf{prBPP} = \mathsf{prP}$" proves the following sentence for $b \in \{0,1\}$. Let $n, q, \sigma, t, r, k, w \in \mathsf{Log}$, $\varepsilon > 0$, and $m_b \in \{0,1\}^k$. Assume that $\mathsf{LWE}^{ct^c,\varepsilon}_{n,kw(n+1),q,\sigma}$ holds, then $\mathcal{H}^{\mathsf{base}}_{b,1} \approx_{t,\varepsilon} \mathcal{H}^{\mathsf{base}}_{b,2}$.*

Proof. We argue in APC_1 + "$\mathsf{prBPP} = \mathsf{prP}$". Let $c > 0$ be a constant to be determined later. Fix $n, q, \sigma, t, r, k, w \in \mathsf{Log}$ and $\varepsilon > 0$.

Let $\mathsf{Gen}_1, \mathsf{Gen}_2 : \{0,1\}^r \to \mathbb{Z}_q^{(n+1)\times kw(n+1)}$ be the samplers of LWE and uniformly random distributions, i.e.,

$$\mathsf{Gen}_1(\mathcal{U}_{(k+1)r}) = \begin{pmatrix} \mathbf{A} \\ \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal \end{pmatrix}, \quad \mathsf{Gen}_2(\mathcal{U}_{(k+1)r}) = \begin{pmatrix} \mathbf{A} \\ \mathbf{u}^\intercal \end{pmatrix},$$

Algorithm 1 The base Gentry-Sahai-Waters [GSW13] encryption scheme for circuits of depth at most d. We will use this base scheme to construct a leveled FHE scheme in Algorithm 2.

Let $w := \lceil \log q \rceil$ and $\mathbf{g} := (1, 2, 4, \ldots, 2^{w-1})^\intercal$.
Let $\mathbf{G} := (\mathbf{I}_{n+1} \otimes \mathbf{g}) \in \mathbb{Z}_q^{(n+1)\times(n+1)w}$.
Let $\mathbf{G}^-$ denotes the bit-decomposition inverse mapping; i.e., $\mathbf{G}^-(\mathbf{A}) = \mathbf{B}$ if $\mathbf{w}$ has $\{0,1\}$ entries, and $\mathbf{GB} = \mathbf{A}$.

- **Key generation:** $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Gen}(1^\lambda, 1^D)$ takes as input a security parameter 1^λ and a circuit depth bound 1^D. It set up parameters as follows.
  - **Perfect correctness constraint.** Set $D = \max(2, D)$ to be at least 2 and $\lambda = \max(\lambda, 64 \log D)$ such that the condition $\lambda > 8 + 7 \log \lambda + 6 \log D$ holds.
  - **LWE parameters.** Set the LWE parameters $n := (\lambda D)^2$, $q := 2^{\lambda D}$, $\sigma := 3\lambda D$. Note that by the choice of λ and D, it holds that $q \geq 2^{2D}(n+1)^{2D} w^{2D} \sigma$.

  The algorithm samples secret vector $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ and outputs $\mathsf{sk} := \mathbf{s}$.
- **Encryption:** $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Enc}_{\mathsf{sk}}(m; \mathsf{sd})$ takes as input a one bit message $m \in \{0,1\}$ and sample randomness $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times(n+1)w}$ and $\mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{(n+1)w}$. Note that our sampler ensures that $\|\mathbf{e}\|_\infty \leq \sigma\sqrt{(n+1)w}$ (see Remark 4.19). It then outputs:

$$\mathbf{C} = \begin{pmatrix} \mathbf{A} \\ \mathbf{s}^\intercal \mathbf{A} + \mathbf{e}^\intercal \end{pmatrix} + m\mathbf{G}.$$

  For $m \in \{0,1\}^k$, we define $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Enc}_{\mathsf{sk}}(m; \mathsf{sd})$ to be the concatenation of encryptions of each bit in m.
- **Decryption:** On input $\mathbf{C} \in \mathbb{Z}_q^{(n+1)\times(n+1)w}$, $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Dec}_{\mathsf{s}}$ proceeds as follows:
  - Let $\mathbf{w} \leftarrow \mathbb{Z}_q^{n+1}$ such that $\mathbf{w} = (0, 0, \ldots, 0, \lfloor q/2 \rfloor)^\intercal$.
  - Compute $\mu = (-\mathbf{s}^\intercal | 1) \cdot \mathbf{C} \cdot \mathbf{G}^-(\mathbf{w})$.
  - If $\mu \in (-q/4, q/4)$, output 0. Else, output 1.

  Note that the decryption circuit has depth $\mathsf{poly}(\log n, \log\log q) = \mathsf{poly}(\log \lambda, \log D)$.
- **Homomorphic computation:**
  - $\mathsf{GSW}_{\mathsf{base}}.\mathsf{NANDEval}(\mathbf{C}_1, \mathbf{C}_2)$: On input two ciphertext $C_1, C_2 \in \mathbb{Z}_q^{(n+1)\times(n+1)w}$, the algorithm outputs $\mathbf{G} - \mathbf{C}_1 \cdot \mathbf{G}^-(\mathbf{C}_2)$.
  - $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Eval}(f, \mathbf{C}_1, \ldots, \mathbf{C}_n)$: For a NAND circuit f of depth at most d, first label each input of f with $\mathbf{C}_i$ appropriately. Then, implement $\mathsf{GSW}_{\mathsf{base}}.\mathsf{NANDEval}$ in topological ordering of f. Output the ciphertext associated the output wires of f.

Algorithm 2 Leveled GSW encryption scheme for circuits of depth d, satisfying malicious gate correctness. This scheme uses the base scheme in Algorithm 1 as a subroutine.

Let $D(\lambda)$ be a polynomial such that the base scheme $\mathsf{GSW}_{\mathsf{base}}$ on parameter $(\lambda, D = D(\lambda))$ has a decryption circuit of depth less than $D - D_0$, where D_0 is a fixed constant representing the maximal depth of NAND-circuit representation of 2-bit to 1-bit functions. Note that such $D(\lambda)$ exists since the decryption circuit of $\mathsf{GSW}_{\mathsf{base}}$ has depth $\mathsf{poly}(\log \lambda, \log D)$.

Key generation: $\mathsf{GSW.Gen}(1^\lambda, 1^d)$:

- Run $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Gen}(1^\lambda, 1^D)$ for $d+1$ times to obtain secret keys $\mathsf{sk}_0, \ldots, \mathsf{sk}_d$.
- Compute $\mathsf{ek}_i \leftarrow \mathsf{GSW}_{\mathsf{base}}.\mathsf{Enc}_{\mathsf{sk}_{i+1}}(\mathsf{sk}_i)$ for $i = 0, \ldots, d-1$.
- Output $(\mathsf{sk} = (\mathsf{sk}_0, \ldots, \mathsf{sk}_d), \mathsf{ek} = (\mathsf{ek}_0, \ldots, \mathsf{ek}_{d-1}))$.

Encryption: $\mathsf{GSW.Enc}_{\mathsf{sk}}(m) = \mathsf{GSW}_{\mathsf{base}}.\mathsf{Enc}_{\mathsf{sk}_0}(m)$ (i.e. encrypt under key sk_0 using the base scheme).
Decryption: $\mathsf{GSW.Dec}_{\mathsf{sk}}(i, \mathsf{ct}) = \mathsf{GSW}_{\mathsf{base}}.\mathsf{Dec}(\mathsf{sk}_i, \mathbf{C})$. (i.e. decrypt under the i-th level secret key sk_i using the base scheme).
Homomorphic Gate Evaluation: $\mathsf{GSW.GateEval}_{\mathsf{ek}}(f, i, \mathsf{ct}_1, \mathsf{ct}_2)$,

- Rewrite f as a NAND circuit (of depth at most D_0).
- Compute $f_{\mathbf{C}_0, \mathbf{C}_1}(\mathbf{s})$ be the NAND circuit that does the following:
  - Compute $m_b \leftarrow \mathsf{GSW}_{\mathsf{base}}.\mathsf{Dec}_{\mathbf{s}}(\mathbf{C}_b)$.
  - Output $f(m_0, m_1)$.
- Output $\mathbf{P} \leftarrow \mathsf{GSW}_{\mathsf{base}}.\mathsf{Eval}(f_{\mathbf{C}_0, \mathbf{C}_1}, \mathsf{ek}_i)$ (i.e. treat ek_i as a FHE ciphertext, and homomorphically evaluate the circuit $f_{\mathbf{C}_0, \mathbf{C}_1}$ on ek_i).

Homomorphic Circuit Evaluation: $\mathsf{GSW.Eval}(f, C_1, \ldots, C_n)$: Given a circuit f of depth at most d and ciphertexts $C_1, \ldots, C_n$ for the input bits, evaluate gate by gate using $\mathsf{GSW.GateEval}_{\mathsf{ek}}(\cdot, \cdot)$.

where $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times kw(n+1)}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{kw(n+1)}$, and $\mathbf{u} \leftarrow \mathbb{Z}_q^{kw(n+1)}$. Let $\mathcal{D}_1, \mathcal{D}_2$ be the distributions defined by $\mathsf{Gen}_1, \mathsf{Gen}_2$, respectively. By $\mathsf{LWE}_{n,kw(n+1),q,\sigma}^{ct^c,\varepsilon}$, recall that $\mathcal{D}_1 \approx_{ct^c,\varepsilon} \mathcal{D}_2$.

Let $\mathcal{A}$ be the following circuit:

$$\mathcal{A}\begin{pmatrix}\mathbf{A}\\ \mathbf{v}^\intercal\end{pmatrix} := \begin{pmatrix}\mathbf{A}\\ \mathbf{v}^\intercal\end{pmatrix} + (m_{b,1}\mathbf{G} \mid m_{b,2}\mathbf{G} \mid \cdots \mid m_{b,k}\mathbf{G}).$$

It can then be proved that $\mathcal{A} \circ \mathsf{Gen}_1, \mathcal{A} \circ \mathsf{Gen}_2$ are functionally equivalent to $H_{b,1}, H_{b,2}$, respectively. Choose a constant $c > 1$ such that the size of $\mathcal{A}$ is at most $(c-1)t^c$. Therefore, by the Reduction Lemma, we have that $\mathcal{H}_{b,1} \approx_{t,\varepsilon} \mathcal{H}_{b,2}$. □

Lemma 8.6. $\mathsf{PV}_1 \vdash \mathcal{H}_{0,2}^{\mathsf{base}} \cong \mathcal{H}_{1,2}^{\mathsf{base}}$.

Proof. We argue in the theory PV_1. Note that both $H_{0,2}$ and $H_{1,2}$ sample $\mathsf{sk} := \mathbf{s}^\intercal$ from sd_1, and sample $\mathbf{A}_1, \ldots, \mathbf{A}_k \in \mathbb{Z}_q^{n \times (n+1)w}$ and $\mathbf{u}_1^\intercal, \ldots, \mathbf{u}_k^\intercal \in \{0,1\}^{(n+1)w}$ from sd_2. In the rest of the proof, we will identify sd_1 and $\mathbf{s}^\intercal \in \{0,1\}^{(n+1)w}$, and identify $\mathbf{A}_1, \ldots, \mathbf{A}_k, \mathbf{u}_1^\intercal, \ldots, \mathbf{u}_k^\intercal$ and sd_2 for simplicity. It can be verify that this is without loss of generality.

Let $f, g : \{0,1\}^{(k+1)r} \to \{0,1\}^{(k+1)r}$ be the following circuits:

$$f(\mathbf{s}, \mathbf{A}, \mathbf{u}) := (\mathbf{s}, \hat{\mathbf{A}}, \hat{\mathbf{u}}); \quad g(\mathbf{s}, \hat{\mathbf{A}}, \hat{\mathbf{u}}) := (\mathbf{s}, \mathbf{A}, \mathbf{u}), \tag{8.3}$$

where

$$\begin{pmatrix}\hat{\mathbf{A}}\\ \hat{\mathbf{u}}^\intercal\end{pmatrix} := \begin{pmatrix}\mathbf{A}\\ \mathbf{u}^\intercal\end{pmatrix} + ((m_{1,1}-m_{0,1})\mathbf{G} \mid (m_{1,2}-m_{0,2})\mathbf{G} \mid \cdots \mid (m_{1,k}-m_{0,k})\mathbf{G}),$$
$$\begin{pmatrix}\mathbf{A}\\ \mathbf{u}^\intercal\end{pmatrix} := \begin{pmatrix}\hat{\mathbf{A}}\\ \hat{\mathbf{u}}^\intercal\end{pmatrix} - ((m_{1,1}-m_{0,1})\mathbf{G} \mid (m_{1,2}-m_{0,2})\mathbf{G} \mid \cdots \mid (m_{1,k}-m_{0,k})\mathbf{G}).$$

By the definition, it is easy to verify that $f \circ g = g \circ f = \mathsf{id}_{(k+1)r}$. Moreover, one can see that $H_{0,2} \circ f$ and $H_{1,2}$ are functionally equivalent and it is provable in PV_1. This concludes that $\mathcal{H}^{\mathsf{base}}_{0,2} \cong \mathcal{H}^{\mathsf{base}}_{1,2}$. □

Applying hybrid argument. We are now ready to prove the security of $\mathsf{GSW}_{\mathsf{base}}$. Let $D(\lambda), t(\lambda), k(\lambda) \in \mathsf{poly}(\lambda)$ be PV functions. Recall that we set $n(\lambda) = (\lambda D)^2$, $q(\lambda) = 2^{\lambda D}$, and $\sigma = 3\lambda D$. Let $\ell = \ell(\lambda) = (n+1)^2 \log q$ be the length of the secret key and ciphertext for one-bit encryption, and $r = r(\lambda, D)$ be the seed length of $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Gen}$ and $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Enc}$.

Let $\mathsf{Gen}_m : \{0,1\}^{(k+1)r} \to \{0,1\}^{\ell k}$ be the circuit that, given $(\mathsf{sd}_1, \mathsf{sd}_2)$, runs $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Gen}(1^\lambda, 1^d; \mathsf{sd}_1)$ to obtain sk, and outputs $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Enc}_{\mathsf{sk}}(m; \mathsf{sd}_2)$. Let $\mathcal{H}^{\mathsf{base}}_m$ be the distribution defined by Gen_m.

Theorem 8.7 (Security of GSW Base Encryption)**.** *For all* PV *functions $D(\lambda), t(\lambda), k(\lambda) : \mathsf{Log} \to \mathsf{Log}$ that are polynomials in λ, there exists a number $\lambda_0 \in \mathbb{N}$ such that the following sentence is provable in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$: For every $\lambda > \lambda_0$ and strings $m_0, m_1 \in \{0,1\}^{k(\lambda)}$, $\mathcal{H}^{\mathsf{base}}_{m_0} \approx_{t(\lambda),1/t(\lambda)} \mathcal{H}^{\mathsf{base}}_{m_1}$.*

Proof. We argue in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$. Let $D = D(\lambda), t = t(\lambda), k = k(\lambda) : \mathsf{Log} \to \mathsf{Log}$ be polynomials in λ and let $\lambda_0 \in \mathbb{N}$ to be determined later. Fix $\lambda > \lambda_0$ and $m_1, m_2 \in \{0,1\}^k$. We can observe that $\mathcal{H}^{\mathsf{base}}_{m_b}$ and $\mathcal{H}^{\mathsf{base}}_{b,1}$ are defined by exactly the same circuits, so it suffices to prove that $\mathcal{H}^{\mathsf{base}}_{0,1} \approx_{t,1/t} \mathcal{H}^{\mathsf{base}}_{1,1}$.

Let $c \in \mathbb{N}$ be the constant in Lemma 8.5. Recall that in the theory $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$, we have sentences in $\mathsf{LWE}_{\lambda'_0}$ as axioms for some function $\lambda'_0[p_1, p_2, p_3, p_4, p_5]$. Let $t' := c \cdot (5t)^c$. We set $\lambda_0 := \lambda'_0[n, kw(n+1), q, \sigma, t'] \in \mathbb{N}$. It follows that the sentence

$$\forall \lambda > \lambda'_0 \; \mathsf{LWE}^{t',1/t'}_{n,kw(n+1),q,\sigma}$$

is in $\mathsf{LWE}_{\lambda_0} \subseteq \mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$. Therefore, we have that $\mathsf{LWE}^{t',1/t'}_{n,kw(n+1),q,\sigma}$ holds and by Lemma 8.5, we can conclude that $\mathcal{H}^{\mathsf{base}}_{b,1} \approx_{5t,1/(5t)} \mathcal{H}^{\mathsf{base}}_{b,2}$ for $b \in \{0,1\}$.

Moreover, as $\mathcal{H}^{\mathsf{base}}_{0,2} \cong \mathcal{H}^{\mathsf{base}}_{1,2}$, we know by the Isomorphism Lemma that $\mathcal{H}^{\mathsf{base}}_{0,2} \approx_{5t,1/(5t)} \mathcal{H}^{\mathsf{base}}_{1,2}$. Then by Hybrid Argument, we have that $\mathcal{H}^{\mathsf{base}}_{0,1} \approx_{5t,4/(5t)} \mathcal{H}^{\mathsf{base}}_{1,1}$, which immediately implies that $\mathcal{H}^{\mathsf{base}}_{0,1} \approx_{t,1/t} \mathcal{H}^{\mathsf{base}}_{1,1}$. □

8.5 Security Analysis for Leveled GSW

We are now ready to prove the security of leveled GSW. Let $d, t, k, \lambda \in \mathsf{Log}$. Let $R = R(\lambda, d) \in \mathsf{Log}$ be the seed length of GSW.Gen and GSW.Enc, and $D(\lambda)$ be a polynomial such that the base scheme $\mathsf{GSW}_{\mathsf{base}}$ on parameter $(\lambda, D = D(\lambda))$ has a decryption circuit of depth less than $D - D_0$, where D_0 is a fixed constant representing the maximal depth of NAND-circuit representation of 2-bit to 1-bit functions. We use $r = r(\lambda, D) \in \mathsf{Log}$ to denote the seed length of $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Gen}$ and $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Enc}$. Recall that we choose $n = (\lambda D)^2$, $q = 2^{\lambda D}$, and $\sigma = 3\lambda D$. Our goal is to prove that there exists a function $\lambda_0[p_1, p_2, p_3]$ such that every sentence in $\mathsf{FHESecure}_{\lambda_0}$ is true, where FHE.$*$ is implemented by GSW.$*$ (see Algorithm 2).

We will first define a sequence of hybrids used in the security proof. Recall that d is the depth of the circuit to evaluate. For $i \in \{0, 1, \ldots, d\}$ and $m \in \{0,1\}^k$, we define a hybrid $\mathcal{H}_{m,i}$ as follows.

Hybrid $\mathcal{H}_{m,i}$. We define $H_{m,i} : \{0,1\}^{(k+1)r} \to (\mathbb{Z}_q^{(n+1)\times w(n+1)})^{dn+k}$ as the following circuit. The input is parsed as $(\mathsf{sd}_1, \mathsf{sd}_2) \in \{0,1\}^r \times \{0,1\}^{kr}$, where sd_1 is the seed for $\mathsf{GSW.Gen}$ and sd_2 is the seed for $\mathsf{GSW.Enc}$ (to encrypt a k-bit message). Recall that the secret key of $\mathsf{GSW}_{\mathsf{base}}$ is of form $\mathbb{Z}_q^n$. The circuit $H_{m,i}$ generates:

- Secret keys $\mathsf{sk}_0, \mathsf{sk}_1, \ldots, \mathsf{sk}_d \in \mathbb{Z}_q^n$ for $\mathsf{GSW}_{\mathsf{base}}$ independently from $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Gen}(1^\lambda, 1^D)$.
- Real evaluation keys: $\mathsf{ek}_0, \ldots, \mathsf{ek}_{i-1}$, where $\mathsf{ek}_j := \mathsf{GSW}_{\mathsf{base}}.\mathsf{Enc}_{\mathsf{sk}_{j+1}}(\mathsf{sk}_j)$ for $0 \le j < i$.
- Fake evaluation keys: $\mathsf{ek}_i, \ldots, \mathsf{ek}_{d-1}$, where $\mathsf{ek}_j := \mathsf{GSW}_{\mathsf{base}}.\mathsf{Enc}_{\mathsf{sk}_{j+1}}(0^n)$ for $i \le j < d$.

It then runs $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Enc}_{\mathsf{sk}_0}(m)$ to obtain the ciphertext ct_m, and outputs $(\mathsf{ek} = (\mathsf{ek}_0, \mathsf{ek}_1, \ldots, \mathsf{ek}_{d-1}), \mathsf{ct}_m)$. Let $\mathcal{H}_{m,i}$ be the distribution defined by the circuit $H_{m,i}$.

Lemma 8.8. *For every* PV *functions $d(\lambda), t(\lambda), k(\lambda) : \mathsf{Log} \to \mathsf{Log}$ that are polynomials in λ, there is a constant $\lambda_0 \in \mathbb{N}$ such that $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ proves the following sentence. For every $\lambda > \lambda_0$ and $i \in [d(\lambda)]$ and message $m \in \{0,1\}^{k(\lambda)}$, $\mathcal{H}_{m,i-1} \approx_{t(\lambda),1/t(\lambda)} \mathcal{H}_{m,i}$.*

Proof. Fix $d = d(\lambda)$, $t = t(\lambda)$, and $k = k(\lambda)$. Let $\lambda_0 \in \mathbb{N}$ be a constant to be determined later. We argue in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$. Fix $\lambda > \lambda_0$, $i \in [d(\lambda)]$, and message $m \in \{0,1\}^k$.

Let $\alpha \subseteq [(k+1)r]$ be all but the parts of the seed used to sample sk_i and the encryption randomness of ek_{i-1}. By the Averaging Argument for Indistinguishable Distribution, it suffices to prove that for every assignment ρ to bits in α, $\mathcal{H}_{m,i-1}|_\rho \approx_{t,1/t} \mathcal{H}_{m,i}|_\rho$. Notice that both $\mathcal{H}_{m,i-1}|_\rho$ and $\mathcal{H}_{m,i}|_\rho$ can be viewed as the direct product of two distributions: ek_{i-1} and $(\mathsf{ek}_0, \ldots, \mathsf{ek}_{i-2}, \mathsf{ek}_i, \ldots, \mathsf{ek}_{d-1}, \mathsf{ct})$. The second part takes no seed, and $\mathcal{H}_{m,i-1}$ and $\mathcal{H}_{m,i}$ are identical on the second part. By the Independent Side-Information Corollary, it suffices to prove that the following two distributions are $(t, 1/t)$-indistinguishable:

- $\mathcal{H}^*_{m,i-1}$ is the distribution defined by the circuit $H^*_{m,i-1} : \{0,1\}^{(n+1)r} \to (\mathbb{Z}_q^{(n+1)\times w(n+1)})^n$. Given $n+1$ independent seeds of length r, it runs $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Gen}(1^\lambda, 1^D)$ to generate sk_i, and then outputs $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Enc}_{\mathsf{sk}_i}(0^n)$ using n independent seeds (as $|\mathsf{sk}_{i-1}| = n$).
- $\mathcal{H}^*_{m,i}$ is the distribution defined by the circuit $H^*_{m,i} : \{0,1\}^{(n+1)r} \to (\mathbb{Z}_q^{(n+1)\times w(n+1)})^n$. Given $n+1$ independent seeds of length r, it runs $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Gen}(1^\lambda, 1^D)$ to generate sk_i, and then outputs $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Enc}_{\mathsf{sk}_i}(\mathsf{sk}_{i-1})$ using n independent seeds. Note that sk_{i-1} has been fixed given ρ.

Notice that $\mathcal{H}^*_{m,i-1}, \mathcal{H}^*_{m,i}$ are exactly the distributions $\mathcal{H}^{\mathsf{base}}_{0^n}, \mathcal{H}^{\mathsf{base}}_{\mathsf{sk}_{i-1}}$ in the Security of GSW Base Encryption. Therefore, there exists a constant λ_0 such that when $\lambda > \lambda_0$,

$$\mathcal{H}^*_{m,i-1}|_\rho \approx_{t,1/t} \mathcal{H}^*_{m,i}|_\rho.$$

This completes the proof. □

Lemma 8.9. *For every* PV *functions $d(\lambda), t(\lambda), k(\lambda) : \mathsf{Log} \to \mathsf{Log}$ that are polynomials in λ, there is a constant $\lambda_0 \in \mathbb{N}$ such that $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ proves the following sentence. For every $m_0, m_1 \in \{0,1\}^{k(\lambda)}$, $\mathcal{H}_{m_0,0} \approx_{t,1/t} \mathcal{H}_{m_1,0}$.*

Proof Sketch. The proof is similar to that of Lemma 8.8, so we will only sketch the proof. We argue in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$. Fix $d = d(\lambda), t = t(\lambda), k = k(\lambda)$ and $\lambda > \lambda_0$, where λ_0 is to be determined later. Let $m_0, m_1 \in \{0,1\}^{k(\lambda)}$. As all evaluation keys are fake in $\mathcal{H}_{m_0,0}$ and $\mathcal{H}_{m_1,0}$, both distributions can be viewed as the direct product of $(\mathsf{ek}_0, \ldots, \mathsf{ek}_{d-1})$ and ct. By the Product Hybrid Lemma, it suffices to prove that these two distributions are indistinguishable on the second part.

Notice that the second parts are exactly the encryptions of m_0 and m_1. We can then conclude by the Security of GSW Base Encryption that for sufficiently large $\lambda > \lambda_0$, the second parts of the two distributions are $(t, 1/t)$-indistinguishable. □

Wrapping up with hybrid argument. We can then conclude the security of the FHE scheme $\mathsf{GSW}.*$ by hybrid argument. Recall that $\mathsf{FHESecure}_{\lambda_0}$ states that for every PV functions $d, t, k : \mathsf{Log} \to \mathsf{Log}$ that are polynomials and any $\lambda > \lambda_0[d, t, k]$, any $m_0, m_1 \in \{0,1\}^{k(\lambda)}$, the encryption distributions of m_0 and m_1 (see Definition 8.3) are $(t, 1/t)$-indistinguishable.

Theorem 8.10 (Leveled GSW Security Theorem)**.** *There exists a function $\lambda[p_1, p_2, p_3]$ from* PV *functions $p_1, p_2, p_3 : \mathsf{Log} \to \mathsf{Log}$ such that $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ proves every sentence in $\mathsf{FHESecure}_{\lambda_0}$, when $\mathsf{FHE}.*$ are implemented by $\mathsf{GSW}.*$.*

Proof. Fix any PV functions $d = d(\lambda), t = t(\lambda), k = k(\lambda)$ that are polynomials and let $\lambda_0 = \lambda_0[d, t, k] \in \mathbb{N}$ be a constant to be determined later. Fix any $m_0, m_1 \in \{0,1\}^k$, and let $t' = 2dt \in \mathsf{poly}(\lambda)$.

Note that the encryption distributions of m_0 and m_1 are exactly the hybrids $\mathcal{H}_{m_0,d}$ and $\mathcal{H}_{m_1,d}$ defined above. Notice that by Lemma 8.8 and Lemma 8.9, there exists a constant λ_0 such that when $\lambda > \lambda_0$,

$$\mathcal{H}_{m_0,d} \approx_{t',1/t'} \mathcal{H}_{m_0,d-1} \approx_{t',1/t'} \cdots \approx_{t',1/t'} \mathcal{H}_{m_0,0} \approx_{t',1/t'} \mathcal{H}_{m_1,0} \approx_{t',1/t'} \cdots \approx_{t',1/t'} \mathcal{H}_{m_1,d}.$$

By Hybrid Argument, we have that $\mathcal{H}_{m_0,d} \approx_{t',2d/t'} \mathcal{H}_{m_1,d}$. This immediately implies that $\mathcal{H}_{m_0,d} \approx_{t,1/t} \mathcal{H}_{m_1,d}$ as $2d/t' = 1/t$ and $t' \geq t$. □

8.6 Efficiency and Correctness

Finally, we briefly discuss why the efficiency and correctness of the leveled GSW scheme (Algorithm 2) can be proved. The efficiency directly follows from the fact that in the leveled GSW scheme, the key for each level, the ciphertext, and the GateEval operations all corresponds to elements in $\mathsf{GSW}_{\mathsf{base}}$ on a fixed parameter $D(\lambda)$. Similarly, the encryption correctness of the leveled GSW scheme corresponds to the correctness of $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Enc}$ and $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Dec}$, while the malicious gate correctness of the leveled GSW scheme corresponds to the correctness of $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Eval}$ and $\mathsf{GSW}_{\mathsf{base}}.\mathsf{Dec}$. All these correctness properties of $\mathsf{GSW}_{\mathsf{base}}$ was shown to be provable in PV in [JKLM25].

9 Somewhere Extractable Hashing

Syntax. A somewhere statistically binding (SEH) hash scheme is a type of algorithms $(\mathsf{SEH.Gen}, \mathsf{SEH.TGen}, \mathsf{SEH.Hash}, \mathsf{SEH.Open}, \mathsf{SEH.Ver}, \mathsf{SEH.Ext})$ with the following syntax:

$\mathsf{Gen}(1^\lambda, 1^N) \to \mathsf{hk}$. On input a security parameter 1^λ, message length 1^N, a locality parameter 1^{loc}, and outputs a hash key hk.

$\mathsf{TGen}(1^\lambda, 1^N, i \in [N]) \to (\mathsf{hk}^*, \mathsf{td})$. On input a security parameter 1^λ, message length 1^N, a locality parameter 1^{loc} and in index $i \in [N]$, outputs a hash key hk^* along with a trapdoor td.

$\mathsf{Hash}(\mathsf{hk}, \mathbf{x} \in \{0,1\}^N) \to \tau$. On input a hash key hk and a string $\mathbf{x}$, output a hash value τ.

$\mathsf{Open}(\mathsf{hk}, \mathbf{x}, i) \to \rho$. On input a hash key hk, a string $\mathbf{x} \in \{0,1\}^N$ and an index $i \in [N]$, output a "local opening" ρ.

$\mathsf{Ver}(\mathsf{hk}, \tau, i, y, \rho) \to 0/1$. On input a hash key hk, hash value τ, index $i \in [N]$, symbol $y \in \{0,1\}$ and opening ρ, the verification algorithm decides to accept or reject the local opening.

$\mathsf{Ext}(\mathsf{hk}^*, \mathsf{td}, \tau) \to \mathbf{y}$. On input the hash key hk^*, trapdoor td and hash value τ, the extraction algorithm outputs the bit $\mathbf{x}_i$ (the index which was chosen during the TGen algorithm).

Definition 9.1. A somewhere statistically binding (SEH) family ($\mathsf{SEH.Gen}$, $\mathsf{SEH.TGen}$, $\mathsf{SEH.Hash}$, $\mathsf{SEH.Open}$, $\mathsf{SEH.Ver}$, $\mathsf{SEH.Ext}$) is required to satisfy the following properties:

Succinct Key. The size of the key is bounded by $\mathsf{poly}(\lambda, |S|, \log N)$.

Succinct Hash. The size of the hash value c is bounded by $\mathsf{poly}(\lambda, |S|, \log N)$.

Succinct Local Opening. The size of the local opening $\pi_i \leftarrow \mathsf{Open}(K, m, i, r)$ is at most $\mathsf{poly}(\lambda, |S|, \log N)$.

Succinct Verification. The running time of the verification algorithm is bounded by $\mathsf{poly}(\lambda, |S|, \log N)$.

Key Indistinguishability. For any non-uniform PPT adversary $\mathcal{A}$ and any polynomial $N = N(\lambda)$, there exists a negligible function $\nu(\lambda)$ such that for all $i^* \in [N]$

$$\Big| \Pr\left[S \leftarrow \mathcal{A}(1^\lambda, 1^N), \mathsf{hk} \leftarrow \mathsf{Gen}(1^\lambda, 1^N) : \mathcal{A}(\mathsf{hk}) = 1\right] - \Pr\left[S \leftarrow \mathcal{A}(1^\lambda, 1^N), (\mathsf{hk}^*, \mathsf{td}) \leftarrow \mathsf{TGen}(1^\lambda, 1^N, i^*) : \mathcal{A}(\mathsf{hk}^*) = 1\right] \Big| \leq \nu(\lambda).$$

Opening Completeness. For any hash key hk, any message $\mathbf{x} = (x_1, \ldots, x_N) \in \{0,1\}^N$, any randomness r, and any index $i \in [N]$, we have

$$\Pr\left[\tau \leftarrow \mathsf{Hash}(\mathsf{hk}, \mathbf{x}), \pi_i \leftarrow \mathsf{Open}(\mathsf{hk}, \mathbf{x}, i) : \mathsf{Ver}(\mathsf{hk}, \tau, x_i, i, \rho_i) = 1\right] = 1.$$

Extraction Correctness. For any subset $i^* \in [N]$, any trapdoor key $(\mathsf{hk}^*, \mathsf{td}) \leftarrow \mathsf{TGen}(1^\lambda, 1^N, S)$, any hash τ, any bit x_{i^*}, and any proof π_{i^*}, we have

$$\Pr\left[\mathsf{Ver}(\mathsf{hk}, \tau, x_{i^*}, i^*, \rho_{i^*}) = 1 \Rightarrow \mathsf{Ext}(\tau, \mathsf{td})|_{i^*} = x_{i^*}\right] = 1.$$

Since the extracted value $\mathsf{Ext}(\tau, \mathsf{td})|_{i^*}$ is unique, the extraction correctness implies statistical binding property.

9.1 Formalization of SEH in APC_1 + "prBPP = prP"

Formalization of functions. We formalize the deterministic functions $\mathsf{SEH.Hash}$, $\mathsf{SEH.Open}$, $\mathsf{SEH.Ver}$, $\mathsf{SEH.Ext}$ as PV functions. We formalize the probabilistic functions $\mathsf{SEH.Gen}$ and $\mathsf{SEH.TGen}$ as PV functions that additionally take a random seed sd. For simplicity, suppose that the seed length for both of these algorithms are $r(\lambda, N)$.

Formalization of completeness and correctness. In this work, we will focus on schemes which have perfect *opening completeness and extraction correctness*. Essentially, we will require that these equations can be formalized and proven in PV_1.

Formalization of security. It remains to formalize the security of SEH. Informally, it states that for all $i \in [N]$, an honestly generated hash key $\mathsf{hk} \leftarrow \mathsf{Gen}(1^\lambda)$ and a trapdoor mode hash key hk^* generated via $(\mathsf{hk}^*, \mathsf{td}) \leftarrow \mathsf{TGen}(1^\lambda, i)$ are indistinguishable.

We parameterize SEH security via λ. Let $\ell = \ell(\lambda, N)$ be length of the hash key and $r = r(\lambda, N)$ be the number of random bits required by TGen, where $\ell, r \in \mathsf{Log}$. We define $\mathsf{Gen}_i : \{0,1\}^r \to \{0,1\}^\ell$ be the following circuit:

- For $i = 0$, $\mathsf{Gen}_0(\mathsf{sd})$ calls $\mathsf{SEH.Gen}(1^\lambda; \mathsf{sd})$ and outputs hk.
- For $i \in [N]$, $\mathsf{Gen}_i(\mathsf{sd})$ calls $\mathsf{SEH.TGen}(1^\lambda, i; \mathsf{sd})$ to generate $(\mathsf{hk}, \mathsf{td})$, and outputs hk.

The security of the scheme states that for all $i \in [N]$, the distributions of Gen_0 and Gen_i are indistinguishable.

Definition 9.2 (Parametrized SEH security, in $\mathsf{APC}_1 +$ "prBPP = prP".)**.** Fix PV functions SEHash.Gen and SEHash.TGen, and $\lambda \in \mathbb{N}$, $\varepsilon > 0$. Let $r = r(\lambda, N)$ be the number of random bits used by Gen and TGen, and let $\ell(\lambda, N)$ denote the key length, where $\ell, r \in \mathsf{Log}$. Let $\mathsf{Gen}_0, \mathsf{Gen}_i : \{0,1\}^r \to \{0,1\}^\ell$ for $i \in [N]$ be defined as above.

Let $\mathcal{D}_j$ for $j \in \{0\} \cup [N]$ be the distribution corresponding to Gen_j. The formula $\mathsf{SEHSecure}^{t,\varepsilon}_{\lambda,N}$ in the language of $\mathsf{APC}_1 +$ "prBPP = prP" is defined as

$$\forall i \in [N], \mathcal{D}_0 \approx_{t,\varepsilon} \mathcal{D}_i.$$

Definition 9.3 (SEH security, in $\mathsf{APC}_1 +$ "prBPP = prP")**.** Fix PV functions (SEH.Gen, SEH.TGen, SEH.Hash, SEH.Open, SEH.Ver, SEH.Ext) and polynomials $t, N : \mathsf{Log} \to \mathsf{Log}$. Let $\lambda_0[p_1, p_2]$ be a mapping from polynomials to $\mathbb{N}$ which specifies the largest security parameter for which the SEH scheme is not secure.

We define $\mathsf{SEHSecure}_{\lambda_0}$ as the following set of sentences: For all PV functions that are polynomials $N(\lambda)$, it includes the sentence

$$\forall \lambda > \lambda_0[t, N], \mathsf{SEHSecure}^{t(\lambda), 1/t(\lambda)}_{\lambda, N(\lambda)}.$$

9.2 HW SEH Construction

In this section, we recall the SEH construction of Hubáček and Wichs [HW15]. We slightly modified the construction so that we can build it from secret key FHE formulated in Definition 8.1.

Construction 9.4 (Somewhere extractable Hash, [HW15])**.** In the following, we interchangeably parse an integer $i \in [N]$ as a bit string $(i_1, i_2, \ldots, i_L)$ of length $L = \lceil \log N \rceil$. For a binary tree of depth L, we view the root node as level 0 and the leaf nodes as level L. For a node at level ℓ, we index the nodes from left to right by bit strings $\{0,1\}^\ell$, and the children of a node v are indexed by $v\|0$ and $v\|1$. The root node is indexed by the empty string ε.

Let $\mathsf{FHE} = (\mathsf{FHE.Gen}, \mathsf{FHE.Enc}, \mathsf{FHE.Eval}, \mathsf{FHE.Dec})$ be a (secret-key) fully homomorphic encryption scheme (Definition 8.1). For all $\lambda \in \mathbb{N}$, let d_λ be an efficiently computable bound such that the circuit depth of FHE.Dec on parameter (d_λ, λ) is smaller than d_λ. Let $s = s(\lambda)$ be the secret key length of the FHE scheme on parameter (λ, d_λ), and $r = r(\lambda)$ be the encryption randomness length of the same parameters.

We now construct a SEH scheme (SEH.Gen, SEH.TGen, SEH.Hash, SEH.Open, SEH.Ver, SEH.Ext) as follows:

- $\mathsf{Gen}(1^\lambda, 1^N)$: Set tree depth parameter $L = \lceil \log N \rceil$. For each $\ell = 0, \ldots, L$, generate FHE key pairs $(\mathsf{sk}_\ell, \mathsf{ek}_\ell) \leftarrow \mathsf{FHE.Gen}(1^\lambda, 1^{d_\lambda})$ for $\ell = 0, \ldots, L$. For all $\ell \in [L]$, compute ciphertexts $c_\ell = \mathsf{FHE.Enc}_{\mathsf{sk}_{\ell-1}}(0^{s+1})$. Finally, compute two ciphertexts $\hat{c}_0 = \mathsf{FHE.Enc}_{\mathsf{sk}_L}(0)$ and $\hat{c}_1 = \mathsf{FHE.Enc}_{\mathsf{sk}_L}(1)$. Output the hash key $\mathsf{hk} = (\mathsf{ek}_0, \ldots, \mathsf{ek}_L, c_1, \ldots, c_L, \hat{c}_0, \hat{c}_1)$.
- $\mathsf{TGen}(1^\lambda, 1^N, i \in [N])$: Set tree depth parameter $L = \lceil \log N \rceil$. Generate $L+1$ pairs of keys $(\mathsf{sk}_\ell, \mathsf{ek}_\ell) \leftarrow \mathsf{FHE.Gen}(1^\lambda)$ for $\ell = 0, \ldots, L$. Parse $i \in [N]$ as a bit string $(i_1, i_2, \ldots, i_L)$. For all $\ell \in [L]$, compute ciphertexts $c_\ell = \mathsf{FHE.Enc}_{\mathsf{sk}_{\ell-1}}(\mathsf{sk}_\ell \| i_\ell)$. Finally, compute two ciphertexts $\hat{c}_0 = \mathsf{FHE.Enc}_{\mathsf{sk}_L}(0)$ and $\hat{c}_1 = \mathsf{FHE.Enc}_{\mathsf{sk}_L}(1)$. Output the hash key $\mathsf{hk}^* = (\mathsf{ek}_0, \ldots, \mathsf{ek}_L, c_1, \ldots, c_L, \hat{c}_0, \hat{c}_1)$ and the trapdoor $\mathsf{td} = \mathsf{sk}_L$.
- $\mathsf{Hash}(\mathsf{hk}, \mathbf{x} \in \{0,1\}^N)$: Parse $\mathsf{hk} = (\mathsf{ek}_0, \ldots, \mathsf{ek}_L, c_1, \ldots, c_L, \hat{c}_0, \hat{c}_1)$. The algorithm computes hash values along a binary tree of depth L bottom-up through the following procedure.
  - For each leaf node $i \in \{0,1\}^L$, set $h_i = \hat{c}_{x_i}$ as the ciphertext for bit x_i precomputed in the hash key. We set $x_i = 0$ if $i > N$.
  - For the ℓ-th level of the tree, for each node $j \in \{0,1\}^\ell$, let $\mathsf{ct}_0 = h_{j\|0}$ and $\mathsf{ct}_1 = h_{j\|1}$ be the ciphertexts associated with the left and right children. Compute $h_j = \mathsf{FHE.Eval}_{\mathsf{ek}_{\ell-1}}(F_{\mathsf{ct}_0,\mathsf{ct}_1}, c_\ell)$, where the circuit F is defined as:

    $$F_{\mathsf{ct}_0,\mathsf{ct}_1}(\mathsf{sk}, b) = \mathsf{FHE.Dec}_{\mathsf{sk}}(\mathsf{ct}_b)$$

    Note that the FHE evaluation of $F_{\mathsf{ct}_0,\mathsf{ct}_1}$ is a deterministic operation.

  The algorithm outputs the root hash value $\tau = h_\varepsilon$.
- $\mathsf{Open}(\mathsf{hk}, \mathbf{x}, i)$: Parse $\mathsf{hk} = (\mathsf{ek}_0, \ldots, \mathsf{ek}_L, c_1, \ldots, c_L, \hat{c}_0, \hat{c}_1)$. Parse $i \in [N]$ as a bit string $(i_1, i_2, \ldots, i_L)$. The algorithm first computes the hash tree as in $\mathsf{Hash}(\mathsf{hk}, \mathbf{x})$ to obtain all the intermediate hash values h_v for each node $v \in \{0,1\}^{\le L}$. For $\ell \in [1, L]$, let $\bar{i}_\ell = (i_1, \ldots, 1 - i_\ell)$ be the nodes on the copath of the path from the root to the leaf i. Output the local opening $\rho = (h_{\bar{i}_1}, \ldots, h_{\bar{i}_L})$.
- $\mathsf{Ver}(\mathsf{hk}, \tau, i, y, \rho)$: Parse $\mathsf{hk} = (\mathsf{ek}_0, \ldots, \mathsf{ek}_L, c_1, \ldots, c_L, \hat{c}_0, \hat{c}_1)$, index $i \in [N]$ as a bit string $(i_1, i_2, \ldots, i_L)$, and local opening $\rho = (h_{\bar{i}_1}, \ldots, h_{\bar{i}_L})$. The algorithm first compute the hash value on index i by $h_i = \hat{c}_y$. Then, for $\ell = L, L-1, \ldots, 1$, it computes the hash value on the path from the leaf i to the root as follows:

  $$h_{i_1,\ldots,i_{\ell-1}} = \begin{cases} \mathsf{FHE.Eval}_{\mathsf{ek}_{\ell-1}}(F_{h_{i_1,\ldots,i_\ell}, h_{\bar{i}_\ell}}, c_\ell), \text{ if } i_\ell = 0 \\ \mathsf{FHE.Eval}_{\mathsf{ek}_{\ell-1}}(F_{h_{\bar{i}_\ell}, h_{i_1,\ldots,i_\ell}}, c_\ell), \text{ if } i_\ell = 1 \end{cases}$$

  Finally, the algorithm accepts if and only if $h_\varepsilon = \tau$.
- $\mathsf{Ext}(\mathsf{hk}^*, \mathsf{td}, \tau)$: Parse $\mathsf{hk}^* = (\mathsf{ek}_0, \ldots, \mathsf{ek}_L, c_1, \ldots, c_L, \hat{c}_0, \hat{c}_1)$, $\mathsf{td} = \mathsf{sk}_L$. The algorithm outputs $y = \mathsf{FHE.Dec}_{\mathsf{sk}_L}(\tau)$.

9.3 Security Analysis of SEH

In the following, let $\ell(\lambda, N)$ be the length of the hash key on parameter (λ, N), and let $r(\lambda, N)$ be the length of the randomness used by Gen and TGen.

Theorem 9.5 (SEH Opening Completeness Theorem). PV_1 *proves the following sentence:*

$$\forall \lambda, N \in \mathsf{Log}, \forall \mathsf{hk} \in \{0,1\}^{\ell(\lambda,N)}, \mathbf{x} \in \{0,1\}^N, i \in [N],\ \mathsf{Ver}(\mathsf{hk}, \mathsf{Hash}(\mathsf{hk}, \mathbf{x}), i, \mathbf{x}_i, \mathsf{Open}(\mathsf{hk}, \mathbf{x}, i)) = 1$$

The completeness proof follows by inductively expanding the deterministic algorithms Hash, Open and Ver. It is nearly identical to the proof in [JKLM25, Section9.3].

Theorem 9.6 (SEH Extraction Correctness Theorem). *Let* $\mathsf{FHE}_{\mathsf{Cor}}$ *be the correctness properties of FHE defined in Definition 8.1, including malicious gate correctness. Then,* $\mathsf{PV}_1 + \mathsf{FHE}_{\mathsf{Cor}}$ *proves the following sentence:*

$$\begin{aligned}&\forall \lambda, N \in \mathsf{Log}, \forall i^* \in [N], \mathsf{sd} \in \{0,1\}^{r_{\mathsf{Gen}}}, \forall \tau \in \{0,1\}^{\ell(\lambda,N)}, y \in \{0,1\}, \rho \in \{0,1\}^L,\\ &(\mathsf{hk}^*, \mathsf{td}) = \mathsf{TGen}(1^\lambda, 1^N, i^*; \mathsf{sd}) \wedge \mathsf{Ver}(\mathsf{hk}^*, \tau, i^*, y, \rho) = 1 \Rightarrow \mathsf{Ext}(\mathsf{hk}^*, \mathsf{td}, \tau) = y\end{aligned}$$

where r_{Gen} *is the randomness length of* TGen *on parameter* (λ, N).

Proof. We first prove that, in the execution of $\mathsf{Ver}(\mathsf{hk}^*, \tau, i^*, y, \rho)$, the intermediate hash values $h_{i_1,\ldots,i_\ell}$ for all $\ell = 0, \ldots, L$ satisfies $\mathsf{FHE.Dec}_{\mathsf{sk}_\ell}(h_{i_1,\ldots,i_\ell}) = y$.

For the base case $\ell = L$, $h_{i_1,\ldots,i_L} = \hat{c}_y$, and by the definition of $\hat{c}_y$ in TGen, $\hat{c}_y$ is an encryption of y under sk_L. Therefore by $\mathsf{FHE}_{\mathsf{Cor}}$ we have $\mathsf{FHE.Dec}_{\mathsf{sk}_L}(h_{i_1,\ldots,i_L}) = y$.

For the inductive case, suppose that for some $\ell \in [L]$, we have $\mathsf{FHE.Dec}_{\mathsf{sk}_\ell}(h_{i_1,\ldots,i_\ell}) = y$. By the definition of Ver, we have

$$h_{i_1,\ldots,i_{\ell-1}} = \begin{cases}\mathsf{FHE.Eval}_{\mathsf{ek}_{\ell-1}}(F_{h_{i_1,\ldots,i_\ell}, h_{\bar{i}_\ell}}, c_\ell), & \text{if } i_\ell = 0\\ \mathsf{FHE.Eval}_{\mathsf{ek}_{\ell-1}}(F_{h_{\bar{i}_\ell}, h_{i_1,\ldots,i_\ell}}, c_\ell), & \text{if } i_\ell = 1\end{cases}$$

where $c_\ell = \mathsf{FHE.Enc}_{\mathsf{sk}_{\ell-1}}(\mathsf{sk}_\ell \| i_\ell)$ as defined in TGen. By $\mathsf{FHE}_{\mathsf{Cor}}$, we know $\mathsf{FHE.Dec}_{\mathsf{sk}_{\ell-1}}(c_\ell) = \mathsf{sk}_\ell \| i_\ell$. Furthermore, the circuit $F_{h_{i_1,\ldots,i_\ell}, h_{\bar{i}_\ell}}$ has depth 1 plus the depth of FHE.Dec, and thus by definition is no more than d_λ. Therefore, by applying the malicious evaluation correctness in $\mathsf{FHE}_{\mathsf{Cor}}$, we have

$$\mathsf{FHE.Dec}_{\mathsf{sk}_{\ell-1}}(h_{i_1,\ldots,i_{\ell-1}}) = \begin{cases}F_{h_{i_1,\ldots,i_\ell}, h_{\bar{i}_\ell}}(\mathsf{sk}_\ell \| 0), & \text{if } i_\ell = 0\\ F_{h_{\bar{i}_\ell}, h_{i_1,\ldots,i_\ell}}(\mathsf{sk}_\ell \| 1), & \text{if } i_\ell = 1,\end{cases}$$

which, by the definition of F, can be rewritten as

$$\mathsf{FHE.Dec}_{\mathsf{sk}_{\ell-1}}(h_{i_1,\ldots,i_{\ell-1}}) = \mathsf{FHE.Dec}_{\mathsf{sk}_\ell}(h_{i_1,\ldots,i_\ell}) = y.$$

By induction, we have proved that for all $\ell = 0, \ldots, L$, $\mathsf{FHE.Dec}_{\mathsf{sk}_\ell}(h_{i_1,\ldots,i_\ell}) = y$. In particular, for the root node, we have $\mathsf{FHE.Dec}_{\mathsf{sk}_0}(h_\varepsilon) = y$. Since $\mathsf{Ver}(\mathsf{hk}^*, \tau, i^*, y, \rho) = 1$ implies $h_\varepsilon = \tau$, we have $\mathsf{FHE.Dec}_{\mathsf{sk}_0}(\tau) = y$, and thus by the definition of Ext, $\mathsf{Ext}(\mathsf{hk}^*, \mathsf{td}, \tau) = y$. □

Theorem 9.7 (SEH Security Theorem). *Let* FHESecure *be the semantic security property of* FHE *defined in Definition 8.4. Then,* $\mathsf{APC}_1 + \text{“prBPP} = \mathsf{prP}\text{”} + \mathsf{FHESecure}_{\lambda_0}$ *proves the* SEHash *security property* $\mathsf{SEHSecure}_{\lambda_0'}$ *(Definition 9.3) of Construction 9.4 for some suitable* λ_0'.

Proof. Fix any $\lambda, N \in \mathsf{Log}$ and any index $i^* \in [N]$, let $d = d_\lambda$ be an efficiently computable bound such that the circuit depth of FHE.Dec on parameter (d_λ, λ) is smaller than d_λ. Let $s = s(\lambda)$ be the secret key length of the FHE scheme on parameter (λ, d_λ), as defined in Construction 9.4. We define the hybrid circuits $\mathcal{H}_j$ for $j = 0, \ldots, L$, which operates as follows:

- Set tree depth parameter $L = \lceil \log N \rceil$. Generate $L+1$ pairs of keys $(\mathsf{sk}_\ell, \mathsf{ek}_\ell) \leftarrow \mathsf{FHE.Gen}(1^\lambda)$ for $\ell = 0, \ldots, L$. Parse $i \in [N]$ as a bit string $(i_1, i_2, \ldots, i_L)$.
- For all $\ell \in [j]$, compute ciphertexts $c_\ell = \mathsf{FHE.Enc}_{\mathsf{sk}_{\ell-1}}(0^{s+1})$.
- For all $\ell \in [j+1, L]$, compute ciphertexts $c_\ell = \mathsf{FHE.Enc}_{\mathsf{sk}_{\ell-1}}(\mathsf{sk}_\ell \| i_\ell)$.
- Finally, compute two ciphertexts $\hat{c}_0 = \mathsf{FHE.Enc}_{\mathsf{sk}_L}(0)$ and $\hat{c}_1 = \mathsf{FHE.Enc}_{\mathsf{sk}_L}(1)$.

It is easy to see that $\mathcal{H}_L$ is identically distributed to the output of $\mathsf{Gen}(1^\lambda, 1^N)$, and $\mathcal{H}_0$ is identically distributed to the crs outputted by $\mathsf{TGen}(1^\lambda, 1^N, i^*)$.

We now prove that each neighboring hybrids are indistinguishable assuming the semantic security of FHE (Definition 8.3).

Lemma 9.8. *For all $j = 0, \ldots, L-1$, APC_1 + "$\mathsf{prBPP} = \mathsf{prP}$"+ $\mathsf{FHESecure}^{t-t_{\mathsf{Gen}},\varepsilon}_{d,s+1,\lambda}$ proves the following sentence:*

$$\mathcal{H}_j \approx_{t,\varepsilon} \mathcal{H}_{j+1}.$$

where t_{Gen} is the upper bound of the circuit size of Gen *and* TGen *on parameter (λ, N).*

Proof. Observe that the only difference between $\mathcal{H}_j$ and $\mathcal{H}_{j+1}$ is in the computation of c_{j+1}. In $\mathcal{H}_j$, c_{j+1} is an encryption of $\mathsf{sk}_{j+1} \| i_{j+1}$ under sk_j, while in $\mathcal{H}_{j+1}$, c_{j+1} is an encryption of 0^{s+1} under sk_j. Also note that not that in both hybrids, c_j is an encryption of 0^{s+1}, independent of the j-th level secret key sk_j. Therefore, we can consider the two following circuits generating c_{j+1}:

- $G_0(\mathsf{sd} = (\mathsf{sd}_j, \mathsf{sd}_{j+1}, \mathsf{sd}_{\mathsf{Enc},j}))$: Sample (from seed sd')

$$(\mathsf{sk}_j, \mathsf{ek}_j) \leftarrow \mathsf{FHE.Gen}(1^\lambda; \mathsf{sd}_j),$$
$$(\mathsf{sk}_{j+1}, \mathsf{ek}_{j+1}) \leftarrow \mathsf{FHE.Gen}(1^\lambda; \mathsf{sd}_{j+1}),$$

  and compute $c_{j+1} = \mathsf{FHE.Enc}_{\mathsf{sk}_j}(\mathsf{sk}_{j+1} \| i_{j+1}; \mathsf{sd}_{\mathsf{Enc},j})$. Output $(\mathsf{ek}_j, c_{j+1}, \mathsf{ek}_{j+1}, \mathsf{sk}_{j+1})$.
- $G_1(\mathsf{sd}')$: Same as $G_0(\mathsf{sd}')$, but compute $c_{j+1} = \mathsf{FHE.Enc}_{\mathsf{sk}_j}(0^{s+1})$.

It is obvious that $H_j = C \circ G_0$ and $H_{j+1} = C \circ G_1$ for some appropriate circuit C of size at most t_{Gen}. Therefore by the Reduction Lemma, we have

$$G_0 \approx_{t-t_{\mathsf{Gen}},\varepsilon} G_1 \vdash \mathcal{H}_j \approx_{t,\varepsilon} \mathcal{H}_{j+1}.$$

Finally, for every partial assignment of sd_{j+1} in G_0 and G_1, the two circuits exactly corresponds to the distributions for the parameterized FHE security Definition 8.3 where $(m_0, m_1) = (\mathsf{sk}_{j+1} \| i_{j+1}, 0^{s+1})$. Therefore

$$\mathsf{FHESecure}^{t-t_{\mathsf{Gen}},\varepsilon}_{d,s+1,\lambda} \vdash G_0|_{\mathsf{sd}_{j+1}} \approx_{t-t_{\mathsf{Gen}},\varepsilon} G_1|_{\mathsf{sd}_{j+1}},$$

The lemma now follows by applying the Averaging Argument for Indistinguishable Distribution. □

By combining the above lemma for all $j = 0, \ldots, L-1$ with the Hybrid Argument, we have

$$\mathsf{FHESecure}^{t-t_{\mathsf{Gen}},L\varepsilon}_{d,s+1,\lambda} \vdash \mathcal{H}_0 \approx_{t-t_{\mathsf{Gen}},\varepsilon} \mathcal{H}_L.$$

Therefore, by setting $\lambda'_0[t, N] = \lambda_0[d, \max(t - t_{\mathsf{Gen}}, Lt), s+1]$, every sentence in $\mathsf{SEHSecure}_{\lambda'_0}$ can be proven in APC_1 + "$\mathsf{prBPP} = \mathsf{prP}$" + $\mathsf{FHESecure}_{\lambda_0}$. □

10 Batch Arguments for NP

Syntax. A Batch Argument scheme (BARG) for NP consists of algorithms $\mathsf{BARG}.(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver}, \mathsf{TGen}, \mathsf{Ext})$ with the following syntax:

- $\mathsf{Gen}(1^\lambda, 1^m, 1^s, 1^L) \to \mathsf{crs}$: On input the security parameter λ, batch size m, the circuit size upperbound s, and the extraction size upperbound L, outputs a common reference string crs.
- $\mathsf{TGen}(1^\lambda, 1^m, 1^s, 1^L, \mathcal{S} \subseteq [m]) \to (\mathsf{crs}^*, \mathsf{td})$: On input the security parameter λ, batch size m, the circuit size upperbound s, the extraction size upperbound L, and a set of indices $\mathcal{S} \subseteq [m]$ with size at most L, outputs a common reference string crs^* and a trapdoor td.
- $\mathsf{Prove}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), (\mathbf{y}_1, \ldots, \mathbf{y}_m)) \to \pi$: On input the common reference string crs, a relation circuit $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ of size at most s, m statements $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \{0,1\}^n$, and m witnesses $\mathbf{y}_1, \ldots, \mathbf{y}_m \in \{0,1\}^h$ such that $C(\mathbf{x}_j, \mathbf{y}_j) = 1$ for all $j \in [m]$, outputs a proof π.
- $\mathsf{Ver}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi) \to \{0,1\}$: On input the common reference string crs, a relation circuit $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ of size at most s, m statements $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \{0,1\}^n$, and a proof π, outputs a bit indicating whether to accept or reject the proof.
- $\mathsf{Extract}(\mathsf{crs}^*, \mathsf{td}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi) \to \{\mathbf{y}_i^*\}_{i \in \mathcal{S}}$: On input the common reference string crs^*, the trapdoor td, a relation circuit $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ of size at most s, m statements $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \{0,1\}^n$, and a proof π, outputs the extracted witness $\mathbf{y}_i^* = \mathbf{y}_i$ for each $i \in \mathcal{S}$ (the indices set which was chosen during the TGen algorithm).

We say that a BARG scheme supports *single point extraction* if the TGen and $\mathsf{Extract}$ algorithms only take a single index $i^* \in [m]$ as input/output instead of a set $\mathcal{S} \subseteq [m]$. In this case, we omit the 1^L input in Gen and TGen. In Section 10.5, we show that BARG with single point extraction implies general BARG via parallel repetition.

Definition 10.1 (Batch Argument)**.** A Batch Argument scheme $\mathsf{BARG}.(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver}, \mathsf{TGen}, \mathsf{Ext})$ is required to satisfy the following properties:

Succinct Proof. The size of the proof π is bounded by $\mathsf{poly}(\lambda, s, \log m, L)$.

Correctness. For all $\lambda, m, s, L \in \mathbb{N}$, all relation circuit $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ of size at most s, and any instances $(\mathbf{x}_1, \ldots, \mathbf{x}_m)$ and witnesses $(\mathbf{y}_1, \ldots, \mathbf{y}_m)$ such that $C(\mathbf{x}_i, \mathbf{y}_i) = 1$ for all $i \in [m]$, and all $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^m, 1^s, 1^L)$, it holds that

$$\mathsf{Ver}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \mathsf{Prove}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), (\mathbf{y}_1, \ldots, \mathbf{y}_m))) = 1.$$

CRS indistinguishability. For all polynomial sized adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\lambda)$ such that for all $\lambda \in \mathbb{N}$, $m, s, L \in \mathsf{poly}(\lambda)$, and all $\mathcal{S} \subseteq [m]$ where $|\mathcal{S}| \leq L(\lambda)$, it holds that

$$\left| \begin{array}{l} \Pr[\mathcal{A}(\mathsf{crs}) = 1 \mid \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^m, 1^s, 1^L)] \\ -\Pr[\mathcal{A}(\mathsf{crs}) = 1 \mid (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{TGen}(1^\lambda, 1^m, 1^s, 1^L, \mathcal{S})] \end{array} \right| \leq \mathsf{negl}(\lambda).$$

Somewhere Extractability. For all $\lambda, m, s, L \in \mathbb{N}$, all indices set $\mathcal{S} \subseteq [m]$, all relation circuit $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ of size at most s, all instances $(\mathbf{x}_1, \ldots, \mathbf{x}_m)$, all proof π, and all $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{TGen}(1^\lambda, 1^m, 1^s, 1^L, \mathcal{S})$, it holds that

$$\mathsf{Ver}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi) = 1 \implies \forall i^* \in \mathcal{S}, C(\mathbf{x}_{i^*}, \mathsf{Extract}_{i^*}(\mathsf{td}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi)) = 1,$$

where $\mathsf{Extract}_{i^*}(\cdot)$ denotes the extracted value on index i^*.

10.1 Formalization of BARG in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$

In this section, we formalize the syntax and security properties of BARG in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. We note that all the formalization extends to BARG with single point extraction by fixing $L = 1$.

Formalization of functions. Similar to Section 8.2, we formalize deterministic algorithms BARG.Prove, BARG.Ver, BARG.Extract as PV functions, and BARG.Gen, BARG.TGen as PV functions that additionally takes random seed sd of length $r(\lambda, m, s, L) = \mathsf{poly}(\lambda, m, s, L)$ as input.

Formalization of correctness. For simplicity, we focus on perfectly secure BARG schemes in this work. Correctness can thus be formalized as the following sentence in the language of PV_1, denoted by BARGcor: For all $\lambda, m, s \in \mathsf{Log}$, all relation circuit $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ of size at most s, all instances $(\mathbf{x}_1, \ldots, \mathbf{x}_m)$ and witnesses $(\mathbf{y}_1, \ldots, \mathbf{y}_m)$, all random seed $\mathsf{sd} \in \{0,1\}^{r(\lambda,m,s)}$, if $C(\mathbf{x}_i, \mathbf{y}_i) = 1$ for every $i \in [m]$, then

$$\mathsf{Ver}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \mathsf{Prove}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), (\mathbf{y}_1, \ldots, \mathbf{y}_m))) = 1$$

where $\mathsf{crs} := \mathsf{Gen}(1^\lambda, 1^m, 1^s, 1^L; \mathsf{sd})$. Note that this is a universal sentence in the language of PV_1, thus it can also be formulated as an equivalent PV equation (see [Li25, Chapter 3]).

Formalization of somewhere extractability. In this work, we adopt perfect extractability by adding appropriate rejection in the construction of BARG.TGen. The somewhere extractability property can thus be formalized as the following universal sentence in PV_1 (or equivalently, a PV equation), denoted by BARGext: For all $\lambda, m, s, L \in \mathsf{Log}$, all relation circuit $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ of size at most s, all instances $(\mathbf{x}_1, \ldots, \mathbf{x}_m)$, all proof π, all indices set $\mathcal{S} \subseteq [m]$, and all random seed $\mathsf{sd} \in \{0,1\}^{r(\lambda,m,s,L)}$,

$$\mathsf{Ver}(\mathsf{crs}, C, (x_1, \ldots, x_m), \pi) = 1 \implies \forall i^* \in \mathcal{S}, C\big(\mathbf{x}_{i^*}, \mathsf{Extract}_{i^*}(\mathsf{crs}, \mathsf{td}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi)\big) = 1$$

where $(\mathsf{crs}, \mathsf{td}) := \mathsf{TGen}(1^\lambda, 1^m, 1^s, 1^L, \mathcal{S}; \mathsf{sd})$, and $\mathsf{Extract}_{i^*}$ is the subroutine of Extract that only outputs the extracted witness on index i^*.

Formalization of CRS indistinguishability. Similar to Section 4.3 and 8.2, we start by formalizing the parameterized CRS indistinguishability for BARG.

Definition 10.2 (Parameterized CRS Indistinguishability for BARG, in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$)**.** Let $r(\lambda, m, s, L)$ be the length of the random seed taken by BARG.Gen and BARG.TGen, and $\ell(\lambda, m, s, L)$

be the length of the common reference string generated by BARG.Gen and BARG.TGen. For all indices set $\mathcal{S} \subseteq [m]$ where $|\mathcal{S}| \leq L$, we define $\mathsf{Gen}_{\mathcal{S}}^{\lambda,m,s,L} : \{0,1\}^{r(\lambda,m,s,L)} \to \{0,1\}^{\ell(\lambda,m,s,L)}$ be the following circuits:

- $\mathsf{Gen}_{\varnothing}^{\lambda,m,s,L}(\mathsf{sd}) = \mathsf{Gen}(1^\lambda, 1^m, 1^s, 1^L; \mathsf{sd})$,
- For $\mathcal{S} \subseteq [m]$, $\mathsf{Gen}_{\mathcal{S}}^{\lambda,m,s,L}(\mathsf{sd})$ computes $(\mathsf{crs}, \mathsf{td}) = \mathsf{TGen}(1^\lambda, 1^m, 1^s, 1^L, \mathcal{S}; \mathsf{sd})$ and outputs crs.

Let $\mathcal{D}_{\mathcal{S}}$ for be the distribution corresponding to $\mathsf{Gen}_{\mathcal{S}}^{\lambda,m,s,L}(\mathsf{sd})$. The sentence $\mathsf{BARGind}_{\lambda,m,s,L}^{t,\varepsilon}$ in the language of $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$" is defined as

$$\forall \mathcal{S} \subseteq [m] \text{ where } |\mathcal{S}| \leq L, \mathcal{D}_{\varnothing} \approx_{t,\varepsilon} \mathcal{D}_{\mathcal{S}}.$$

Definition 10.3 (CRS Indistinguishability for BARG, in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$")**.** Let $\lambda_0[t, m, s]$ be a mapping from polynomials to $\mathbb{N}$ which specifies the largest security parameter for which CRS indistinguishability does not hold. We define $\mathsf{BARGind}_{\lambda_0}$ as the following set of sentences: For all PV functions $t(\lambda), m(\lambda), s(\lambda), L(\lambda) : \mathsf{Log} \to \mathsf{Log}$ that are polynomials, it includes the sentence

$$\forall \lambda > \lambda_0[t, m, s, L], \mathsf{BARGind}_{\lambda,m(\lambda),s(\lambda),L(\lambda)}^{t(\lambda),1/t(\lambda)}.$$

10.2 WW BARG Construction

In this section, we recall the BARG construction for single point extraction of Waters and Wu [WW22]. The construction is mainly identical to the presentation in [WW22], but simplified to be based on SXDH assumption for easier formalization in $\mathsf{APC}_1 +$"$\mathsf{prBPP} = \mathsf{prP}$". See Remark 10.5 for the full list of differences with the original construction.

Notation. Fix a bilinear group description $\mathcal{G} = (1^\lambda, 1^\kappa, p, \mathsf{Map}, \mathsf{Val}, \mathsf{Add}, \mathsf{Mul}, \mathsf{Pair})$ as described in Section 4.4. We denote the group elements $\mathsf{Map}(a, b)$ by $[a]_b \in \{0,1\}^\kappa$ for $b \in \{1, 2, T\}$. We use $[a]_b + [a']_b$ to denote $\mathsf{Add}([a]_b, [a']_b)$, $c[a]_b$ to denote $\mathsf{Mul}(c, [a]_b)$, and $[a]_1[b]_2$ or $[a]_1 \cdot [b]_2$ to denote $\mathsf{Pair}([a]_1, [b]_2)$. For matrix $\mathbf{A} \in \mathbb{Z}_p^{n \times m}$, we use $[\mathbf{A}]_b$ to denote elementwise mapping and extend the remaining operations naturally.

Construction 10.4 (WW BARG [WW22])**.** In the following, the relation circuit C is always parsed as a circuit consisting only NAND gates.

Let $\mathsf{GroupGen}$ be a prime-order bilinear group generator. The BARG is constructed as follows.

- $\mathsf{Gen}(1^\lambda, 1^m, 1^s)$: On input security parameter λ, the number of instances m, and the circuit size upper bound s, the algorithm does the following:
  - Run $\mathsf{GroupGen}(1^\lambda)$ to obtain $\mathcal{G} = (1^\kappa, p, \mathsf{Map}, \mathsf{Val}, \mathsf{Add}, \mathsf{Mul}, \mathsf{Pair})$.
  - Sample $\mathbf{m}, \hat{\mathbf{m}} \leftarrow \mathbb{Z}_p^2$.
  - For each $i \in [m]$, sample $\alpha_i, \hat{\alpha}_i \leftarrow \mathbb{Z}_p$ and set $\mathbf{a}_i = \alpha_i \mathbf{m}, \hat{\mathbf{a}}_i = \hat{\alpha}_i \hat{\mathbf{m}}$. Set $\mathbf{a} = \sum_{i \in [m]} \mathbf{a}_i, \hat{\mathbf{a}} = \sum_{i \in [m]} \hat{\mathbf{a}}_i$.
  - For each $i, j \in [m]$ where $i \neq j$, sample $r_{i,j} \leftarrow \mathbb{Z}_p$ and set

$$\mathbf{b}_{i,j} = (\alpha_i \hat{\alpha}_j + r_{i,j})\mathbf{m}, \qquad \hat{\mathbf{b}}_{i,j} = -r_{i,j}\hat{\mathbf{m}}. \tag{10.1}$$

- Output $\mathsf{crs} = (\mathcal{G}, [\mathbf{m}]_1, [\hat{\mathbf{m}}]_2, [\mathbf{a}]_1, [\hat{\mathbf{a}}]_2, \{[\mathbf{a}_i]_1, [\hat{\mathbf{a}}_i]_2\}_{i\in[m]}, \{[\mathbf{b}_{i,j}]_1, [\hat{\mathbf{b}}_{i,j}]_2\}_{i,j\in[m],i\neq j})$

- $\mathsf{TGen}(1^\lambda, 1^m, 1^s, i^*)$: The trapdoor CRS generation algorithm samples a different CRS as follows. The difference with the normal setup is highlighted.

  - Run $\mathsf{GroupGen}(1^\lambda)$ to obtain $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e)$.
  - Sample $\mathbf{m}, \hat{\mathbf{m}} \leftarrow \mathbb{Z}_p^2$.
  - For each $i \in [m]\backslash\{i^*\}$, sample $\alpha_i, \hat{\alpha}_i \leftarrow \mathbb{Z}_p$ and set $\mathbf{a}_i = \alpha_i \mathbf{m}$, $\hat{\mathbf{a}}_i = \hat{\alpha}_i \hat{\mathbf{m}}$. Sample $\mathbf{a}_{i^*}, \hat{\mathbf{a}}_{i^*} \leftarrow \mathbb{Z}_p^2$. Set $\mathbf{a} = \sum_{i\in[m]} \mathbf{a}_i$, $\hat{\mathbf{a}} = \sum_{i\in[m]} \hat{\mathbf{a}}_i$.
  - For each $i, j \in [m]$ where $i \neq j$, sample $r_{i,j} \leftarrow \mathbb{Z}_p$. For $i, j \neq i^*$, set

$$\mathbf{b}_{i,j} = (\alpha_i \hat{\alpha}_j + r_{i,j})\mathbf{m}, \qquad \hat{\mathbf{b}}_{i,j} = -r_{i,j}\hat{\mathbf{m}}. \tag{10.2}$$

    Finally, for $i, j \in [m] \setminus \{i^*\}$, set

$$\mathbf{b}_{i,i^*} = r_{i,i^*}\mathbf{m}, \qquad \hat{\mathbf{b}}_{i,i^*} = -r_{i,i^*}\hat{\mathbf{m}} + \alpha_i \hat{\mathbf{a}}_{i^*}, \tag{10.3}$$

    and set

$$\mathbf{b}_{i^*,j} = \hat{\alpha}_j \mathbf{a}_{i^*} + r_{i^*,j}\mathbf{m}, \qquad \hat{\mathbf{b}}_{i^*,j} = -r_{i^*,j}\hat{\mathbf{m}}. \tag{10.4}$$

  - If $(\mathbf{m}, \mathbf{a}_{i^*})$ is not a basis of $\mathbb{Z}_p^2$, i.e., if $\mathbf{a}_{i^*} = (a_{i^*,1}, a_{i^*,2})^\mathsf{T}$ and $\mathbf{m} = (m_1, m_2)^\mathsf{T}$ satisfies $a_{i^*,1}m_2 - a_{i^*,2}m_1 = 0$, output $(\mathsf{crs}, \mathsf{td}) = (\perp, \perp)$. Otherwise, Compute normal vector $\tau = (a_{i^*,1}m_2 - a_{i^*,2}m_1)^{-1}(-m_2, m_1)^\mathsf{T}$ such that $\tau^\mathsf{T}\mathbf{a}_{i^*} = 1$ and $\tau^\mathsf{T}\mathbf{m} = 0$.
    Similarly, if $(\hat{\mathbf{m}}, \hat{\mathbf{a}}_{i^*})$ is not a basis of $\mathbb{Z}_p^2$, output $(\mathsf{crs}, \mathsf{td}) = (\perp, \perp)$. Otherwise, compute $\hat{\tau}$ such that $\hat{\tau}^\mathsf{T}\hat{\mathbf{a}}_{i^*} = 1$ and $\hat{\tau}^\mathsf{T}\hat{\mathbf{m}} = 0$.
  - Output $\mathsf{crs} = (\mathcal{G}, [\mathbf{m}]_1, [\hat{\mathbf{m}}]_2, [\mathbf{a}]_1, [\hat{\mathbf{a}}]_2, \{[\mathbf{a}_i]_1, [\hat{\mathbf{a}}_i]_2\}_{i\in[m]}, \{[\mathbf{b}_{i,j}]_1, [\hat{\mathbf{b}}_{i,j}]_2\}_{i,j\in[m],i\neq j})$ along with trapdoor $\mathsf{td} = (\tau, \hat{\tau})$.

- $\mathsf{Prove}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), (\mathbf{y}_1, \ldots, \mathbf{y}_m))$: On input the common reference string crs, the relation circuit $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$, instances $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \{0,1\}^n$, and witnesses $\mathbf{y}_1, \ldots, \mathbf{y}_m \in \{0,1\}^h$, the algorithm does the following:

  - If $\mathsf{crs} = \perp$, output $\pi = \perp$.
  - Let $t \leq s$ be the number of wires in C, where the first n wires are inputs wires corresponding to the instance and the following h wires are input wires corresponding to the witness. Let $w_{i,d} \in \{0,1\}$ be the value of wire $d \in [t]$ in circuit evaluation $C(\mathbf{x}_i, \mathbf{y}_i)$.
  - **Wire commitment**. For each wire $d \in [t]$, compute

$$[\mathbf{u}_d]_1 = \sum_{i\in[m]} w_{i,d}[\mathbf{a}_i]_1, \qquad [\hat{\mathbf{u}}_d]_2 = \sum_{i\in[m]} w_{i,d}[\hat{\mathbf{a}}_i]_2 \tag{10.5}$$

  - **Input validation.** For each wire $d \in \{n+1 \ldots n+h\}$ corresponding to the witness input, compute

$$[\mathbf{v}_{d,1}]_1 = \sum_{i\neq j}(1 - w_{i,d})w_{j,d}[\mathbf{b}_{i,j}]_1, \qquad [\hat{\mathbf{v}}_{d,1}]_2 = \sum_{i\neq j}(1 - w_{i,d})w_{j,d}[\hat{\mathbf{b}}_{i,j}]_2 \tag{10.6}$$

along with

$$[\mathbf{v}_{d,2}]_1 = \sum_{i \neq j} w_{i,d}(1 - w_{j,d})[\mathbf{b}_{i,j}]_1, \qquad [\hat{\mathbf{v}}_{d,2}]_2 = \sum_{i \neq j} w_{i,d}(1 - w_{j,d})[\hat{\mathbf{b}}_{i,j}]_2. \tag{10.7}$$

– **Gate validation.** For each NAND gate g_ℓ (where $\ell \in [s]$) with input wire (d_1, d_2) and output wire d_3, compute

$$[\mathbf{w}_{\ell,1}]_1 = \sum_{i \neq j}(1 - w_{i,d_1}w_{j,d_2} - w_{j,d_3})[\mathbf{b}_{i,j}]_1, \quad [\hat{\mathbf{w}}_{\ell,1}]_2 = \sum_{i \neq j}(1 - w_{i,d_1}w_{j,d_2} - w_{j,d_3})[\hat{\mathbf{b}}_{i,j}]_2 \tag{10.8}$$

along with

$$[\mathbf{w}_{\ell,2}]_1 = \sum_{i \neq j}(1 - w_{i,d_1}w_{j,d_2} - w_{i,d_3})[\mathbf{b}_{i,j}]_1, \quad [\hat{\mathbf{w}}_{\ell,2}]_2 = \sum_{i \neq j}(1 - w_{i,d_1}w_{j,d_2} - w_{i,d_3})[\hat{\mathbf{b}}_{i,j}]_2. \tag{10.9}$$

– Output the proof

$$\pi = (\{[\mathbf{u}_d]_1, [\hat{\mathbf{u}}_d]_2\}_{d\in[t]}, \{[\mathbf{v}_{d,i}]_1, [\hat{\mathbf{v}}_{d,i}]_2\}_{d\in[n+1\ldots n+h],i\in 1,2}, \{[\mathbf{w}_{\ell,i}]_1, [\hat{\mathbf{w}}_{\ell,i}]_2\}_{\ell\in[s],i\in 1,2})$$

- $\mathsf{Ver}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi)$: On input the common reference string crs , the relation circuit $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$, instances $(\mathbf{x}_1, \ldots, \mathbf{x}_m)$, and proof π, the algorithm executes the following checks:

  – **Aborting CRS.** If $\mathsf{crs} = \perp$ or $\pi = \perp$, reject.

  – **Group element well-formedness.** For each group element X in the proof π, check that $\mathsf{Val}(X, b) = 1$ for appropriate $b \in \{1, 2\}$. If all check passed, parse the proof π as

$$\pi = (\{[\mathbf{u}_d]_1, [\hat{\mathbf{u}}_d]_2\}_{d\in[t]}, \{[\mathbf{v}_{d,i}]_1, [\hat{\mathbf{v}}_{d,i}]_2\}_{d\in[n+1\ldots n+h],i\in 1,2}, \{[\mathbf{w}_{\ell,i}]_1, [\hat{\mathbf{w}}_{\ell,i}]_2\}_{\ell\in[s],i\in 1,2}),$$

  – **Statement consistency.** For each $d \in [n]$, check that

$$[\mathbf{u}_d]_1 = \sum_{i\in[m]} x_{i,d}[\mathbf{a}_i]_1, \qquad [\hat{\mathbf{u}}_d]_2 = \sum_{i\in[m]} x_{i,d}[\hat{\mathbf{a}}_i]_2 \tag{10.10}$$

  – **Witness validity.** For each wire $d \in \{n + 1 \ldots n + h\}$ corresponding to the witness input, check that

$$[\mathbf{a}]_1 \cdot [\hat{\mathbf{u}}_d^\mathsf{T}]_2 = [\mathbf{u}_d]_1 \cdot [\hat{\mathbf{u}}_d^\mathsf{T}]_2 + [\mathbf{m}]_1 \cdot [\hat{\mathbf{v}}_{d,1}^\mathsf{T}]_2 + [\mathbf{v}_{d,1}]_1 \cdot [\hat{\mathbf{m}}^\mathsf{T}]_2 \tag{10.11}$$

and that

$$[\mathbf{u}_d]_1 \cdot [\hat{\mathbf{a}}^\mathsf{T}]_2 = [\mathbf{u}_d]_1 \cdot [\hat{\mathbf{u}}_d^\mathsf{T}]_2 + [\mathbf{m}]_1 \cdot [\hat{\mathbf{v}}_{d,2}^\mathsf{T}]_2 + [\mathbf{v}_{d,2}]_1 \cdot [\hat{\mathbf{m}}^\mathsf{T}]_2 \tag{10.12}$$

  – **Gate validity.** For each NAND gate g_ℓ (where $\ell \in [s]$) with input wire (d_1, d_2) and output wire d_3, check that

$$[\mathbf{a}]_1 \cdot [\hat{\mathbf{a}}^\mathsf{T}]_2 = [\mathbf{u}_{d_1}]_1 \cdot [\hat{\mathbf{u}}_{d_2}^\mathsf{T}]_2 + [\mathbf{a}]_1 \cdot [\hat{\mathbf{u}}_{d_3}^\mathsf{T}]_2 + [\mathbf{m}]_1 \cdot [\hat{\mathbf{w}}_{\ell,1}^\mathsf{T}]_2 + [\mathbf{w}_{\ell,1}]_1 \cdot [\hat{\mathbf{m}}^\mathsf{T}]_2 \tag{10.13}$$

and that

$$[\mathbf{a}]_1 \cdot [\hat{\mathbf{a}}^\mathsf{T}]_2 = [\mathbf{u}_{d_1}]_1 \cdot [\hat{\mathbf{u}}_{d_2}^\mathsf{T}]_2 + [\mathbf{u}_{d_3}]_1 \cdot [\hat{\mathbf{a}}^\mathsf{T}]_2 + [\mathbf{m}]_1 \cdot [\hat{\mathbf{w}}_{\ell,2}^\mathsf{T}]_2 + [\mathbf{w}_{\ell,2}]_1 \cdot [\hat{\mathbf{m}}^\mathsf{T}]_2 \tag{10.14}$$

– **Output validity.** Check that $[\mathbf{u}_t]_1 = [\mathbf{a}]_1$ and $[\hat{\mathbf{u}}_t]_2 = [\hat{\mathbf{a}}]_2$.

The algorithm outputs 0 if any of the checks fail, and 1 otherwise.

- $\mathsf{Extract}(\mathsf{crs}, \mathsf{td}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi)$: On input the crs, the trapdoor $\mathsf{td} = (\tau, \hat{\tau})$, the relation circuit $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$, instances $(\mathbf{x}_1, \ldots, \mathbf{x}_m)$, and proof

$$\pi = (\{[\mathbf{u}_d]_1, [\hat{\mathbf{u}}_d]_2\}_{d\in[t]}, \{[\mathbf{v}_{d,i}]_1, [\hat{\mathbf{v}}_{d,i}]_2\}_{d\in[n+1\ldots n+h],i\in 1,2}, \{[\mathbf{w}_{\ell,i}]_1, [\hat{\mathbf{w}}_{\ell,i}]_2\}_{\ell\in[s],i\in 1,2}),$$

the extraction algorithm computes $\tau^\mathsf{T}[\mathbf{u}_d]_1$ for each $d \in [n+1 \ldots n+h]$. For each of the wire corresponding to the witness input, it set $w_d = 0$ if $\tau^\mathsf{T}[\mathbf{u}_d]_1 = [0]_1$, $w_d = 1$ if $\tau^\mathsf{T}[\mathbf{u}_d]_1 = [1]_1$, and $w_d = \perp$ otherwise. If any of the w_d is $\perp$, output $\perp$. Otherwise, output $\mathbf{y} = (w_{n+1}, \ldots, w_{n+h})$.

Remark 10.5 (Differences with WW BARG [WW22])**.** To simplify the later proofs in the language of APC_1, we made the following simplifications to the original construction:

- The work [WW22] builds BARG from the k-Linear assumption for arbitrary k. We restrict the construction by setting $k = 1$, i.e., the SXDH assumption.
- For perfect extractability, we modify the $\mathsf{BARG.TGen}$ algorithm to reject invalid crs.
- To simplify the argument of $\mathsf{BARG.TGen}$ as a PV function, we ensure that the trapdoor vector τ is chosen deterministically such that $\tau^\mathsf{T}\mathbf{m} = 0, \tau^\mathsf{T}\mathbf{a} \neq 0$.

The remaining of this construction aligns with [WW22].

10.3 Preparations: Number-Theoretic Lemmas

In this section, we prove several number-theoretic lemmas in PV_1 and $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$", which will be useful in the security analysis of WW BARG scheme. We first start with the lemma that every vector in $\mathbb{Z}_p^2$ can be uniquely decomposed into a linear combination of the two basis vectors.

Lemma 10.6. *The following sentence holds in* PV_1*. Let* $p \geq 2$ *be a primer number (i.e.* $\forall a \in (1,p),\ a \nmid p$*). Then for every* $\mathbf{a} = (a_1, a_2), \mathbf{m} = (m_1, m_2), \mathbf{u} = (u_1, u_2) \in \mathbb{Z}_p^2$ *such that* $a_1 m_2 - a_2 m_1 \neq 0$*,* $(\xi, \zeta) = \mathsf{Decomp}((\mathbf{a}, \mathbf{m}), \mathbf{u})$ *is the unique pair in* $\mathbb{Z}_p^2$ *such that*

$$\xi\mathbf{a} + \zeta\mathbf{m} = \mathbf{u},$$

where $\mathsf{Decomp}((\mathbf{a}, \mathbf{m}), \mathbf{u})$ *is the circuit computing*

$$\mathsf{Decomp}((\mathbf{a}, \mathbf{m}), \mathbf{u}) := \left((u_1 m_2 - u_2 m_1)(a_1 m_2 - a_2 m_1)^{-1}, (a_1 u_2 - a_2 u_1)(a_1 m_2 - a_2 m_1)^{-1}\right).$$

Proof Sketch. We argue in PV_1. It can be verified that $(\xi, \zeta) = \mathsf{Decomp}((\mathbf{a}, \mathbf{m}), \mathbf{u})$ satisfies $\xi\mathbf{a} + \zeta\mathbf{m} = \mathbf{u}$. For uniqueness, if (ξ', ζ') also satisfies $\xi'\mathbf{a} + \zeta'\mathbf{m} = \mathbf{u}$, then we have $(\xi - \xi')\mathbf{a} + (\zeta - \zeta')\mathbf{m} = 0$. If $\xi \neq \xi'$, then we have $\mathbf{a} = c\mathbf{m}$ for $c = -(\zeta - \zeta')(\xi - \xi')^{-1}$, therefore $a_1 m_2 - a_2 m_1 = c m_1 m_2 - c m_2 m_1 = 0$, contradicting the assumption. Similarly, if $\zeta \neq \zeta'$, we also reach a contradiction. Therefore, $\xi = \xi'$ and $\zeta = \zeta'$, completing the proof. □

We need the fact that the multiplicative inverse in $\mathbb{Z}_p$ is efficiently computable by an algorithm $\mathsf{Inv}(\cdot,\cdot)$, and its correct is provable in PV_1. Formally:

Lemma 10.7 ($\mathbb{Z}_p$ Multiplicative-Inverse Lemma)**.** *There exist a* PV *function* $\mathsf{Inv}(\cdot,\cdot)$ *such that we can prove the following sentence in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$*: Suppose that* p *is a prime number (i.e.* $\forall a \in (1,p),\ a \nmid p$*), then for every* $a \in (1,p)$*,* $b = \mathsf{Inv}(a,p)$ *is the unique number in* $\mathbb{Z}_p$ *such that* $ab = 1$*.*

Proof Sketch. The lemma follows immediately from the fact that the correctness of the Extended Euclidean Algorithm can be proven in PV_1 [Jeř05, Section 4.3.1]. In particular, for all a, b, we can prove that

$$\mathsf{ExtGCD}(a,b) = (g,x,y) \implies g|a \wedge g|b \wedge ax + by = g.$$

Therefore, we can prove that

$$\forall a \in (1,p),\ a \nmid p \implies \forall b < p, \mathsf{ExtGCD}(b,p) = (1,x,y)$$

By defining $\mathsf{Inv}(a,p)$ to be $x \bmod p$, where $\mathsf{ExtGCD}(a,p) = (1,x,y)$, we can prove that

$$\forall a \in (1,p),\ a \nmid p \implies \forall b < p, b \cdot \mathsf{Inv}(b,p) = 1 \pmod p.$$

Finally, if $c \neq \mathsf{Inv}(b,p)$ satisfies $c \cdot b \equiv 1 \pmod p$, then we have $(c - \mathsf{Inv}(b,p)) \cdot b = 0 \pmod p$, which implies either $c - \mathsf{Inv}(b,p) = 0$ or $b \in (1,p) \wedge b|p$. Either case contradicts the assumption. □

The following lemma shows in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ that random vectors form a basis of $\mathbb{Z}_p^2$ with high probability. Note that random vectors are sampled using the imperfect sampler $\mathsf{Samp}_p(\cdot)$.

Lemma 10.8 (Full-Rank Lemma)**.** *The following sentence is provable in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$*. Let* $R(\mathsf{sd} = (\mathsf{sd}_1, \mathsf{sd}_2, \mathsf{sd}_3, \mathsf{sd}_4))$ *be the circuit sampling random* $\mathbb{Z}_p$ *elements* (s_1, s_2, s_3, s_4) *from* $\mathsf{Samp}_p(\mathsf{sd}_i)$*, outputting* 1 *if* $s_1 s_4 - s_2 s_3 = 0$ *and* 0 *otherwise.*

Then for all $p > 1$ *and* $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$*, if* p *is a prime number (i.e.* $\forall a \in (1,p),\ a \nmid p$*), then*

$$\Pr_\delta[R] \leq \delta + \beta + 2p^{-1}.$$

Proof. We argue in APC_1. Fix $p > 1$ and $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$. The acceptance set of R can be separated by two cases, either $s_1 = 0$ and $s_2 = s_3^{-1}$, or $s_1 \neq 0$ and $s_4 = s_1^{-1}(1 - s_2 s_3)$. Let C_1 be the circuit that that outputs 1 if $s_1 = 0$, C_2 be the circuit that outputs 1 if $s_1 \neq 0 \wedge s_4 = s_1^{-1}(1 - s_2 s_3)$, and $\eta^{-1} \in \mathsf{Log}$ be a parameter to be determined. By the Monotonicity of CAPP and the Union Bound, we have

$$\Pr_\eta[R] \leq \Pr_\eta[C_1 \vee C_2] + 3\eta \leq \Pr_\eta[C_1] + \Pr_\eta[C_2] + 7\eta. \tag{10.15}$$

We will then argue that both $\Pr_\eta[C_1]$ and $\Pr_\eta[C_2]$ are small.

The circuit C_1. Note that C_1 samples $s_1, \ldots, s_4$ but only reads the part of s_1. Let α be the set of indices corresponding to the seeds of s_2, s_3, s_4. Note that for any assignment ρ to variables in α, by the correctness of Samp (the $\mathbb{Z}_p$ Sampling Lemma) with $y := 0$, we have $\Pr_\eta[C_1|_\rho] \leq 2\eta + p^{-1}$. By the Averaging Argument, it follows that

$$\Pr_\eta[C_1] \leq p^{-1} + 5\eta. \tag{10.16}$$

The circuit C_2. Similar to the first case, let α be the set of indices corresponding to the seeds of s_1, s_2, s_3. For any assignment ρ to variables in α, by the correctness of Samp (the $\mathbb{Z}_p$ Sampling Lemma) with $y := s_1^{-1}(1 - s_2 s_3)$, we have $\Pr_\eta[C_2|_\rho] \leq 2\eta + p^{-1}$. By the Averaging Argument, it follows that

$$\Pr_\eta[C_2] \leq p^{-1} + 5\eta. \tag{10.17}$$

Wrapping up. Finally, we combine Equations (10.15) to (10.17) and Precision Consistency of CAPP to obtain

$$\Pr_\delta[R] \leq \Pr_\eta[R] + (\delta + 2\eta) \leq 2p^{-1} + \delta + 19\eta. \tag{10.18}$$

This completes the proof by setting $\eta := \beta/20$. □

10.4 Security Analysis of BARG

In this section, we formalize the security of the WW BARG construction. The correctness and extractability properties are proved in PV_1, while the CRS indistinguishability is proved in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$.

10.4.1 Correctness

Theorem 10.9 (WW BARG Correctness Theorem)**.** *The correctness property (Definition 10.1) of the WW BARG construction (Construction 10.4) can be proved in* $\mathsf{PV}_1 + \mathsf{Cor}_{\mathsf{GroupGen}}$.

Proof (Sketch). The correctness follows immediately from the correctness of the group operations. Honest proofs can be shown to satisfy all the verification equations via PV through expanding the constructions of the Prove algorithm. In particular,

- The statement consistency (Equation (10.10)) follows from the construction of the wire commitment elements (Equation (10.5)).
- The witness validity (Equations (10.11) and (10.12)) follows from the construction of the input validation elements (Equations (10.6) and (10.7)).
- The gate validity (Equations (10.13) and (10.14)) follows from the construction of the gate validation elements (Equations (10.8) and (10.9)).
- The output validity follows from the construction of the wire commitment elements (Equation (10.5)), along with the guarantee that $C(x_i, y_i) = 1$ for all $i \in [m]$. □

10.4.2 Somewhere Extractability

Theorem 10.10 (WW BARG Somewhere Extractability Theorem)**.** *The somewhere extractability property (Definition 10.1) of the WW BARG construction (Construction 10.4) can be proved in* $\mathsf{PV}_1 + \mathsf{Cor}_{\mathsf{GroupGen}}$.

Proof. Let $(\mathsf{crs}, \mathsf{td}) := \mathsf{TGen}(1^\lambda, 1^m, 1^s, 1^{i^*}; \mathsf{sd})$. By construction, whenever $\mathsf{crs} = \bot$, the verification algorithm always rejects. In this case, the sentence

$$\mathsf{Ver}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi) = 1 \implies C(\mathbf{x}_{i^*}, \mathsf{Extract}(\mathsf{crs}, \mathsf{td}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi)) = 1$$

holds trivially as the premise is false. Therefore, we only need to consider the case when $\mathsf{crs} \neq \bot$.

Suppose that $\mathsf{crs} \neq \bot$. By the construction of the TGen algorithm, we have that $(\mathbf{m}, \mathbf{a}_{i^*})$ is a basis of $\mathbb{Z}_p^2$, and so is $(\hat{\mathbf{m}}, \hat{\mathbf{a}}_{i^*})$. Note that by the construction of the TGen algorithm, the trapdoor $\mathsf{td} = (\tau, \hat{\tau})$ satisfies $\tau^\mathsf{T}\mathbf{u} = \mathsf{Decomp}((\mathbf{a}_{i^*}, \mathbf{m}), \mathbf{u})_1$, the first coordinate of the decomposition of $\mathbf{u}$ under basis $(\mathbf{a}_{i^*}, \mathbf{m})$. Similarly, $\hat{\tau}^\mathsf{T}\mathbf{u} = \mathsf{Decomp}((\hat{\mathbf{a}}_{i^*}, \hat{\mathbf{m}}), \mathbf{u})_1$.

Now, consider any $\mathsf{crs} \neq \bot$ and proof π such that $\mathsf{Ver}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi) = 1$. Our goal is to show that $C(\mathbf{x}_{i^*}, \mathsf{Extract}(\mathsf{crs}, \mathsf{td}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi)) = 1$.

First, since the proof π passes the group element well-formedness check, by the correctness axioms of GroupGen, we know that there exists some suitable $\mathbf{u}_d, \hat{\mathbf{u}}_d, \mathbf{v}_{d,i}, \hat{\mathbf{v}}_{d,i}, \mathbf{w}_{\ell,i}, \hat{\mathbf{w}}_{\ell,i} \in \mathbb{Z}_p^2$, such that we can parse π as

$$\pi = (\{[\mathbf{u}_d]_1, [\hat{\mathbf{u}}_d]_2\}_{d\in[t]}, \{[\mathbf{v}_{d,i}]_1, [\hat{\mathbf{v}}_{d,i}]_2\}_{d\in[n+1\ldots n+h],i\in 1,2}, \{[\mathbf{w}_{\ell,i}]_1, [\hat{\mathbf{w}}_{\ell,i}]_2\}_{\ell\in[s],i\in 1,2}).$$

Let $(\xi_d, \zeta_d) = \mathsf{Decomp}((\mathbf{a}_{i^*}, \mathbf{m}), \mathbf{u}_d)$ and $(\hat{\xi}_d, \hat{\zeta}_d) = \mathsf{Decomp}((\hat{\mathbf{a}}_{i^*}, \hat{\mathbf{m}}), \hat{\mathbf{u}}_d)$ for each $d \in [t]$ be the decomposition of $\mathbf{u}_d, \hat{\mathbf{u}}_d$ under basis $(\mathbf{a}_{i^*}, \mathbf{m})$ and $(\hat{\mathbf{a}}_{i^*}, \hat{\mathbf{m}})$ respectively.

Next, since the proof π passes the statement consistency check (Equation (10.10)), by the correctness axioms of GroupGen, we have that for each $d \in [n]$,

$$\mathbf{u}_d = \sum_{i\in[m]} x_{i,d}\mathbf{a}_i = x_{i^*,d}\mathbf{a}_{i^*} + \left(\sum_{i\neq i^*} x_{i,d}\alpha_i\right)\mathbf{m}, \quad \hat{\mathbf{u}}_d = \sum_{i\in[m]} x_{i,d}\hat{\mathbf{a}}_i = x_{i^*,d}\hat{\mathbf{a}}_{i^*} + \left(\sum_{i\neq i^*} x_{i,d}\hat{\alpha}_i\right)\hat{\mathbf{m}}.$$

By the uniqueness of decomposition (Lemma 10.6), we thus have $\xi_d = \hat{\xi}_d = x_{i^*,d}$ for each $d \in [n]$.

Furthermore, for each wire $d \in [n+1\ldots n+h]$ corresponding to the witness input, since the proof π passes the witness validity checks (Equations (10.11) and (10.12)), by the correctness axioms of GroupGen, we have the following equations in $\mathbb{Z}_p$:

$$\mathbf{a}\hat{\mathbf{u}}_d^\mathsf{T} = \mathbf{u}_d\hat{\mathbf{u}}_d^\mathsf{T} + \mathbf{m}\hat{\mathbf{v}}_{d,1}^\mathsf{T} + \mathbf{v}_{d,1}\hat{\mathbf{m}}^\mathsf{T}, \qquad \mathbf{u}_d\hat{\mathbf{a}}^\mathsf{T} = \mathbf{u}_d\hat{\mathbf{u}}_d^\mathsf{T} + \mathbf{m}\hat{\mathbf{v}}_{d,2}^\mathsf{T} + \mathbf{v}_{d,2}\hat{\mathbf{m}}^\mathsf{T}.$$

Multiplying τ^T on the left and $\hat{\tau}$ on the right canceling out $\tau^\mathsf{T}\mathbf{m} = \hat{\mathbf{m}}^\mathsf{T}\hat{\tau} = 0$, we have

$$(\tau^\mathsf{T}\mathbf{a})(\hat{\mathbf{u}}_d^\mathsf{T}\hat{\tau}) = (\tau^\mathsf{T}\mathbf{u}_d)(\hat{\mathbf{u}}_d^\mathsf{T}\hat{\tau}), \qquad (\tau^\mathsf{T}\mathbf{u}_d)(\hat{\mathbf{a}}^\mathsf{T}\hat{\tau}) = (\tau^\mathsf{T}\mathbf{u}_d)(\hat{\mathbf{u}}_d^\mathsf{T}\hat{\tau}).$$

Observe that

$$\tau^\mathsf{T}\mathbf{a} = \tau^\mathsf{T}\left(\mathbf{a}_{i^*} + \sum_{i\neq i^*}\alpha_i\mathbf{m}\right) = 1, \qquad \hat{\mathbf{a}}^\mathsf{T}\hat{\tau} = \left(\hat{\mathbf{a}}_{i^*} + \sum_{i\neq i^*}\hat{\alpha}_i\hat{\mathbf{m}}\right)^\mathsf{T}\hat{\tau} = 1.$$

Also, by Lemma 10.6, we have $\tau^\mathsf{T}\mathbf{u}_d = \xi_d$ and $\hat{\mathbf{u}}_d^\mathsf{T}\hat{\tau} = \hat{\xi}_d$. Therefore the verification check implies that $\hat{\xi}_d = \xi_d\hat{\xi}_d, \xi_d = \xi_d\hat{\xi}_d$, Hence $\xi_d = \hat{\xi}_d \in \{0,1\}$, corresponding to a valid witness bit.

Note that the extraction algorithm Extract computes its output $\mathbf{y} = (w_{n+1}, \ldots, w_{n+h})$ by checking whether $\tau^\mathsf{T}[\mathbf{u}_d]_1$ is $[0]_1$ or $[1]_1$ for each $d \in [n+1, \ldots, n+h]$. By the correctness axioms of GroupGen, $w_d = \xi_d$ if $\xi_d \in \{0,1\}$, and $w_d = \perp$ otherwise. Since we proved that $\xi_d \in \{0,1\}$, we have $w_d = \xi_d \neq \perp$ for each $d \in [n+1\ldots n+h]$.

Let $C_d(\mathbf{x}_{i^*}, \mathbf{y})$ be the value of wire d in circuit evaluation $C(\mathbf{x}_{i^*}, \mathbf{y})$, where $\mathbf{y}$ is the extracted witness as described above. In the rest of the proof, we prove by induction on $d \leq [t]$ that $C_d(\mathbf{x}_{i^*}, \mathbf{y}) = \xi_d$ for every $d \leq \ell$. This induction principle is available in PV_1, as the property is decidable by a straightforward polynomial-time algorithm. The base case $d \leq [n+h]$ follows directly from the arguments above.

Now we consider the induction case. Specifically, for each NAND gate g_ℓ (where $\ell \in [s]$) with input wire (d_1, d_2) and output wire d_3, we show that if $\xi_{d_1} = \hat{\xi}_{d_1} = C_{d_1}(\mathbf{x}_{i^*}, \mathbf{y})$ and $\xi_{d_2} = \hat{\xi}_{d_2} = C_{d_2}(\mathbf{x}_{i^*}, \mathbf{y})$, and that the gate validity check (Equations (10.13) and (10.14)) passes on the gate, then we can prove $\xi_{d_3} = \hat{\xi}_{d_3} = C_{d_3}(\mathbf{x}_{i^*}, \mathbf{y})$. By the correctness axioms of GroupGen, the gate validity checks (Equations (10.13) and (10.14)) imply that

$$\mathbf{a}\hat{\mathbf{a}}^\mathsf{T} = \mathbf{u}_{d_1}\hat{\mathbf{u}}_{d_2}^\mathsf{T} + \mathbf{a}\hat{\mathbf{u}}_{d_3}^\mathsf{T} + \mathbf{m}\hat{\mathbf{w}}_{\ell,1}^\mathsf{T} + \mathbf{w}_{\ell,1}\hat{\mathbf{m}}^\mathsf{T}, \qquad \mathbf{a}\hat{\mathbf{a}}^\mathsf{T} = \mathbf{u}_{d_1}\hat{\mathbf{u}}_{d_2}^\mathsf{T} + \mathbf{u}_{d_3}\hat{\mathbf{a}}^\mathsf{T} + \mathbf{m}\hat{\mathbf{w}}_{\ell,2}^\mathsf{T} + \mathbf{w}_{\ell,2}\hat{\mathbf{m}}^\mathsf{T}.$$

Similar to the witness validity checks, by multiplying τ^T from the left and $\hat{\tau}$ from the right, we have

$$(\tau^\mathsf{T}\mathbf{a})(\hat{\mathbf{a}}^\mathsf{T}\hat{\tau}) = (\tau^\mathsf{T}\mathbf{u}_{d_1})(\hat{\mathbf{u}}_{d_2}^\mathsf{T}\hat{\tau}) + (\tau^\mathsf{T}\mathbf{a})(\hat{\mathbf{u}}_{d_3}^\mathsf{T}\hat{\tau}), \qquad (\tau^\mathsf{T}\mathbf{a})(\hat{\mathbf{a}}^\mathsf{T}\hat{\tau}) = (\tau^\mathsf{T}\mathbf{u}_{d_1})(\hat{\mathbf{u}}_{d_2}^\mathsf{T}\hat{\tau}) + (\tau^\mathsf{T}\mathbf{u}_{d_3})(\hat{\mathbf{a}}^\mathsf{T}\hat{\tau}).$$

Which implies that

$$1 = \xi_{d_1}\hat{\xi}_{d_2} + \hat{\xi}_{d_3}, \qquad 1 = \xi_{d_1}\hat{\xi}_{d_2} + \xi_{d_3}.$$

Therefore, $\xi_{d_3} = \hat{\xi}_{d_3} = 1 - \xi_{d_1}\xi_{d_2} = \mathrm{NAND}(\xi_{d_1}, \xi_{d_2}) = \mathrm{NAND}(C_{d_1}, C_{d_2})(\mathbf{x}_{i^*}, \mathbf{y}) = C_{d_3}(\mathbf{x}_{i^*}, \mathbf{y})$.

This completes the induction proof. In particular, it shows that $C_t(\mathbf{x}_{i^*}, \mathbf{y}) = \xi_t = \hat{\xi}_t$. Finally, since the proof π passes the output validity check, we have $\mathbf{u}_t = \mathbf{a}$ and $\hat{\mathbf{u}}_t = \hat{\mathbf{a}}$. Therefore $\xi_t = \tau^\mathsf{T}\mathbf{a} = 1$, and $\hat{\xi}_t = \hat{\mathbf{a}}^\mathsf{T}\hat{\tau} = 1$. Then we have $C(\mathbf{x}_{i^*}, \mathbf{y}) = C_t(\mathbf{x}_{i^*}, \mathbf{y}) = \xi_t = 1$, completing the proof. □

10.4.3 CRS Indistinguishability

We start with the following lemma proving a parallel-version of the SXDH assumption. The following lemma shows that it is provable in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ that the SXDH assumption implies the k parallel version.

Lemma 10.11 (Parallel SXDH Lemma)**.** *The following sentence is provable in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$*. Let* $t, k \in \mathsf{Log}$*,* $\varepsilon \in [0, 1]$*, and* $\mathcal{G}$ *is a bilinear group satisfying* $\mathsf{Cor}_\mathcal{G}$*. Let* $s \in \mathsf{Log}$ *be the maximum size of the challenge generation circuits* $\mathsf{Gen}_0, \mathsf{Gen}_1$ *in Definition 4.25. Then* $\mathsf{SXDH}_\mathcal{G}^{t,\varepsilon}$ *implies* k-$\mathsf{SXDH}_\mathcal{G}^{t-ks,k\varepsilon}$*.*

Proof Sketch. This is achieved by a standard Hybrid Argument, where the indistinguishability of hybrids are proved using the Reduction Lemma. We omit the proof, as a very similar formalization has been done for the security proof of the leveled FHE construction (see Section 8.5). □

Theorem 10.12 (WW BARG CRS Indistinguishability Theorem)**.** *In* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} + \mathsf{SXDH}_{\lambda_0,\mathsf{GroupGen}}$*, the CRS indistinguishability property (Definition 10.1)*[28] *of the WW BARG construction (Construction 10.4) can be proved. In particular, there exists some* λ_0' *such that for all* PV *functions* $m, s, t : \mathsf{Log} \to \mathsf{Log}$ *that are polynomials and* $\lambda > \lambda_0'[m, s, t]$*, the sentence* $\mathsf{BARGind}^{t(\lambda),1/t(\lambda)}_{\lambda,m(\lambda),s(\lambda)}$ *is provable in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} + \mathsf{SXDH}_{\lambda_0,\mathsf{GroupGen}}$*.*

Proof. Fix the function λ_0 (in SXDH) and any PV functions $m, s, t : \mathsf{Log} \to \mathsf{Log}$ that are polynomials, we will prove that there exists a constant $\lambda_0' \in \mathbb{N}$ to be determined later such that for every $\lambda > \lambda_0'$, $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} + \mathsf{SXDH}_{\lambda_0,\mathsf{GroupGen}}$ proves $\mathsf{BARGind}^{t(\lambda),1/t(\lambda)}_{\lambda,m(\lambda),s(\lambda)}$ (see Definition 10.2

Fix any $\lambda > \lambda_0'$ and let $m = m(\lambda), s = s(\lambda), t = t(\lambda)$, our goal is to prove that for every $i^* \in [m]$, the distributions $\mathcal{D}_0$ and $\mathcal{D}_{i^*}$ defined in Definition 10.2 are $(t, 1/t)$-indistinguishable. The distribution $\mathcal{D}_0$ is the real CRS distribution and $\mathcal{D}_i$ is the trapdoor CRS distribution.

Let $\mathcal{G} = (1^\kappa, p, \mathsf{Map}, \mathsf{Val}, \mathsf{Add}, \mathsf{Mul}, \mathsf{Pair})$ be the group generated by $\mathsf{GroupGen}(1^\lambda)$. Recall that as p is a prime by $\mathsf{Cor}_\mathcal{G}$, the correctness of computing multiplicative inverse is provable in PV_1 (the $\mathbb{Z}_p$ Multiplicative-Inverse Lemma). In the rest of the proof, we use a^{-1} to denote $\mathsf{Inv}(a, p)$.

[28] Note that we always set $L(\lambda) = 1$ and omit the parameter for BARG with single point extraction.

Hybrids. Now, we are ready to prove the CRS indistinguishability property. Let $\mathsf{Gen}_i^{\lambda,m,s} : \{0,1\}^{r(\lambda,m,s)} \to \{0,1\}^{\ell(\lambda,m,s)}$ be the circuits in Definition 10.1, instantiated with Construction 10.4.

- $\mathsf{Gen}_0^{\lambda,m,s}(\mathsf{sd}) = \mathsf{Gen}(1^\lambda, 1^m, 1^s; \mathsf{sd})$,
- For $i \in [m]$, $\mathsf{Gen}_i^{\lambda,m,s}(\mathsf{sd})$ computes $(\mathsf{crs}, \mathsf{td}) = \mathsf{TGen}(1^\lambda, 1^m, 1^s, i; \mathsf{sd})$ and outputs crs.

We now argue the indistinguishability between $\mathsf{Gen}_0^{\lambda,m,s}$ and $\mathsf{Gen}_{i^*}^{\lambda,m,s}$ for any $i^* \in [m]$ via the following hybrid circuits.

$\boxed{\mathcal{H}_0}$ This is the original $\mathsf{Gen}_0^{\lambda,m,s}$ algorithm. In particular, the relevant variables are generated as follows:

- For all $i \in [m]$, $\mathbf{a}_i = \alpha_i \mathbf{m}$, $\hat{\mathbf{a}}_i = \hat{\alpha}_i \hat{\mathbf{m}}$.
- For $(i,j) \in [m]^2$ where $i \neq j$, $\mathbf{b}_{i,j} = (\alpha_i \hat{\alpha}_j + r_{i,j})\mathbf{m}$, $\hat{\mathbf{b}}_{i,j} = -r_{i,j}\hat{\mathbf{m}}$.

$\boxed{\mathcal{H}_1}$ This hybrid changes how $\mathbf{b}_{i,i^*}, \hat{\mathbf{b}}_{i,i^*}$ are generated. In particular, they are generated by:

- For $i \neq i^*$, $\mathbf{b}_{i,i^*} = r_{i,i^*}\mathbf{m}$, $\hat{\mathbf{b}}_{i,i^*} = (-r_{i,i^*} + \alpha_i \hat{\alpha}_{i^*})\hat{\mathbf{m}}$.

$\boxed{\mathcal{H}_2}$ In the computation of $\mathbf{b}$ and $\hat{\mathbf{b}}$, this hybrid changes all appearance of $\alpha_i^* \mathbf{m}$ to $\mathbf{a}_{i^*}$, and all appearance of $\hat{\alpha}_{i^*}\hat{\mathbf{m}}$ to $\hat{\mathbf{a}}_{i^*}$. In particular,

- For $j \neq i^*$, $\mathbf{b}_{i^*,j} = \hat{\alpha}_j \mathbf{a}_{i^*} + r_{i^*,j}\mathbf{m}$, $\hat{\mathbf{b}}_{i^*,j} = -r_{i^*,j}\hat{\mathbf{m}}$.
- For $i \neq i^*$, $\mathbf{b}_{i,i^*} = r_{i,i^*}\mathbf{m}$, $\hat{\mathbf{b}}_{i,i^*} = -r_{i,i^*}\hat{\mathbf{m}} + \alpha_i \hat{\mathbf{a}}_{i^*}$.

$\boxed{\mathcal{H}_3}$ This hybrid computes variables dependent on $(\mathbf{m}, \hat{\mathbf{m}}, \mathbf{a}_{i^*}, \hat{\mathbf{a}}_{i^*})$ directly in group elements. In particular, after sampling $\mathbf{m}, \hat{\mathbf{m}}$ and computing $[\mathbf{a}_{i^*}]_1 = \alpha_{i^*}[\mathbf{m}]_1$, $[\hat{\mathbf{a}}_{i^*}]_2 = \hat{\alpha}_{i^*}[\hat{\mathbf{m}}]_2$, the remaining variables are generated as follows:

- $[\mathbf{a}_i]_1 = \alpha_i[\mathbf{m}]_1$, $[\hat{\mathbf{a}}_i]_2 = \hat{\alpha}_i[\hat{\mathbf{m}}]_2$ for $i \neq i^*$.
- $[\mathbf{a}]_1 = \sum_{i\in[m]}[\mathbf{a}_i]_1$, $[\hat{\mathbf{a}}]_2 = \sum_{i\in[m]}[\hat{\mathbf{a}}_i]_2$.
- For $i, j \neq i^*$, $[\mathbf{b}_{i,j}]_1 = (\alpha_i \hat{\alpha}_j + r_{i,j})[\mathbf{m}]_1$, $[\hat{\mathbf{b}}_{i,j}]_2 = -r_{i,j}[\hat{\mathbf{m}}]_2$.
- For $j \neq i^*$, $[\mathbf{b}_{i^*,j}]_1 = \hat{\alpha}_j[\mathbf{a}_{i^*}]_1 + r_{i^*,j}[\mathbf{m}]_1$, $[\hat{\mathbf{b}}_{i^*,j}]_2 = -r_{i^*,j}[\hat{\mathbf{m}}]_2$.
- For $i \neq i^*$, $[\mathbf{b}_{i,i^*}]_1 = r_{i,i^*}[\mathbf{m}]_1$, $[\hat{\mathbf{b}}_{i,i^*}]_2 = -r_{i,i^*}[\hat{\mathbf{m}}]_2 + \alpha_i[\hat{\mathbf{a}}_{i^*}]_2$.

$\boxed{\mathcal{H}_4}$ This hybrid replaces $[\mathbf{a}_{i^*}]_1$ and $[\hat{\mathbf{a}}_{i^*}]_2$ with random group elements (sampled using $\mathsf{Samp}_p(\cdot)$). The remaining variables are generated as in $\mathcal{H}_3$.

$\boxed{\mathcal{H}_5}$ This hybrid is identical with $\mathcal{H}_4$, except that every variables are computed as $\mathbb{Z}_p$ elements until the output.

$\boxed{\mathcal{H}_6}$ This is hybrid adds the check that $(\mathbf{m}, \mathbf{a}_{i^*})$ and $(\hat{\mathbf{m}}, \hat{\mathbf{a}}_{i^*})$ form bases of $\mathbb{Z}_p^2$. If not, it outputs $\mathsf{crs} = \bot$. This is identical to the algorithm $\mathsf{Gen}_{i^*}^{\lambda,m,s}$ algorithm.

We note that some of these hybrids may have different seed length. Nevertheless, we can pad dummy input to the circuits generating the distributions so that indistinguishability between them is well-defined.

Indistinguishability between hybrids. We now argue the indistinguishability between each two consecutive hybrids.

Lemma 10.13. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves that for every* $t \in \mathsf{Log}$, $\mathcal{H}_0 \approx_{t,mp^{-1}} \mathcal{H}_1$.

Proof. We argue in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. Fix any $t \in \mathsf{Log}$. Recall that Gen_0 and Gen_1 output (the group encoding of) $\mathbf{m}, \hat{\mathbf{m}} \leftarrow \mathbb{Z}_p^2$, $\mathbf{a}_i = \alpha_i \mathbf{m}, \hat{\mathbf{a}}_i = \hat{\alpha}_i \hat{\mathbf{m}}$ (for $\alpha_i, \hat{\alpha}_i \leftarrow \mathbb{Z}_p$), $\mathbf{a} = \sum_{i\in[m]} \mathbf{a}$, $\hat{\mathbf{a}} = \sum_{i\in[m]} \hat{\mathbf{a}}$, and $\mathbf{b}_{i,j}, \hat{\mathbf{b}}_{i,j}$. Let G_0, G_1 be subcircuits of the circuits $\mathsf{Gen}_0, \mathsf{Gen}_1$ for $\mathcal{H}_0$ and $\mathcal{H}_1$ that generate the variables $\{(\mathbf{b}_{i,i^*}, \hat{\mathbf{b}}_{i,i^*})\}_{i\neq i^*}$. Specifically,

$$G_0(\mathbf{m}, \hat{\mathbf{m}}, \hat{\alpha}_{i^*}, \{\alpha_i\}_{i\neq i^*}; \{r_{i,i^*}\}_{i\neq i^*}) \to \left(\mathbf{b}_{i,i^*} = (r_{i,i^*} + \alpha_i \hat{\alpha}_{i^*})\mathbf{m}, \hat{\mathbf{b}}_{i,i^*} = -r_{i,i^*}\hat{\mathbf{m}}\right),$$
$$G_1(\mathbf{m}, \hat{\mathbf{m}}, \hat{\alpha}_{i^*}, \{\alpha_i\}_{i\neq i^*}; \{r_{i,i^*}\}_{i\neq i^*}) \to \left(\mathbf{b}_{i,i^*} = r_{i,i^*}\mathbf{m}, \hat{\mathbf{b}}_{i,i^*} = (-r_{i,i^*} + \alpha_i \hat{\alpha}_{i^*})\hat{\mathbf{m}}\right).$$

Let $\mathcal{G}_0, \mathcal{G}_1$ be the distributions corresponding to G_0 and G_1, respectively. Note that other parts of $\mathsf{Gen}_0, \mathsf{Gen}_1$ are identical. By the Averaging Argument for Indistinguishable Distribution, it suffices to prove that for every assignment ρ to all random bits except those for r_{i,i^*}, $i \neq i^*$, we have

$$\mathcal{H}_0|_\rho \approx_{t,mp^{-1}} \mathcal{H}_1|_\rho.$$

In the rest of the proof, we fix an assignment ρ. After fixing ρ, the output bits of $\mathsf{Gen}_0, \mathsf{Gen}_1$ besides those of G_0, G_1 are fixed strings that is independent of the input. Let $t' \in \mathsf{Log}$ be the length of the fixed parts. Thus, by the Reduction Lemma, it suffices to prove that $cG_0 \approx_{t+t',mp^{-1}} \mathcal{G}_1$, where both $\mathcal{G}_0, \mathcal{G}_1$ take random bits for r_{i,i^*}, $i \neq i^*$.

Note that by the Hybrid Argument, and by applying the Reduction Lemma, it suffices to prove that for any $t'' \in \mathsf{Log}$, the following two distributions are $(t + t' + t'', p^{-1})$-indistinguishable:

- $\mathcal{G}'_0$: $(r + \alpha, -r)$;
- $\mathcal{G}'_1$: $(r, -r + \alpha)$.

Both distributions take random bits for $r \leftarrow \mathbb{Z}_p$ (implemented by $\mathsf{Samp}_p(\cdot)$), where $\alpha \in \mathbb{Z}_p$ is a fixed element. Here, the Hybrid Argument is to focus on a fixed $i \neq i^*$, and the Reduction Lemma is applied to remove the identical parts (i.e. the parts for $j \neq i$) and unwrap the group encoding.

Again, by applying the Reduction Lemma, it suffices to prove that for any $t'' \in \mathsf{Log}$, the following two distributions are $(t + t' + t'', p^{-1})$:

- $\mathcal{G}''_0$: $r + \alpha$;
- $\mathcal{G}''_1$: r.

This is because the second coordinate of $\mathcal{G}'_0$ and $\mathcal{G}'_1$ can be computed from the first coordinate by the mapping $u \mapsto \alpha - u$.

Finally, $\mathcal{G}''_0 \simeq_{p^{-1}} \mathcal{G}''_1$ follows from the $\mathbb{Z}_p$ Shift-Invariance Lemma, which implies that $\mathcal{G}''_0$ and $\mathcal{G}''_1$ are (t, p^{-1})-indistinguishable for *any* $t \in \mathsf{Log}$, by the Isomorphism Lemma. This completes the proof. □

Lemma 10.14. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves that* $\mathcal{H}_1 \equiv \mathcal{H}_2$.

This lemma is immediate since the two hybrids only differs on a change of variables.

Lemma 10.15. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} + \mathsf{Cor}_{\mathsf{GroupGen}}(\lambda)$ *proves that* $\mathcal{H}_2 \equiv \mathcal{H}_3$.

This lemma is immediate since the two hybrids can be shown to be identical from the correctness of the group operations.

Lemma 10.16. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} + \mathsf{SXDH}^{t+3t_{\mathsf{Gen}},\varepsilon/2}_{\lambda,\mathsf{GroupGen}(\lambda)}$ *proves that* $\mathcal{H}_3 \approx_{t,\varepsilon} \mathcal{H}_4$*, where* $t_{\mathsf{Gen}} \in \mathsf{Log}$ *is the upper bound of the size of the circuit generating* $\mathcal{H}_3$*,* $\mathcal{H}_4$*.*

Proof. We argue in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} + \mathsf{SXDH}^{t+3t_{\mathsf{Gen}},\varepsilon/2}_{\lambda,\mathsf{GroupGen}(\lambda)}$. First, by the Parallel SXDH Lemma, we have $\mathsf{SXDH}^{t_0,\varepsilon_0}_{\mathcal{G}} \vdash \text{2-}\mathsf{SXDH}^{t_0-2t_{\mathsf{Gen}},2\varepsilon_0}_{\mathcal{G}}$, where $t_{\mathsf{Gen}} \in \mathsf{Log}$ is the upper bound of the size of the SXDH challenge generation circuits $\mathsf{Gen}_0, \mathsf{Gen}_1$ with respect to group $\mathcal{G}$.

Next, observe that $\mathcal{H}_3$ and $\mathcal{H}_4$ can be constructed by the circuit composition $C \circ \mathsf{Gen}^2_b$ for $b \in \{0,1\}$, where vectors of group elements $[\mathbf{a}_{i^*}]_1, [\hat{\mathbf{a}}_{i^*}]_2, [\mathbf{m}]_1, [\hat{\mathbf{m}}]_2$ are generated by Gen^2_b through

$$\begin{aligned}
&\{[u_b]_b, [v_{i,b}]_b, [w_{i,b}]_b\}_{i\in\{1,2\},b\in\{1,2\}} \leftarrow \mathsf{Gen}^2_b,\\
&[\mathbf{m}]_1 = ([v_{1,1}]_1, [v_{2,1}]_1), [\hat{\mathbf{m}}]_2 = ([v_{1,2}]_2, [v_{2,2}]_2).\\
&[\mathbf{a}_{i^*}]_1 = ([w_{1,1}]_1, [w_{2,1}]_1), [\hat{\mathbf{a}}_{i^*}]_2 = ([w_{1,2}]_2, [w_{2,2}]_2),
\end{aligned}$$

and the rest of the variables are generated by some circuit C, identical in both hybrids. Note that by implicitly setting $\alpha_{i^*} = u_1, \hat{\alpha}_{i^*} = u_2$, it is clear that Gen^2_0 provides group elements with distribution identical to $\mathcal{H}_3$. By the Reduction Lemma, we have

$$\text{2-}\mathsf{SXDH}^{t_0-2t_{\mathsf{Gen}},2\varepsilon_0}_{\mathcal{G}} \vdash \mathcal{H}_3 \approx_{t-2t_{\mathsf{Gen}}-t_C,2\varepsilon_0} \mathcal{H}_4,$$

where t_C is the size of C. The lemma immediately follows by observing $t_{\mathsf{Gen}}, t_C \le t_{\mathsf{Gen}}$. □

Lemma 10.17. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} + \mathsf{Cor}_{\mathsf{GroupGen}}(\lambda)$ *proves that* $\mathcal{H}_4 \equiv \mathcal{H}_5$*.*

This lemma is immediate since the two hybrids can be shown to be identical from the correctness of the group operations.

Lemma 10.18. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} + \mathsf{SXDH}^{t,\varepsilon}_{\lambda,\mathsf{GroupGen}(\lambda)}$ *proves that* $\mathcal{H}_5 \approx_{8\cdot 2^{-\lambda}} \mathcal{H}_6$*.*

From the Full-Rank Lemma, we know that for all $\beta^{-1}, \delta^{-1} \in \mathsf{Log}$, we can prove in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ that $\Pr_\delta[(\mathbf{a}_{i^*}, \mathbf{m}) \text{ is not a basis}] \le \delta + \beta + 2p^{-1}$. Therefore, by the Union Bound and Mixture Lemma, $\mathcal{H}_4 \approx_{8p^{-1}} \mathcal{H}_5$. The lemma now follows from the group largeness axiom in SXDH, stating that $p > 2^\lambda$.

Wrapping up. Combining the above lemmas, we know that $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} + \mathsf{SXDH}^{t+3t_{\mathsf{Gen}},1/4t}_{\lambda,\mathsf{GroupGen}\lambda}$ proves the sentence $\mathcal{H}_0 \approx_{t,1/2t+(m+8)\cdot 2^{-\lambda}} \mathcal{H}_6$. Since $t_{\mathsf{Gen}} \in \mathsf{Log}$ is bounded by a fixed polynomial of the setup parameters λ, m, s, by setting $\hat{\lambda}_0[m,t]$ to be the suitable parameters where

$$\forall \lambda > \hat{\lambda}_0[m,t], 1/2t(\lambda) + (m(\lambda)+8)\cdot 2^{-\lambda} < 1/t(\lambda)$$

and set $\lambda'_0[m,s,t]$ to be

$$\lambda'_0[m,s,t] = \max(\lambda_0[t'], \hat{\lambda}_0[m,t]), \quad t'(\lambda) = \max\big(t(\lambda) + 3t_{\mathsf{Gen}}(\lambda,m,s), 4t)\lambda\big),$$

we conclude that $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} + \mathsf{SXDH}_{\lambda_0,\mathsf{GroupGen}}$ proves every sentence $\mathsf{BARGind}_{\lambda'_0}$. □

10.5 BARG with Set Extraction

The BARG construction given in Construction 10.4 can be easily extended to allow polynomially many extraction indices via combining polynomially many parallel instances. The correctness, somewhere extractability, and CRS indistinguishability property can be proved in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ and SXDH by combining Section 10.4 with the Hybrid Argument.

Construction 10.19 (BARG with Set Extraction)**.** Let BARG_1 be the batch argument scheme supporting single point extraction. We construct BARG with set extraction as follows.

- $\mathsf{BARG.TGen}(1^\lambda, 1^m, 1^s, 1^L, \mathcal{S})$: On input the security parameter λ, the number of instance m, the circuit size upper bound s, the extraction set size upper bound L, and a set $\mathcal{S} = \{i_1, \ldots, i_\ell\} \subseteq [m]$ of size $\ell \leq L$, the algorithm does the following.
  - For $j \in [\ell]$, sample $(\mathsf{crs}_j, \mathsf{td}_j) \leftarrow \mathsf{BARG}_1.\mathsf{TGen}(1^\lambda, 1^m, 1^s, i_j)$.
  - For $j \in [L] \setminus [\ell]$, sample $(\mathsf{crs}_j, \mathsf{td}_j) \leftarrow \mathsf{BARG}_1.\mathsf{Gen}(1^\lambda, 1^m, 1^s)$.
  - Output $\mathsf{crs} = (\mathsf{crs}_1, \ldots, \mathsf{crs}_L), \mathsf{td} = (\mathsf{td}_1, \ldots, \mathsf{td}_\ell)$
- $\mathsf{BARG.Gen}(1^\lambda, 1^m, 1^s, 1^L)$ is defined by $\mathsf{BARG.TGen}(1^\lambda, 1^m, 1^s, 1^L, \varnothing)$, dropping the (empty) trapdoor td.
- $\mathsf{BARG.Prove}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), (\mathbf{y}_1, \ldots, \mathbf{y}_m))$ computes $\pi_i = \mathsf{BARG}_1.\mathsf{Prove}(\mathsf{crs}_j, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), (\mathbf{y}_1, \ldots, \mathbf{y}_m))$ for $j \in [L]$, and outputs $\pi = (\pi_1, \ldots, \pi_L)$.
- $\mathsf{BARG.Ver}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi)$ checks $\mathsf{BARG}_1.\mathsf{Ver}(\mathsf{crs}_j, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi)$ for $j \in [L]$, and accepts if all checks accept.
- $\mathsf{BARG.Extract}(\mathsf{crs}, \mathsf{td}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi)$ computes $\mathbf{y}_i = \mathsf{Extract}(\mathsf{crs}_j, \mathsf{td}_J, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi)$ for all $j \in [\ell]$, and outputs $\mathbf{y} = (\mathbf{y}_1, \ldots, \mathbf{y}_j)$.

Lemma 10.20. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves that if* BARG_1 *is a batch argument scheme supporting single point extraction with correctness, somewhere extractability, and CRS indistinguishability properties, then the* BARG *construction in Construction 10.19 is a batch argument scheme supporting set extraction with correctness, somewhere extractability, and CRS indistinguishability properties.*

Proof Sketch. Since the construction in Construction 10.19 is simply L parallel instances of the single-point-extraction BARG_1 construction, the correctness and somewhere extractability properties follow directly from those of BARG_1. The CRS indistinguishability property follows from the Hybrid Argument over the L instances. In particular, for any (ordered) set $|\mathcal{S}| \leq L$, let $\mathcal{S}_\ell$ be the first ℓ elements in $\mathcal{S}$. It is easy to see that the distribution $D_{\mathcal{S}_\ell}$ generated by $\mathsf{BARG.TGen}(1^\lambda, 1^m, 1^s, 1^L, \mathcal{S}_\ell)$ and the distribution $D_{\mathcal{S}_{\ell-1}}$ generated by $\mathsf{BARG.TGen}(1^\lambda, 1^m, 1^s, 1^L, \mathcal{S}_{\ell-1})$ differs only on the ℓ-th BARG_1 crs instance, where crs_ℓ is either generated using the normal generation algorithm $\mathsf{BARG}_1.\mathsf{Gen}$ or the trapdoor generation algorithm $\mathsf{BARG}_1.\mathsf{TGen}(\mathcal{S}[\ell])$. Therefore, by the Reduction Lemma, the parameterized CRS indistinguishability property of BARG_1 (i.e., $\mathsf{BARG}_1\mathsf{ind}^{t,\varepsilon}_{\lambda,m,s}$) implies that $D_{\mathcal{S}_\ell} \approx_{t-t_{\mathsf{Gen}},\varepsilon} D_{\mathcal{S}_{\ell-1}}$, where t_{Gen} is the upper bound of the size of the BARG crs generation circuits. By applying the Hybrid Argument over $\ell \in [|\mathcal{S}|]$, we conclude that for any set $\mathcal{S}$ of size at most L, $D_\varnothing \approx_{t-L\cdot t_{\mathsf{Gen}},L\varepsilon} D_{\mathcal{S}}$. This completes the proof. □

Remark 10.21 (Efficiency of Construction 10.19)**.** The construction in Construction 10.19 has

- CRS size $O(Lm^2) \cdot \mathsf{poly}(\lambda)$,
- Proof size $|C| \cdot O(L) \cdot \mathsf{poly}(\lambda)$,
- Verification time $(|C| + O(nm^2)) \cdot O(L) \cdot \mathsf{poly}(\lambda)$,

11 SNARG Construction and Analysis

In this section, we present a variant of the EF-SNARG construction from Jin, Kalai, Lombardi and Vaikuntanathan [JKLV24], and analyze its security in APC_1 + "$\mathsf{prBPP} = \mathsf{prP}$". We first introduce some tools and formalization in Section 11.1. Then, we present the modified encrypt-hash and BARG construction of JKLV in Section 11.3.

11.1 Definitions and Tools

11.1.1 Circuits

In this section, we will be precise about how our Boolean circuits are expressed. This section is almost verbatim from [JKLV24, Section 3.1].

Circuits. We formally describe such a circuit with s gates via a string $\langle C \rangle$ given by $(g_1, \ldots, g_s)$, where each gate $g_i = (i, j, k, f)$ is a tuple consisting of:

- The output wire name i.
- The two children wires are j and k.
- A boolean function $f : \{0,1\} \times \{0,1\} \to \{0,1\}$.

Moreover, the last m gates $g_{s-m+1}, \ldots, g_s$ correspond to the output of the circuit. To evaluate $C(x)$, we define:

- for $g_1, \ldots, g_n$, set $g_i(x) = x_i$.
- for $g_i = (i, j, k, f)$ where $i > n$, iteratively evaluate $g_i(x) = f(g_j(x), g_k(x))$.
- Output $g_{s-m+1}(x), \ldots, g_s(x)$.

Universal circuits. Consider a family $\mathcal{C}_{s,n,m}$ of size s circuits with n-bit inputs and m-bit outputs. We then define the following explicit universal circuit U for $\mathcal{C}_{s,n,m}$ such that $U(x, \langle C \rangle) = C(x)$. We construct U via the following sequence of subcircuits $\{U_i\}_{i \in [s]}$, where we define $U_i : \{0,1\}^{i-1} \times \mathcal{G} \to \{0,1\}$ (where $\mathcal{G}$ denotes the gate description alphabet) as follows: For every $1 \le j < i$, let w_j^{out} denote the output wire of U_j. This will be the jth input to U_i. U_i will additionally take as input the ith gate of $\langle C \rangle$.

- Suppose the input to U_i is $(w_1, \ldots, w_{i-1}, g = (i, j_0, j_1, f))$.
- For $b \in \{0,1\}, \alpha \in [i-1]$, construct a subcircuit $\Lambda_{b,\alpha}$ which takes the same input $(w_\alpha, g = (i, j, k, f))$ and outputs w_α if $\alpha = j_b$ (and $\perp$ otherwise). Denote the output wire of $\Lambda_{b,\alpha}$ by $\omega_{b,\alpha}$ (and $\perp$ otherwise).

- Let Λ_b take as input $(\omega_{b,1}, \ldots, \omega_{b,i-1})$ and output the single input which is not equal to $\perp$.
- Construct a subcircuit Λ_{out} which takes as input ω_0, ω_1 and $g = (i, j_0, j_1, f)$, and outputs $f(\omega_0, \omega_1)$.
- The output of U_i corresponds to the output of Λ_{out}.

The output wires correspond to the output wires of $U_{s-m+1}, \ldots, U_s$. It is easy to see that the above universal circuit has size and depth $\mathsf{poly}(s)$. Moreover, each U_i has depth at most $O(\log s)$.

Remark 11.1. As noted in JKLV, there are more efficient constructions of universal circuits. Here, we choose a simple explicit construction that is sufficient for the construction and analysis.

11.2 Local assignment generators

Definitions 11.2 and 11.4 are taken nearly verbatim from [JKLV24].

Definition 11.2 (Local assignment generator)**.** Let $C : \{0,1\}^n \to \{0,1\}^m$ denote a Boolean circuit of size s. For parameters ℓ and ε, we say that C has an (ℓ, ε)-local assignment generator if there is an algorithm $\mathsf{LocalGen}$ with the following properties.

- **Syntax:** The input to $\mathsf{LocalGen}$ is a subset of wires $T \subseteq [s]$ of size at most ℓ and its output is an assignment $(\sigma_i)_{i \in T} \in \{0,1\}^T$ to the corresponding wires of C.
- **Local consistency:** for any gate $g_i = (i, j, k, f)$ of C, and any set $T \supseteq \{i, j, k\}$ of size at most ℓ, we have
$$\Pr\left[\ \sigma_i \neq f(\sigma_j, \sigma_k) \ \ : \ \ (\sigma_\alpha)_{\alpha \in T} \leftarrow \mathsf{LocalGen}(T)\ \right] \leq \varepsilon.$$
- **Computational non-signaling:** for any sets $T_0, T_1 \subseteq [s]$ of size at most ℓ, the following distributions are ε-computationally indistinguishable:
$$(\{\sigma_i\}_{i \in U} : \{\sigma_i\}_{i \in T_0} \leftarrow \mathsf{LocalGen}(T_0)) \approx_\varepsilon (\{\sigma_i\}_{i \in U} : \{\sigma_i\}_{i \in T_1} \leftarrow \mathsf{LocalGen}(T_1))$$
where $U = T_0 \cap T_1$.

Definition 11.3 (Accepting Local Assignment Generator)**.** We say that an (ℓ, ε)-local assignment generator $\mathsf{LocalGen}$ for circuit C is *accepting* if
$$\Pr[\sigma \neq 1 : \sigma \leftarrow \mathsf{LocalGen}(\{\mathsf{out}\})] \leq \varepsilon.$$
where out corresponds to the output wire of C.

The following is an equivalent formulation of the local assignment generator that will simplify some of the later proofs. Essentially, the first definition allowed for queries on *sets*, whereas the following definition allows for queries on *ordered tuples*.

Definition 11.4 (Local tuple generator)**.** In the same setting as in Definition 11.2, a (ℓ, ε)-local *tuple* generator $\mathsf{LocalTupGen}$ for a circuit C has the following properties.

- **Syntax:** The input to $\mathsf{LocalTupGen}$ is the description of an ℓ-tuple of wires $(i_1, \ldots, i_\ell) \in ([s] \cup \{\perp\})^\ell$. Its output is an assignment $(\sigma_{i_1}, \sigma_{i_2}, \ldots, \sigma_{i_\ell}) \in (\{0,1\} \cup \{\perp\})^\ell$ to the corresponding wires of C (and $\sigma_{i_k} = \perp$ if $i_k = \perp$).

- **Wire consistency:** For any ℓ-tuple $(i_1, \ldots, i_\ell)$ and any pair j_1, j_2 such that $i_{j_1} = i_{j_2}$, we have

$$\Pr\left[\ \sigma_{i_{j_1}} \neq \sigma_{i_{j_2}} \ : \ (\sigma_{i_j})_{j=1}^{\ell} \leftarrow \mathsf{LocalTupGen}(i_1, \ldots, i_\ell)\ \right] \leq \varepsilon.$$

- **Gate consistency:** For any tuple $(i_1, \ldots, i_\ell)$ and triple (j_1, j_2, j_3), if $(i_{j_1}, i_{j_2}, i_{j_3})$ form a gate C with corresponding function f, we have

$$\Pr\left[\ \sigma_{i_{j_1}} \neq f(\sigma_{i_{j_2}}, \sigma_{i_{j_3}}) \ : \ (\sigma_{i_j})_{j\in[\ell]} \leftarrow \mathsf{LocalTupGen}(i_1, \ldots, i_\ell)\ \right] \leq \varepsilon.$$

- **Computational non-signaling:** For any pair of ℓ-tuples $(i_1, \ldots, i_\ell)$ and $(i'_1, \ldots, i'_\ell)$, let $T = \{j \in [\ell] : (i_j = i'_j) \wedge (i_j \neq \perp)\}$. Then, the following distributions are ε-computationally indistinguishable:

$$\begin{aligned}&(\{\sigma_{i_j}\}_{j\in T} : \{\sigma_{i_j}\}_{j\in[\ell]} \leftarrow \mathsf{LocalTupGen}(i_1, \ldots, i_\ell))\\ &\quad\approx_\varepsilon (\{\sigma_{i'_j}\}_{i\in T} : \{\sigma_{i'_j}\}_{j\in[\ell]} \leftarrow \mathsf{LocalTupGen}(i'_1, \ldots, i'_\ell)).\end{aligned}$$

Definition 11.5 (Accepting Local Tuple Generator)**.** We say that an (ℓ, ε)-local tuple generator $\mathsf{LocalTupGen}$ for circuit C is *accepting* if

$$\Pr[\sigma_1 = 1 : \{\sigma_j\}_{j\in[\ell]} \leftarrow \mathsf{LocalTupGen}((\mathsf{out}, \perp, \perp, \ldots, \perp))] \geq 1 - \varepsilon.$$

where out corresponds to the output wire of C.

Formalization in $\mathsf{APC}_1 + $ "$\mathsf{prBPP} = \mathsf{prP}$". We will now formalize the properties of local assignment and tuple generators in $\mathsf{APC}_1 + $ "$\mathsf{prBPP} = \mathsf{prP}$".

We treat (ℓ, ε)-local assignment (tuple resp.) generator $\mathsf{LocalGen}$ for a circuit C of size s, as a randomized algorithm which takes r bits as input. Hence we formalize it as a PV symbol which takes as input a set (tuple resp.) of size ℓ. We will introduce an additional parameter t to denote the adversary size in the computational non-signaling game.

Definition 11.6 (Local assignment generator in $\mathsf{APC}_1 + $ "$\mathsf{prBPP} = \mathsf{prP}$")**.** Let $\ell, t, s \in \mathsf{Log}$, and $\varepsilon \in [0, 1]$. An (ℓ, ε, t)-local assignment generator for a circuit C is a randomized algorithm $\mathsf{LocalGen}$ of size s with the same syntax as in Definition 11.2 with the following properties:

- **Local consistency:** Let $g_i = (i, j, k, f)$ be a gate of C, and let $T \supseteq \{i, j, k\}$ be a subset of wires of C of size at most ℓ. Define the following game $\mathcal{G}_{g_i,T}$:
  - $\mathcal{C}_1$ is the circuit which simply outputs T.
  - $\mathcal{C}_2$ takes as input a set of wire assignments $\{\sigma_\alpha\}_{\alpha\in T}$. If $\sigma_i \neq f(\sigma_j, \sigma_k)$, output 1. Else, output 0.

  We say $\mathsf{LocalGen}$ satisfies ε-local consistency if for all $g_i = (i, j, k, f)$ and $T \supseteq \{i, j, k\}$ of size at most ℓ, we have that $\mathsf{Adv}_{\mathcal{G}_{g_i,T}}[\mathsf{LocalGen}] \leq \varepsilon$.
- **Computational non-signaling:** Define sets $T_0, T_1 \subseteq [s]$ with size at most ℓ, and let $U = T_0 \cap T_1$. Let $G_{T_0,T_1,b}$ correspond to the circuit which takes as input a seed $\mathsf{sd} \in \{0, 1\}^r$ and does the following:
  - Sample $\{\sigma_i\}_{i\in T_b} \leftarrow \mathsf{LocalGen}(T_b; \mathsf{sd})$.
  - Output $\{\sigma_i\}_{i\in U}$.

Let $\mathcal{H}_b$ be the distribution corresponding to G_b. We say that $\mathsf{LocalGen}$ satisfies (t, ε)-non-signaling if $\mathcal{H}_0 \approx_{t,\varepsilon} \mathcal{H}_1$ for all (T_0, T_1).

Additionally, we say that the local assignment generator is accepting if the following holds. Let $\mathcal{G}_{\mathsf{accept}} = (\mathcal{C}_1, \mathcal{C}_2, 0)$ be the game defined by:

- $\mathcal{C}_1$ outputs the set $\{\mathsf{out}\}$, where out corresponds to the output gate of C.
- $\mathcal{C}_2$ takes as input σ, and outputs 1 if $\sigma = 1$, and 0 otherwise.

We say that $\mathsf{LocalGen}$ is ε-almost accepting if $\mathsf{Adv}_{\mathcal{G}_{\mathsf{accept}}}[\mathsf{LocalGen}] \geq 1 - \varepsilon$.

Definition 11.7 (Local tuple generator in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$")**.** Let $\ell, t, s, u \in \mathsf{Log}$, and $\varepsilon \in [0, 1]$. A (ℓ, ε, t)-local tuple generator for a circuit C of size s is a randomized algorithm $\mathsf{LocalTupGen}$ of size u with the same syntax as in Definition 11.4 with the following properties:

- **Wire consistency:** For any ℓ-tuple $T = (i_1, \ldots, i_\ell)$, and any pair j_1, j_2 such that $i_{j_1} = i_{j_2}$, define the following search game $\mathcal{G}^{\mathsf{wirecon}}_{T,j_1,j_2} = (C_1, C_2, 0)$:
  - C_1 is the circuit which simply outputs the tuple T.
  - C_2 takes as input a wire assignment $\{\sigma_j\}_{j\in[\ell]}$. If $\sigma_{j_1} \neq \sigma_{j_2}$ output 1, and output 0 otherwise.

  We say that $\mathsf{LocalTupGen}$ satisfies (ε-)wire consistency if for all T, j_1, j_2,

  $$\mathsf{Adv}_{\mathcal{G}^{\mathsf{wirecon}}_{T,j_1,j_2}}[\mathsf{LocalTupGen}] \leq \varepsilon.$$

  By definition, C_1 and C_2 have size $O(\ell)$.

- **Gate consistency:** For any ℓ-tuple $T = (i_1, \ldots, i_\ell)$ and triple (j_1, j_2, j_3), if $g_k = (i_{j_1}, i_{j_2}, i_{j_3}, f)$ is a gate in the C, define the following search game $\mathcal{G}^{\mathsf{gatecon}}_{T,(j_1,j_2,j_3)} = (C_1, C_2, 0)$:
  - C_1 is the circuit which simply outputs the tuple T.
  - C_2 is the circuit which takes as input a wire assignment $\{\sigma_j\}_{j\in[\ell]}$. If $g_{i_{j_1}} = (i_{j_1}, i_{j_2}, i_{j_3}, f)$, output 1 if $\sigma_{j_1} \neq f(\sigma_{j_2}, \sigma_{j_3})$, and 0 otherwise.

  We say that $\mathsf{LocalTupGen}$ satisfies ε-gate consistency if for all ℓ-tuples T and triples (j_1, j_2, j_3) such that $i_{j_1}, i_{j_2}, i_{j_3}$ form a gate,

  $$\mathsf{Adv}_{\mathcal{G}^{\mathsf{gatecon}}_{T,(j_1,j_2,j_3)}}[\mathsf{LocalTupGen}] \leq \varepsilon.$$

  By definition, C_1 and C_2 have size $O(\ell)$.

- **Computational non-signaling:** Consider two ℓ-tuples $T_0 = \{i_1^{(0)}, \ldots, i_\ell^{(0)}\}$ and $T_1 = \{i_1^{(1)}, \ldots, i_\ell^{(1)}\}$. Let $U = \{j \in [\ell] : (i_j^{(0)} = i_j^{(1)}) \wedge (i_j^{(0)} \neq \perp)\}$. Let $C_{T_0,T_1,b}$ correspond to the circuit which takes as input a seed $\mathsf{sd} \in \{0,1\}^r$ and does the following:
  - Call $\{\sigma_{i_j}\}_{j\in[\ell]} \leftarrow \mathsf{LocalTupGen}(T_b; \mathsf{sd})$.
  - Output $\{\sigma_{i_j}\}_{j\in U}$.

  Let $\mathcal{H}_b$ be the distribution corresponding to $C_{T_0,T_1,b}$. We say that $\mathsf{LocalTupGen}$ satisfies (t, ε)-non-signaling if $\mathcal{H}_0 \approx_{t,\varepsilon} \mathcal{H}_1$ for all (T_0, T_1).

Additionally, we say that the local tuple generator is accepting if the following holds. Let $\mathcal{G}_{\mathsf{accept}} = (\mathcal{C}_1, \mathcal{C}_2, 0)$ be the game defined by:

- $\mathcal{C}_1$ outputs the tuple $(\mathsf{out}, \bot, \bot, \ldots, \bot)$, where out corresponds to the output gate of C.
- $\mathcal{C}_2$ takes as input $\{\sigma_i\}_{i\in[\ell]}$, and outputs 1 if $\sigma_1 = 1$, and 0 otherwise.

We say that LocalTupGen is ε-almost accepting if $\mathsf{Adv}_{\mathcal{G}_{\mathsf{accept}}}[\mathsf{LocalTupGen}] \geq 1 - \varepsilon$.

Here, we give a simpler proof than in [JKLV24] (although with slightly worse parameters) that a local tuple generator can be used to derive a local assignment generator.

Claim 11.8. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves the following sentence. Let $t, \ell, s \in \mathsf{Log}$. Let $c \in \mathbb{N}$ be a large constant. If a circuit C of size s has a $(\ell, \varepsilon, ct^c)$-local tuple generator, then it also has a $(\ell/2, O(\ell \cdot \varepsilon), t)$-local assignment generator. Furthermore, if it has an* accepting *$(\ell, \varepsilon, ct^c)$-local tuple generator, then it also has an accepting $(\ell/2, O(\ell \cdot \varepsilon), t)$-local assignment generator.*

Proof. Suppose LocalTupGen is an $(\ell, \varepsilon, ct^c)$-local tuple generator. We define LocalGen as follows.

> $\mathsf{LocalGen}(T)$: On input a set T of size at most $\ell/2$, do the following.
>
> - Let $\{i_1, \ldots, i_{|T|}\}$ be the indices of the wires in T in lexicographic order, and let $\widehat{T} = (i_1, \ldots, i_{|T|}, \bot, \ldots, \bot)$ be an ℓ-tuple.
> - Call $\{\sigma_{i_j}\}_{j\in[\ell]} \leftarrow \mathsf{LocalTupGen}(\widehat{T})$.
> - Output $\{\sigma_{i_j}\}_{j\in[|T|]}$.

It is clear that by wire consistency and gate consistency of LocalTupGen, we obtain that LocalGen also satisfies ε-local consistency. Moreover, it is easy to see that if LocalTupGen is ε-almost accepting, so is LocalGen. Moreover, one can prove this in PV_1. It therefore suffices to argue non-signaling.

Fix two sets T_0 and T_1 of sizes $\ell/2$ with indices $\{i_1^{(0)}, \ldots, i_{\ell/2}^{(0)}\}$ and $\{i_1^{(1)}, \ldots, i_{\ell/2}^{(1)}\}$ (ordered in lexicographic order). Let $U = T_0 \cap T_1$. Let $\widehat{T_b}$ denote the ℓ-tuple $\{i_1^{(b)}, \ldots, i_{\ell/2}^{(b)}, \bot, \ldots, \bot\}$.

We now argue in a few hybrids. We describe each hybrid by the corresponding circuits.

$\boxed{\mathcal{H}_1}$ Compute $\{\sigma_i\}_{i\in T_0} \leftarrow \mathsf{LocalGen}(T_0)$. Output $\{\sigma_i\}_{i\in U}$.

$\boxed{\mathcal{H}_2}$ Compute $\{\sigma_{i_j^{(0)}}\}_{j\in[\ell]} \leftarrow \mathsf{LocalTupGen}(\widehat{T_0})$. Let $\rho \subseteq [\ell/2]$ correspond to the indices j such that $i_j^{(0)} \in U$. Output $\{\sigma_{i_j^{(0)}}\}_{j\in\rho}$.

$\boxed{\mathcal{H}_3}$ Consider the tuple $Z = \{z_1, \ldots, z_\ell\}$, where $\{z_1, \ldots, z_\ell\} = \{i_1^{(0)}, \ldots, i_{\ell/2}^{(0)}, i_1^{(1)}, \ldots, i_{\ell/2}^{(1)}\}$. Let $\{\sigma_{z_j}\}_{j\in[\ell]} \leftarrow \mathsf{LocalTupGen}(Z)$. Let $\rho \subseteq [\ell/2]$ correspond to the indices $i_j^{(0)}$ such that $i_j^{(0)} \in U$. Output $\{\sigma_{i_j^{(0)}}\}_{j\in\rho}$.

$\boxed{\mathcal{H}_4}$ Consider the tuple $Z = \{z_1, \ldots, z_\ell\}$, generated as in the previous hybrid. Let $\{\sigma_{z_j}\}_{j\in[\ell]} \leftarrow \mathsf{LocalTupGen}(Z)$. For every $p \in U$,

- Find $i_j^{(0)}, i_k^{(1)} = p$.
- If $\sigma_{i_j^{(0)}} \neq \sigma_{i_k^{(1)}}$, output $\perp$.

Output $\{\sigma_{z_j}\}_{j\in\rho}$.

$\boxed{\mathcal{H}_5}$ Consider the tuple $Z = \{z_1, \ldots, z_\ell\}$, generated as in the previous hybrid. For every $p \in U$,

- Find $i_j^{(0)}, i_k^{(1)} = p$.
- If $\sigma_{i_j^{(0)}} \neq \sigma_{i_k^{(1)}}$, output $\perp$.

Let $\{\sigma_{z_j}\}_{j\in[\ell]} \leftarrow \mathsf{LocalTupGen}(Z)$. Let $\rho' \subseteq [\ell] \setminus [\ell/2]$ correspond to the indices j such that $z_j \in U$. Output $\{\sigma_{z_j}\}_{j\in\rho'}$.

$\boxed{\mathcal{H}_6}$ Consider the tuple $Z = \{z_1, \ldots, z_\ell\}$, generated as in the previous hybrid. Let $\{\sigma_{z_j}\}_{j\in[\ell]} \leftarrow \mathsf{LocalTupGen}(Z)$. Let $\rho' \subseteq [\ell] \setminus [\ell/2]$ correspond to the indices j such that $z_j \in U$. Output $\{\sigma_{z_j}\}_{j\in\rho'}$.

$\boxed{\mathcal{H}_7}$ Consider the tuple $Z = \{z_1, \ldots, z_\ell\}$, where $\{z_1, \ldots, z_\ell\} = \{i_1^{(1)}, \ldots, i_{\ell/2}^{(1)}, i_1^{(1)}, \ldots, i_{\ell/2}^{(1)}\}$. Let $\{\sigma_{z_j}\}_{j\in[\ell]} \leftarrow \mathsf{LocalTupGen}(Z)$. Let $\rho' \subseteq [\ell] \setminus [\ell/2]$ correspond to the indices z_j such that $z_j \in U$. Output $\{\sigma_{z_j}\}_{j\in\rho'}$.

$\boxed{\mathcal{H}_8}$ Consider the tuple Z generated as in the previous hybrid. Let $\{\sigma_{z_j}\}_{j\in[\ell]} \leftarrow \mathsf{LocalTupGen}(Z)$. Let $\rho \subseteq [\ell/2]$ correspond to the indices j such that $z_j \in U$. Output $\{\sigma_{z_j}\}_{j\in\rho}$.

$\boxed{\mathcal{H}_9}$ Consider the tuple $Z = \widehat{T_1}$. Let $\{\sigma_{z_j}\}_{j\in[\ell]} \leftarrow \mathsf{LocalTupGen}(Z)$. Let $\rho \subseteq [\ell/2]$ correspond to the indices j such that $z_j \in U$. Output $\{\sigma_{z_j}\}_{j\in\rho}$.

$\boxed{\mathcal{H}_{10}}$ Compute $\{\sigma_i\}_{i\in T_1} \leftarrow \mathsf{LocalGen}(T_1)$. Output $\{\sigma_i\}_{i\in U}$.

Lemma 11.9. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves that* $\mathcal{H}_1 \equiv \mathcal{H}_2$.

Proof. By definition, it is easy to show that the distributions in $\mathcal{H}_1$ and $\mathcal{H}_2$ are identical in PV_1. Therefore, we can show in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ that $\mathcal{H}_1 \equiv \mathcal{H}_2$. □

Lemma 11.10. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves that* $\mathcal{H}_2$ *and* $\mathcal{H}_3$ *are* (t, ε)*-indistinguishable.*

Proof. Let $S = \{j \in [\ell] : (i_j = z_j)\}$. Clearly, by definition, $S = \{1, \ldots, \ell/2\}$. Let $\mathcal{D}_0$ denote the distribution corresponding to:

- Let $\{\sigma_j\}_{j\in[\ell]} \leftarrow \mathsf{LocalTupGen}(\widehat{T_b})$
- Output $\{\sigma_j\}_{j\in S}$.

Let $\mathcal{D}_1$ denote the distribution corresponding to:

- Let $\{\sigma_j\}_{j\in[\ell]} \leftarrow \mathsf{LocalTupGen}(Z)$.
- Output $\{\sigma_j\}_{j\in S}$.

By the computational-NS property of $\mathsf{LocalTupGen}$, $\mathcal{D}_0$ and $\mathcal{D}_1$ are (ct^c, ε)-indistinguishable. Moreover, let $\mathcal{A}$ denote the circuit that takes as input $\{\sigma_i\}_{i\in S}$ and outputs $\{\sigma_i\}_{i\in\rho}$. By the Reduction Lemma, we have that $\mathcal{A}\circ\mathcal{D}_0$ and $\mathcal{A}\circ\mathcal{D}_1$ are also (t,ε)-indistinguishable (assuming $t + |\mathcal{A}| \leq ct^c$). Finally, it is easy to see that $\mathcal{A}\circ\mathcal{D}_0$ is identical to $\mathcal{H}_2$, and $\mathcal{A}\circ\mathcal{D}_1$ is identical to $\mathcal{H}_3$. This completes the proof via the Hybrid Argument. □

Lemma 11.11. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves that* $\mathcal{H}_3$ *and* $\mathcal{H}_4$ *are* $(t, \ell\cdot\varepsilon)$*-indistinguishable.*

Proof. Let $Z = \{z_1,\ldots,z_\ell\}$, where $\{z_1,\ldots,z_\ell\} = \{i_1^{(0)},\ldots,i_{\ell/2}^{(0)}, i_1^{(1)},\ldots,i_{\ell/2}^{(1)}\}$ as defined as in $\mathcal{H}_3$ and $U = T_0 \cap T_1$. For each $w \in U$, consider $i_j^{(0)}, i_k^{(1)} = w$. Let $\mathcal{G}_w = \mathcal{G}^{\mathsf{wirecon}}_{Z,i_j^{(0)},i_k^{(1)}}$ denote the wire consistency game corresponding to the tuple Z. Recall that $\mathsf{Adv}_{\mathcal{G}_w}[\mathsf{LocalTupGen}] \leq \varepsilon$ by definition of $\mathsf{LocalTupGen}$. We write $\mathcal{G}_w = (C_1^{(w)}, C_2^{(w)}, 0)$. By definition, for all $w_1, w_2 \in U$, $C_1^{(w)}$ is identical to the circuit C_1 that simply outputs Z. Let $C_2 = \bigvee_{w\in U} C_2^{(w)}$, and define $\mathcal{G} = (C_1, C_2, 0)$. Then, by applying the Game Union Bound Lemma, it is easy to see that $\mathsf{Adv}_{\mathcal{G}}(\mathsf{LocalTupGen}) \leq \ell\cdot\varepsilon$.

Moreover, note that C_2 is PV equivalent to the circuit C_2' with the following functionality:

- Take as input a wire assignment $\{\sigma_j\}_{j\in\ell}$. For every $p \in U$,
  - Find $i_j^{(0)}, i_k^{(1)} = p$.
  - If $\sigma_{i_j^{(0)}} \neq \sigma_{i_k^{(1)}}$, output $\bot$.

Let $T_{\mathcal{G},\mathcal{A}}$ be the circuit as defined in Definition 5.2. Let B be the circuit that always outputs $\bot$, let $\mathcal{D}$ be the corresponding distribution.

Then, we can rewrite $\mathcal{H}_4$ as $\mathcal{H}_3|_{\neg T_{\mathcal{G},\mathcal{A}}} + \mathcal{B}|_{T_{\mathcal{G},\mathcal{A}}}$. By invoking the Mixture Lemma, we have that the two distributions are $(t, \ell\cdot\varepsilon)$-indistinguishable. □

Lemma 11.12. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves that* $\mathcal{H}_4 \equiv \mathcal{H}_5$.

Proof. It suffices to show that the two distributions are identical. Indeed, if the output is not $\bot$, we have that for every $p \in U$, if $i_j^{(0)}, i_k^{(1)} = p$, then $\sigma_{i_j^{(0)}} = \sigma_{i_k^{(1)}}$. Therefore, the two outputs are $\{\sigma_{z_j}\}_{j\in\rho}$ and $\{\sigma_{z_j}\}_{j\in\rho'}$ are indeed identical (where ρ and ρ' are as defined in $\mathcal{H}_4$ and $\mathcal{H}_5$ respectively). Hence, the two distributions are indeed equivalent. □

Lemma 11.13. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves that* $\mathcal{H}_5$ *and* $\mathcal{H}_6$ *are* $(t, \ell\cdot\varepsilon)$*-indistinguishable.*

This proof is near identical to the proof of Lemma 11.11, so we will omit it.

Lemma 11.14. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves that* $\mathcal{H}_6$ *and* $\mathcal{H}_7$ *are* (t,ε)*-indistinguishable.*

The proof is near identical to the proof of Lemma 11.10, so we will omit it.

Lemma 11.15. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves that* $\mathcal{H}_7$ *and* $\mathcal{H}_8$ *are* $(t, O(\ell\cdot\varepsilon))$*-indistinguishable.*

This proof follows from a similar argument as was done in Lemmas 11.11 to 11.13.

Lemma 11.16. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves that* $\mathcal{H}_8$ *and* $\mathcal{H}_9$ *are* (t,ε)*-indistinguishable.*

The proof is near identical to the proof of Lemma 11.10, so we will omit it.

Lemma 11.17. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves that* $\mathcal{H}_9 \equiv \mathcal{H}_{10}$.

Same as Lemma 11.9, the two hybrids can be shown identical in PV_1

Therefore, via the Hybrid Argument, we have that $\mathcal{H}_1$ and $\mathcal{H}_{10}$ are $(t, O(\ell \cdot \varepsilon))$-indistinguishable, therefore completing the proof. □

11.3 Encrypt-Hash-and-BARG and Construction

In this work, we will give a slightly simplified construction[29] of the JKLV SNARG. We note that much of the writing and algorithm is identical to JKLV.

Definition 11.18 (Augmented circuit)**.** Let C be a circuit with k wires, and let D be a circuit that takes as input k bits, and outputs nothing (one can think of this as a circuit which always outputs 0). Let $\mathsf{Aug}(C, D)$ denote the following circuit: on input x,

- Compute $C(x)$, and let τ be the corresponding wire assignment.
- Evaluate $D(\tau)$.
- Output $C(x)$.

We will use this notation throughout this section.

Construction 11.19 (Simplified JKLV SNARG)**.** The following construction is a simplification from JKLV.

Fix an NP language $\mathcal{L}$ with witness for x of length $m(|x|)$. In the following, n will be the input length parameter, and $L = \mathsf{poly}(\ell)$, where ℓ will be a bound on the length of the propositional proof of non-membership (we choose this exact polynomial in Section 11.5). We will use the following ingredients:

- A secret-key leveled FHE scheme $\mathsf{FHE}.(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval}, \mathsf{GateEval})$ whose correctness, malicious gate correctness and malicious evaluation correctness can be proven in PV_1. Moreover, suppose there exists λ_0^{FHE} such $\mathsf{FHESecure}_{\lambda_0^{\mathsf{FHE}}}$ holds for $\mathsf{FHE}.*$.
- A BARG scheme $\mathsf{BARG}.(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver}, \mathsf{TGen}, \mathsf{Ext})$ with set extraction whose correctness and somewhere extractability can be proven in PV_1. Moreover, suppose there exists $\lambda_0^{\mathsf{BARG}}$ such that $\mathsf{BARGind}_{\lambda_0^{\mathsf{BARG}}}$ holds for $\mathsf{BARG}.*$.
- A SEH scheme $(\mathsf{SEH.Gen}, \mathsf{SEH.TGen}, \mathsf{SEH.Hash}, \mathsf{SEH.Open}, \mathsf{SEH.Ver}, \mathsf{SEH.Ext})$ whose opening completeness and extraction correctness can be proven in PV_1. Moreover, suppose there exists λ_0^{SEH} such that every sentence in $\mathsf{SEHSecure}_{\lambda_0^{\mathsf{SEH}}}$ holds when implemented by $\mathsf{SEH}.*$.

We now define the algorithm TGen, Gen, Prove and Ver. Note that while the SNARG itself does not need a trapdoor mode, we define TGen for the security analysis.

- **Trapdoor CRS algorithm.** $\mathsf{TGen}(1^\lambda, (C, 1^n, 1^m), 1^L, T_{\mathsf{SEH}}, S_{\mathsf{BARG}}, \mathsf{aux}; \mathsf{sd})$ operates as follows: On input a security parameter 1^λ, an input length 1^n, a parameter 1^L, a tuple $T_{\mathsf{SEH}} \subseteq [2^\lambda]$ of size loc, a set $S_{\mathsf{BARG}} \subseteq [2^\lambda]$ of size at most loc (where $\mathsf{loc} = \mathsf{poly}(\lambda, \log n, \log m, \log s)$, and the explicit value will be chosen in the proof), a string $\mathsf{aux} \in \{0,1\}^L$, and randomness $\mathsf{sd} = (\mathsf{sd}_{\mathsf{FHE}}, \mathsf{sd}_{\mathsf{ct}}, \mathsf{sd}_{\mathsf{SEH}}, \mathsf{sd}_{\mathsf{BARG}})$

[29] In essence, the CRS in JKLV grows with the size and computation depth of any machine that outputs the propositional proof of non-membership. In this work, we simplify their construction to allow for a simpler analysis, albeit having a CRS that grows with the total length of the propositional proof of non-membership. We believe their version can also be formalized in our theory.

1. Set the FHE depth parameter $d_{\mathsf{FHE}} = \mathsf{poly}(L, |C|)$ for some large polynomial.
2. **Generate FHE key:** Sample FHE keys $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{FHE.Gen}(1^\lambda, 1^{d_{\mathsf{FHE}}}; \mathsf{sd}_{\mathsf{FHE}})$.
3. **Generate FHE ciphertext:** Sample a ciphertext $\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{sk}, \mathsf{aux}; \mathsf{sd}_{\mathsf{ct}})$.
4. **Generate SEH keys:** Let $T_{\mathsf{SEH}} = \{i_1, \ldots, i_{\mathsf{loc}}\}$. Let N^* be as defined in Step (5) of the prover algorithm.
   - Parse $\mathsf{sd}_{\mathsf{SEH}} = \{\mathsf{sd}_j\}_{j \in \mathsf{loc}}$.
   - For $j \in [\mathsf{loc}]$, sample a hash key $(\mathsf{hk}_j, \mathsf{td}_j) \leftarrow \mathsf{SEH.TGen}(1^\lambda, i_j; \mathsf{sd}_j)$.
   - Set $\mathsf{hk} = \{\mathsf{hk}_1, \ldots, \mathsf{hk}_{\mathsf{loc}}\}$ and $\mathsf{td}_{\mathsf{SEH}} = \{\mathsf{td}_1, \ldots, \mathsf{td}_{\mathsf{loc}}\}$.
   - In the remaining algorithms, we use the shorthand $\mathsf{SEH.Hash}(\mathsf{hk}, \cdot), \mathsf{SEH.Open}(\mathsf{hk}, \cdot), \mathsf{SEH.Ver}(\mathsf{hk}, \cdot)$ to denote the algorithms which carry out the respective algorithms loc times with respect to each of $\mathsf{hk}_1, \ldots, \mathsf{hk}_{\mathsf{loc}}$. We use the shorthand $\mathsf{SEH.Ext}(\mathsf{hk}, \mathsf{td}, \cdot)$ to denote running the Ext algorithm loc times with respect to each of $(\mathsf{hk}_1, \mathsf{td}_1), \ldots, (\mathsf{hk}_{\mathsf{loc}}, \mathsf{td}_{\mathsf{loc}})$.
5. **Generate BARG crs:** Sample $(\mathsf{crs}_{\mathsf{BARG}}, \mathsf{td}_{\mathsf{BARG}}) \leftarrow \mathsf{BARG.TGen}(1^\lambda, 1^{k^*}, 1^{s^*}, 1^{\mathsf{loc}}, S_{\mathsf{BARG}}; \mathsf{sd}_{\mathsf{BARG}})$, where we define k^* and s^* in Step (6) of the prover algorithm.
6. Output $(\mathsf{crs} = (\mathsf{hk}, \mathsf{ek}, \mathsf{ct}, \mathsf{crs}_{\mathsf{BARG}}), \mathsf{td} = (\mathsf{sk}, \mathsf{td}_{\mathsf{SEH}}, \mathsf{td}_{\mathsf{BARG}}))$.

- **CRS algorithm.** $\mathsf{Gen}(1^\lambda, 1^n, 1^\ell)$ operates as follows:
  1. Let C denote the relation circuit for the language $\mathcal{L}$ on inputs of length n, and witnesses of length m.
  2. Let $L = \mathsf{poly}(\ell)$, where the exact poly will be chosen via Theorem 11.23.
  3. Set $T_{\mathsf{SEH}} = (0, \ldots, 0)$, $S_{\mathsf{BARG}} = \varnothing$ and $\mathsf{aux} = 1^L$.
  4. Sample $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{TGen}(1^\lambda, (C, 1^n, 1^m), 1^L, T_{\mathsf{SEH}}, S_{\mathsf{BARG}}, \mathsf{aux})$, and output crs.
- **Prove algorithm.** $\mathcal{P}(\mathsf{crs}, C, x, w)$ operates as follows:
  1. Parse $(\mathsf{hk}, \mathsf{ek}, \mathsf{ct}, \mathsf{crs}_{\mathsf{BARG}}) \leftarrow \mathsf{crs}$.
  2. Compute the transcript of $C_x(w)$, and let $\tau = \tau_x(w)$ denote the wire assignment for this computation.
  3. Run the circuit $\mathsf{FHE.Eval}_{\mathsf{ek}}(U_\tau, \cdot)$ on ct gate-by-gate, where $U(\tau, \langle D \rangle)$ denotes the universal circuit that interprets $\langle D \rangle$ as a circuit and runs it on τ (see Section 11.1.1 for details on the description of U). Denote this circuit by $\Gamma_{\mathsf{ek},\mathsf{ct}}$.
  4. Let $\widehat{C_{x,\mathsf{ek},\mathsf{ct}}} = \mathsf{Aug}(C_x, \Gamma_{\mathsf{ek},\mathsf{ct}})$ denote the computation done in steps (2) and (3) (i.e. the gate-by-gate FHE evaluation), and let $\tilde{\tau}$ denote the wire assignment.
  5. Let N^* be the length $\tilde{\tau}$. Let $v_{\mathcal{P}} = \mathsf{SEH.Hash}(\mathsf{hk}, \tilde{\tau})$.
  6. Compute a BARG proof that essentially proves that every gate of $\widehat{C_{x,\mathsf{ek},\mathsf{ct}}}$ was computed correctly. Formally, let $R_{\mathsf{hk},v_{\mathcal{P}}}$ be the following relation circuit:
     - **Instance:** Description of some gate $g_i = (i, j_0, j_1, f)$, along with $\mathsf{aux}_i = (\beta_0, \beta_1)$ where β_b is the hardcoded value for wire j_b if any (e.g. if gate g_i takes as input any bit of $x, \mathsf{ek}, \mathsf{ct}$), and $\perp$ otherwise.
     - **Witness:** Wire values w_i, w_{j_0}, w_{j_1} and openings $\rho_i, \rho_{j_0}, \rho_{j_1}$.

     Then, $R_{\mathsf{hk}}((g_i, \mathsf{aux}_i), (w_i, w_{j_0}, w_{j_1}, \rho_i, \rho_{j_0}, \rho_{j_1}))$ does the following:

- Parse $g_i = (i, j_0, j_1, f)$.
- Verify that $\mathsf{SEH.Ver}(\mathsf{hk}, v_{\mathcal{P}}, i, w_i, \rho_i) = 1$.
- Parse $\mathsf{aux}_i = (\beta_0, \beta_1)$.
- For $b \in \{0, 1\}$, if $\beta_b \neq \perp$, check that $j_b = \beta_b$.
- Else, verify that $\mathsf{SEH.Ver}(\mathsf{hk}, v_{\mathcal{P}}, j_b, w_{j_b}, \rho_{j_b})$.
- Verify that $f(w_{j_0}, w_{j_1}) = w_i$.
- Output 1 if all of the above checks pass, and output 0 otherwise.

Let k^* denote the number of gates in $\widehat{C_{x,\mathsf{ek},\mathsf{ct}}}$. Let s^* be the size of the circuit $R_{\mathsf{hk},v}$. It is easy to see that $s^* = \mathsf{poly}(\lambda, \log |C|)$ since it simply verifies SEH openings. Note that one can derive the batch of instances $\{(g_i, \mathsf{aux}_i)\}_{i\in[k^*]}$ from the description of $\widehat{C_{x,\mathsf{ek},\mathsf{ct}}}$ efficiently, and the prover can compute the corresponding $\{(w_i, w_{j_0}, w_{j_1}, \rho_i, \rho_{j_0}, \rho_{j_1})\}_{i\in[k^*]}$ efficiently.
Let

$$\pi_{\mathsf{BARG}} \leftarrow \mathsf{BARG.Prove}(\mathsf{crs}, R_{\mathsf{hk},v_{\mathcal{P}}}, \{(g_i, \mathsf{aux}_i)\}_{i\in[k^*]}, \{(w_i, w_{j_0}, w_{j_1}, \rho_i, \rho_{j_0}, \rho_{j_1})\}_{i\in[k^*]})$$

be the corresponding proof.

7. Let ρ_{out} be the opening to the output wire of $\widehat{C_{x,\mathsf{ek},\mathsf{ct}}}$.
8. Output $(v_{\mathcal{P}}, \pi_{\mathsf{BARG}}, \rho_{\mathsf{out}})$.

- **The Verifier Algorithm.** $\mathsf{SNARG}.\mathcal{V}(\mathsf{crs}, C, x, \pi)$ does the following:

1. Parse $(\mathsf{hk}, \mathsf{ek}, \mathsf{ct}, \mathsf{crs}_{\mathsf{BARG}}) \leftarrow \mathsf{crs}$ and $(v_{\mathcal{P}}, \pi_{\mathsf{BARG}}, \rho_{\mathsf{out}}) \leftarrow \pi$.
2. Check that ρ_{out} corresponds to an opening to the bit 1.
3. Parse $\widehat{C_{x,\mathsf{ek},\mathsf{ct}}}$ as the batch of instances $\{(g_i, \mathsf{aux}_i)\}_{i\in[k^*]}$, and let $R = R_{\mathsf{hk},v_{\mathcal{P}}}$.
4. $\mathsf{BARG.Ver}(\mathsf{crs}_{\mathsf{BARG}}, R, \{(g_i, \mathsf{aux}_i)\}_{i\in[k^*]}, \pi_{\mathsf{BARG}}) = 1$.
5. Reject if any of the above checks fail, accept otherwise.

We will denote by $\mathsf{SNARG} = \mathsf{SNARG}.(\mathsf{Gen}, \mathcal{P}, \mathcal{V})$, and we will use $\mathsf{SNARG.TGen}$ only in the security proof.

Remark 11.20 (Changes from JKLV)**.** We point out the main changes made in this algorithm relative to the original EF-SNARG construction of [JKLV24] in order to simplify our analysis. The most prominent change is that we will instantiate our BARG via SXDH rather than LWE. Moreover, the JKLV SNARG construction has a shorter CRS in the case where the propositional proof of non-membership has a small time-bounded Kolmogorov complexity. However, one would need to rely on unleveled FHE in order to achieve this efficiency, which we do not know from plain LWE. Therefore, we omit this optimization. Finally, our scheme results in a verifier whose runtime is $\mathsf{poly}(\lambda, n, \ell)$, after which we rely on Remark 3.6 to improve this dependence to be $n \cdot \mathsf{poly}(\lambda, \log \ell)$. The construction of JKLV does this optimization inline.

Completeness and Efficiency. Completeness of the scheme follows directly from the BARG completeness and SEH opening correctness. Recall that the proof π consists of a hash $v_{\mathcal{P}}$ and opening ρ_{out} which both have size at most $\mathsf{poly}(\lambda, \mathsf{loc}, \log n) = \mathsf{poly}(\lambda, \log n, \log \ell)$, and a BARG proof $|\pi_{\mathsf{BARG}}| \leq \mathsf{poly}(\lambda, s^*, \log k^*) = \mathsf{poly}(\lambda, \mathsf{loc}, \log n, \log \ell) = \mathsf{poly}(\lambda, \log n)$. Therefore, $|\pi| = \mathsf{poly}(\lambda, \log n, \log \ell)$, which is indeed succinct. The CRS on the other hand has size $\mathsf{poly}(\lambda, n, \ell)$.

While the verifier itself is inefficient (takes time $\mathsf{poly}(\lambda, n, \ell)$ rather than logarithmic in ℓ), one can generically boost this efficiency, as described in Remark 3.6.

Security. We now prove that the above construction is secure, and show our main result in Corollary 11.22.

Theorem 11.21 (JKLV Security Theorem). *There exists a function $\lambda_0[p_1, p_2, p_3]$ from* PV *functions $p_1, p_2, p_3 :$* Log $\to$ Log *such that* $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ *proves every sentence in* $\mathsf{EF\text{-}SNARG}_{\lambda_0}$*, when* EF-SNARG.$*$ *is implemented by* $\mathsf{SNARG}.(\mathsf{Gen}, \mathcal{P}, \mathcal{V})$.

By combining the JKLV Security Theorem and the security lifting lemma (see Lemma 5.6), it immediately follows that there are SNARGs for all NP, under our assumptions.

Corollary 11.22. *Suppose that* $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ *is sound (i.e. the non-logical axioms hold in the standard model) and Assumption 3 is true, for every $L \in$* NP*, there exists a sound and complete* SNARG *scheme for L.*

We prove the JKLV Security Theorem in two parts. First, we show that if an adversary $\mathcal{A}$ is able to create accepting proofs for some $x \in \{0,1\}^n$ with probability at least ε, then one can construct an *accepting* local assignment generator for any augmented circuit $\mathsf{Aug}(C_x, E_x)$ with description size at most L. Then, we will construct an E_x for which there is *no* local assignment generator which is also *accepting* via the following theorem.

Theorem 11.23. $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$" *proves the following sentence. There exists a constant $c \in \mathbb{N}$, and* PV*-definable polynomial h, p, p_1, p_2 such that the following holds. Let $s, s', n, \ell \in$* Log*. Let $t = h(n, \ell, s)$. Consider some $x \in \{0,1\}^n$ and circuit C of size s' such that there exists an* EF *proof Φ of size ℓ proving that $\forall w, C_x(w) = 0$. Then, there exists a circuit E_x of size $p(\ell)$ with the following property: For any $(p_1(\log n, \log \ell), \varepsilon, ct^c)$-local assignment generator of size s' for $\mathsf{Aug}(C_x, E_x)$, it holds that*

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{accept}}}[\mathsf{LocalGen}] \leq \varepsilon \cdot p_2(n, \ell)$$

where $\mathcal{G}^{\mathsf{accept}}$ is as defined in Definition 11.6.

We will prove this in Section 11.5.

11.4 Proof of JKLV Security Theorem (Theorem 11.21)

In this section, we prove the JKLV Security Theorem in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ assuming Theorem 11.23 is true. We delay the proof of Theorem 11.23 to Section 11.5.

Following Sections 8 to 10, there exist an FHE scheme, a BARG scheme and a somewhere extractable hash scheme satisfying correctness provable in PV_1, along with security $\mathsf{FHESecure}_{\lambda_0^{\mathsf{FHE}}}$, $\mathsf{BARGind}_{\lambda_0^{\mathsf{BARG}}}$, $\mathsf{SEHSecure}_{\lambda_0^{\mathsf{SEH}}}$ provable in $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$.

Fix an NP language $\mathcal{L}$, and suppose that the relation circuit $C(x, w)$ has size $s = q_1(|x|)$ and witnesses of length $m = q_2(|x|)$ for PV-definable polynomials q_1 and q_2. Fix a PV-definable polynomial t^*. Let n, L be PV-definable polynomials, and let $s = q_1(n)$ and $m = q_2(n)$.

Choosing parameters. Let t be the PV definable polynomial defined by $t = t^* + n + \ell + s + m$. Fix a large $c \in \mathbb{N}$ be a large constant. Looking forward, we will pick c to be large enough so that ct^c is much larger than any of the reductions constructed in the proof. It is easy to verify that all the reductions in the proof have size at most $\mathsf{poly}(t^*, n, \ell)$.

Set $t_{\mathsf{FHE}}, t_{\mathsf{BARG}}, t_{\mathsf{SEH}} = ct^c$. Now, we pick $\lambda_0 \in \mathbb{N}$ such that for any $\lambda > \lambda_0$, it is bigger than the following quantities:

- $\lambda_0^{\mathsf{FHE}}[d_{\mathsf{FHE}}, t_{\mathsf{FHE}}, L]$ (see Definition 8.4), where d_{FHE} is as defined in the SNARG algorithm.
- $\lambda_0^{\mathsf{BARG}}[t_{\mathsf{BARG}}, k^*, s^*]$ (see Definition 10.3) where k^* and s^* are as defined in Step (6) of the prover algorithm.
- $\lambda_0^{\mathsf{SEH}}[t_{\mathsf{SEH}}, N^*]$, where N^* is as defined in Step (5) of the prover algorithm.

For convenience, we will write $\varepsilon_{\mathsf{FHE}} = 1/t_{\mathsf{FHE}}$, $\varepsilon_{\mathsf{BARG}} = 1/t_{\mathsf{BARG}}$ and $\varepsilon_{\mathsf{SEH}} = 1/t_{\mathsf{SEH}}$.

Suppose there exists some $\lambda > \lambda_0[n, \ell, t^*]$ such that $\mathsf{EF\text{-}SNARG}_{\lambda,n(\lambda),\ell(\lambda)}^{t^*(\lambda),1/t^*(\lambda)}$ is false. Let $\mathcal{A}$ be the adversary of size $t^*(\lambda)$ such that

$$\mathsf{Adv}_{\mathcal{G}_x}[\mathcal{A}] \geq \frac{1}{t^*(\lambda)},$$

where $x \in \{0,1\}^n$ is some string with a proof $\tau \in \{0,1\}^\ell$ that $\forall w, C_x(w) = 0$. For convenience, we write $\varepsilon^* = \frac{1}{t^*(\lambda)}$.

For readability, from here on, we will suppress λ, n, ℓ, L, C from inputs of the algorithms when it is clear from context.

Lemma 11.24 (Indistinguishability of CRS, based on Lemma 4.10 of [JKLV24])**.** APC_1 + "prBPP = prP"+FHESecure+SEHSecure+BARGind *proves the following sentence. Let* $\mathcal{D}_{T,S,\mathsf{aux}}$ *denote the distribution corresponding to* $\mathsf{TGen}(T, S, \mathsf{aux})$*. For all tuples* T, T' *of size* loc*, sets* S, S' *of size at most* loc *and* $\mathsf{aux}, \mathsf{aux}' \in \{0,1\}^L$*, we have that* $\mathcal{D}_{T,S,\mathsf{aux}}$ *and* $\mathcal{D}_{T',S',\mathsf{aux}'}$ *are* (t_1, ε_1)*-indistinguishable, where* $\varepsilon_1 = \mathsf{loc} \cdot \varepsilon_{\mathsf{SEH}} + \varepsilon_{\mathsf{BARG}} + \varepsilon_{\mathsf{FHE}}$ *and* $t_1 = c_1 t^{c_1}$ *for some constant* $c_1 \in \mathbb{N}$ *greater than 1.*

Proof. By construction, one can show in PV that the CRS output by $\mathsf{TGen}(T_{\mathsf{SEH}}, S_{\mathsf{BARG}}, \mathsf{aux}; \mathsf{sd})$ can be written as a direct product of the distributions corresponding to the following circuit:

- For $j \in [\mathsf{loc}]$, let $\mathsf{Gen}_{\mathsf{SEH},i_j}$ output $\mathsf{SEH.Gen}(i_j; \mathsf{sd}_j)$. Let $\mathcal{H}_{\mathsf{SEH},i_j}$ denote the corresponding distribution.
- Let $\mathsf{Gen}_{\mathsf{FHE},\mathsf{ct}}$ be the circuit which takes as input $\mathsf{sd}_{\mathsf{FHE}}, \mathsf{sd}_{\mathsf{ct}}$:
  - Sample $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{FHE.Gen}(\mathsf{sd}_{\mathsf{FHE}})$.
  - Sample $\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{sk}, \mathsf{aux}; \mathsf{sd}_{\mathsf{ct}})$.
  - Output $(\mathsf{ek}, \mathsf{ct})$.

  Let $\mathcal{H}_{\mathsf{FHE},\mathsf{aux}}$ denote the corresponding distribution.
- Let $\mathsf{Gen}_{\mathsf{BARG}}$ be the circuit which takes as input a set S and $\mathsf{sd}_{\mathsf{BARG}}$ and outputs $\mathsf{BARG.TGen}(S; \mathsf{sd}_{\mathsf{BARG}})$. Let $\mathcal{H}_{\mathsf{BARG},S}$ denote the corresponding distributions.

In other words, one can prove in PV that $\mathcal{D}_{T,S,\mathsf{aux}} \equiv \mathcal{H}_{i_1} \times \mathcal{H}_{i_2} \times \ldots \mathcal{H}_{i_{\mathsf{loc}}} \times \mathcal{H}_{\mathsf{FHE},\mathsf{aux}} \times \mathcal{H}_{\mathsf{BARG},S}$.

Recall that by FHE security, we have that for all $\mathsf{aux}, \mathsf{aux}'$, $\mathcal{H}_{\mathsf{FHE},\mathsf{aux}}$ and $\mathcal{H}_{\mathsf{FHE},\mathsf{aux}'}$ are $(t_{\mathsf{FHE}}, \varepsilon_{\mathsf{FHE}})$-indistinguishable. Suppose $T = (i_1, \ldots, i_{\mathsf{loc}})$ and $T' = (i'_1, \ldots, i'_{\mathsf{loc}})$. By SEH security, we have that each $\mathcal{H}_{\mathsf{SEH},i_j}$ and $\mathcal{H}_{\mathsf{SEH},i'_j}$ are $(t_{\mathsf{SEH}}, \varepsilon_{\mathsf{SEH}})$-indistinguishable. By BARG crs-indistinguishability, we have that $\mathcal{H}_{\mathsf{BARG},S}$ and $\mathcal{H}_{\mathsf{BARG},S'}$ are $(t_{\mathsf{BARG}}, \varepsilon_{\mathsf{BARG}})$-indistinguishable. Therefore, by applying the Product Hybrid Lemma, we have that $\mathcal{D}_{T,S,\mathsf{aux}}$ and $\mathcal{D}_{T',S',\mathsf{aux}'}$ are $(c_1 t^{c_1}, \varepsilon_1)$-indistinguishable, where $\varepsilon_1 = \mathsf{loc} \cdot \varepsilon_{\mathsf{SEH}} + \varepsilon_{\mathsf{BARG}} + \varepsilon_{\mathsf{FHE}}$, and we choose $c_1 \in \mathbb{N}$ such that $c_1 t^{c_1} + s_1 \leq ct^c$, where $s_1 = \mathsf{poly}(\lambda, n, L)$ is the total size of the circuit generating $\mathcal{D}_{T,S,\mathsf{aux}}$. By choosing t to be large enough, we can choose c_1 such that this equation holds. □

Let $\mathcal{G}^{\mathsf{SNARG}}_{T,S,\mathsf{aux}} = (C_1, C_2, 0)$ denote the security game defined as follows:

- $C_1(\mathsf{sd})$ runs $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{TGen}(T, S, \mathsf{aux}; \mathsf{sd})$ and outputs $\mathsf{chall} = \mathsf{crs}$ and $\mathsf{state} = \mathsf{crs}$.
- $C_2(\pi, \mathsf{crs})$ outputs $\mathsf{Ver}(\mathsf{crs}, x, \pi)$.

Note that the soundness game $\mathcal{G}_x$ of SNARG is equivalent to $\mathcal{G}^{\mathsf{SNARG}}_{\varnothing,\varnothing,0^L}$ We can thus rewrite the condition $\mathsf{Adv}_{\mathcal{G}_x}[\mathcal{A}] \geq \varepsilon^*$ as $\mathsf{Adv}_{\mathcal{G}^{\mathsf{SNARG}}_{\varnothing,\varnothing,0^L}}[\mathcal{A}] \geq \varepsilon^*$.

We now analyze the advantage of $\mathcal{A}$ against games with modified inputs.

Lemma 11.25. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} + \mathsf{FHESecure} + \mathsf{SEHSecure} + \mathsf{BARGind}$ *proves that for all sets* S, T *of size at most* loc *and* $\mathsf{aux} \in \{0,1\}^L$,

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{SNARG}}_{T,S,\mathsf{aux}}} \geq \varepsilon^* - \varepsilon_1$$

where $\varepsilon_1 := \mathsf{loc} \cdot \varepsilon_{\mathsf{SEH}} + \varepsilon_{\mathsf{BARG}} + \varepsilon_{\mathsf{FHE}}$*, as defined in Lemma 11.24.*

Proof. From Lemma 11.24, we have that the outputs of C_1 in $\mathcal{G}^{\mathsf{SNARG}}_{T,S,\mathsf{aux}}$ and $\mathcal{G}^{\mathsf{SNARG}}_{\varnothing,\varnothing,0^L}$ are $(c_1 t^{c_1}, \varepsilon_1)$-indistinguishable. By choosing t to be large enough, one can ensure that $|C_2| + |\mathcal{A}| \leq c_1 t^{c_1}$. Therefore, we get the result by invoking the Game Composition Lemma. □

Let $\langle E_x \rangle \in \{0,1\}^L$ be the description of a circuit E_x (following the syntax of Section 11.1.1) that takes as input a wire assignment τ for C_x, and outputs a bit b. Let D_x denote the circuit that computes C_x, along with the gates of E_x (i.e. the output of D_x is the output of C_x, but it contains additional gates that correspond to the gates of E_x). We now choose $\mathsf{aux}^* = \langle E_x \rangle$. By Lemma 11.25, for any subsets T, S, $\mathsf{Adv}_{\mathcal{G}^{\mathsf{SNARG}}_{T,S,\mathsf{aux}^*}}[\mathcal{A}] \geq \varepsilon - \varepsilon_1$.

Hardcoding the FHE randomness. Define $\mathcal{G}^{\mathsf{SNARG}}_{T,S,\mathsf{aux}^*,\mathsf{sd}^*_{\mathsf{FHE}},\mathsf{sd}^*_{\mathsf{ct}}} = (C_1, C_2, 0)$ as the game $\mathcal{G}^{\mathsf{SNARG}}_{T,S,\mathsf{aux}^*}$ hardcoding the FHE randomness $(\mathsf{sd}^*_{\mathsf{FHE}}, \mathsf{sd}^*_{\mathsf{ct}})$. The game is formally defined as follows:

- $C_1(\mathsf{sd}')$ first calls $(\mathsf{crs} = (\mathsf{hk}, \mathsf{ek}, \mathsf{ct}, \mathsf{crs}_{\mathsf{BARG}}), \mathsf{td} = (\mathsf{sk}, \mathsf{td}_{\mathsf{SEH}}, \mathsf{td}_{\mathsf{BARG}})) \leftarrow \mathsf{TGen}(T, S, \mathsf{aux}^*; (\mathsf{sd}^*_{\mathsf{FHE}}, \mathsf{sd}^*_{\mathsf{ct}}, \mathsf{sd}'))$. Output $\mathsf{chall}, \mathsf{state} = \mathsf{crs}$.
- $C_2(\pi, \mathsf{crs}^*)$ outputs $\mathsf{Ver}(\mathsf{crs}^*, x, \pi)$.

Lemma 11.26. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} + \mathsf{FHESecure} + \mathsf{SEHSecure} + \mathsf{BARGind}$ *proves that for all sets* S, T *of size at most* loc*, there exist* $(\mathsf{sd}^*_{\mathsf{FHE}}, \mathsf{sd}^*_{\mathsf{ct}})$ *such that*

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{SNARG}}_{T,S,\mathsf{sd}^*_{\mathsf{FHE}},\mathsf{sd}^*_{\mathsf{ct}}}}[\mathcal{A}] \geq \varepsilon_2,$$

where $\varepsilon_2 := \varepsilon - \varepsilon_1$.

The lemma follows by applying the Averaging Argument over Lemma 11.25.

Fix TGen^* to be the algorithm that runs TGen but always uses $\mathsf{aux}^*, \mathsf{sd}^*_{\mathsf{FHE}}, \mathsf{sd}^*_{\mathsf{ct}}$ (and random $\mathsf{sd}_{\mathsf{SEH}}, \mathsf{sd}_{\mathsf{BARG}}$). Let $(\mathsf{sk}^*, \mathsf{ek}^*) \leftarrow \mathsf{FHE.Gen}(\mathsf{sd}^*_{\mathsf{FHE}})$ and $\mathsf{ct}^* \leftarrow \mathsf{FHE.Enc}(\mathsf{sk}^*, \mathsf{aux}^*; \mathsf{sd}^*_{\mathsf{ct}})$.

Constructing local tuple and local assignment generators. We now use $\mathcal{A}$ to construct local assignment generators. We will do this in a few steps.

- In Lemma 11.27, we show that there exists a local tuple generator for $\widehat{C_{x,\mathsf{ek}^*,\mathsf{ct}^*}} = \mathsf{Aug}(C_x, \Gamma_{\mathsf{ek}^*,\mathsf{ct}^*})$ (as described in Step (4).)
- In Lemma 11.28, we show that there exists a local assignment generator for $\mathsf{Aug}(C_x, U(E_x, \cdot))$.
- In Lemma 11.29, we show that there exists a local assignment generator for $\mathsf{Aug}(C_x, E_x)$. We will then use Theorem 11.23 to reach a contradiction.

Lemma 11.27 (Based on [JKLV24, Lemma 4.11]). $\mathsf{APC}_1[\mathsf{BPP}, \mathsf{LWE}, \mathsf{SXDH}]$ *proves that there exists a* $(\ell_1, \varepsilon_3, c_3 t^{c_3})$*-accepting local tuple generator for* $\widehat{C_{x,\mathsf{ek}^*,\mathsf{ct}^*}}$ *(as described in Step (4).), where* $\ell_1 = \mathsf{loc}, \varepsilon_3 = \max(\varepsilon', \mathsf{loc} \cdot \varepsilon_{\mathsf{SEH}})$*, where* $\varepsilon' = \frac{1}{p(\lambda,n,\ell)}$ *for some* PV *definable (and tunable) polynomial* p*.*

Proof. Let $Q = \mathsf{poly}(\lambda, n, \ell, t^*)$, where the exact value will be determined in the following proof. We first construct the local-tuple generator.

$\mathsf{LocalTupGen}(T; \{\mathsf{sd}_{1,i}\}_{i\in[Q]}, \{\mathsf{sd}_{2,i}\}_{i\in[Q]})$:

1. For $i \in [Q]$:
   - Obtain $(\mathsf{crs}_i, \mathsf{td}_i) \leftarrow \mathsf{TGen}^*(T_{\mathsf{SEH}} = T, S_{\mathsf{BARG}} = \varnothing; \mathsf{sd}_{1,i})$, where TGen^* is the algorithm $\mathsf{SNARG.TGen}$ hardcoding the input aux^*, and seeds $\mathsf{sd}^*_{\mathsf{FHE}}, \mathsf{sd}^*_{\mathsf{ct}}$.
   - Call $\pi_i \leftarrow \mathcal{A}(\mathsf{crs}; \mathsf{sd}_{2,i})$.
2. If for all π_i, $\mathsf{SNARG}.\mathcal{V}(\mathsf{crs}_i, C, x, \pi_i) = 0$, output $\bot$.
3. Else, let $i \in [Q]$ be such that $\mathsf{SNARG}.\mathcal{V}(\mathsf{crs}_i, C, x, \pi_i) = 1$, and let $\pi = \pi_i$, $\mathsf{crs} = \mathsf{crs}_i$ and $\mathsf{td} = \mathsf{td}_i$.
4. Parse $\pi = (v_{\mathcal{P}}, \pi_{\mathsf{BARG}}, \rho_{\mathsf{out}})$ and $\mathsf{td} = (\mathsf{sk}^*, \mathsf{td}_{\mathsf{SEH}}, \mathsf{td}_{\mathsf{BARG}})$.
5. Let $\{\sigma_j\}_{j\in[\mathsf{loc}]} = \mathsf{SEH.Ext}(\mathsf{hk}, \mathsf{td}_{\mathsf{SEH}}, v_{\mathcal{P}})$.
6. Output $(\sigma_1, \ldots, \sigma_{\ell_1})$.

We now prove the properties of the local tuple generator.

Computational Non-Signaling. We will show this via the key indistinguishability property of SEH. Consider two tuples T_0 and T_1, and let $U = \{j \in [\ell_1] : (i_j^{(0)} = i_j^{(1)}) \wedge (i_j \neq \bot)\}$. As argued in Lemma 11.24, we can rewrite crs distribution output by $\mathsf{TGen}^*(T_b, \varnothing; \mathsf{sd}_1)$ as the product:

$$\mathcal{H}_{i_1^{(b)}} \times \mathcal{H}_{i_2^{(b)}} \times \ldots \mathcal{H}_{i_{\mathsf{loc}}^{(b)}} \times (\mathsf{sk}^*, \mathsf{ct}^*) \times \mathcal{H}_{\mathsf{BARG},S},$$

where we use $(\mathsf{sk}^*, \mathsf{ct}^*)$ as shorthand for the circuit that always outputs $(\mathsf{sk}^*, \mathsf{ct}^*)$. Now, consider the circuit $\mathcal{C}_{T_0,T_1,b}$ (defined relative to $\mathsf{LocalTupGen}$ as defined in Definition 11.7), and let $\mathcal{D}_b$ be the corresponding distributions. Let $U = \{j \in [\mathsf{loc}] : (i_j^{(0)} = i_j^{(1)}) \wedge (i_j \neq \bot)\}$. By definition of $\mathsf{LocalTupGen}$ and $\mathcal{C}_{T_0,T_1,b}$, it is easy to see that $\mathcal{C}_{T_0,T_1,b}$ is PV-equivalent to the following distribution:

$\mathcal{C}_{T_0,T_1,T_b}(\{\mathsf{sd}_{1,i}\}_{i\in[Q]}, \{\mathsf{sd}_{2,i}\}_{i\in[Q]})$

- For $i \in [Q]$:
  - Parse $\mathsf{sd}_{1,i} = (\mathsf{sd}_{\mathsf{SEH}}, \mathsf{sd}_{\mathsf{BARG}})$, and $\mathsf{sd}_{\mathsf{SEH}} = \{\mathsf{sd}_i\}_{i\in[\mathsf{loc}]}$.
  - For $j \notin U$, sample $\mathsf{hk}_j \leftarrow \mathcal{H}_{i_j^{(b)}}$.
  - For $j \in U$, sample $(\mathsf{hk}_j, \mathsf{td}_j) \leftarrow \mathsf{SEH.TGen}(i_j^{(0)}; \mathsf{sd}_j)$.
  - Sample $\mathsf{crs}_{\mathsf{BARG}} \leftarrow \mathsf{BARG.Gen}(\mathsf{sd}_{\mathsf{BARG}})$.
  - Set $\mathsf{crs}_i = (\mathsf{hk}, \mathsf{ek}^*, \mathsf{ct}^*, \mathsf{crs}_{\mathsf{BARG}})$ and $\mathsf{td}_i = \{\mathsf{sk}^*, \{\mathsf{td}_j\}_{j\in U}\}$.
  - Call $\pi_i \leftarrow \mathcal{A}(\mathsf{crs}_i)$.
- If for all π_i, $\mathsf{SNARG}.\mathcal{V}(\mathsf{crs}_i, C, x, \pi_i) = 0$, output $\perp$.
- Else, let $i \in [Q]$ be such that $\mathsf{SNARG}.\mathcal{V}(\mathsf{crs}_i, C, x, \pi_i) = 1$, and let $\pi = \pi_i$, $\mathsf{crs} = \mathsf{crs}_i$ and $\mathsf{td} = \mathsf{td}_i$.
- Parse $\pi = (v_{\mathcal{P}}, \pi_{\mathsf{BARG}}, \rho_{\mathsf{out}})$, and parse $v_{\mathcal{P}} = (v_1, \ldots, v_{\mathsf{loc}})$ and $\mathsf{td} = (\mathsf{sk}^*, \{\mathsf{td}_j\}_{j\in U})$.
- For $j \in U$, let $\sigma_j = \mathsf{SEH.Ext}(\mathsf{hk}_j, \mathsf{td}_j, v_j)$.
- Output $\{\sigma_j\}_{j\in U}$.

Let $s_{\mathsf{SEH}} \le \mathsf{poly}(\lambda, n, \ell)$ be the size of each $\mathcal{H}_{i_j}^{(b)}$. Recall that

$$\prod_{i\in[Q]} \left(\prod_{j\notin U} \mathcal{H}_{i_j^{(0)}} \right) \text{ and } \prod_{i\in[Q]} \left(\prod_{j\notin U} \mathcal{H}_{i_j^{(b)}} \right)$$

$(t_{\mathsf{SEH}} - Q \cdot \mathsf{loc} \cdot s_{\mathsf{SEH}}, Q \cdot \mathsf{loc} \cdot \varepsilon_{\mathsf{SEH}})$-indistinguishable by invoking the Product Hybrid Lemma. By ensuring we chose $ct^c \gg C$, we can ensure that ct^c is much larger than $\mathsf{LocalTupGen}'$, by the Reduction Lemma, we have that $\mathcal{C}_{T_0,T_1,0}$ and $\mathcal{C}_{T_0,T_1,1}$ indeed describe $(c_3 t^{c_3}, C \cdot \mathsf{loc} \cdot \varepsilon_{\mathsf{SEH}})$-indistinguishable distributions for some $c_3 \in \mathbb{N}$.

Wire consistency. Consider an ℓ_1-tuple $T = (i_1, \ldots, i_{\ell_1})$, and any pair j_1, j_2 such that $i_{j_1} = i_{j_2}$. Consider $\mathcal{G}^{\mathsf{wirecon}}_{T,j_1,j_2}$ as defined in Definition 11.7. Let g_k be an arbitrary gate in the computation which touches the wire w corresponding to i_{j_1}.

$\boxed{\mathcal{H}_1}$ Output $\mathsf{LocalTupGen}(T)$.

$\boxed{\mathcal{H}_2}$ Let $\mathsf{LocalTupGen}_1(T)$ be identical to $\mathsf{LocalTupGen}$, except that each crs_i is instead sampled as $\mathsf{TGen}^*(T, \{k\}; \mathsf{sd}_{1,i})$.

The two hybrids are $(t_{\mathsf{BARG}} - s, Q \cdot \varepsilon_{\mathsf{BARG}})$-indistinguishable, where $s = \mathsf{poly}(\lambda, n, \ell)$ bounds the sizes of $\mathsf{LocalTupGen}$ and $\mathsf{LocalTupGen}'$. The indistinguishability follows from combining the BARG CRS indistinguishability, the Reduction Lemma, and the Product Hybrid Lemma.

We now claim that if $\mathsf{LocalTupGen}_1(T)$ does not output $\bot$ (which it only does if for all i, π_i is an invalid proof), then the output satisfies wire consistency for gate g_k.

In fact, it suffices to argue in PV_1 that $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{TGen}^*(T, \{k\}; \mathsf{sd}_1)$ and the corresponding $\mathsf{LocalTupGen}'$ outputs $\{\sigma_j\}_{j\in[\ell_1]}$,

$$\left(\ \mathsf{SNARG}.\mathcal{V}(\mathsf{crs}, x, \pi) = 1 \ \right) \to (\sigma_{j_1} = \sigma_{j_2}).$$

We argue in PV as follows:

1. Parse $\mathsf{crs} = (\mathsf{hk}, \mathsf{ek}^*, \mathsf{ct}^*, \mathsf{crs}_{\mathsf{BARG}})$ and $\pi = (v_{\mathcal{P}}, \pi_{\mathsf{BARG}}, \rho_{\mathsf{out}})$, and let $R = R_{\mathsf{hk}, v_{\mathcal{P}}}$.
2. Parse $\widehat{C_{x,\mathsf{ek},\mathsf{ct}}}$ as the list $G = \{g_i, \mathsf{aux}_i\}_{i\in[k^*]}$.
3. $\mathsf{SNARG}.\mathcal{V}(\mathsf{crs}, x, \pi) = 1$ implies that $\mathsf{BARG.Ver}(\mathsf{crs}_{\mathsf{BARG}}, R, G, \pi_{\mathsf{BARG}}) = 1$.
4. Let $\mathsf{crs}_{\mathsf{BARG}} \leftarrow \mathsf{TGen}^*(T, \{k\}; \mathsf{sd}_1)$. Compute

   $$(b_1, b_2, b_3, \rho_1, \rho_2, \rho_3) \leftarrow \mathsf{BARG.Extract}(\mathsf{crs}_{\mathsf{BARG}}, \mathsf{td}_{\mathsf{BARG}}, R, G, \pi_{\mathsf{BARG}}).$$

   By correctness of the extraction, we have that

   $$R((g_k, \mathsf{aux}_k), (b_1, b_2, b_3, \rho_1, \rho_2, \rho_3)) = 1.$$

   In other words,

   - b_1, b_2, b_3 is consistent with aux_k,
   - for $j \in \{1, 2, 3\}$ (which are not fixed by aux_k), ρ_j is an opening to bit b_j at location corresponding to wire w_{i_j},
   - $f_i(b_1, b_2) = b_3$.
5. Suppose b_α and ρ_α correspond to wire w of g_k.
6. Parse $\rho_\alpha = (\rho_1, \ldots, \rho_{\mathsf{loc}})$.
7. For $\gamma \in \{1, 2\}$, since $(\mathsf{hk}_{j_\gamma}, \mathsf{td}_{j_\gamma}) \leftarrow \mathsf{SEH.Hash}(i_{j_\gamma}; \mathsf{sd}_{j_\gamma})$, we have that by the extraction correctness,

   $$\mathsf{SEH.Ver}(\mathsf{hk}_{j_\gamma}, v_{j_\gamma}, b_\alpha, i_{j_\gamma}, \rho_{j_\gamma}) = 1 \to \mathsf{SEH.Ext}(v_{j_\gamma}, \mathsf{td}_{j_\gamma}) = b_\alpha.$$

8. Therefore, we have that $\mathsf{SEH.Ext}(v_{j_0}, \mathsf{td}_{j_0}) = \mathsf{SEH.Ext}(v_{j_1}, \mathsf{td}_{j_1})$.
9. By definition of $\mathsf{LocalTupGen}'$, $\sigma_{j_\gamma} = \mathsf{SEH.Ext}(v_{j_\gamma}, \mathsf{td}_{j_\gamma})$. Therefore, we have that $\sigma_{j_0} = \sigma_{j_1}$.

For notational convenience, let $\mathcal{G}$ denote $\mathcal{G}^{\mathsf{SNARG}}_{T,S,\mathsf{sd}^*_{\mathsf{FHE}},\mathsf{sd}^*_{\mathsf{ct}}}$. Following the notation in Lemma 7.22, let $\mathcal{G}^{\vee Q} = (\mathcal{C}_1^{\otimes Q}, \mathcal{C}_2^{\vee Q}, 0)$ denote the Q-fold repetition of the game $\mathcal{G}$, and let $\mathcal{A}^{\otimes Q}$ denote the adversary that simulates $\mathcal{A}$ in parallel Q times. By instantiating the Error Reduction Lemma by picking $\beta = \varepsilon/2$, we have that

$$\mathsf{Adv}_{\mathcal{G}^{\vee Q}}[\mathcal{A}^{\otimes Q}] \leq 1 - (1 - \varepsilon/2)^Q.$$

For convenience, we write $\varepsilon' = (1 - \varepsilon/2)^Q$.

Now, it is easy to prove in PV_1 that if $\mathsf{LocalTupGen}_1(T; \{\mathsf{sd}_{1,i}, \mathsf{sd}_{2,i}\})$ does not output $\bot$, then

$$\mathcal{C}_2^{\mathsf{repeat}} \circ \mathcal{A}^{\mathsf{repeat}}(\{\mathsf{sd}_{2,i}\}_{i\in[C]}) \circ \mathcal{C}_1^{\mathsf{repeat}}(\{\mathsf{sd}_{1,i}\}_{i\in[C]}) = 1.$$

Therefore, by the Game Reduction Lemma, we have that:

$$\mathsf{Adv}_{\overline{\mathcal{G}^{\mathsf{wirecon}}_{T,j_1,j_2}}}[\mathsf{LocalTupGen}_1] \geq \mathsf{Adv}_{\mathcal{G}^{\mathsf{repeat}}}[\mathcal{A}^{\mathsf{repeat}}] \geq 1 - \varepsilon'.$$

By invoking the Adversary Indistinguishability Lemma, we have that

$$\mathsf{Adv}_{\overline{\mathcal{G}^{\mathsf{wirecon}}_{T,j_1,j_2}}}[\mathsf{LocalTupGen}] \geq 1 - \varepsilon' - Q \cdot \varepsilon_{\mathsf{BARG}}.$$

By applying the Game Complement Lemma, we have that

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{wirecon}}_{T,j_1,j_2}}[\mathsf{LocalTupGen}'] \leq \varepsilon' + Q \cdot \varepsilon_{\mathsf{BARG}}.$$

Finally, since we set $\varepsilon_{\mathsf{BARG}} = 1/t_{\mathsf{BARG}}$ to be small enough, the advantage $\varepsilon' + Q \cdot \varepsilon_{\mathsf{BARG}} = (1 - \varepsilon/2)^Q + Q \cdot \varepsilon_{\mathsf{BARG}}$ can be tuned to be smaller than $1/p(\lambda, m\ell)$ for some appropriately large polynomial p.

Gate consistency. The following argument is very similar to the case of wire-consistency. For any ℓ_1-tuple $T = (i_1, \ldots, i_{\ell_1})$ and triple (j_1, j_2, j_3), if $g_k = (i_{j_1}, i_{j_2}, i_{j_3}, f)$ is a gate in the C, define the following search game $\mathcal{G}^{\mathsf{gatecon}}_{T,(j_1,j_2,j_3)}$. Consider $\mathsf{LocalTupGen}'$, where the crs is instead sampled as $\mathsf{TGen}^*(T, \{g_k\}; \mathsf{sd}_1)$.

Now, we argue in PV_1 that $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{TGen}^*(T, \{g_k\}; \mathsf{sd}_1)$ and the corresponding $\mathsf{LocalTupGen}'$ outputs $\{\sigma_j\}_{j \in [\ell_1]}$,

$$\left(\ \mathsf{SNARG}.\mathcal{V}(\mathsf{crs}, x, \pi) = 1) \ \right) \rightarrow (\sigma_{j_3} = f(\sigma_{j_1}, \sigma_{j_2})).$$

1. Argue Steps (1)-(4) identically to the case of wire-consistency.
2. For $\gamma \in \{1, 2, 3\}$, since $(\mathsf{hk}_{j_\gamma}, \mathsf{td}_{j_\gamma}) \leftarrow \mathsf{SEH.Gen}(i_{j_\gamma}; \mathsf{sd}_{j_\gamma})$, we have that by the extraction correctness,
$$\mathsf{SEH.Ver}(\mathsf{hk}_{j_\gamma}, v_{j_\gamma}, b_\gamma, i_{j_\gamma}, \rho_{j_\gamma}) = 1 \rightarrow \mathsf{SEH.Ext}(v_{j_\gamma}, \mathsf{td}_{j_\gamma}) = b_\gamma.$$
3. Moreover, by correctness of BARG extraction, we have that b_1, b_2, b_3 satisfy $b_3 = f(b_1, b_2)$.
4. Therefore, by substitution, we have that $\sigma_{j_3} = f(\sigma_{j_1}, \sigma_{j_2})$, completing the proof.

Therefore, by a near identical argument as in the case of wire consistency, we have that

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{gatecon}}_{T,(j_1,j_2,j_3)}}[\mathsf{LocalTupGen}] \leq \varepsilon' + Q \cdot \varepsilon_{\mathsf{BARG}}.$$

Accepting. Let $T = (\mathsf{out}, \bot, \bot, \ldots, \bot)$. Now, we argue in PV_1 that $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{TGen}^*(T, \{g_k\}; \mathsf{sd}_1)$ and the corresponding $\mathsf{LocalTupGen}$ outputs $\{\sigma_j\}_{j \in [\ell_1]}$,

$$\left(\ \mathsf{SNARG}.\mathcal{V}(\mathsf{crs}, x, \pi) = 1) \ \right) \rightarrow (\sigma_1 = 1).$$

We argue this as follows:

- Recall that if $\mathsf{SNARG}.\mathcal{V}(\mathsf{crs}, x, \pi) = 1$, then there exists an opening ρ_{out} such that $\mathsf{SEH.Ver}(\mathsf{hk}, v_{\mathcal{P}}, \mathsf{out}, \rho_{\mathsf{out}}, 1) = 1$.
- Parse $v_{\mathcal{P}} = (v_1, \ldots, v_{\mathsf{loc}})$.

- Recall that $(\mathsf{hk}_1, \mathsf{td}_1) \leftarrow \mathsf{SEH.TGen}(\mathsf{out})$.
- Therefore, by extraction correctness, we have that

$$\mathsf{SEH.Ver}(\mathsf{hk}_1, v_1, 1, i_1, \rho_1) = 1 \rightarrow \mathsf{SEH.Ext}(v_1, \mathsf{td}_1) = 1.$$

In particular, this shows that if $\mathsf{LocalTupGen}$ does not output $\bot$, then π_i corresponds to some accepting SNARG proof, and hence $\sigma_{\mathsf{out}} = 1$.

Therefore, by an identical argument as in the case of wire consistency, we have that

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{gatecon}}_{T,(j_1,j_2,j_3)}}[\mathsf{LocalTupGen}] \leq \varepsilon'.$$

We have that $\mathsf{LocalTupGen}$ is a $(\mathsf{loc}, \max(\varepsilon' + Q \cdot \varepsilon_{\mathsf{BARG}}, \mathsf{loc} \cdot \varepsilon_{\mathsf{SEH}}), c_3 t^{c_3})$-tuple generator. We choose Q such that $\varepsilon' = (1 - \varepsilon/2)^Q \leq \frac{1}{\mathsf{poly}(\lambda,n,\ell)}$ for some poly. □

Let $\beta = \mathsf{poly}(\lambda)$ be the size of an $\mathsf{FHE.GateEval}(\mathsf{ek}_i, \cdot)$ circuit which takes as input two ciphertexts and outputs a new ciphertext.

Lemma 11.28 (Based on [JKLV24, Lemma 4.12])**.** APC_1 + "$\mathsf{prBPP} = \mathsf{prP}$" *proves that there exists a* $(\ell_2, \varepsilon_4, c_4 t^{c_4})$ *local assignment generator for* $\mathsf{Aug}(C_x, U(E_x, \cdot))$*, where* $\ell_2 = \ell_1/2\beta, \varepsilon_4 = O(\varepsilon_3 \cdot \ell_2^3), c_4 \in \mathbb{N}$ *is some constant.*

Proof. First, note that there exists a $(\ell_1/2, O(\mathsf{loc} \cdot \varepsilon_3), c' t^{c'})$ local assignment generator $\mathsf{LocalGen}$ for $\widehat{C_{x,\mathsf{ek}^*,\mathsf{ct}^*}}$ by combining Claim 11.8 and Lemma 11.27. Now, we construct a new local assignment generator $\mathsf{LocalGen}'$ for $\mathsf{Aug}(C_x, U(E_x, \cdot))$ as follows.

$\mathsf{LocalGen}'(S)$: Take as input a set of wires S of size at most $\ell \leq \ell_2$, and do the following.

- Parse $S = \{w_1, \ldots, w_\ell\}$ of D_x.
- If w_i is a wire in C_x, let Ω_i correspond to the same wire in $\widehat{C_{x,\mathsf{ek}^*,\mathsf{ct}^*}}$.
- If w_i is a wire in U_{E_x}, Let Δ_i be the subcircuit of $\widehat{C_{x,\mathsf{ek}^*,\mathsf{ct}^*}}$ which homomorphically evaluates w_i. Let Ω_i contain all the wires corresponding to the output of this homomorphic evaluation. Note that Ω_i is the size of an FHE ciphertext.
- Let $W_S = \bigcup_{i=1}^{\ell} \Omega_i$.
- Call $\{\sigma_i\}_{i \in W_S} \leftarrow \mathsf{LocalGen}(W_S)$.
- If w_i is a wire of C_x and $\Omega_i = \{\alpha_i\}$, set $\beta_i = \sigma_{\alpha_i}$.
- If w_i is a wire of U_{E_x}, parse $\{\sigma_j\}_{j \in \Omega_i}$ as an FHE ciphertext γ, and set $\beta_i \leftarrow \mathsf{FHE.Dec}(\mathsf{sk}^*, \gamma)$.
- Output $\{\beta_i\}_{i \in [\ell]}$.

We now show that the necessary properties hold.

Accepting. It is clear that on input $\{\mathsf{out}\}$, $\mathsf{LocalGen}'$ simply calls $\mathsf{LocalGen}(\{\mathsf{out}\})$ (because out is not under an encryption). Therefore, $\mathsf{Adv}_{G_{\mathsf{accept}}}[\mathsf{LocalGen}'] = \mathsf{Adv}_{G_{\mathsf{accept}}}[\mathsf{LocalGen}] \leq \varepsilon_3$.

Computational non-signaling. Consider sets T_0, T_1 of wires of size at most ℓ_2, and let $U = T_0 \cap T_1$. Let $G_{T_0,T_1,b}$ be the circuit as defined in the NS security in Definition 11.6. Consider the sets W_{T_b} as constructed by $\mathsf{LocalGen}'(T_b)$. Now, we argue that $G_{T_0,T_1,b}$ is isomorphic to the following circuit:

- Call $\{\sigma_i\}_{i \in W_{T_b}} \leftarrow \mathsf{LocalGen}(W_{T_b})$.
- Compute $W_U = W_{T_b} \cap W_{T_{1-b}}$.
- Let $L = \{\sigma_i\}_{i \in W_U}$.
- For each $w_i \in U$, let Ω_i be as defined in $\mathsf{LocalGen}'$.
- For each $w_i \in U$, use L to deduce Ω_i and therefore β_i as was done in $\mathsf{LocalGen}'$.
- Output $\{\beta_i\}_{i \in U}$.

Let s_4 be the size of the above circuit. By non-signaling of $\mathsf{LocalGen}$, we have that the distributions corresponding to $G^{\mathsf{LocalGen}}_{W_{T_0},W_{T_1},0}$ and $G^{\mathsf{LocalGen}}_{W_{T_0},W_{T_1},1}$ (for $\mathsf{LocalGen}$) are $(c't^{c'}, O(\mathsf{loc} \cdot \varepsilon_3)$-indistinguishable. Therefore, by invoking the Reduction Lemma, we have that $G_{T_0,T_1,0}$ and $G_{T_0,T_1,1}$ are $(c_4 t^{c_4}, O(\mathsf{loc} \cdot \varepsilon_3))$-indistinguishable, where $c_4 \in \mathbb{N}$ is some constant in $\mathbb{N}$ such that $c_4 t^{c_4} + s_4 \leq c' t^{c'}$.

Local consistency. Let $g = (w_i, w_j, w_k, f)$ be a gate of $\mathsf{Aug}(C_x, U(E_x, \cdot))$, and let $T \supseteq \{w_i, w_j, w_k\}$ be a subset of wires of C of size at most ℓ. Let $\mathcal{G}_{g,T}$ be the local consistency game as defined in Definition 11.6 for $\mathsf{LocalGen}'$.

If w_k is a wire of C_x, then, by local consistency of $\mathsf{LocalGen}$, we have that

$$\mathsf{Adv}_{\mathcal{G}_{g,T}}[\mathcal{A}] \leq \varepsilon_3.$$

If w_k is a wire of $U(E_x, \cdot)$, Ω_k is computed from Ω_i and Ω_j via $\mathsf{FHE.GateEval}$.

Let Γ correspond to the set of wires used to compute Ω_k from Ω_i, Ω_j. We argue in a few steps. First, consider $\mathsf{LocalGen}''(T)$ which is identical to $\mathsf{LocalGen}'$ but queries $\mathsf{LocalGen}(W_T \cup \Gamma)$ instead of $\mathsf{LocalGen}(W_T)$.

By the $(c_3 t^{c_3}, \varepsilon_3)$-non-signaling property of $\mathsf{LocalGen}$, we have that the outputs of $\mathsf{LocalGen}'(T)$ and $\mathsf{LocalGen}''(T)$ are $(c_4 t^{c_4}, \varepsilon_3)$-indistinguishable via the Reduction Lemma, if we choose $c_4 t^{c_4} + s \leq c_3 t^{c_3}$, where s is a size bound on $\mathsf{LocalGen}$ and $\mathsf{LocalGen}'$.

Now, consider the output $\mathsf{LocalGen}(W_T \cup \Gamma)$. Let $\mathcal{G}^{\mathsf{fullcon}} = (\mathcal{C}_1, \mathcal{C}_2, 0)$ denote the following game:

- $\mathcal{C}_1$ outputs $\mathsf{chall} = W_T \cup \Gamma$.
- $\mathcal{C}_2$ iterates over all triples of $W_T \cup \Gamma$. If the triple corresponds to a gate, it checks that the gate is consistent. If any gate is inconsistent, it outputs 1. Else, it outputs 0. (i.e. checks full consistency among every triple in the set).

By local consistency of $\mathsf{LocalGen}$ and invoking the Game Union Bound Lemma, we have that

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{fullcon}}}[\mathsf{LocalGen}] \leq O(\varepsilon_3 \cdot \ell_2^3).$$

Suppose the output of $\{\sigma_i\}_{W_T \cup \Gamma} \leftarrow \mathsf{LocalGen}(W_T \cup \Gamma)$ satisfies full consistency. Interpret the wires corresponding to Ω_i, Ω_j and Ω_k as FHE ciphertexts γ_i, γ_j and γ_k. If all the wires are consistent, γ_k is a

correct $\mathsf{FHE.GateEval}(\mathsf{ek}, f, \gamma_i, \gamma_j)$. Therefore, by the PV_1 malicious gate correctness of FHE, we can prove in PV_1 that $\mathsf{FHE.Dec}(\mathsf{sk}^*, \gamma_k) = f(\mathsf{FHE.Dec}(\mathsf{sk}^*, \gamma_i), \mathsf{FHE.Dec}(\mathsf{sk}^*, \gamma_j)$. Therefore, the corresponding output of $\mathsf{LocalGen}''$ must satisfy local consistency on the outputs w_1, w_2, w_3. Therefore, we have that whenever $\mathsf{LocalGen}''$ succeeds $\overline{\mathcal{G}_{\mathsf{fullcon}}}$, $\mathsf{LocalGen}''$ succeeds in $\overline{\mathcal{G}_{g,T}}$. Invoking the Game Reduction Lemma and Game Complement Lemma, we have that

$$\mathsf{Adv}_{\mathcal{G}_{g,T}}[\mathsf{LocalGen}''] \leq O(\varepsilon_3 \cdot \ell_2^3).$$

By ε_3-indistinguishability of $\mathsf{LocalGen}''$ and $\mathsf{LocalGen}'$, we then have that

$$\mathsf{Adv}_{\mathcal{G}_{g,T}}[\mathsf{LocalGen}''] \leq O(\varepsilon_3 \cdot \ell_2^3) + \varepsilon_3.$$

This completes the proof. □

Lemma 11.29. *$\mathsf{APC}_1 + $ "$\mathsf{prBPP} = \mathsf{prP}$" proves that there exists a $(\ell_3, \varepsilon_5, c_5 t^{c_5})$-accepting local assignment generator for $\mathsf{Aug}(C_x, E_x)$, where $c_5 \in \mathbb{N}$ is some constant, $\ell_3 = \ell_2 - \mathsf{poly}(\log s)$, and $\varepsilon_5 = \mathsf{poly}(s) \cdot \varepsilon_4$.*

Proof. By Lemma 11.28, there exists a $(\ell_2, \varepsilon_4, c_4 t^{c_4})$-accepting local assignment generator $\mathsf{LocalGen}$ for $\mathsf{Aug}(C_x, \mathcal{U}(E_X, \cdot))$. We define $\mathsf{LocalGen}'$ as follows.

$\mathsf{LocalGen}'(S)$: On input a set S of at most ℓ_3 wires of $\mathsf{Aug}(C_x, E_x)$, do the following.

- Parse $S = \{w_1, \ldots, w_{\ell_3}\}$.
- If w_i is a wire of C_x, let the $\widehat{w_i}$ be the corresponding wire in $\mathsf{Aug}(C_x, \mathcal{U}(E_x, \cdot))$.
- If w_i is a wire compute by gate g_j of E_x, let $\widehat{w_i}$ be the wire corresponding to the output of U_j which evaluates w_i, where U_j is defined in Section 11.1.1.
- Let $\widehat{S} = \{\widehat{w_1}, \widehat{w_2}, \ldots, \widehat{w_{\ell_3}}\}$.
- Output $\mathsf{LocalGen}(\widehat{S})$.

By definition, the fact that the circuit satisfies $(c_5 t^{c_5}, \varepsilon_4)$-non-signaling follows from the non-signaling property of $\mathsf{LocalGen}$ and the Reduction Lemma (where we use the fact that the above $\mathsf{LocalGen}'$ has size at most $\mathsf{poly}(t)$ for some fixed poly). Moreover, the fact that $\mathsf{LocalGen}$ is ε_4-accepting implies that $\mathsf{LocalGen}'$ is also ε_4-accepting. It suffices to prove local consistency.

Local consistency. Clearly, for gates g that are contained within C_x, local consistency of $\mathsf{LocalGen}'$ follows directly from the local consistency of $\mathsf{LocalGen}$. It suffices to consider some gate $g_i = (i, j_0, j_1, f)$ contained in E_x, and show that $\mathsf{LocalGen}$ (and hence $\mathsf{LocalGen}'$) is ε_5-consistent on wires i, j_0, j_1. Consider the corresponding gate $\mathcal{U}_i$ in $\mathcal{U}(E_x, \cdot)$ (see Section 11.1.1 for the appropriate definition). Recall that U_i comprises subcircuits $\{\Lambda_{b,\alpha}\}_{b\in\{0,1\},\alpha\in[i-1]}$, $\{\Lambda_b\}_{b\in\{0,1\}}$ and Λ_{out} (following Section 11.1.1). We will make reference to these variables throughout the following proof.

Consider the following game $\mathcal{G}^{\mathsf{transcript}}_{i,j_0,j_1,S}$, where S is some subset of wires of $\Lambda_0, \Lambda_1, \Lambda_{\mathsf{out}}$ of size at most $\ell_2 - \mathsf{poly}(\log s)$:

- $\mathcal{C}_1$: Output $\mathsf{chall} = \mathsf{state} = \{i, j_0, j_1\} \cup S$.

- $\mathcal{C}_2$: Given $(\sigma_i, \sigma_{j_0}, \sigma_{j_1}, \{\sigma_k\}_{k\in S})$:
  - Compute the true transcript of $\Lambda_0, \Lambda_1, \Lambda_{\mathsf{out}}$ given as input σ_{j_0} and σ_{j_1} (note that the transcript does not depend on any other wire inputs to U_i).
  - Check if all $\{\sigma_k\}_{k\in S}$ is consistent with the transcript. Output 1 if any of the wires are inconsistent.

Let $\omega_{b,\alpha}$ denote the output wire of $\Lambda_{b,\alpha}$ for $\alpha \in [i-1]$.

Claim 11.30. $\mathsf{Adv}^{\mathsf{transcript}}_{\mathcal{G}, i, j_0, j_1, \omega_{b,\alpha}}[\mathsf{LocalGen}] \leq \delta$, *where* $\delta = \mathsf{poly}(\log s) \cdot \varepsilon_4 + \varepsilon_4$.

Proof. We proceed in a few hybrids.

$\boxed{\mathcal{H}_1}$ Output $\mathsf{LocalGen}(W)$, where W of is a set of at most $\ell_3 - |\Lambda_{b,\alpha}|$ wires.

$\boxed{\mathcal{H}_2}$ Consider $\mathsf{LocalGen}_2(W)$ (which only accepts W of size at most $\ell_3 - |\Lambda_{b,\alpha}|$ which does the following:

- On input wires W, queries $\{\sigma_i\}_{i\in W} \cup \{\sigma_\omega\}_{\omega\in\Lambda_{b,\alpha}} \leftarrow \mathsf{LocalGen}(W \cup \{\omega\}_{\omega\in\Lambda_{b,\alpha}})$.
- Output $\{\omega_i\}_{i\in W}$.

By the ε_4-non-signaling property of $\mathsf{LocalGen}'$, this hybrid is ε_4 indistinguishable from the previous hybrid.

$\boxed{\mathcal{H}_3}$ Consider $\mathsf{LocalGen}'''$ which additionally does the following:

- On input wires W, queries $\{\sigma_i\}_{i\in W} \cup \{\sigma_\omega\}_{\omega\in\Lambda_{b,\alpha}} \leftarrow \mathsf{LocalGen}(W \cup \{\omega\}_{\omega\in\Lambda_{b,\alpha}})$.
- If any of the wire values corresponding to the gates in $\Lambda_{b,\alpha}$ are inconsistent, output $\perp$.
- Output $\{\sigma_i\}_{i\in W}$.

Recall that $\mathsf{LocalGen}$ is ε_4 consistent, so by the Game Union Bound Lemma, the probability that any of the gates is inconsistent is $|\Lambda_{b,\alpha}|^3 \cdot \varepsilon_4$ (there are at most $|\Lambda_{b,\alpha}|^3$ gates in $\Lambda_{b,\alpha}$).

Therefore, $\mathsf{LocalGen}_3$ is $(|\Lambda_{b,\alpha}|^3 \cdot \varepsilon_4 + \varepsilon_4)$-indistinguishable from $\mathsf{LocalGen}'$. By the Adversary Indistinguishability Lemma, we have that

$$\mathsf{Adv}^{\mathsf{transcript}}_{\mathcal{G}, i, j_0, j_1, \omega_{b,\alpha}}[\mathsf{LocalGen}''] \leq \mathsf{Adv}^{\mathsf{transcript}}_{\mathcal{G}, i, j_0, j_1, \omega_{b,\alpha}}[\mathsf{LocalGen}'''] + |\Lambda_{b,\alpha}|^3 \cdot \varepsilon_4 + \varepsilon_4.$$

We now have two cases. Fix $b \in \{0,1\}$. Suppose $\alpha \neq j_b$. Then, if the wire values of $\Lambda_{b,\alpha}$ are locally consistent, then the output of $\Lambda_{b,\alpha} = \perp$. Therefore, when $\mathsf{LocalGen}_3(\{i, j_0, j_1, \omega_{b,\alpha}\})$ does not abort, $\omega_{b,\alpha} = \perp$, which is indeed consistent with the transcript.

Similarly, suppose $\alpha = j_b$. If the wire values of $\Lambda_{b,\alpha}$ are locally consistent, then the output of $\Lambda_{b,\alpha} = \sigma_{j_b}$, which is indeed consistent with the transcript. Therefore, we deduce that:

$$\mathsf{Adv}^{\mathsf{transcript}}_{\mathcal{G}, i, j_0, j_1, \omega_{b,\alpha}}[\mathsf{LocalGen}_3] = 0.$$

By combining the inequalities with the fact that $|\Lambda_{b,\alpha}| \leq \mathsf{poly}(\log s)$, we have the desired claim. □

Let the subcircuit $\hat{\Lambda} = \Lambda_0 \cup \Lambda_1 \cup \Lambda_{\mathsf{out}}$. Now, we will inductively prove the following claim.

Claim 11.31. *Let η be some wire at depth k (the input wires are depth 0) of circuit $\hat{C}$.*

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{transcript}}_{i,j_0,j_1,\eta}}[\mathsf{LocalGen}] \leq 2^k\delta + 2^{k+1}\varepsilon_4 - 2\varepsilon_4.$$

Proof. By Claim 11.30, we know that the base case is true.

Suppose the claim holds for all wires at depth k of the $\hat{\Lambda}$. Now, consider wire η at depth $k+1$ of gate $g = (\eta, \zeta_0, \zeta_1, f)$ of C_b. We will show that the claim holds for η via induction.

$\boxed{\mathcal{H}_1}$ Output $\mathsf{LocalGen}$ on input $\{i, j_0, j_1, \eta\}$.

$\boxed{\mathcal{H}_2}$ Consider $\mathsf{LocalGen}_1$ which on input $\{i, j_0, j_1, \eta\}$ does the following:

- Query $\{\sigma_i, \sigma_{j_0}, \sigma_{j_1}, \sigma_\eta, \sigma_{\zeta_0}, \sigma_{\zeta_1}\} \leftarrow \mathsf{LocalGen}(\{i, j_0, j_1, \eta, \zeta_0, \zeta_1\})$.
- Output $\{\sigma_i, \sigma_{j_0}, \sigma_{j_1}, \sigma_\eta\}$.

By $(c_4 t^{c_4}, \varepsilon_4)$-non-signaling, we have that this is ε_4 indistinguishable from the previous hybrid.

$\boxed{\mathcal{H}_3}$ Consider $\mathsf{LocalGen}_2$ which on input $\{i, j_0, j_1, \eta\}$, does the following:

- Query $\{\sigma_i, \sigma_{j_0}, \sigma_{j_1}, \sigma_\eta, \sigma_{\zeta_0}, \sigma_{\zeta_1}\} \leftarrow \mathsf{LocalGen}(\{i, j_0, j_1, \eta, \zeta_0, \zeta_1\})$.
- If ζ_0 and ζ_1 is not consistent with η, output $\perp$.
- Output $\{\sigma_i, \sigma_{j_0}, \sigma_{j_1}, \sigma_\eta\}$.

By the ε_4-local consistency of $\mathsf{LocalGen}$, this is ε_4-indistinguishable from the previous hybrid.

$\boxed{\mathcal{H}_4}$ Consider $\mathsf{LocalGen}_3$ which on input $\{i, j_0, j_1, \eta\}$, does the following:

- Query $\{\sigma_i, \sigma_{j_0}, \sigma_{j_1}, \sigma_\eta, \sigma_{\zeta_0}, \sigma_{\zeta_1}\} \leftarrow \mathsf{LocalGen}(\{i, j_0, j_1, \eta, \zeta_0, \zeta_1\})$.
- If ζ_0 or ζ_1 is inconsistent with the transcript (i.e. $\mathcal{G}^{\mathsf{transcript}}_{i,j_0,j_1,\zeta_0}$ or $\mathcal{G}^{\mathsf{transcript}}_{i,j_0,j_1,\zeta_1}$ outputs 1), reject.
- If ζ_0 and ζ_1 are not consistent with η, output $\perp$.
- Output $\{\sigma_i, \sigma_{j_0}, \sigma_{j_1}, \sigma_\eta\}$.

By the inductive hypothesis, Lemmas 7.18 and 7.23, this is $2 \cdot (2^k\delta + 2^k\varepsilon_4 - 2\varepsilon_4)$-indistinguishable from the previous hybrid.

Note that $\mathsf{LocalGen}_3$ never outputs ζ which is inconsistent with the transcript, i.e.

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{transcript}}_{i,j_0,j_1,\eta}}[\mathsf{LocalGen}_3] = 0.$$

Hence, by the Hybrid Argument and the Adversary Indistinguishability Lemma, we have that

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{transcript}}_{i,j_0,j_1,\eta}}[\mathsf{LocalGen}] \leq 2\varepsilon_4 + 2(2^k\delta + 2^{k+1}\varepsilon_4 - 2\varepsilon_4) = 2^{k+1}\delta + 2^{k+2}\varepsilon_4 - 2\varepsilon_4,$$

therefore completing the proof. □

Recall that $\hat{\Lambda}$ has depth $O(\log s)$. Let Λ_{out} correspond to the output of $\hat{\Lambda}$. We then have that

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{transcript}}_{i,j_0,j_1,\Lambda_{\mathsf{out}}}}[\mathsf{LocalGen}] \leq \mathsf{poly}(s) \cdot (\varepsilon_4 + \delta).$$

Moreover, it is easy to show that when $\mathsf{LocalGen}$ satisfies $\overline{\mathcal{G}^{\mathsf{transcript}}_{i,j_0,j_1,\Lambda_{\mathsf{out}}}}$, $\mathsf{LocalGen}'$ satisfies $\overline{\mathcal{G}_{g,\{g\}}}$ (as defined in Definition 11.6), and hence

$$\mathsf{Adv}_{\mathcal{G}_{g,\{g\}}}[\mathsf{LocalGen}'] \leq \mathsf{poly}(s) \cdot \varepsilon_4.$$

By non-signaling, we can also show that for any $T \supseteq \{g\}$,

$$\mathsf{Adv}_{\mathcal{G}_{g,T}}[\mathsf{LocalGen}'] \leq \mathsf{poly}(s) \cdot \varepsilon_4 + \varepsilon_4.$$

This shows that $\mathsf{LocalGen}'$ is $\mathsf{poly}(s) \cdot \varepsilon_4$-locally consistent. □

By picking $c_6 \in \mathbb{N}$ appropriately, our above analysis shows that there exist PV definable polynomials q_1, q_2 so that Lemma 11.29 gives a $(q_1(\log n, \log L, \lambda), \frac{1}{q_2(n,L,\lambda)}, c_6 t^{c_6})$ accepting local assignment generator, i.e.

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{accept}}}[\mathsf{LocalGen}] \geq 1 - \frac{1}{q_2(n, \lambda, \ell)}.$$

However, Theorem 11.23 shows that there exists a polynomial p such that

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{accept}}}[\mathsf{LocalGen}] \leq \frac{p_2(n, \ell)}{q_2(n, \lambda, \ell)}.$$

By picking q_2 to be large enough (which we can do by picking the constant c appropriately at the beginning of this proof), this leads to a contradiction, thus completing the proof.

11.5 Proof of Theorem 11.23

In this section, following [JKLV24], we show that if there is a size ℓ-sized EF that $\forall w, C_x(w) = 0$, then there exists an extension E_x with description size $L = \mathsf{poly}(\ell)$ a constant A such that there is no *accepting* local assignment generator for $\mathsf{Aug}(C_x, E_x)$. As we show in Appendix B, we can without loss of generality (up to a polynomial blow-up in proof size) assume that every line in the EF proof has constant width.

The following part of this section follows [JKLV24] nearly verbatim.

Circuit as a Set of Propositions. For any circuit C, we define a set of propositional formulas $\mathsf{Prop}[C]$, which represents that each gate computation in C is correct. More specifically, we assign each wire in C a propositional variable. Then for each gate g in C with input wires l, r and output wire o, we can use the propositional formula $o \leftrightarrow g(l, r)$ to represent that g is computed correctly. We put all such formulas together as a set, and define it as $\mathsf{Prop}[C]$. One can view a proof of the fact that $\forall w, C_x(w) = 0$ as length $\mathsf{poly}(\ell)$ proof of the fact that:

$$\mathsf{Prop}[C_x] \vdash (\mathsf{out} \leftrightarrow 0)$$

We now describe how to construct an extension E_x for C_x such that Theorem 11.23 holds.

Construction 11.32 (Extension E_x)**.** Let $\phi_1, \ldots, \phi_k, \theta_1, \ldots, \theta_\ell \vdash_{\mathsf{EF}} C_x(w) = 0$, where each $|\phi_i|, |\theta_i| = O(1)$. Suppose that the ϕ_i are all extensions, and the θ_i are all premises and deductions. We construct the extension circuit E_x which takes as input the wires of C_x, and does the following:

- **Variables for gates:** For each wire of C_x, construct a gate g in E_x that simply copies over this wire.
- **Adding extensions:** For each extension ϕ_i of the form $v_i \leftrightarrow \varphi_i$, where v_i is a new variable and φ_i is a propositional formula of $O(1)$ existing variables (including the wire values of C_x). Now, compute a helper circuit Λ_i which takes as input all the variables that appear in φ_i, and computes φ_i. We set the output of Λ_i to be v_i.
- **Adding proofs:** For line θ_i of the proof, recall that it depends on $O(1)$ previous lines of the proof. Construct the circuit P_i which takes as input the wires corresponding to the variables which appear in θ_i, and computes θ_i. Recall that θ_i has one of two forms:
  - **Premise:** This asserts that for some gate g in C_x with input wires l, r and output wire o, we have that $o \leftrightarrow g(l, r)$. Therefore, P_i takes as input the wires of E_x corresponding to the variables o, l and r and simply asserts that $o \leftrightarrow g(l, r)$ holds.
  - **Deduction.** θ_i takes as input some extension variables and asserts that some predicate on these variables holds. Recall that θ_i is deduced via some lines $\{\theta_{i_1}, \dots \theta_{i_c}\}$ of the proof via some axiom of EF.

  It is easy to see that each P_i has size $O(1)$ (because each line θ_i has size $O(1)$).
- **Binary-and-tree:** Without loss of generality, suppose ℓ is a power of 2. Compute a binary tree of ANDs (each gate is and AND of its children), where the leaf nodes corresponds to the output wires of $P_0, \dots, P_{\ell-1}$.

We are now ready to prove Theorem 11.23. The proof strategy is very similar, but we present it directly (JKLV modularizes the analysis of the AND-tree).

Proof of Theorem 11.23. We choose E_x as defined in Construction 11.32. For convenience, we write $\gamma = \log_2 \ell$ and $\mathsf{loc} = p_1(\log n, \log \ell)$, where p_1 is a *linear* function in $\log n$ and $\log \ell$. Consider the subcircuit of E_x which corresponds to the binary AND tree with leaves corresponding to the output wires of $P_1, \dots, P_\ell$. Let the output wire of P_i be ρ_i.

Suppose lines $\theta_0, \dots, \theta_\alpha$ correspond to the premise (i.e. gate consistency of the wires of C_x). Let S be any subset of wires of $\mathsf{Aug}(C_x, E_x)$ of size at most $\mathsf{loc}/2$. Consider the following search game $\mathcal{G}^{\mathsf{premise}}_{k,S} = (\mathcal{C}_1, \mathcal{C}_2, k)$ for $k \leq \alpha$:

- $\mathcal{C}_1$ outputs wire ρ_k corresponding to the output gate of P_k, along with rest of the set S.
- $\mathcal{C}_2$ on input $\{\sigma_{\rho_k}\} \cup \{\sigma_i\}_{i \in S}$, output 1 if $\sigma_{\rho_k} = 0$, and 0 otherwise.

Claim 11.33. $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$" *proves the following.* $\mathsf{Adv}_{\mathcal{G}^{\mathsf{premise}}_{k,S}}[\mathsf{LocalGen}] \leq O(\varepsilon)$.

Proof. We proceed in a few hybrids to first construct an indistinguishable $\mathsf{LocalGen}'$.

$\boxed{\mathcal{H}_1}$ Output $\mathsf{LocalGen}(\rho_k)$.

$\boxed{\mathcal{H}_2}$ Consider $\mathsf{LocalGen}_1$ which does the following:

- Let Γ be the subcircuit of E_x used to compute ρ_k (i.e. the relevant wires of C_x and the subcircuit computing $o \leftrightarrow g(\ell, r)$). Clearly, this has size $O(1)$.

- Compute $\{\sigma_w\} \cup \{\sigma_i\}_{i \in \Gamma} \leftarrow \mathsf{LocalGen}(\{w\} \cup \Gamma)$.
- Output σ_w.

This distribution is (t, ε)-indistinguishable from the previous hybrid via the (ct^c, ε)-indistinguishability of $\mathsf{LocalGen}$.

$\boxed{\mathcal{H}_3}$ Consider $\mathsf{LocalGen}_2$ which does the following.

- Let Γ be the subcircuit used to compute w (i.e. the relevant wires of C_x and the subcircuit computing $o \leftrightarrow g(\ell, r)$).
- Compute $\{\sigma_w\} \cup \{\sigma_i\}_{i \in \Gamma} \leftarrow \mathsf{LocalGen}(\{w\} \cup \Gamma)$.
- If any of the wires in $\{\sigma_w\} \cup \{\sigma_i\}_{i \in \Gamma}$ are inconsistent, output $\perp$.
- Output σ_w.

By the ε-local consistency of $\mathsf{LocalGen}$, by applying Lemmas 7.18 and 7.23, we have that this distribution is $(t, \varepsilon \cdot |\Gamma|^3)$-indistinguishable from the previous hybrid.

By construction, we have that $\mathsf{LocalGen}(w)$ and $\mathsf{LocalGen}_2$ are $(t, 2|\Gamma|^3 \cdot \varepsilon)$-indistinguishable. Moreover, it is easy to see that one can prove in PV_1 that

$$\mathsf{Adv}_{\mathcal{G}_k^{\mathsf{premise}}}[\mathsf{LocalGen}_2] = 0.$$

Since Γ has size at most $O(1)$, by applying the Adversary Indistinguishability Lemma, we have that

$$\mathsf{Adv}_{\mathcal{G}_k^{\mathsf{premise}}}[\mathsf{LocalGen}] \leq O(\varepsilon).$$

Now, we can invoke (ct^c, ε)-non-signaling of $\mathsf{LocalGen}$ to argue that

$$\mathsf{Adv}_{\mathcal{G}_{k \cup S}^{\mathsf{premise}}}[\mathsf{LocalGen}] \leq \mathsf{Adv}_{\mathcal{G}_k^{\mathsf{premise}}}[\mathsf{LocalGen}] + \varepsilon.$$

This completes the proof. □

Now, lines $\theta_{\alpha+1}, \ldots, \theta_{\ell-1}$ correspond to deductions. Consider line k, and let $S_k = \{i_1, \ldots, i_c\} \subseteq [k-1]$ denote the set of $c = O(1)$ lines $\theta_{i_1}, \ldots, \theta_{i_c}$ used to deduce θ_k.

Let S be any subset of wires of $\mathsf{Aug}(C_x, E_x)$ of size at most $p(\log n, \log \ell)/2$. Consider the following search game $\mathcal{G}_{k \cup S}^{\mathsf{deduce}}$ for $k \in \{\alpha + 1, \ldots, \ell - 1\}$:

- $\mathcal{C}_1$ outputs the output wire w_k of P_k, the output wires w_{i_j} of P_{i_j} for $i_j \in S_k$ as well as the set S.
- $\mathcal{C}_2$, on input $\{\sigma_k, \sigma_{i_1}, \ldots \sigma_{i_c}\} \cup \{\sigma_w\}_{w \in S}$ does the following:
  - If any of $\sigma_{i_c} = 0$, output 0.
  - Otherwise, if $\sigma_k = 0$, output 1. Else, output 0.

We can then prove the following claim.

Claim 11.34. $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$" *proves that* $\mathsf{Adv}_{\mathcal{G}_{k \cup S}^{\mathsf{deduce}}}[\mathsf{LocalGen}] \leq O(\varepsilon)$.

Proof. We first prove the claim when $S = \varnothing$. We first construct an indistinguishable program to $\mathsf{LocalGen}$ via a few hybrids.

$\boxed{\mathcal{H}_1}$ Output $\mathsf{LocalGen}(\{w_k, w_{i_1}, \ldots, w_{i_c}\})$.

$\boxed{\mathcal{H}_2}$ Let $\mathsf{LocalGen}_1$ work as follows:

- Let $P = P_k \cup P_{i_1} \cup \cdots \cup P_{i_c}$.
- Call $\{\sigma_k, \sigma_{i_1}, \ldots, \sigma_{i_c}\} \cup \{\sigma_w\}_{w \in P} \leftarrow \mathsf{LocalGen}(\{w_k, w_{i_1}, \ldots, w_{i_c}\} \cup P)$.
- Output $\{\sigma_k, \sigma_{i_1}, \ldots, \sigma_{i_c}\}$.

This is (t, ε)-indistinguishable from the previous hybrid due to the (ct^c, ε)-non-signaling of $\mathsf{LocalGen}$.

$\boxed{\mathcal{H}_3}$ Let $\mathsf{LocalGen}_2$ work as follows:

- Let $P = P_k \cup P_{i_1} \cup \cdots \cup P_{i_c}$.
- Call $\{\sigma_k, \sigma_{i_1}, \ldots, \sigma_{i_c}\} \cup \{\sigma_w\}_{w \in P} \leftarrow \mathsf{LocalGen}(\{w_k, w_{i_1}, \ldots, w_{i_c}\} \cup P)$.
- If any of the wire values in $\{\sigma_k, \sigma_{i_1}, \ldots, \sigma_{i_c}\} \cup \{\sigma_w\}_{w \in P}$ are inconsistent, output 0.
- Output $\{\sigma_k, \sigma_{i_1}, \ldots, \sigma_{i_c}\}$.

By ε-consistency of $\mathsf{LocalGen}$, we can invoke the Game Union Bound Lemma and Mixture Lemma to show that the distributions are $(t, |P|^3 \cdot \varepsilon)$-indistinguishable. Recall that $|P| = O(1)$.

Now, it is easy to show in PV_1 that $\mathsf{Adv}_{\mathcal{G}_k^{\mathsf{deduce}}}[\mathsf{LocalGen}_2] = 0$ because P_k outputs 1 if all of $P_{i_1}, \ldots, P_{i_c} = 1$ by the soundness of EF. Therefore, by the Adversary Indistinguishability Lemma, we have the desired claim.

Now, by relying on (ct^c, ε)-non-signaling of $\mathsf{LocalGen}$ and the Reduction Lemma, we can then deduce that

$$\mathsf{Adv}_{\mathcal{G}_{k \cup S}^{\mathsf{deduce}}}[\mathsf{LocalGen}] \leq \mathsf{Adv}_{\mathcal{G}_k^{\mathsf{deduce}}}[\mathsf{LocalGen}] + \varepsilon.$$

Choosing p_2 appropriately, we complete the proof. □

Consider (disjoint) sets of wires S and T, each of size at most $\mathsf{loc}/4$, where S is contained in the AND tree, and T is any subset of wires in $\mathsf{Aug}(C_x, E_x)$. We define a security game $\mathcal{G}^{\mathsf{AND}}_{S,T} = (\mathcal{C}_1, \mathcal{C}_2, 0)$ as follows:

- $\mathcal{C}_1$ outputs S, T.
- $\mathcal{C}_2$ takes as input a set of wires $\{\sigma_w\}_S \cup \{\sigma_w\}_{w \in T}$, and outputs 1 if any of the wire values in $\{\sigma_w\}_S$ is 0 (ignore the wire values in T). Output 0 otherwise.

We identify every wire in the tree with a string $(0, b_1, \ldots, b'_\gamma)$ for $\gamma' \leq \gamma$ as follows:

- The root wire (AKA the output wire) of the AND tree is labeled by the string (0).
- The string $(0, b_1, \ldots, b'_\gamma)$ corresponds to the string obtained by starting at the root wire, and iterating over $b_1, \ldots, b'_\gamma$, going down the left input if $b_i = 0$ and right input if $b_i = 1$.

Let T be any subset of wires in $\mathsf{Aug}(C_x, E_x)$ of size at most $\mathsf{loc}/4$, and S be a subset of wires in the AND tree of size at most $\mathsf{loc}/8$. We define a security game $\mathcal{G}^{\mathsf{AND\text{-}path}}_{(b_0, b_1, \ldots, b_\gamma), S, T} = (\mathcal{C}_1, \mathcal{C}_2, 0)$ (where $b_i \in \{0, 1\}$) to be the security game $\mathcal{G}^{\mathsf{AND}}_{U,T}$, where U is constructed as follows:

- Initialize $U = \varnothing$.

- Iterate over $i \in [\gamma]$, and for each i such that $b_i = 1$, add $(0, b_1, \ldots, b_i - 1)$ to U.
- Add elements of S to U.
- Output U.

For convenience, we identify the string $(b_0, b_1, \ldots, b_\gamma)$ with $k \in \{0, 1, \ldots, \ell\}$ as follows:

$$k = \sum_{i=0}^{\gamma} 2^i b_{\gamma - i}.$$

We will equivalently write $\mathcal{G}^{\mathsf{AND\text{-}path}}_{(b_1,\ldots,b_\gamma),S,T}$ and $\mathcal{G}^{\mathsf{AND\text{-}path}}_{k,S,T}$ for convenience.

Claim 11.35. $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ *proves the following sentence. For all* $k \in \{0, 1, \ldots, \ell\}$ *and all sets* T *of size at most* $\mathsf{loc}/4$*, we have that*

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{AND\text{-}path}}_{k,\varnothing,T}}[\mathsf{LocalGen}] \leq q(\log n, \log \ell, k) \cdot \varepsilon,$$

where q *is some* PV*-definable polynomial.*

Proof. We prove the above theorem via induction on k.[30] We denote by S_k the output of $\mathcal{C}_1$ in game $\mathcal{G}^{\mathsf{AND}}_k$. Clearly,

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{AND\text{-}path}}_{0,\varnothing,T}}[\mathsf{LocalGen}] = 0$$

by definition because $\mathcal{C}_1$ queries an empty set. Therefore, the claim holds for $k = 0$. Suppose that the claim holds for $k = \kappa$. Now, consider $\kappa + 1$. We will first prove the following.

If ρ_κ corresponds to a premise, via the Game Union Bound Lemma, we have that

$$\begin{aligned}\mathsf{Adv}_{\mathcal{G}^{\mathsf{AND\text{-}path}}_{\kappa,\{\rho_\kappa\},T}}[\mathsf{LocalGen}] &\leq \mathsf{Adv}_{\mathcal{G}^{\mathsf{AND\text{-}path}}_{\kappa,\varnothing,\{\rho_\kappa\}\cup T}}[\mathsf{LocalGen}] + \mathsf{Adv}_{\mathcal{G}^{\mathsf{premise}}_{\rho_\kappa,S_\kappa\cup T}}[\mathsf{LocalGen}] \\ &\leq q(\log n, \log \ell, \kappa) \cdot \varepsilon + O(\varepsilon). \end{aligned} \tag{11.1}$$

Similarly, if ρ_κ corresponds to a line which is a deduction, via the Game Union Bound Lemma, we have that

$$\begin{aligned}\mathsf{Adv}_{\mathcal{G}^{\mathsf{AND\text{-}path}}_{\kappa,\{\rho_\kappa\},T}}[\mathsf{LocalGen}] &\leq \mathsf{Adv}_{\mathcal{G}^{\mathsf{AND\text{-}path}}_{\kappa,\varnothing,\{\rho_\kappa\}\cup T}}[\mathsf{LocalGen}] + \mathsf{Adv}_{\mathcal{G}^{\mathsf{deduce}}_{\rho_\kappa,S_\kappa\cup T}}[\mathsf{LocalGen}] \\ &\leq q(\log n, \log \ell, \kappa) \cdot \varepsilon + O(\varepsilon). \end{aligned} \tag{11.2}$$

We consider two cases based on the parity of κ. If κ is even (i.e. $\kappa + 1$ is odd), it is easy to verify that $S_{\kappa+1} = S_\kappa \cup \{\rho_\kappa\}$. Therefore, by definition,

$$\begin{aligned}\mathsf{Adv}_{\mathcal{G}_{\kappa+1,\varnothing,T}}[\mathsf{LocalGen}] &= \mathsf{Adv}_{\mathcal{G}_{\kappa,\{\rho_\kappa\},T}}[\mathsf{LocalGen}] \\ &\leq q(\log n, \log \ell, \kappa) \cdot \varepsilon + O(\varepsilon),\end{aligned}$$

hence proving the claim. Therefore, this satisfies the inductive hypothesis. It suffices to consider the case where κ is odd. We define sets S, U, U', U'' in two cases in order to be explicit.

[30] Recall that PV_1 (and thus $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$) allows induction on polynomial-time decidable properties. Here, as $\mathsf{loc} = O(\log n + \log \ell)$ and $n, \ell \in \mathsf{Log}$, we know that $|T| \in \mathsf{Log}$ and thus the universal quantification over T is polynomial-time decidable; subsequently the induction is admissible in the theory.

Case 1: $\kappa \leq \ell - 2$. In this case, there exists an index $\gamma' < \gamma$ such that $\kappa = (0, b_1, \ldots, b_{\gamma'}, 0, 1, \ldots, 1)$ and $\kappa + 1 = (0, b_1, \ldots, b_{\gamma'}, 1, 0, \ldots, 0)$. There exists a set S so that we can rewrite:

$$S_\kappa = S \cup \{(0, b_0, b_1, \ldots, b_{\gamma'}, 0, \underbrace{0, \ldots, 0}_{i \text{ zeros}})\}_{i \in \{1, \ldots, \gamma - \gamma' - 1\}} = S \cup U$$
$$S_{\kappa+1} = S \cup \{(b_0, b_1, \ldots, b_{\gamma'}, 0)\} = S \cup U'.$$

We also define the set:

$$U'' = \{(b_1, \ldots, b_{\gamma'}, \underbrace{0, \ldots, 0}_{i \text{ zeros}}, 1)\}_{i \in [\gamma - \gamma' - 1]}$$

Case 2: $\kappa = \ell - 1$. In this case, $\kappa = (0, 1, \ldots, 1)$ and $\kappa + 1 = (1, 0, \ldots, 0)$. Then, we set $S = 0$, and it is clear that:

$$S_\kappa = \{(0, \underbrace{0, \ldots, 0}_{i \text{ zeros}})\}_{i \in [\gamma]} = U,$$
$$S_{\kappa+1} = \{(0)\} = U'.$$

Let the set $U'' = \{(\underbrace{0, \ldots, 0}_{i \text{ zeros}}, 1)\}_{i \in [\gamma]}$.

The following argument works for either case. We proceed in a few hybrids.

$\boxed{\mathcal{H}_1}$ Output $\mathsf{LocalGen}(S_{\kappa+1} \cup T)$.

$\boxed{\mathcal{H}_2}$ $\mathsf{LocalGen}_1(S_{\kappa+1} \cup T)$ works as follows:

- Call $\{\sigma_i\}_{S_{\kappa+1} \cup U \cup U'' \cup \{\rho_k\} \cup T} \leftarrow \mathsf{LocalGen}(S_{\kappa+1} \cup U \cup U'' \cup \{\rho_k\} \cup T)$.
- Output $\{\sigma_i\}_{i \in S_{\kappa+1} \cup T}$.

(t, ε)-indistinguishability from the previous hybrid follows from (ct^c, ε)-non-signaling of $\mathsf{LocalGen}$ (and by picking c to be large enough).

$\boxed{\mathcal{H}_3}$ $\mathsf{LocalGen}_2(S_{\kappa+1} \cup T)$ works as follows:

- Call $\{\sigma_i\}_{S_{\kappa+1} \cup U \cup U'' \cup \{\rho_k\} \cup T} \leftarrow \mathsf{LocalGen}(S_{\kappa+1} \cup U \cup U'' \cup \{\rho_k\} \cup T)$.
- If any of $\{\sigma_i\}_{i \in S_\kappa \cup \{\rho_\kappa\}}$ is equal to 0, abort.
- Output $\{\sigma_i\}_{i \in S_{\kappa+1} \cup T}$.

This follows the Mixture Lemma and (11.1) or (11.2) (based on whether P_κ corresponds to a premise or deduction), we have that this hybrid is $(t, \varepsilon \cdot (p(\log n, \log \ell, \kappa) + O(\varepsilon)))$-indistinguishable from the previous.

$\boxed{\mathcal{H}_4}$ $\mathsf{LocalGen}_3(S_{\kappa+1} \cup T)$ works as follows:

- Call $\{\sigma_i\}_{S_{\kappa+1} \cup U \cup U'' \cup \{\rho_k\} \cup T} \leftarrow \mathsf{LocalGen}(S_{\kappa+1} \cup U \cup U'' \cup \{\rho_k\} \cup T)$.
- If any of $\{\sigma_i\}_{i \in S_\kappa \cup \{\rho_\kappa\}}$ is equal to 0, abort.
- If any of the wire values are inconsistent with each other (among the gates that are being extracted), abort.

- Output $\{\sigma_i\}_{i \in S_{\kappa+1} \cup T}$.

Here, we rely on the Game Union Bound Lemma, Mixture Lemma, and the ε-gate consistency of $\mathsf{LocalGen}$ to argue that the distributions are $(|S| + |U| + |U'| + |U''| + |T| + 1)^3 \cdot \varepsilon$-indistinguishable.

Recall that if all the wires in $U \cup U' \cup U''$ are consistent, and we know that all the wires in $S \cup U$ as well as ρ_k evaluate to 1, then every wire in $S \cup U'$ evaluates to 1. Therefore, it is clear that

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{AND\text{-}path}}_{\kappa+1,\varnothing,0}}[\mathsf{LocalGen}_3] = 0.$$

Recall that if $S_\kappa \cup \{\rho_k\}$ evaluates to 1, by wire consistency of the AND gates, all the wires in $S_{\kappa+1}$ also evaluate to 1. By the Hybrid Argument, we have that

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{AND\text{-}path}}_{\kappa+1,\varnothing,0}}[\mathsf{LocalGen}] \leq q(\log n, \log \ell, \kappa) \cdot \varepsilon + q'(\log n, \log \ell),$$

where the q' is determined by the hybrids. Therefore, by choosing q to be large enough, we have the desired result. □

Let η_{out} be the output wire of the AND-tree. At this point, Claim 11.35, we have that for any subset of wires T of $\mathsf{Aug}(C_x, E_x)$ of size at most $\mathsf{loc}/4$,

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{AND}}_{\{\eta_{\mathsf{out}}\},T}}[\mathsf{LocalGen}] \leq q(\log n, \log \ell, \ell) \cdot \varepsilon. \tag{11.3}$$

Now, let T be the following set of wires:

- Add all the wires in the AND-tree from the root node to the last line $\rho_{\ell-1}$, along with the sibling wires in the AND-tree of all such nodes.
- Add all wires of $P_{\ell-1}$.
- Recall that $P_{\ell-1}$ corresponds to the claim that $\mathsf{out} \leftrightarrow 0$, where out is the output wire of C_x.

It is easy to see that $|T| = O(\log n, \log \ell)$, and hence for a large enough linear function p_1, we do indeed have that $|T| \leq \mathsf{loc}/4$. Let $\mathcal{G}_{\mathsf{accept}}$ be the search game as defined in Definition 11.6. We now show there exists p_2 such that

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{accept}}}[\mathsf{LocalGen}] \leq p_2(\log n, \ell) \cdot \varepsilon. \tag{11.4}$$

We show this via a few hybrids.

$\boxed{\mathcal{H}_1}$ Output $\mathsf{LocalGen}(\{\mathsf{out}\})$.

$\boxed{\mathcal{H}_2}$ Let $\mathsf{LocalGen}_1(\{\mathsf{out}\})$ be the following program:

- Call $\{\sigma_{\mathsf{out}}, \sigma_{\eta_{\mathsf{out}}}\} \cup \{\sigma_w\}_{w \in T} \leftarrow \mathsf{LocalGen}'(\{\mathsf{out}, \eta_{\mathsf{out}}\} \cup T)$.
- Output σ_{out}.

By the (ct^c, ε)-non-signaling of $\mathsf{LocalGen}$, we have that the distributions are (t, ε)-indistinguishable.

$\boxed{\mathcal{H}_3}$ Let $\mathsf{LocalGen}_2(\{\mathsf{out}\})$ be the following program:

- Call $\{\sigma_{\mathsf{out}}, \sigma_{\eta_{\mathsf{out}}}\} \cup \{\sigma_w\}_{w \in T} \leftarrow \mathsf{LocalGen}'(\{\mathsf{out}, \eta_{\mathsf{out}}\} \cup T)$.
- If $\eta_{\mathsf{out}} \neq 1$, output $\bot$.
- Else, output σ_{out}.

By (11.3) and the Mixture Lemma, we have that the hybrids are $(t, q(\log n, \log \ell, \ell) \cdot \varepsilon)$-indistinguishable.

$\boxed{\mathcal{H}_4}$ Let $\mathsf{LocalGen}_3(\{\mathsf{out}\})$ be the following program:

- Call $\{\sigma_{\mathsf{out}}, \sigma_{\eta_{\mathsf{out}}}\} \cup \{\sigma_w\}_{w \in T} \leftarrow \mathsf{LocalGen}'(\{\mathsf{out}, \eta_{\mathsf{out}}\} \cup T)$.
- If $\eta_{\mathsf{out}} \neq 1$, output $\bot$.
- If any of the wires output by LocalGen are inconsistent, output $\bot$.
- Else, output σ_{out}.

By the ε-local consistency of LocalGen, via the Game Union Bound Lemma and Mixture Lemma, we have that the distributions are $(t, |T|^3 \cdot \varepsilon)$-indistinguishable.

$\boxed{\mathcal{H}_5}$ Let $\mathsf{LocalGen}_4(\{\mathsf{out}\})$ be the following program:

- Call $\{\sigma_{\mathsf{out}}, \sigma_{\eta_{\mathsf{out}}}\} \cup \{\sigma_w\}_{w \in T} \leftarrow \mathsf{LocalGen}'(\{\mathsf{out}, \eta_{\mathsf{out}}\} \cup T)$.
- If $\eta_{\mathsf{out}} \neq 1$, output $\bot$.
- If any of the wires output by LocalGen are inconsistent, output $\bot$.
- If $\sigma_{\mathsf{out}} = 1$, output $\bot$.
- Else, output σ_{out}.

By consistency of all the wires in T, we know that $\eta_{\mathsf{out}} = 1$ means that $\rho_{\ell-1} = 1$ (by consistency of the AND gates in T). By construction, $\rho_{\ell-1} = 1$ only if $\mathsf{out} = 0$ is indeed true. Therefore, we can prove in PV that this is a functionally equivalent change.

By definition, we have that

$$\mathsf{Adv}_{\mathcal{G}^{\mathsf{accept}}}[\mathsf{LocalGen}_4] = 0.$$

Therefore, by invoking the Adversary Indistinguishability Lemma and the Hybrid Argument, we have (11.4). This completes the proof. □

Acknowledgements

We thank Jan Krajíček for helpful comments on the introduction.

A. Jain was supported in part by NSF CAREER 1942789, Johns Hopkins University Catalyst award, JP Morgan Faculty Award and research gifts from Ethereum Foundation, Stellar Development Foundation, and Cisco. J. Li received support from National Science Foundation under Grant No. CCF-2420092. S. Mathialagan was supported in part by Jane Street and the Simons Institute for the Theory of Computing.

Part of this work was done while the authors were visiting the Simons Institute for the Theory of Computing, UC Berkeley, and while Yao-Ching Hsieh and Jiatu Li were interns at NTT Research, Sunnyvale.

References

[ABM23] Albert Atserias, Sam Buss, and Moritz Müller. On the consistency of circuit lower bounds for non-deterministic time. In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 1257–1270. ACM, 2023. 7, 10, 42

[Ajt94] Miklós Ajtai. The complexity of the pigeonhole principle. *Comb.*, 14(4):417–433, 1994. 41

[AT25] Albert Atserias and Iddo Tzameret. Feasibly constructive proof of schwartz-zippel lemma and the complexity of finding hitting sets. In *57th ACM STOC*, pages 1096–1107. ACM Press, June 2025. 6, 10

[AW09] Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *ACM Trans. Comput. Theory*, 1(1):2:1–2:54, 2009. 42

[BBK+23] Zvika Brakerski, Maya Farber Brodsky, Yael Tauman Kalai, Alex Lombardi, and Omer Paneth. SNARGs for monotone policy batch NP. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part II*, volume 14082 of *LNCS*, pages 252–283. Springer, Cham, August 2023. 1, 8

[BCC+17] Nir Bitansky, Ran Canetti, Alessandro Chiesa, Shafi Goldwasser, Huijia Lin, Aviad Rubinstein, and Eran Tromer. The hunting of the SNARK. *Journal of Cryptology*, 30(4):989–1066, October 2017. 8

[BCCT13] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. Recursive composition and bootstrapping for SNARKS and proof-carrying data. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 111–120. ACM Press, June 2013. 8

[BCI+13] Nir Bitansky, Alessandro Chiesa, Yuval Ishai, Rafail Ostrovsky, and Omer Paneth. Succinct non-interactive arguments via linear interactive proofs. In Amit Sahai, editor, *TCC 2013*, volume 7785 of *LNCS*, pages 315–333. Springer, Berlin, Heidelberg, March 2013. 8

[BGI+01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 1–18. Springer, Berlin, Heidelberg, August 2001. 1

[BGS75] Theodore Baker, John Gill, and Robert Solovay. Relativizations of the p=?np question. *SIAM Journal on computing*, 4(4):431–442, 1975. 42

[BHK17] Zvika Brakerski, Justin Holmgren, and Yael Tauman Kalai. Non-interactive delegation and batch NP verification from standard computational assumptions. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *49th ACM STOC*, pages 474–482. ACM Press, June 2017. 8

[BKK+18] Saikrishna Badrinarayanan, Yael Tauman Kalai, Dakshita Khurana, Amit Sahai, and Daniel Wichs. Succinct delegation for low-space non-deterministic computation. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *50th ACM STOC*, pages 709–721. ACM Press, June 2018. 8

[BKO20] Jan Bydzovsky, Jan Krajíček, and Igor C. Oliveira. Consistency of circuit lower bounds with bounded theories. *Log. Methods Comput. Sci.*, 16(2), 2020. 10, 42

[BKT14] Samuel R. Buss, Leszek Aleksander Kolodziejczyk, and Neil Thapen. Fragments of approximate counting. *J. Symb. Log.*, 79(2):496–525, 2014. 5

[BM20] Jan Bydzovsky and Moritz Müller. Polynomial time ultrapowers and the consistency of circuit lower bounds. *Arch. Math. Log.*, 59(1-2):127–147, 2020. 10, 42

[Bus85] Samuel R Buss. *Bounded arithmetic*. Princeton University, 1985. 5, 23, 24, 45

[Bus87] Samuel R. Buss. Polynomial size proofs of the propositional pigeonhole principle. *J. Symb. Log.*, 52(4):916–927, 1987. 42

[CCH+19] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, Ron D. Rothblum, and Daniel Wichs. Fiat-Shamir: from practice to theory. In Moses Charikar and Edith Cohen, editors, *51st ACM STOC*, pages 1082–1090. ACM Press, June 2019. 8

[Cel22] Carlo Cellucci. The theory of gödel. *Synthese Library*, 2022. 5

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594, July 2004. 8

[CGJ+23] Arka Rai Choudhuri, Sanjam Garg, Abhishek Jain, Zhengzhong Jin, and Jiaheng Zhang. Correlation intractability and SNARGs from sub-exponential DDH. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part IV*, volume 14084 of *LNCS*, pages 635–668. Springer, Cham, August 2023. 1, 8

[CHO+22] Lijie Chen, Shuichi Hirahara, Igor Carboni Oliveira, Ján Pich, Ninad Rajgopal, and Rahul Santhanam. Beyond natural proofs: Hardness magnification and locality. *J. ACM*, 69(4):25:1–25:49, 2022. 42

[CJ25] Liyan Chen and Zhengzhong Jin. On the impossibility of snargs with short CRS : (or: Revisiting gentry-wichs barrier in the non-adaptive setting). In *66th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2025, Sydney, Australia, December 14-17, 2025*, pages 1741–1760. IEEE, 2025. 6

[CJJ21] Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. Non-interactive batch arguments for NP from standard assumptions. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 394–423, Virtual Event, August 2021. Springer, Cham. 1, 8

[CJJ22] Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. SNARGs for $\mathcal{P}$ from LWE. In *62nd FOCS*, pages 68–79. IEEE Computer Society Press, February 2022. 1, 7, 8

[CK07] Stephen A. Cook and Jan Krajíček. Consequences of the provability of $NP \subseteq P/poly$. *J. Symb. Log.*, 72(4):1353–1371, 2007. 10, 42

[CKK+25] Marco Carmosino, Valentine Kabanets, Antonina Kolokolova, Igor C. Oliveira, and Dimitrios Tsintsilidas. Provability of the circuit size hierarchy and its consequences. In Raghu Meka, editor, *16th Innovations in Theoretical Computer Science Conference, ITCS 2025, January 7-10, 2025, Columbia University, New York, NY, USA*, volume 325 of *LIPIcs*, pages 30:1–30:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025. 10

[CKKO21] Marco Carmosino, Valentine Kabanets, Antonina Kolokolova, and Igor C. Oliveira. Learn-uniform circuit lower bounds and provability in bounded arithmetic. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 770–780. IEEE, 2021. 7, 10, 14, 42

[CLO24] Lijie Chen, Jiatu Li, and Igor C. Oliveira. Reverse mathematics of complexity lower bounds. In *Symposium on Foundations of Computer Science* (FOCS), pages 505–527, 2024. 6, 10, 14, 45

[CLO25] Lijie Chen, Jiatu Li, and Igor C. Oliveira. On the unprovability of circuit size bounds in intuitionistic S^1_2. *Log. Methods Comput. Sci.*, 21(3), 2025. 10, 42

[CLOW26] Lijie Chen, Jiatu Li, Igor C Oliveira, and Ryan Williams. A theory for probabilistic polynomial-time reasoning. *STOC (to appear)*, 2026. 7

[CN10] Stephen Cook and Phuong Nguyen. *Logical foundations of proof complexity*, volume 31. Cambridge University Press Cambridge, 2010. 5, 6, 10

[Cob65] Alan Cobham. The intrinsic computational difficulty of functions. 1965. 24

[Coh63] Paul J Cohen. The independence of the continuum hypothesis. *Proceedings of the National Academy of Sciences*, 50(6):1143–1148, 1963. 2, 5

[Coh64] Paul J Cohen. The independence of the continuum hypothesis, ii. *Proceedings of the National Academy of Sciences*, 51(1):105–110, 1964. 2, 5

[Coo75] Stephen A. Cook. Feasibly constructive proofs and the propositional calculus (preliminary version). In William C. Rounds, Nancy Martin, Jack W. Carlyle, and Michael A. Harrison, editors, *Proceedings of the 7th Annual ACM Symposium on Theory of Computing, May 5-7, 1975, Albuquerque, New Mexico, USA*, pages 83–97. ACM, 1975. 1, 5, 9, 10, 24, 40, 41, 42, 45, 130

[CP90] Stephen A. Cook and Toniann Pitassi. A feasibly constructive lower bound for resolution proofs. *Inf. Process. Lett.*, 34(2):81–85, 1990. 10

[CR79] Stephen A Cook and Robert A Reckhow. The relative efficiency of propositional proof systems. *The journal of symbolic logic*, 44(1):36–50, 1979. 1, 2, 41, 130

[CRT25] Lijie Chen, Ron D. Rothblum, and Roei Tell. Fiat-shamir in the plain model from derandomization (or: Do efficient algorithms believe that NP = PSPACE?). In *57th ACM STOC*, pages 977–985. ACM Press, June 2025. 10, 45

[CS88] Vasek Chvátal and Endre Szemerédi. Many hard examples for resolution. *J. ACM*, 35(4):759–768, 1988. 41

[CU93] Stephen A. Cook and Alasdair Urquhart. Functional interpretations of feasibly constructive arithmetic. *Ann. Pure Appl. Log.*, 63(2):103–200, 1993. 5

[DGKV22] Lalita Devadas, Rishab Goyal, Yael Kalai, and Vinod Vaikuntanathan. Rate-1 non-interactive arguments for batch-NP and applications. In *63rd FOCS*, pages 1057–1068. IEEE Computer Society Press, October / November 2022. 8

[DHK+26] Lalita Devadas, Samuel B. Hopkins, Yael Tauman Kalai, Pravesh K. Kothari, Alex Lombardi, and Surya Mathialagan. SNARGs for NP and non-signaling PCPs, revisited. Cryptology ePrint Archive, Paper 2026/006, to appear at STOC 2026, 2026. 8

[GC25] Stefan Grosser and Marco Carmosino. Student-teacher constructive separations and (un)provability in bounded arithmetic: Witnessing the gap. In *57th ACM STOC*, pages 1341–1347. ACM Press, June 2025. 10

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 467–476. ACM Press, June 2013. 1

[GK15] Shafi Goldwasser and Yael Tauman Kalai. Cryptographic assumptions: A position paper. Cryptology ePrint Archive, Report 2015/907, 2015. 35

[Göd31] Kurt Gödel. Über formal unentscheidbare sätze der principia mathematica und verwandter systeme i. *Monatshefte für mathematik und physik*, 38(1):173–198, 1931. 2

[Gol11] Oded Goldreich. In a world of P=BPP. In Oded Goldreich, editor, *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation - In Collaboration with Lidor Avigad, Mihir Bellare, Zvika Brakerski, Shafi Goldwasser, Shai Halevi, Tali Kaufman, Leonid Levin, Noam Nisan, Dana Ron, Madhu Sudan, Luca Trevisan, Salil Vadhan, Avi Wigderson, David Zuckerman*, volume 6650 of *Lecture Notes in Computer Science*, pages 191–232. Springer, 2011. 27

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008. 31

[Gro10] Jens Groth. Short pairing-based non-interactive zero-knowledge arguments. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 321–340. Springer, Berlin, Heidelberg, December 2010. 1, 8

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 75–92. Springer, Berlin, Heidelberg, August 2013. 6, 10, 17, 18, 59, 63, 64, 129

[GW11] Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In Lance Fortnow and Salil P. Vadhan, editors, *43rd ACM STOC*, pages 99–108. ACM Press, June 2011. 6

[Hak85] Armin Haken. The intractability of resolution. *Theor. Comput. Sci.*, 39:297–308, 1985. 41

[Hås86] Johan Håstad. Almost optimal lower bounds for small depth circuits. In *18th ACM STOC*, pages 6–20. ACM Press, May 1986. 41

[HJKS22] James Hulett, Ruta Jawale, Dakshita Khurana, and Akshayaram Srinivasan. SNARGs for P from sub-exponential DDH and QR. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part II*, volume 13276 of *LNCS*, pages 520–549. Springer, Cham, May / June 2022. 1

[HW15] Pavel Hubacek and Daniel Wichs. On the communication complexity of secure function evaluation with long output. In Tim Roughgarden, editor, *ITCS 2015*, pages 163–172. ACM, January 2015. 10, 17, 18, 70

[Ila25] Rahul Ilango. Gödel in cryptography: Effectively zero-knowledge proofs for NP with no interaction, no setup, and perfect soundness. In *66th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2025, Sydney, Australia, December 14-17, 2025*, pages 1100–1127. IEEE, 2025. 3, 4, 9

[ILW23] Rahul Ilango, Jiatu Li, and R. Ryan Williams. Indistinguishability obfuscation, range avoidance, and bounded arithmetic. In Barna Saha and Rocco A. Servedio, editors, *55th ACM STOC*, pages 1076–1089. ACM Press, June 2023. 5, 10, 45

[Jeř05] Emil Jeřábek. *Weak pigeonhole principle, and randomized computation.* PhD thesis, Faculty of Mathematics and Physics, Charles University, Prague, 2005. 10, 80

[Jeř07a] Emil Jeřábek. Approximate counting in bounded arithmetic. *Journal of Symbolic Logic*, 72(3):959–993, 2007. 3, 5, 6, 7, 10, 12, 13, 23, 25, 27, 28, 49, 50

[Jeř07b] Emil Jeřábek. On independence of variants of the weak pigeonhole principle. *J. Log. Comput.*, 17(3):587–604, 2007. 5

[JJ22] Abhishek Jain and Zhengzhong Jin. Indistinguishability obfuscation via mathematical proofs of equivalence. In *63rd FOCS*, pages 1023–1034. IEEE Computer Society Press, October / November 2022. 7, 8, 9, 12

[JJ25] Abhishek Jain and Zhengzhong Jin. Sometimes-decryptable homomorphic encryption from sub-exponential DDH. In *CRYPTO 2025, Part III*, LNCS, pages 406–439. Springer, Cham, August 2025. 1, 8, 9

[JJMP25] Abhishek Jain, Zhengzhong Jin, Surya Mathialagan, and Omer Paneth. On succinct obfuscation via propositional proofs. In *66th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2025, Sydney, Australia, December 14-17, 2025*, pages 1703–1740. IEEE, 2025. 9, 12

[JKKZ21] Ruta Jawale, Yael Tauman Kalai, Dakshita Khurana, and Rachel Yun Zhang. SNARGs for bounded depth computations and PPAD hardness from sub-exponential LWE. In Samir Khuller and Virginia Vassilevska Williams, editors, *53rd ACM STOC*, pages 708–721. ACM Press, June 2021. 1, 8

[JKLM25] Zhengzhong Jin, Yael Tauman Kalai, Alex Lombardi, and Surya Mathialagan. Universal SNARGs for NP from proofs of correctness. In *57th ACM STOC*, pages 933–943. ACM Press, June 2025. 1, 8, 9, 12, 68, 72

[JKLV24] Zhengzhong Jin, Yael Kalai, Alex Lombardi, and Vinod Vaikuntanathan. SNARGs under LWE via propositional proofs. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *56th ACM STOC*, pages 1750–1757. ACM Press, June 2024. i, 1, 2, 3, 4, 6, 7, 8, 9, 11, 12, 17, 19, 20, 37, 88, 89, 92, 97, 99, 101, 105, 110, 130

[JLS21] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In Samir Khuller and Virginia Vassilevska Williams, editors, *53rd ACM STOC*, pages 60–73. ACM Press, June 2021. 8

[JLS22] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from LPN over $\mathbb{F}_p$, DLIN, and PRGs in NC^0. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part I*, volume 13275 of *LNCS*, pages 670–699. Springer, Cham, May / June 2022. 8

[Kil92] Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *24th ACM STOC*, pages 723–732. ACM Press, May 1992. 1

[KLV23] Yael Tauman Kalai, Alex Lombardi, and Vinod Vaikuntanathan. SNARGs and PPAD hardness from the decisional Diffie-Hellman assumption. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part II*, volume 14005 of *LNCS*, pages 470–498. Springer, Cham, April 2023. 8

[KLVW23] Yael Kalai, Alex Lombardi, Vinod Vaikuntanathan, and Daniel Wichs. Boosting batch arguments and RAM delegation. In Barna Saha and Rocco A. Servedio, editors, *55th ACM STOC*, pages 1545–1552. ACM Press, June 2023. 8

[KO17] Jan Krajíček and Igor C. Oliveira. Unprovability of circuit upper bounds in cook's theory PV. *Log. Methods Comput. Sci.*, 13(1), 2017. 10, 42

[Kor25] Oliver Korten. Range avoidance and the complexity of explicit constructions. *Bull. EATCS*, 145, 2025. 5, 12

[KP89] Jan Krajíček and Pavel Pudlák. Propositional proof systems, the consistency of first order theories and the complexity of computations. *J. Symb. Log.*, 54(3):1063–1079, 1989. 7, 9, 41, 44, 45

[KP98] Jan Krajíček and Pavel Pudlák. Some consequences of cryptographical conjectures fors12and ef. *Information and Computation*, 140(1):82–94, 1998. 10

[KP16] Yael Tauman Kalai and Omer Paneth. Delegating RAM computations. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 91–118. Springer, Berlin, Heidelberg, October / November 2016. 8

[KPW95] Jan Krajíček, Pavel Pudlák, and Alan Woods. An exponential lower bound to the size of bounded depth frege proofs of the pigeonhole principle. *Random structures & algorithms*, 7(1):15–39, 1995. 41

[KPY19] Yael Tauman Kalai, Omer Paneth, and Lisa Yang. How to delegate computations publicly. In Moses Charikar and Edith Cohen, editors, *51st ACM STOC*, pages 1115–1124. ACM Press, June 2019. 1, 8

[KR09] Yael Tauman Kalai and Ran Raz. Probabilistically checkable arguments. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 143–159. Springer, Berlin, Heidelberg, August 2009. 8

[Kra94] Jan Krajíček. Lower bounds to the size of constant-depth propositional proofs. *J. Symb. Log.*, 59(1):73–86, 1994. 41

[Kra95] Jan Krajicek. *Bounded arithmetic, propositional logic and complexity theory*, volume 60. Cambridge University Press, 1995. 5, 9, 10, 12, 23, 41, 45

[Kra97] Jan Krajíček. Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic. *J. Symb. Log.*, 62(2):457–486, 1997. 10

[Kra05] Jan Krajıcek. Hardness assumptions in the foundations of theoretical computer science. *Archive for Mathematical Logic*, 44(6):667–675, 2005. 2

[Kra11a] Jan Krajíček. *Forcing with Random Variables and Proof Complexity*, volume 382 of *London Mathematical Society lecture note series*. Cambridge University Press, 2011. 10

[Kra11b] Jan Krajíček. On the proof complexity of the nisan-wigderson generator based on a hard NP ∩ conp function. *J. Math. Log.*, 11(1), 2011. 10, 42

[Kra19] Jan Krajíček. *Proof complexity*, volume 170. Cambridge University Press, 2019. 5, 9, 10, 23, 24, 41, 42, 45, 46, 130

[Kra24] Jan Krajíček. On the existence of strong proof complexity generators. *Bulletin of Symbolic Logic*, 30(1):20–40, 2024. 10

[Kra25] Jan Krajíček. *Proof Complexity Generators*. London Mathematical Society Lecture Note Series. Cambridge University Press, 2025. 10, 42

[KRR14] Yael Tauman Kalai, Ran Raz, and Ron D. Rothblum. How to delegate computations: the power of no-signaling proofs. In David B. Shmoys, editor, *46th ACM STOC*, pages 485–494. ACM Press, May / June 2014. 8

[KVZ21] Yael Tauman Kalai, Vinod Vaikuntanathan, and Rachel Yun Zhang. Somewhere statistical soundness, post-quantum security, and SNARGs. In Kobbi Nissim and Brent Waters, editors, *TCC 2021, Part I*, volume 13042 of *LNCS*, pages 330–368. Springer, Cham, November 2021. 1

[Le19] Dai Tri Man Le. *Bounded Arithmetic and Formalizing Probabilistic Proofs*. PhD thesis, University of Toronto, Canada, 2019. 6, 10

[Li25] Jiatu Li. An introduction to feasible mathematics and bounded arithmetic for computer scientists. *Electron. Colloquium Comput. Complex.*, TR25-086, 2025. 24, 75

[Lip12] Helger Lipmaa. Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments. In Ronald Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 169–189. Springer, Berlin, Heidelberg, March 2012. 8

[LLL82] AK Lenstra, HW Lenstra, and L Lovász. Factoring polynomials with rational coefficients. *Math. ann*, 261(4):515–534, 1982. 26

[LLR24] Jiawei Li, Yuhao Li, and Hanlin Ren. Metamathematics of resolution lower bounds: A TFNP perspective. *Electron. Colloquium Comput. Complex.*, TR24-190, 2024. 10

[LO23] Jiatu Li and Igor C. Oliveira. Unprovability of strong complexity lower bounds in bounded arithmetic. In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 1051–1057. ACM, 2023. 7, 10, 14, 42, 45

[LP21] Yanyi Liu and Rafael Pass. Cryptography from sublinear-time average-case hardness of time-bounded kolmogorov complexity. In Samir Khuller and Virginia Vassilevska Williams, editors, *53rd ACM STOC*, pages 722–735. ACM Press, June 2021. 9

[MDS25] Yaohua Ma, Chenxin Dai, and Elaine Shi. Quasi-linear indistinguishability obfuscation via mathematical proofs of equivalence and applications. In Serge Fehr and Pierre-Alain Fouque, editors, *EUROCRYPT 2025, Part III*, volume 15603 of *LNCS*, pages 157–186. Springer, Cham, May 2025. 9

[Mic94] Silvio Micali. CS proofs (extended abstracts). In *35th FOCS*, pages 436–453. IEEE Computer Society Press, November 1994. 1, 8

[MP20] Moritz Müller and Ján Pich. Feasibly constructive proofs of succinct weak circuit lower bounds. *Ann. Pure Appl. Log.*, 171(2), 2020. 6, 10, 14

[NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994. 12

[NWW24] Shafik Nassar, Brent Waters, and David J. Wu. Monotone policy BARGs from BARGs and additively homomorphic encryption. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024, Part II*, volume 15365 of *LNCS*, pages 399–430. Springer, Cham, December 2024. 1

[Oja04] Kerry Ojakian. *Combinatorics in bounded arithmetic*. Carnegie Mellon University, 2004. 10

[Oli25] Igor C. Oliveira. SIGACT news complexity theory column 124 meta-mathematics of computational complexity theory. *SIGACT News*, 56(1):41–68, 2025. 2, 7, 10, 23, 42, 44

[Par71] Rohit Parikh. Existence and feasibility in arithmetic. *J. Symb. Log.*, 36(3):494–508, 1971. 5

[PBI93] Toniann Pitassi, Paul Beame, and Russell Impagliazzo. Exponential lower bounds for the pigeonhole principle. *Comput. Complex.*, 3:97–140, 1993. 41

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 333–342. ACM Press, May / June 2009. 26

[PH77] Jeff Paris and Leo Harrington. A mathematical incompleteness in Peano arithmetic. In *Handbook of Mathematical Logic*, pages 1133–1142. Amsterdam: North-Holland, 1977. 5

[Pic15a] Ján Pich. Circuit lower bounds in bounded arithmetics. *Ann. Pure Appl. Log.*, 166(1):29–45, 2015. 10, 42

[Pic15b] Ján Pich. Logical strength of complexity theory and a formalization of the PCP theorem in bounded arithmetic. *Log. Methods Comput. Sci.*, 11(2), 2015. 6, 42

[PP22] Omer Paneth and Rafael Pass. Incrementally verifiable computation via rate-1 batch arguments. In *63rd FOCS*, pages 1045–1056. IEEE Computer Society Press, October / November 2022. 8

[PR17] Omer Paneth and Guy N. Rothblum. On zero-testable homomorphic encryption and publicly verifiable non-interactive arguments. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part II*, volume 10678 of *LNCS*, pages 283–315. Springer, Cham, November 2017. 19

[PS21] Ján Pich and Rahul Santhanam. Strong co-nondeterministic lower bounds for NP cannot be proved feasibly. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 223–233. ACM, 2021. 7, 10, 42, 45

[Pud97] Pavel Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. *The Journal of Symbolic Logic*, 62(3):981–998, 1997. 41

[PWW88] Jeff B. Paris, A. J. Wilkie, and Alan R. Woods. Provability of the pigeonhole principle and the existence of infinitely many primes. *J. Symb. Log.*, 53(4):1235–1244, 1988. 10

[Raz95a] Alexander A Razborov. Bounded arithmetic and lower bounds in boolean complexity. In *Feasible Mathematics II*, pages 344–386. Springer, 1995. 10

[Raz95b] Alexander A Razborov. Unprovability of lower bounds on circuit size in certain fragments of bounded arithmetic. *Izvestiya: mathematics*, 59(1):205, 1995. 10

[Raz01] Alexander A. Razborov. Proof complexity of pigeonhole principles. In Werner Kuich, Grzegorz Rozenberg, and Arto Salomaa, editors, *Developments in Language Theory, 5th International Conference, DLT 2001, Vienna, Austria, July 16-21, 2001, Revised Papers*, volume 2295 of *Lecture Notes in Computer Science*, pages 100–116. Springer, 2001. 41

[Raz15] Alexander A Razborov. Pseudorandom generators hard for k-dnf resolution and polynomial calculus resolution. *Annals of Mathematics*, pages 415–472, 2015. 2, 40, 42

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005. 26

[RR97] Alexander A. Razborov and Steven Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997. 42, 43

[Rud97] Steven Rudich. Super-bits, demi-bits, and np/qpoly-natural proofs. In José D. P. Rolim, editor, *Randomization and Approximation Techniques in Computer Science, International Workshop, RANDOM'97, Bolognna, Italy, July 11-12. 1997, Proceedings*, volume 1269 of *Lecture Notes in Computer Science*, pages 85–93. Springer, 1997. 42, 43

[RVV24] Seyoon Ragavan, Neekon Vafa, and Vinod Vaikuntanathan. Indistinguishability obfuscation from bilinear maps and LPN variants. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024, Part IV*, volume 15367 of *LNCS*, pages 3–36. Springer, Cham, December 2024. 8

[Sup72] Patrick Suppes. *Axiomatic set theory*. Courier Corporation, 1972. 5

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In David B. Shmoys, editor, *46th ACM STOC*, pages 475–484. ACM Press, May / June 2014. 1, 8

[Tha05] Neil Thapen. Structures interpretable in models of bounded arithmetic. *Ann. Pure Appl. Log.*, 136(3):247–266, 2005. 12

[Urq87] Alasdair Urquhart. Hard examples for resolution. *J. ACM*, 34(1):209–219, 1987. 41

[Woo81] Alan Woods. Some problems in logic and number theory and their connections. *Ph. D. Thesis, Univ. of Manchester*, 1981. 10

[WW22] Brent Waters and David J. Wu. Batch arguments for NP and more from standard bilinear group assumptions. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part II*, volume 13508 of *LNCS*, pages 433–463. Springer, Cham, August 2022. 1, 3, 6, 7, 8, 10, 17, 18, 76, 79

[WW24] Brent Waters and David J. Wu. Adaptively-sound succinct arguments for NP from indistinguishability obfuscation. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *56th ACM STOC*, pages 387–398. ACM Press, June 2024. 1, 8, 26

[WW25] Brent Waters and David J. Wu. A pure indistinguishability obfuscation approach to adaptively-sound SNARGs for NP. In *CRYPTO 2025, Part VII*, LNCS, pages 292–326. Springer, Cham, August 2025. 8, 129

[WZ24] Brent Waters and Mark Zhandry. Adaptive security in SNARGs via iO and lossy functions. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part X*, volume 14929 of *LNCS*, pages 72–104. Springer, Cham, August 2024. 8

A List of Internal Theorems in Bounded Arithmetic

In this section, we provide a list of important internal theorems in bounded arithmetic that we formalized for future references categorized in four types: basic properties of CAPP (which is used to formalize probability), basic probability theory, cryptographic tools, and security analysis of concrete cryptographic constructions. For each theorem, we provide an informal explanation and the theory to formalize it.

A.1 Infrastructure: Consistency of CAPP

Precision Consistency of CAPP (Proposition 7.1) *(in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$*)*

The acceptance probability of a circuit changes by only a small amount when computed with different precision parameters.

Global Consistency of CAPP (Proposition 7.2) *(in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$*)*

Two circuits computing the same Boolean function have essentially the same acceptance probability.

Complementation Consistency of CAPP (Proposition 7.3) *(in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$*)*

The acceptance probabilities of a circuit and its negation sum essentially to one.

Monotonicity of CAPP (Proposition 7.4) *(in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$*)*

If acceptance of one circuit always implies acceptance of another, the first has at most the acceptance probability of the second.

A.2 Probability Theory

Union Bound (Proposition 7.5) *(in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$*)*

The acceptance probability of the disjunction of two circuits is essentially at most the sum of their individual probabilities.

Union Bound (general form) (Proposition 7.6) *(in $\mathsf{APC}_1 + \text{“prBPP} = \text{prP”}$)*

The union bound generalized to disjunctions of $m \in \mathsf{Log}$ circuits.

Averaging Argument on Two Circuits (Lemma 7.9) *(in $\mathsf{APC}_1 + \text{“prBPP} = \text{prP”}$)*

If two circuits differ in their acceptance probabilities, then some assignment to a chosen subset of input variables preserves (most of) that difference.

Averaging Argument (Corollary 7.10) *(in $\mathsf{APC}_1 + \text{“prBPP} = \text{prP”}$)*

For a single circuit, there is a partial assignment to a chosen variable subset such that, after applying the assignment, the acceptance probability is essentially at most (or at least) the original acceptance probability.

Averaging Argument for Indistinguishable Distribution (Lemma 7.11) *(in $\mathsf{APC}_1 + \text{“prBPP} = \text{prP”}$)*

If two distributions are indistinguishable under every partial assignment to a chosen subset of seed bits, they are indistinguishable overall.

$\mathbb{Z}_p$ Sampling Lemma (Lemma 4.23) *(in $\mathsf{APC}_1 + \text{“prBPP} = \text{prP”}$)*

An efficient sampler produces an output statistically close to uniform on $\mathbb{Z}_p$ from a uniform binary seed, for any binary integer p.

$\mathbb{Z}_p$ Shift-Invariance Lemma (Lemma 4.24) *(in $\mathsf{APC}_1 + \text{“prBPP} = \text{prP”}$)*

Shifting the $\mathbb{Z}_p$-sampler's output by any fixed element of $\mathbb{Z}_p$ yields a statistically close distribution.

$\mathbb{Z}_p$ Multiplicative-Inverse Lemma (Lemma 10.7) *(in PV_1)*

Multiplicative inverses in $\mathbb{Z}_p$ for prime p are computable by a PV function with provable correctness.

A.3 Cryptographic Tools

Hybrid Argument (Lemma 4.15) *(in $\mathsf{APC}_1 + \text{“prBPP} = \text{prP”}$)*

In a chain of distributions where every two consecutive ones are indistinguishable, the first and the last are also indistinguishable.

Reduction Lemma (Lemma 7.12) *(in $\mathsf{APC}_1 + \text{“prBPP} = \text{prP”}$)*

Applying a fixed circuit to two indistinguishable distributions yields indistinguishable distributions, at the cost of including the reduction's size in the adversary size bound.

Isomorphism Lemma (Lemma 4.14) *(in $\mathsf{APC}_1 + \text{“prBPP} = \text{prP”}$)*

Distributions that are statistically close via a circuit-witnessed bijection are computationally indistinguishable for any adversary.

Independent Side-Information Corollary (Corollary 7.13) *(in $\mathsf{APC}_1 + \text{“prBPP} = \text{prP”}$)*

Concatenating an independent distribution to two indistinguishable distributions preserves indistinguishability.

Product Hybrid Lemma (Corollary 7.14) *(in $\mathsf{APC}_1 + \text{“prBPP} = \text{prP”}$)*

Two pairs of independently indistinguishable distributions remain indistinguishable when paired together as products, with errors summed.

Mixture Lemma (Lemma 7.23) *(in $\mathsf{APC}_1 + \text{“prBPP} = \text{prP”}$)*

Mixing two distributions according to a circuit that accepts almost every input gives a distribution close to the first.

Game Complement Lemma (Lemma 7.15) *(in* APC_1 + "prBPP = prP"*)*
An adversary's advantages in a search game and in its complement game sum essentially to one.

Game Composition Lemma (Lemma 7.16) *(in* APC_1 + "prBPP = prP"*)*
Replacing the challenge sampler in a search game by an indistinguishable one preserves the advantage of the adversary up to the indistinguishability error.

Adversary Indistinguishability Lemma (Lemma 7.17) *(in* APC_1 + "prBPP = prP"*)*
Replacing the adversary in a search game by an indistinguishable one preserves its advantage up to the indistinguishability error.

Game Union Bound Lemma (Lemma 7.18) *(in* APC_1 + "prBPP = prP"*)*
The disjunction of secure search games is secure with security parameter equal to the sum of those for all games.

Game Reduction Lemma (Lemma 7.19) *(in* APC_1 + "prBPP = prP"*)*
If winning one search game implies winning a second via a fixed mapping of seeds, then the advantage in the second game is at least the advantage in the first.

Error Reduction Lemma (Lemma 7.22) *(in* APC_1 + "prBPP = prP"*)*
Running an adversary independently on k-fold parallel repetitions of a search game amplifies its advantage in the standard way.

A.4 Cryptographic Theorems

Leveled GSW Security Theorem (Theorem 8.10) *(in* APC_1 + "prBPP = prP"*)*
The secret-key variant of the leveled Gentry-Sahai-Waters scheme [GSW13] is a semantically secure leveled FHE, assuming LWE.

SEH Opening Completeness Theorem (Theorem 9.5) *(in* PV_1*)*
The construction of SEH from FHE is complete: every honestly produced opening is accepted.

SEH Extraction Correctness Theorem (Theorem 9.6) *(in* PV_1*)*
Assuming FHE correctness and malicious gate correctness, the SEH construction from FEH is somewhere extractable.

SEH Security Theorem (Theorem 9.7) *(in* APC_1 + "prBPP = prP" + FHESecure*)*
The SEH construction is computationally hiding, assuming FHE semantic security.

Parallel SXDH Lemma (Lemma 10.11) *(in* APC_1 + "prBPP = prP"*)*
The SXDH assumption implies its k-fold version (with a polynomial security loss).

WW BARG Correctness Theorem (Theorem 10.9) *(in* PV_1*)*
Assuming the group underlying SXDH is correctly generated ($\mathsf{Cor}_{\mathsf{GroupGen}}$), the honest prover of the Waters-Wu BARG [WW25] is always accepted.

WW BARG Somewhere Extractability Theorem (Theorem 10.10) *(in* PV_1*)*
Assuming the underlying group is correctly generated, the Waters-Wu BARG is somewhere extractable.

WW BARG CRS Indistinguishability Theorem (Theorem 10.12) *(in* APC_1 + "prBPP = prP"*)*
Waters-Wu BARG is CRS indistinguishable assuming SXDH.

JKLV Security Theorem (Theorem 11.21) *(in* $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$*)*

The JKLV SNARG construction [JKLV24] is sound, assuming LWE and SXDH.

B Extended Frege with Constant Width

The definition of Extended Frege can be found in [CR79] as well as standard textbooks (see, e.g., [Kra19]). In this work, it will be easier to work with Extended Frege system of constant *width*, i.e., each proof line is of constant size. This is without loss of generality, as any Extended Frege proof can be transformed into an essentially equivalent Extended Frege proof with constant width, and the transformation is provably correct in PV. For completeness, we describe the transformation and sketch the proof.

For simplicity, we consider the set of connectives $\{\to, \bot\}$, where $\bot$ denotes false and $\to$ is right associative. We define $\neg\varphi$ as an abbreviation of $\varphi \to \bot$, $\varphi \vee \psi$ as an abbreviation of $\neg\varphi \to \psi$, $\varphi \wedge \psi$ as an abbreviation of $\neg(\varphi \to \neg\psi)$. We consider Modus Ponens $\varphi \to \psi, \varphi \vdash \psi$ as the only derivation rule, and fix the following complete and sound set of axioms:

- $\alpha \to \beta \to \alpha$
- $(\gamma \to \alpha \to \beta) \to (\gamma \to \alpha) \to \gamma \to \beta$.
- $\neg\neg\alpha \to \alpha$

Note that the choices of connectives and axioms do not affect the strength of the system up to a polynomial size overhead, and this is provable in PV (see, e.g., [CR79, Coo75]). The *width* of an Extended Frege proof is the maximum size of proof lines in the proof.

Definition B.1 (Cook-Levin representation of formulas)**.** Let φ be any propositional formula. The Cook-Levin representation of φ, denoted by $[\varphi]_{\mathsf{CL}}$, is a pair $(\langle\Gamma_\varphi\rangle, z_\varphi)$ for a sequence Γ_φ of constant-size formula and a variable z_φ. It is inductively defined as follows.

- $[\bot]_{\mathsf{CL}} = (\langle z \leftrightarrow \bot\rangle, z)$.
- $[x]_{\mathsf{CL}} = (\langle\rangle, x)$.
- $[\varphi \to \psi]_{\mathsf{CL}} = (\langle\Gamma_\varphi, \Gamma_\psi, z \leftrightarrow (z_\varphi \to z_\psi)\rangle, z)$, where $[\varphi]_{\mathsf{CL}} = (\langle\Gamma_\varphi\rangle, z_\varphi)$, $[\psi]_{\mathsf{CL}} = (\langle\Gamma_\psi\rangle, z_\psi)$, and z is a fresh variable that is not used in $\Gamma_\varphi, \Gamma_\psi, z_\varphi, z_\psi$.

The variables $z, z_\varphi, z_\psi, \ldots$ introduced are called *auxiliary variables*.

In other words, we construct a new variable for each subformula of φ, and add appropriate constraints to the variables in Γ_φ, each of constant size, that correspond to the evaluation of the formula. The following proposition is left as an exercise.

Proposition B.2. PV_1 *proves the following statement: For every formula* φ*, let* $[\varphi]_{\mathsf{CL}} = (\langle\Gamma_\varphi\rangle, z_\varphi)$*, then* φ *is a tautology if and only if* $\bigwedge\Gamma_\varphi \to z_\varphi$ *is a tautology.*

We need the following lemma.

Lemma B.3. *There is a constant* $c \geq 1$ *and a* PV *function* T *such that* PV_1 *proves the following. Let* φ *be a formula and* $\tau = (\langle\Gamma_\varphi\rangle, z_\varphi)$*,* $\tau' = (\langle\Gamma'_\varphi\rangle, z'_\varphi)$ *be two Cook-Levin representation of formulas (i.e. identical up to a change of variable names). Then* $T(\varphi, \tau, \tau')$ *outputs an Extended Frege proof from* $\Gamma_\varphi, \Gamma'_\varphi$ *to* $z_\varphi \leftrightarrow z'_\varphi$ *of width* c*.*

Proof Sketch. The algorithm T is a recursive algorithm that works as follows. Suppose that $\varphi = \bot$ or $\varphi = x$ for a variable x, there is a constant size proof; in these cases, it suffices to choose c to be larger than the width of the proofs.

Consider the case that φ is of form $\alpha \to \beta$. By the definition of $[\cdot]_{\mathsf{CL}}$, we know that

- $\Gamma_\varphi = \Gamma_\alpha, \Gamma_\beta, z_\varphi \leftrightarrow (z_\alpha \to z_\beta)$, where $(\langle \Gamma_\alpha \rangle, z_\alpha)$ and $(\langle \Gamma_\beta \rangle, z_\beta)$ are Cook-Levin transformations of α and β, respective.
- $\Gamma'_\varphi = \Gamma'_\alpha, \Gamma'_\beta, z'_\varphi \leftrightarrow (z'_\alpha \to z'_\beta)$, where $(\langle \Gamma_\alpha \rangle, z_\alpha)$ and $(\langle \Gamma_\beta \rangle, z_\beta)$ are Cook-Levin transformations of α and β, respective.

The algorithm first recursively generates proofs of $z_\alpha \leftrightarrow z'_\alpha$, $z_\beta \leftrightarrow z'_\beta$ from $\Gamma_\alpha, \Gamma'_\alpha, \Gamma_\beta, \Gamma'_\beta$. Moreover, there is a constant size Extended Frege proof of

$$\frac{z_\alpha \leftrightarrow z'_\alpha \quad z_\beta \leftrightarrow z'_\beta \quad z_\varphi \leftrightarrow (z_\alpha \to z_\beta) \quad z'_\varphi \leftrightarrow (z'_\alpha \to z'_\beta)}{z_\varphi \leftrightarrow z'_\varphi}.$$

Note that the size is an absolute constant as there are only six variables, so we can set c to be larger than the width of the proof. Combining the proofs above, we can obtain an Extended Frege proof of $z_\varphi \leftrightarrow z'_\varphi$ from $\Gamma_\varphi, \Gamma'_\varphi$ of width at most c.

The correctness proof of the algorithm follows by induction on the structure of φ. This is available in PV_1, as the induction property is specified by a straightforward polynomial-time algorithm that checked the correctness of the proof generated by T. □

Now we are ready to state the transformation of arbitrary Extended Frege proofs into constant width Extended Frege proofs.

Lemma B.4. *There is a constant $c \geq 1$ and a PV function G such that the following is provable in PV_1. Let φ be a formula and τ be an Extended Frege proof of φ. Then $G(\varphi, \tau)$ outputs an Extended Frege proof from Γ_φ to z_φ of width at most c, where $(\langle \Gamma_\varphi \rangle, z_\varphi)$ is a Cook-Levin representation of φ.*

Proof Sketch. The algorithm G translates the proof τ line by line. In more detail, let $\tau = \varphi_1, \varphi_2, \ldots, \varphi_\ell$, in the i-th round, the algorithm produces an Extended Frege proof of width c from Γ_{φ_i} to z_{φ_i}, where $(\langle \Gamma_{\varphi_i} \rangle, z_{\varphi_i})$ is a Cook-Levin representation of φ_i. For simplicity, we assume that all Cook-Levin representations use disjoint set of auxiliary variables.

It starts at an empty proof. Let $(\langle \Gamma_{\varphi_i} \rangle, z_{\varphi_i})$ be a Cook-Levin representation of φ_i. In the i-th round, the algorithm considers how the proof line φ_i is introduced in the Extended Frege proof τ.

- (*Axiom*). Suppose that φ_i is introduced as an axiom. We only consider $\varphi =$ "$\alpha \to \beta \to \alpha$". In the $[\cdot]_{\mathsf{CL}}$ transformation of φ, there are variables $z^1_\alpha, z^2_\alpha, z_{\beta\to\alpha}, z_\beta$, corresponding to the first and second occurrences of α, $\beta \to \alpha$, and β in φ. By the definition of Cook-Levin representation, there are formulas
$$z_{\beta\to\alpha} \leftrightarrow (z_\beta \to z^2_\alpha) \quad \text{and} \quad z_{\varphi_i} \leftrightarrow (z^1_\alpha \to z_{\beta\to\alpha})$$
in Γ_{φ_i}. Moreover, by Lemma B.3, we can derive an Extended Frege proof of $z^1_\alpha \leftrightarrow z^2_\alpha$ when c is sufficiently large. Notice that there is a constant size proof of
$$\frac{z_{\beta\to\alpha} \leftrightarrow (z_\beta \to z^2_\alpha) \quad z_{\varphi_i} \leftrightarrow (z^1_\alpha \to z_{\beta\to\alpha}) \quad z^1_\alpha \leftrightarrow z^2_\alpha}{z_{\varphi_i}}.$$
By setting c to be sufficient large, we can combine the proofs above to obtain a proof from Γ_{φ_i} to z_{φ_i} of width at most c.

- (*Extension*). The case for extension rules is similar to that of axioms, and is omitted.

- (*Modus Ponens*). Suppose that φ_i is introduced via the Modus Ponens rule, where $\alpha \to \beta$ and α are available in $\varphi_1, \ldots, \varphi_{i-1}$. Let $(\langle \Gamma_\alpha \rangle, z_\alpha)$, $(\langle \Gamma_\beta \rangle, z_\beta)$, and $(\langle \Gamma_{\alpha\to\beta} \rangle, z_{\alpha\to\beta})$ be Cook-Levin representations of α, β, and $\alpha \to \beta$. By the invariant of the algorithm, we have produced width-c Extended Frege proofs:

  - $\Gamma_{\alpha\to\beta} \vdash z_{\alpha\to\beta}$;
  - $\Gamma_\alpha \vdash z_\alpha$.

  Note that by definition, $\Gamma_{\alpha\to\beta} = \Gamma_\alpha, \Gamma_\beta, z_{\alpha\to\beta} \leftrightarrow (z_\alpha \to z_\beta)$. Our goal is to prove z_β from Γ_β.

  The proof is as follows. Assume without loss of generality that $(\langle \Gamma_\alpha \rangle, z_\alpha)$ and $(\langle \Gamma_\beta \rangle, z_\beta)$ use disjoint set of variable variables. Then by applying the extension rules, we can write proof lines Γ_α and $z_{\alpha\to\beta} \leftrightarrow (z_\alpha \to z_\beta)$, so all formulas in $\Gamma_{\alpha\to\beta}$ are written down. Using the width-c proofs $\Gamma_{\alpha\to\beta} \vdash z_{\alpha\to\beta}$ and $\Gamma_\alpha \vdash z_\alpha$, we can further write down formulas $z_{\alpha\to\beta}$ and z_α. Finally, observe that there is a constant size proof of

  $$\frac{z_{\alpha\to\beta} \quad z_\alpha \quad z_{\alpha\to\beta} \leftrightarrow (z_\alpha \to z_\beta)}{z_\beta},$$

  and we can set c to be sufficient large so that the entire proof has width as most c.

The correctness proof of the algorithm follows by induction on the length of the proof τ. This is available in PV_1, as the induction property is specified by a straightforward polynomial-time algorithm that checked the correctness of the proof generated by G. □