Incrementally Verifiable Computation without Extraction

Abhishek Jain* Surya Mathialagan† Brent Waters‡

Abstract

Incrementally verifiable computation (IVC) [Valiant, TCC '08] allows one to iteratively prove that a configuration x_0 reaches a configuration x_T via T repeated applications of a (possibly non-deterministic) machine $\mathcal{M}$. An IVC scheme is *fully succinct* if the proof size is independent of both T and the size of the intermediate configurations.

In this work, we develop a new indistinguishability obfuscation ($i\mathcal{O}$)-based approach to IVC that avoids the *extraction-based* security analyses central to prior constructions. Assuming subexponential hardness of $i\mathcal{O}$ and one-way functions, we construct an adaptively sound fully succinct IVC scheme for deterministic computations. This yields the first IVC for deterministic computations that does *not* rely on algebraic assumptions.

Under the same assumptions, we further obtain a fully succinct *two-hop* IVC scheme for NP with non-adaptive soundness, allowing one to prove that x_0 reaches x_2 via an intermediate configuration x_1. This is the first IVC scheme for NP achieving full succinctness.

Our constructions are based on a new connection between IVC and *secret sharing* for s-t connectivity in graphs.

*NTT Research and Johns Hopkins
†NTT Research
‡NTT Research and UT Austin

Contents

1 Introduction **3**
- 1.1 Our Results . . . 4

2 Technical Overview **5**
- 2.1 Detour: Secret Sharing for Undirected s-t Connectivity . . . 6
- 2.2 Analysis Inspired by Secret Sharing . . . 7
- 2.3 Construction and Analysis . . . 9
- 2.4 Related Work . . . 11

3 Preliminaries **12**
- 3.1 Pseudorandom Generators . . . 13
- 3.2 Puncturable Pseudorandom Functions . . . 13
- 3.3 Indistinguishability Obfuscation . . . 14
- 3.4 Incrementally Verifiable Computation . . . 14

4 Fully Succinct IVC for P via $i\mathcal{O}$ **16**
- 4.1 Proof of Theorem 4.2 . . . 22
- 4.2 Proof of Theorem 4.3 . . . 25
- 4.3 Proof of Theorem 4.5 . . . 30

5 Two-Hop IVC for NP with Non-Adaptive Soundness **32**
- 5.1 Proof of Theorem 5.2 . . . 39
- 5.2 Proof of Theorem 5.3 . . . 42
- 5.3 Proof of Theorem 5.4 . . . 44

6 References **47**

1 Introduction

Incrementally verifiable computation (IVC) [Val08] allows one to incrementally verify the correctness of a long computation, showing that a starting configuration x_0 is transformed into x_T by T repeated applications of a (possibly non-deterministic) transition function $\mathcal{M}$. The key efficiency requirements are that the prover runtime (per update), proof size, and verification time should be sublinear in T. IVC has a wide range of applications, including enforcing language semantics [CTV13], authenticating images [NT16], and blockchains [BMRS20, KS22, CCDW20], among others.

IVC via Extraction. IVC generalizes the notion of succinct non-interactive arguments (SNARGs) [Mic94]; in particular, SNARGs for NP can be viewed as 1-step IVC for NP. Indeed, Valiant's original proposal for constructing IVC relies on recursive composition of SNARGs: Given a SNARG proof π_1 that x_0 reaches x_1 after t_1 (deterministic) steps, and a proof π_2 that x_1 reaches x_2 in t_2 steps, one can "merge" π_1 and π_2 to obtain a proof π for the statement x_0 reaches x_2 in $t_1 + t_2$ steps. This proof merging process can be applied recursively, for example, over a binary tree structure. The soundness of this approach is argued via a recursive *extraction* procedure, where given a final proof, we recursively extract intermediate "merged" proofs until we arrive at the "base" layer, where we can derive a contradiction by observing a cheating proof for an invalid step of computation.

Earlier works on IVC implemented Valiant's template via SNARGs of knowledge, i.e., SNARKs [BCCT13, BCTV14, COS20, KST22, BCMS20, BCL+21, BDFG21, CCG+23]. However, SNARKs are only known from non-standard assumptions or in non-standard models [CGKS23, BCG25]. A recent line of work has implemented Valiant's strategy in the *standard model* by replacing the use of SNARKs with rate-1 *somewhere-extractable* batch arguments (BARGs) [DGKV22, PP22, DJJ+25]. In a nutshell, a rate-1 BARG allows one to prove the validity of multiple NP statements via a short proof of size proportional to a single witness size. The somewhere-extractability property allows extraction of a witness for one of the statements from the proof, given a secret trapdoor.

The works of [DGKV22, PP22] constructed IVC for deterministic computations assuming rate-1 somewhere-extractable BARGs. Subsequently, Datta et al. [DJJ+25] extended these works to obtain IVC for NP by additionally relying on indistinguishability obfuscation ($i\mathcal{O}$).

Pitfalls of Extraction. We highlight two limitations of the extraction-based proof strategy employed by prior works. We focus on results in the standard model.

First, all of these results rely on algebraic assumptions such as learning with errors (LWE) or subexponential Decisional Diffie-Hellman (DDH). This is because BARGs with the somewhere-extractability property are only known from such assumptions. Note that, in contrast, (adaptively sound) SNARGs for NP are known from a "pure iO" approach [SW14, WW24, WW25], namely, assumptions that do not imply cryptographic primitives with the somewhere-extractability property [AS15, BDV17].

Second, for the case of NP, the proof size in the scheme of Datta et al. [DJJ+25] grows polynomially with the size of the intermediate configurations x_i. This is because their extraction-based proof strategy requires extracting an entire intermediate configuration to derive a contradiction.[1] Once again, this should be contrasted with SNARGs for NP where we know how to achieve proof size independent of the configuration (i.e., statement) size [SW14, WW24].

We ask whether there exist *fully succinct* IVC where the proof size is independent of the intermediate configuration size. Datta et al. construct such an IVC scheme for a subclass of NP languages

[1] For the case of deterministic computations, this issue can be bypassed by hashing the intermediate configurations using the notion of RAM delegation (see Section 2.4). This approach, however, does not extend to NP.

where there exists an efficient trapdoor circuit for deciding reachability of any intermediate configuration. This, however, is a limited class that does not even capture deterministic computations.

1.1 Our Results

In this work, we devise a new approach for constructing (fully succinct) IVC where the soundness analysis does *not* rely on recursive extraction. In particular, ours is a "pure iO" approach relying on assumptions that do not imply any cryptographic primitive with the somewhere-extractability property.

Fully Succinct IVC for P. Our first result is a construction of fully succinct IVC for deterministic computations based on subexponentially secure $i\mathcal{O}$ and one-way functions. Our construction achieves adaptive soundness.

Theorem 1.1. *Consider an IVC language corresponding to a deterministic machine $\mathcal{M}$ that takes as input configurations of length n. Assuming subexponentially secure $i\mathcal{O}$ and one-way functions, there exists an adaptively sound IVC for $\mathcal{M}$ with a common reference string of size $\mathsf{poly}(\lambda, n)$ and proofs of size $\mathsf{poly}(\lambda)$, where λ is the security parameter.*

As discussed earlier, while fully succinct IVC for P was known, ours is the first construction that does not rely on algebraic assumptions such as LWE or DDH.

Fully Succinct Two-Hop IVC for NP. We next investigate a natural intermediate goal towards the problem of fully succinct IVC for NP. Specifically, we consider *two-hop* fully succinct IVC, where a prover can prove that some configuration x_0 reaches x_2 via some intermediate configuration x_1. Intuitively, this problem can be viewed as a SNARG for NP that supports a *single* update.

We believe that this problem, despite its seeming simplicity at first glance, already captures a key challenge in achieving full succinctness. To see this, note that without the full succinctness requirement, two-hop IVC can be easily constructed by simply concatenating two separately computed proofs for each hop. This strategy, however, does *not* work in the fully succinct setting. Indeed, to verify such a proof, one needs access to the statement for each of the two proofs, which includes the intermediate configuration. Thus, the proof needs to explicitly include the intermediate configuration, violating full succinctness.

We show the following result:

Theorem 1.2. *Consider an IVC language corresponding to the non-deterministic machine $\mathcal{M}$ which takes as input configurations of length n and witnesses of length m. Assuming subexponentially secure $i\mathcal{O}$ and one-way functions, there exists a* non-adaptively *sound two-hop IVC scheme for non-deterministic machines $\mathcal{M}$ with a common reference string of size $\mathsf{poly}(\lambda, n, m)$ and proofs of size $\mathsf{poly}(\lambda)$. In the above, λ is the security parameter.*

A Secret Sharing Lens for IVC. To implement our "pure iO" approach to IVC, we take a fresh perspective on the problem of IVC. Our starting observation is that IVC can be rephrased as a graph reachability problem. Namely, can a configuration x_0 reach a configuration x_T in T steps? To leverage this perspective, we draw a connection to the problem of constructing *secret sharing schemes* for s-t connectivity.

Recall that undirected (resp. directed) s-t connectivity asks whether, in a given undirected (resp. directed) graph G, a vertex t is reachable from a vertex s. In our setting, s corresponds to the starting configuration x_0 and t to the ending configuration x_T. A secret sharing scheme for s-t connectivity associates a share with each *edge* of the graph G. The secret can be recovered only from a collection of edges that contains a *path* from s to t.

Guided by this analogy, the high-level view of our construction and analysis is as follows. Our construction associates a secret bit b with each pair consisting of a starting configuration x_0 and an ending configuration x_T, and secret-shares this bit over the corresponding configuration graph. Any intermediate prover certifying a valid transition from x_i to x_{i+1} can derive the share corresponding to that edge. Therefore, any honest computation for x_0 to x_T allows one to recover the corresponding secret b. On the other hand, we show that any share corresponding to an edge (x_i, x_{i+1}) for which x_i does not validly transition to x_{i+1} remains *hidden*. This property underlies the soundness of the scheme. To instantiate this template, we employ the elegant secret sharing scheme for undirected s-t connectivity due to Benaloh and Rudich (as cited in [Bei25]). Our analysis carefully combines the above connection to secret sharing with the techniques of [WW24, WW25] for constructing adaptively sound SNARGs. We refer the reader to Section 2 for a more thorough overview of our techniques and construction.

We believe that extending our techniques to construct IVC for NP beyond two hops would likely require a secret sharing scheme for directed s-t connectivity which is compatible with the template described above.

2 Technical Overview

In this overview, we will motivate our construction as well as our security analysis. First, we will recap the two-challenge approach of Waters and Wu [WW24, WW25] for proving adaptive soundness of SNARGs for NP. Then, we detail the s-t connectivity secret sharing scheme of Rudich and Benaloh in Section 2.1. Finally, we will further expand on the connection between our work and other existing works in Section 2.4.

Recapping the Two-Challenge Approach to SNARGs. The technical starting point of our work is the "two-challenge approach" of Waters and Wu [WW24, WZ24, WW25] (WW) for constructing adaptively sound SNARGs for NP. Their construction can be seen as a modification of the non-adaptive SNARG of Sahai and Waters [SW14], where they instead allow for *two* possible proofs for any statement.

We recap their construction here. Consider an NP language $\mathcal{L}$ with relation R. Sample three puncturable PRF[2] keys $k_0, k_1, k_{\mathsf{sel}}$. The CRS contains obfuscations of the following programs:

- $\mathsf{Verify}(x, (b, \pi))$: If $\pi = F_{k_b}(x)$, accept.
- $\mathsf{Prove}(x, w)$: If $R(x, w) = 1$, compute $b \leftarrow F_{k_{\mathsf{sel}}}(x)$, and output $(b, F_{k_b}(x))$.

At a high level, the construction assigns a "special bit" $b_x = F_{k_{\mathsf{sel}}}(x)$ to every statement x. We say that an accepting proof (b, π) for x is "on-path" if $b = b_x$, and "off-path" otherwise.

WW then argue *adaptive* security via the following high-level template (we gloss over many details):

Step 1: For all $x \notin \mathcal{L}$, one can argue that the corresponding special bit $b_x \leftarrow F_{k_{\mathsf{sel}}}(x)$ is **hidden**. Indeed, for $x \notin \mathcal{L}$, the Prove algorithm never computes b_x. Therefore, one can argue b_x is hidden by a standard puncturable PRF argument.

Step 2: Therefore, if an adversary creates cheating proofs of the form $(x, (b, \pi))$ for $x \notin \mathcal{L}$ with probability p, with probability $\approx \frac{p}{2}$, it must be the case that the proof is "off-path".

[2]Recall that a puncturable PRF is one where a PRF key k can be punctured on some point x^* to obtain a new key $k\{x^*\}$ with the property that $F_k(x) = F_{k\{x^*\}}(x)$ for $x \neq x^*$. Moreover, given just $k\{x^*\}$, the value of $F_k(x)$ is pseudorandom. Refer to Theorem 3.5 for a detailed definition.

Step 3: One can now argue that the CRS is indistinguishable from one which *rejects all off-path proofs*[3], even for $x \in \mathcal{L}$. Therefore, any adversary can only produce "on-path" proofs $(x, (b, \pi))$ where $b = b_x$. By indistinguishability, we therefore have that $p/2 \leq \mathsf{negl}(\lambda)$.

The two-challenge approach proved especially useful in Step 3 because it no longer required the reduction to decide the language.

Extending Their Proof Strategy to the Setting of IVC. However, it is not clear how to extend this analysis directly to the setting of IVC. For example, one could define a "special bit" for any tuple (x_0, x_i, i) to be $F_{k_{\mathsf{sel}}}(x_0, x_i, i)$. One would then hope to argue that if x_0 does not reach x_i, then the corresponding bit is hidden.

However, recall that the IVC prover is defined recursively: if one provides *any* proof for $(x_0, x_1, 1)$, one can create a valid proof for $(x_0, x_2, 2)$ if x_1 reaches x_2. Consider the following case: suppose x_0 does not reach x_1, but x_1 does reach x_i. While one can use the WW argument to argue that $\mathsf{secret}(x_0, x_1, 1)$ is hidden, there still exists an accepting "on-path" proof. Since $i\mathcal{O}$ does not give any knowledge soundness guarantee, it is not entirely clear how to inductively argue that $\mathsf{secret}(x_0, x_2, 2)$ is hidden.

A New Perspective: Secret Sharing. As discussed in the introduction, we instead draw inspiration from secret sharing. We can view the "special bit" associated with (x_0, x_T, T) as a "secret" that can be recovered if one follows some path from x_0 to x_T. This secret is then used to create on-the-fly "shares" corresponding to any (x_i, x_{i+1}). If there is a path from x_0 to x_T, then one can recover the corresponding secret.

With this analogy in mind, it is natural to draw inspiration from secret sharing constructions for s-t connectivity.

2.1 Detour: Secret Sharing for Undirected s-t Connectivity

We now recap the undirected s-t connectivity secret sharing scheme of Rudich and Benaloh [Bei25]. We first recap their construction and security analysis.

Construction. Fix an undirected graph $G = (V, E)$ with n vertices. Choose nodes $s, t \in V$. Suppose $V = \{v_1, \ldots, v_n\}$, and without loss of generality, suppose $v_1 = s$ and $v_n = t$. To secret share a bit $b \in \{0, 1\}$, we perform the following sharing algorithm with randomness $\mathbf{r} = \{r_1, \ldots, r_{n-1}\} \in \{0, 1\}^{n-1}$:

$\mathsf{Share}(b; \{r_1, r_2, \ldots, r_{n-1}\})$

- For every vertex $v_i \in V$, associate a random bit as follows:
  - If $i = 1$, set $b_1 = r_1$.
  - If $i = n$, set $b_n = b \oplus r_1$.
  - Else, set $b_i = r_i$.
- For each edge $(v_i, v_j) \in E$, associate the value $r_{i,j} = b_i \oplus b_j$.
- Output $\{r_{i,j}\}_{(v_i, v_j) \in E}$.

Let $r_{i,j}$ denote the bit associated with edge (v_i, v_j). To reconstruct the secret:

[3] We note that while this is not exactly how the reduction is presented in [WW25], one could achieve this by replacing their use of injective one-way functions with injective PRGs, as was done in [GSWW22, DWW24].

- Find a path $v_{i_0} = s, v_{i_1}, \ldots, v_{i_{\ell-1}}, v_{i_\ell} = t$ from s to t.
- Compute $b \leftarrow \sum_{k=0}^{\ell-1} r_{i_k, i_{k+1}} \pmod 2$.

Correctness follows from the fact that:

$$\sum_{k=0}^{\ell-1} r_{i_k, i_{k+1}} \pmod 2 = \sum_{k=0}^{\ell-1} (b_{i_k} \oplus b_{i_{k+1}}) \pmod 2 = b_{i_0} \oplus b_{i_\ell} = b,$$

where we identify $\oplus$ with mod 2 addition. The above scheme is indeed *information-theoretically* secure.

2.2 Analysis Inspired by Secret Sharing

We now discuss our construction for IVC for P. In order to crystallize the connection to secret sharing, we first begin by describing an **intermediate hybrid** in our proof strategy. (We defer the construction to Section 2.3.) We will now explicitly detail the parallels with the secret sharing

$\mathsf{Verify}(x_0, x_i, i, (b, \pi))$:

- If $\pi = F_{k_b}(x_0, x_i, i)$, accept. Else, reject.

$\mathsf{Prove}(x_0, x_i, i, (b_i, \pi_i))$:

- If $i \geq 1$, verify the proof (b_i, π_i). If $i = 0$, set $b_i = 0$.
- Let x_{i+1} be the next configuration after x_i.
- Let $b_{i+1} = b_i \oplus F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
- Output $(b_{i+1}, F_{k_{b_{i+1}}}(x_0, x_{i+1}, i+1))$.

Figure 1: Intermediate hybrid in IVC for P analysis.

scheme presented in Section 2.1. Consider the reachability graph $G = (V, E)$ where $V = \{0,1\}^n \times \{0, 1, \ldots, T\}$, where G contains an edge between (x_i, i) and $(x_{i+1}, i+1)$ if x_i reaches x_{i+1} in one step. We can then view the above scheme as using PRFs to pseudorandomly implement a secret sharing scheme for $s = (x_0, 0)$ and $t = (x_T, T)$ where:

- The secret associated with the instance is $\mathsf{secret}(x_0, x_i, i) = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$.
- The randomness associated with each (x_i, i) is the PRF value $r_{(x_i,i)} = F_{k_{\mathsf{sel}}}(x_0, x_i, i)$.
- The share associated with the edge between (x_i, i) and $(x_{i+1}, i+1)$ is the XOR of $b_{(x_i,i),(x_{i+1},i+1)} = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1) = r_{(x_i,i)} \oplus r_{(x_{i+1},i+1)}$.

Consider an honest execution of $\mathcal{M}$ with configurations $x_0^*, x_1^*, \ldots, x_T^*$. The corresponding IVC proofs $(b_1, \pi_1), \ldots, (b_T, \pi_T)$ satisfy that $b_i = F_{k_{\mathsf{sel}}}(x_0^*, x_0^*, 0) \oplus F_{k_{\mathsf{sel}}}(x_0^*, x_i^*, i)$. Indeed, one can view each b_i as a partial sum in the reconstruction algorithm in Section 2.1. Therefore, b_T is indeed equal to the share corresponding to (x_0, x_T, T).

Arguing the Secret Bit Is Hidden. We will first argue the simpler case where the instance (y_0, y'_T, T) corresponds to one where T is maximal (the IVC does not support any more hops). If y_0

does not reach y'_T in exactly T steps, then $(y_0, 0)$ does not reach (y'_T, T) in the *undirected version* of the graph G. Since the computation is deterministic, every (x_i, i) is adjacent to *at most one vertex* in layer $i+1$. By leveraging this fact, we have that the only vertices (x_i, i) that reach (y'_T, T) are those where the computation $\mathcal{M}$ takes x_i to y'_T in $T - i$ steps. Therefore, if y_0 does not reach y'_T in T steps, then the corresponding nodes in the graph are not reachable.

Suppose for a moment that the adversary only has *black-box* access to the prover algorithm. Then, we have that if x_i does not reach x_{i+1}, the corresponding share $F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$ is hidden. Therefore, one can employ PRF security along with the information-theoretic security of the secret sharing to argue that $\mathsf{secret}(y_0, y'_T, T)$ is hidden. However, since the prover receives an *obfuscation* of the program, one cannot allude to PRF security as mentioned above.

Instead, we will use a careful analysis via *puncturable PRFs* inspired by the following proof of information-theoretic security of the scheme in Section 2.1, specialized for the above graph G induced by a deterministic machine $\mathcal{M}$. We follow the notation used in Section 2.1. The shares associated to edges of G are independent of the secret $b_{(x_0,0)} \oplus b_{(x'_T,T)}$ by the following "reprogramming" argument:

- For every node $(x_0, 0)$, the corresponding node value $b_{(x_0,0)}$ is only used to construct the share corresponding to **exactly one** valid edge of G between $(x_0, 0)$ and $(x_1 = \mathcal{M}(x_0), 1)$. Therefore, we can instead randomly sample the share $r_{(x_0,0),(x_1,1)}$ first, and then redefine $b_{(x_0,0)} = b_{(x_1,1)} \oplus r_{(x_0,0),(x_1,1)}$.
- Now, suppose we have redefined all the edges between layers 0 and i, so that the corresponding node values satisfy

$$b_{(x_0,0)} = b_{(x_i,i)} \oplus \left(\bigoplus_{j=0}^{i-1} r_{(x_j,j),(x_{j+1},j+1)} \right)$$

  where $x_1, \ldots, x_i$ correspond to unique nodes reachable from x_0.
- Let x_{i+1} be the next node in the path from x_i. Now, we can similarly redefine $b_{(x_i,i)}$ by first sampling $r_{(x_i,i),(x_{i+1},i+1)}$, and setting $b_{(x_i,i)} = r_{(x_i,i),(x_{i+1},i+1)} \oplus b_{(x_{i+1},i+1)}$. This now redefines:

$$b_{(x_0,0)} = b_{(x_{i+1},i+1)} \oplus \left(\bigoplus_{j=0}^{i} r_{(x_j,j),(x_{j+1},j+1)} \right)$$

By conducting the above algorithm, we show that one can first sample the values associated to the edges of G independently, and then assign the values to the nodes. Consider x_0 and x'_T such that x_0 does not reach x'_T. Suppose $x_0, \ldots, x_T$ correspond to the nodes on the true path. Since

$$b_{(x_0,0)} = b_{(x_T,T)} \oplus \left(\bigoplus_{j=0}^{T-1} r_{(x_j,j),(x_{j+1},j+1)} \right)$$

the secret corresponding to x_0 and x'_T requires knowledge of $b_{(x_T,T)} \oplus b_{(x'_T,T)}$. However, since these values are hidden from the adversary's view, the adversary cannot predict the secret.

We imitate the above proof via puncturable PRFs. Consider (y_0, y'_T), and suppose $y_1, \ldots, y_{T-1}$, y_T denotes the actual sequence of configurations reached from y_0. We will show the following indistinguishability:

$\boxed{\mathsf{H}_1}$ Sample $b \leftarrow F_{k_{\mathsf{sel}}}(y_0, y_0, 0) \oplus F_{k_{\mathsf{sel}}}(y_0, y'_T, T)$ (i.e., $\mathsf{secret}(y_0, y'_T, T)$). Sample the CRS as in Fig. 1.

$\boxed{\mathsf{H}_2}$ In this hybrid, sample

$$b \leftarrow F_{k_{\mathsf{sel}}}(y_0, y'_T, T) \oplus F_{k_{\mathsf{sel}}}(y_0, y_T, T) \oplus \left(\bigoplus_{i=0}^{T-1} F_{k_{\mathsf{new}}}(y_i, y_{i+1}, i+1) \right).$$

Construct the prover algorithm instead as follows using a new PRF key k_{new}.

$\mathsf{Prove}(x_0, x_i, i, (b_i, \pi_i))$:

- If $i \geq 1$, verify the proof (b_i, π_i). If $i = 0$, set $b_i = 0$.
- Let x_{i+1} be the next configuration after x_i.
- If $x_0 = y_0$, let $b_{i+1} = \boxed{b_i \oplus F_{k_{\mathsf{new}}}(x_i, x_{i+1}, i+1)}$.
- Otherwise, let $b_{i+1} = b_i \oplus F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
- Output $(b_{i+1}, F_{k_{b_{i+1}}}(x_0, x_{i+1}, i+1))$.

To show the above indistinguishability, we fix the origin y_0 and perform this reprogramming only for prover inputs with $x_0 = y_0$. We iterate over $i = 0, \ldots, T-1$ and for each edge $((x_i, i), (x_{i+1}, i+1))$, we reprogram the share from $F_{k_{\mathsf{sel}}}(y_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(y_0, x_{i+1}, i+1)$ to $F_{k_{\mathsf{new}}}(x_i, x_{i+1}, i+1)$ via a puncturable PRF argument. We then reprogram the bit b accordingly, just as was done in our above security proof sketch.

Once we reach H_2, note that the prover algorithm does not use the value of $F_{k_{\mathsf{sel}}}(y_0, y_T, T)$ explicitly. Therefore, we use puncturable PRF security to argue that b looks pseudorandom given the CRS. Therefore, no adversary can predict the bit b given just the CRS in the last hybrid. In turn, by indistinguishability from H_1, we have that no adversary can predict b given only the CRS in H_1.

2.3 Construction and Analysis

We now state our main construction for IVC for P. Sample puncturable PRF keys $\mathsf{sk}_0, \mathsf{sk}_1, \kappa_{\mathsf{sel}}$. The CRS of our construction will contain obfuscations of the following programs:

- $\mathsf{Verify}((x_0, x_i, i), (b, \pi))$: If $b = F_{\kappa_{\mathsf{sel}}}(x_0, x_i, i)$, and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept.
- $\mathsf{Prove}((x_0, x_i, i), (b_i, \pi_i))$:
  - Let x_{i+1} be the configuration after x_i.
  - If $i = 0$ or (b_i, π_i) is an accepting proof for (x_0, x_i, i), output $b_{i+1} \leftarrow F_{\kappa_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$ and $\pi_{i+1} \leftarrow F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.

Note that unlike the Waters-Wu SNARG scheme, we begin from a hybrid where there is only *one* accepting proof per statement. We now argue essentially as was done in Waters-Wu.

Step 1: Change the selector bit. First, show that the real CRS is indistinguishable from one where $F_{\kappa_{\mathsf{sel}}}(x_0, x_i, i)$ is instead replaced with $F_{k_{\mathsf{sel}}}(x_0, x_0, 0) \oplus F_{k_{\mathsf{sel}}}(x_0, x_i, i)$. This can be shown by a standard puncturable PRF argument over $O(2^{2n} \cdot T)$ hybrids.

Step 2: Change how the prover algorithm computes the selector bit. If $i = 0$, first set $b_i = 0$. Modify the prover algorithm to compute b_{i+1} instead as $b_i \oplus F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$. Since (b_i, π_i) is only accepted if $b_i = F_{k_{\mathsf{sel}}}(x_0, x_0, 0) \oplus F_{k_{\mathsf{sel}}}(x_0, x_i, i)$, the functionality of the prover algorithm does not change. Therefore, we can invoke $i\mathcal{O}$ security.

Step 3: Enable the second proof. Now, we mimic Step 3 of the Waters-Wu proof to modify the verifier algorithm to accept both on-path and off-path proofs. This step relies on the injective PRG[4] security. We do this via the following "sparse trigger" template:

- Sample z as a random string. Check the off-path proof $G(\pi) = z$. With high probability, z is not in the image, so this does not change the functionality of the program.
- Sample z in the range of G. This now introduces a new off-path proof.

We implement the above idea carefully to ensure that the security of G is only used $\mathsf{poly}(T)$ times, as was done in [WW25, DWW24]. This ensures that one can pick the pre-image range of G (and hence, the proof π) to be small.

This resulting hybrid exactly corresponds to Fig. 1.

At this point, we can reach a contradiction as follows. Consider an adversary that produces cheating proofs with probability p. First, since $\mathsf{secret}(x_0, x_i, i)$ for false statements are hidden in Step 3, any cheating adversary must cheat on-path and off-path with roughly equal probability. Therefore, a cheating adversary must produce "off-path" proofs with probability $\approx p/2$. On the other hand, by indistinguishability from the real construction, we have that any cheating proofs correspond to "off-path" proofs with $\approx p/2 \leq \mathsf{negl}(\lambda)$. Therefore, $p \leq \mathsf{negl}(\lambda)$.

Two-Hop IVC for NP. The construction and analysis for two-hop IVC for NP are essentially identical to the above, except that the prover algorithm now also takes an additional witness as input.

- $\mathsf{Verify}((x_0, x_i, i), (b, \pi))$: If $b = F_{\kappa_{\mathsf{sel}}}(x_0, x_i, i)$, and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept.
- $\mathsf{Prove}((x_0, x_i, i), w_{i+1}, (b_i, \pi_i))$:
  - Let x_{i+1} be the configuration reached from x_i via w_{i+1}.
  - If $i = 0$ or (b_i, π_i) is an accepting proof for (x_0, x_i, i), output $b_{i+1} \leftarrow F_{\kappa_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$ and $\pi_{i+1} \leftarrow F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.

The proof also follows in a nearly identical manner, with two main differences.

- We only achieve *non-adaptive soundness*. This restriction arises from the fact that we need the PRG security loss to be independent of the statement length when enabling the "off-path" proofs in Step 3. Therefore, we can no longer let the reduction decide whether a given $(x_0, x_2, 2)$ is in the language. Invoking PRG security $2^{|x|}$ times would incur a $2^{|x|}$ security loss. Compensating for this loss would require the PRG seed length, and hence the IVC proof length, to depend on $|x|$, violating full succinctness.

[4]This use of injective PRGs does not add an assumption: assuming subexponentially secure $i\mathcal{O}$ and one-way functions, one obtains one-way permutations [SZ25], and applying the Goldreich-Levin hard-core predicate [GL89] gives injective PRGs (Theorem 3.3).

- We argue that the real construction is indistinguishable from a hybrid similar to Fig. 1 via a variation of the same argument. While the previous proof relied on determinism, here we rely on the fact that, if x_0 does not reach x_2, then for any intermediate x_1^*, either x_0 does not reach x_1^*, or x_1^* does not reach x_2. Thus, at least one of $F_{k_{\mathsf{sel}}}(x_0, x_0, 0) \oplus F_{k_{\mathsf{sel}}}(x_0, x_1^*, 1)$ or $F_{k_{\mathsf{sel}}}(x_0, x_1^*, 1) \oplus F_{k_{\mathsf{sel}}}(x_0, x_2, 2)$ is hidden. We can use this to reprogram the shares and argue that $\mathsf{secret}(x_0, x_2, 2) = F_{k_{\mathsf{sel}}}(x_0, x_0, 0) \oplus F_{k_{\mathsf{sel}}}(x_0, x_2, 2)$ is hidden.

Towards Fully Succinct IVC for NP. While the reachability interpretation also holds for IVC for NP, it is more directly analogous to the setting of reachability in *directed* graphs rather than *undirected* graphs. Indeed, the reachability problem for deterministic computations (when carefully defined) is equivalent in both directed and undirected settings - x_0 reaches x_T in the directed graph if and only if they are reachable in the undirected graph. Similarly, in the case of two-hop IVC for NP, x_0 reaches x_2 if and only if there is some x_1^* which is adjacent to both. In the graph restricted to these two endpoints and all intermediate configurations, directed and undirected reachability coincide. This allowed us to leverage a secret sharing scheme for *undirected* connectivity. However, such an equivalence does not seem to hold in general for NP (for example, see Fig. 2). We believe that ex-

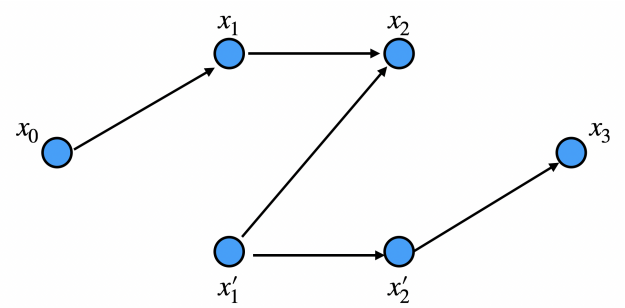

Figure 2: While x_3 is reachable from x_0 in the underlying undirected graph, it is not reachable in the directed graph.

tending our techniques to construct full-fledged IVC for NP likely requires a secret sharing scheme for directed s-t connectivity with an appropriate efficiency requirement which is also compatible with our security analysis and efficiency requirements.

2.4 Related Work

In this section, we present the techniques used in related works in more detail.

IVC for P and NP from Standard Assumptions. The works of [DGKV22, PP22] demonstrate how to use rate-1 batch arguments to construct IVC for deterministic computations. The construction is as follows: Given a proof π_1 that a state x_0 reaches x_1, and proof π_2 that state x_1 reaches x_2, one can use a rate-1 BARG to merge the two proofs to create a single proof π of size $|\pi_b| + \mathsf{poly}(\lambda)$. Now, one can treat (x_1, π) as the new proof for the fact that x_0 reaches x_2. A prover performs the above proof merging in a binary tree fashion. The resulting proof size is therefore $\mathsf{poly}(|x_i|, \log T)$.

The subsequent work of [DJJ+25] show that the above template can be adopted in the setting of NP. At a high level, they use a SNARG for NP at the base level to create "one-hop" proofs. Then, they aggregate the proofs in the same manner as in the constructions of [DGKV22, PP22]. This allows them to achieve $\mathsf{poly}(|x_i|, \log T)$ size proofs assuming $i\mathcal{O}$ and rate-1 BARGs.

In the special case of deterministic computations, the work of [PP22] additionally show how to achieve full succinctness by instead *hashing* the intermediate configurations, i.e., the aggregated

proof contains a hash instead of an explicit configuration. At a high level, their analysis relies on the fact that given the starting configuration x_0, one can predict the correct configuration x_i at time i. Therefore, one can inductively check in the security analysis that the intermediate hashes h_i are in fact honest hashes of x_i. With that, they can now allude to RAM SNARG soundness to complete their proof. However, this technique does not extend to the NP setting since the intermediate states x_i are not unique.

IVC for Trapdoor NP Languages. Datta et al. [DJJ+25] additionally construct IVC for trapdoor NP languages via $i\mathcal{O}$ and injective PRGs. Informally, an IVC language has a trapdoor if there *exists* a (not necessarily easy to find) polynomial-sized circuit C which takes as input (i, x_0, x_i), and determines if x_i is reachable from x_0 in i steps. While their construction is similar to ours, their proof analysis crucially relies on this circuit C. By embedding C into the obfuscated prover and verifier programs, one can construct a new indistinguishable CRS which accepts *no* proofs for false statements. However, such a circuit does not exist in general.

Instead, we rely on the Waters-Wu two-challenge approach, and remove all "off-path" proofs, regardless of whether (x_0, x_i, i) is in the language. Therefore, we do not require the existence of such a small circuit in our proof.

Fully Succinct BARGs for NP. The works of [GSWW22, DWW24] construct fully succinct BARGs for NP assuming $i\mathcal{O}$ and other assumptions. Here, a fully succinct BARG is one whose proof length is $\mathsf{poly}(\lambda)$ and is independent of the batch and witness sizes. We note that their BARGs do not immediately imply IVC - their construction requires the whole statement to be declared upfront. In IVC, the prover needs to compute the statement incrementally. The work of [GSWW22] additionally present an update BARG - a BARG where each instance in the batch is given to the prover on the fly. While this is very close to the setting of IVC, the verifier in their construction still requires access to the full statement in order to verify the proof. In our setting, this would correspond to the full IVC transcript. Moreover, their proof utilizes SSBs, and therefore requires additional algebraic assumptions.

On the other hand, our techniques are highly inspired by their security analysis. Specifically, they use an injective PRG in a similar way to iteratively remove proofs from the obfuscated program in the CRS.

3 Preliminaries

The following preliminaries are largely borrowed from [DJJ+25].

Throughout, we will use λ to denote the security parameter.

- We say that a function $f(\lambda)$ is negligible in λ if $f(\lambda) = \lambda^{-\omega(1)}$, and we denote it by $f(\lambda) = \mathsf{negl}(\lambda)$.
- We say that a function $g(\lambda)$ is polynomial in λ if $g(\lambda) \leq p(\lambda)$ for some fixed polynomial p, and we denote it by $g(\lambda) = \mathsf{poly}(\lambda)$.
- For $n \in \mathbb{N}$, we use $[n]$ to denote $\{1, \ldots, n\}$.
- If R is a random variable, then $r \leftarrow R$ denotes sampling r from R. If T is a set, then $i \leftarrow T$ denotes sampling i uniformly at random from T.

Security Notions. We will use the parameters (s, ε) to describe the security parameters of our primitive. In particular, we say that a primitive is (s, ε)-secure if for all sufficiently large λ and all

adversaries $\mathcal{A}$ of size at most $s(\lambda)$, the advantage of $\mathcal{A}$ is at most $\varepsilon(\lambda)$. For a function $\Gamma(\lambda) \geq 1$, we say that a primitive is Γ-secure if it is $(\Gamma, 1/\Gamma)$-secure.

- **Polynomial Security**: We say that a primitive is polynomially secure if it is p-secure for every polynomial $p = p(\lambda) \geq 1$.
- **Subexponential Security**: We say that a primitive is subexponentially secure if it is 2^{λ^c}-secure for some $0 < c < 1$.

3.1 Pseudorandom Generators

Definition 3.1 (Pseudorandom generators)**.** *Let t and m be polynomials with $t(\lambda) \leq m(\lambda)$. A function $G : \{0,1\}^{t(\lambda)} \to \{0,1\}^{m(\lambda)}$ is a (s, ε)-secure pseudorandom generator if it is efficiently computable, and if for all adversaries $\mathcal{A}$ of size $s(\lambda)$, we have that*

$$\left| \Pr\left[\mathcal{A}(y) = 1 \ : \ \begin{matrix} x \leftarrow \{0,1\}^{t(\lambda)} \\ y \leftarrow G(x) \end{matrix} \right] - \Pr\left[\mathcal{A}(y) = 1 \ : \ y \leftarrow \{0,1\}^{m(\lambda)} \right] \right| \leq \varepsilon(\lambda).$$

In this work, we will require the existence of *injective* PRGs. Towards this, we will first require the following result.

Theorem 3.2 ([SZ25])**.** *Assuming subexponential hardness of $i\mathcal{O}$ and one-way functions, there exists a subexponentially secure one-way permutation.*

By directly applying Goldreich-Levin [GL89], we then obtain the following immediate corollary.

Corollary 3.3. *Assuming subexponential hardness of $i\mathcal{O}$ and one-way functions, there exists a subexponentially secure injective PRG.*

Remark 3.4 (Public parameters for injective PRGs)**.** *Our proofs also allow publicly keyed injective PRGs. In this case, we independently sample a public key pk once in each experiment and suppress it from the notation, writing G for G_{pk}. All evaluations within the experiment use the same key. We require injectivity for every generated key and pseudorandomness even given the sampled public key.*

3.2 Puncturable Pseudorandom Functions

Definition 3.5 (Puncturable PRF)**.** *A puncturable PRF family on key space $\mathcal{K} = \{\mathcal{K}_\lambda\}_\lambda$, domain $\mathcal{X} = \{\mathcal{X}_\lambda\}_\lambda$, output space $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_\lambda$ and polynomial $t(\lambda)$ is a tuple of* PPT *algorithms* ($\mathsf{PPRF.Gen}$, $\mathsf{PPRF.Punc}$, $\mathsf{PPRF.Eval}$) *with the following interface and properties.*

- $\mathsf{PPRF.Gen}(1^\lambda)$ *: On input the security parameter 1^λ, outputs a key $k \in \mathcal{K}_\lambda$.*
- $\mathsf{PPRF.Punc}(k, T)$*: On input a key $k \in \mathcal{K}_\lambda$ and a set $T \subseteq \mathcal{X}_\lambda$ such that $|T| \leq t(\lambda)$, outputs a new key $k\{T\} \in \mathcal{K}_\lambda$.*
- $\mathsf{PPRF.Eval}(k, x)$*: On input $k \in \mathcal{K}_\lambda$ and $x \in \mathcal{X}_\lambda$, outputs some value $y \in \mathcal{Y}_\lambda$.*
- ***Functionality preserving:*** *For all sets $T \subseteq \mathcal{X}_\lambda$ of size $|T| \leq t(\lambda)$, for all $x \in \mathcal{X}_\lambda \setminus T$,*

$$\Pr[k \leftarrow \mathsf{Gen}(1^\lambda), k\{T\} \leftarrow \mathsf{Punc}(k, T) : \mathsf{Eval}(k\{T\}, x) = \mathsf{Eval}(k, x)] = 1 \ .$$

- (s, ε)-***pseudorandomness at punctured points:*** *Consider the following experiment* $\mathsf{Expt}_{\mathcal{A}}^{\mathsf{PPRF}}(1^\lambda, b)$ *between a challenger* $\mathcal{C}$ *and adversary* $\mathcal{A}$*:*
  - *First,* $\mathcal{A}$ *chooses a set* $T \subseteq \mathcal{X}_\lambda$ *such that* $|T| \leq t(\lambda)$*, and sends it to* $\mathcal{C}$*.*
  - $\mathcal{C}$ *samples* $k \leftarrow \mathsf{PPRF.Gen}(1^\lambda)$*, and computes* $k\{T\} \leftarrow \mathsf{PPRF.Punc}(k, T)$*.*
    - *If* $b = 0$*, compute* $L_0 = \{x_i, \mathsf{PPRF.Eval}(k, x_i)\}_{x_i \in T}$*.*
    - *If* $b = 1$*, compute* $L_1 = \{x_i, y_i\}_{x_i \in T}$ *where each* $y_i \leftarrow \mathcal{Y}_\lambda$*.*

    Give $k\{T\}, L_b$ *to* $\mathcal{A}$*.*
  - $\mathcal{A}$ *produces a bit* b'*.*

  We say that the PPRF *is pseudorandom at punctured points if for all size* $s = s(\lambda)$ *adversaries* $\mathcal{A}$*,*

$$\left|\Pr[\mathsf{Expt}_{\mathcal{A}}^{\mathsf{PPRF}}(1^\lambda, 0) = 1] - \Pr[\mathsf{Expt}_{\mathcal{A}}^{\mathsf{PPRF}}(1^\lambda, 1) = 1]\right| \leq \varepsilon(\lambda)$$

Theorem 3.6 ([GGM84, BW13, BGI14, KPTZ13])**.** *Assuming the existence of polynomially secure (resp. subexponentially secure) one-way functions, then there exists polynomially secure (resp. subexponentially secure) puncturable PRFs.*

3.3 Indistinguishability Obfuscation

Definition 3.7 (Indistinguishability obfuscation ($i\mathcal{O}$), [BGI+01])**.** *An* indistinguishability obfuscator $i\mathcal{O}$ *for a circuit class* $\mathcal{C} = \{\mathcal{C}_\lambda\}_\lambda$ *is a* PPT *algorithm that satisfies the following conditions:*

- ***Correctness:*** *For all security parameters* $\lambda \in \mathbb{N}$*, for all* $C \in \mathcal{C}_\lambda$*, for all inputs* x*, we have that*

$$\Pr[C' \leftarrow i\mathcal{O}(1^\lambda, C) : C'(x) = C(x)] = 1.$$

- $(s, \varepsilon)-$***Security:*** *For all pairs of circuits* $C_0, C_1 \in \mathcal{C}_\lambda$ *such that* $|C_0| = |C_1|$ *and* $C_0(x) = C_1(x)$ *for all* x*, for all* $s = s(\lambda)$*-sized adversaries* $\mathcal{A}$*,*

$$\left|\Pr[\mathcal{A}(i\mathcal{O}(1^\lambda, C_0)) = 1] - \Pr[\mathcal{A}(i\mathcal{O}(1^\lambda, C_1)) = 1]\right| \leq \varepsilon(\lambda)$$

3.4 Incrementally Verifiable Computation

Definition 3.8 (IVC Language)**.** *We define the* IVC *language* $\mathcal{L}_{\mathsf{IVC}}$ *to be*

$$\mathcal{L}_{\mathsf{IVC}} = \left\{ (\mathcal{C}, x_0, x_t, t) \;:\; \begin{array}{c} \mathcal{C} : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n, x_0, x_t \in \{0,1\}^n \\ \exists (x_1, \ldots, x_{t-1}, w_1, \ldots, w_t) \text{ such that } \forall i \in [t], \mathcal{C}(x_{i-1}, w_i) = x_i \end{array} \right\}$$

We define $\mathcal{L}_C$ *to be the restriction of* $\mathcal{L}_{\mathsf{IVC}}$ *to circuit* $\mathcal{C} = C$*, i.e.,*

$$\mathcal{L}_C = \{(x_0, x_i, i) : (C, x_0, x_i, i) \in \mathcal{L}_{\mathsf{IVC}}\}$$

We define $\mathcal{L}_{C,i}$ *to be the restriction of* $\mathcal{L}_{\mathsf{IVC}}$ *to circuit* $\mathcal{C} = C$ *and iteration bound* $t = i$*, i.e.,*

$$\mathcal{L}_{C,i} = \{(x_0, x_i) : (C, x_0, x_i, i) \in \mathcal{L}_{\mathsf{IVC}}\}$$

Definition 3.9 (IVC for $\mathcal{L}_{\mathsf{IVC}}$ (adapted from [HAN23]))**.** *An incrementally verifiable computation (IVC) scheme for* $\mathcal{L}_{\mathsf{IVC}}$ *is a tuple of* PPT *algorithms* $\mathsf{IVC} = (\mathsf{IVC.Gen}, \mathsf{IVC}.\mathcal{P}, \mathsf{IVC}.\mathcal{V})$ *where*

- $\mathsf{IVC.Gen}(1^\lambda, T, 1^n, 1^m, C)$*: takes as input the security parameter* λ*, an iteration bound* T *in binary, an input length* n*, a witness length* m*, and a circuit* $C : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$*, and outputs a common reference strings* crs*.*
- $\mathsf{IVC}.\mathcal{P}(\mathsf{crs}, (x_0, x_{i-1}, i-1), \pi_{i-1}, w_i)$*: takes as input the common reference string* crs*, inputs* x_0*,* $x_{i-1} \in \{0,1\}^n$*, an index* $i - 1 \in \{0, \ldots, T-1\}$*, a proof* π_{i-1}*, and a witness* $w_i \in \{0,1\}^m$*, and outputs a proof* π_i*.*
- $\mathsf{IVC}.\mathcal{V}(\mathsf{crs}, (x_0, x_t, t), \pi_t)$*: takes as input the common reference string* crs*, inputs* $x_0, x_t \in \{0,1\}^n$*, an index* $t \in [T]$*, and a proof* π_t*, and outputs a value in* $\{0,1\}$*.*

The scheme is additionally required to satisfy completeness and efficiency as defined below.

Completeness: *For all* $\lambda \in \mathbb{N}$*, all iteration bounds* $T \leq 2^\lambda$*, all* $t \in [T]$*, all polynomials* $n = n(\lambda), m = m(\lambda)$*, all polynomial-sized circuits* $C : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$*, all* $x_0 \in \{0,1\}^n$*, and all* $w_1, \ldots, w_t \in \{0,1\}^m$*,*

$$\Pr\left[\begin{array}{c} \mathsf{IVC}.\mathcal{V}(\mathsf{crs}, (x_0, x_t, t), \pi_t) = 1 \\ : \\ \end{array} \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{IVC.Gen}(1^\lambda, T, 1^n, 1^m, C) \\ \forall i \in [t] : x_i = C(x_{i-1}, w_i) \\ \forall i \in [t] : \\ \pi_i \leftarrow \mathsf{IVC}.\mathcal{P}(\mathsf{crs}, (x_0, x_{i-1}, i-1), \pi_{i-1}, w_i) \end{array}\right] = 1.$$

Here, π_0 *denotes a fixed dummy initial proof, which is ignored by* $\mathsf{IVC}.\mathcal{P}$ *on the first invocation.*

Efficiency: *There exists polynomials* p_1 *and* p_2 *such that on setup parameters* $(1^\lambda, T, 1^n, 1^m, C)$ *where* C *is a polynomial-sized circuit* $C : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$*, all algorithms run in time* $p_1(\lambda, \log T, |C|)$ *and the proof size is* $p_2(\lambda, n, m, \log T, \log |C|)$*.*

We say that the scheme satisfies (s, ε)*-soundness if the following holds:*

(s, ε)**-Adaptive Soundness:** *For all sufficiently large* λ*, all iteration bounds* $T \leq 2^\lambda$*, all polynomials* $n = n(\lambda), m = m(\lambda)$*, all polynomial-sized circuits* $C : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$*, and all* $s = s(\lambda)$*-sized adversaries* $\mathcal{A}$*,*

$$\Pr\left[\begin{array}{c} \mathsf{IVC}.\mathcal{V}(\mathsf{crs}, (x_0, x_t, t), \pi_t) = 1 \\ (C, x_0, x_t, t) \notin \mathcal{L}_{\mathsf{IVC}} \\ t \in [T] \end{array} : \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{IVC.Gen}(1^\lambda, T, 1^n, 1^m, C) \\ (x_0, x_t, t, \pi_t) \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right] \leq \varepsilon(\lambda)$$

(s, ε)**-Non-Adaptive Soundness:** *For all sufficiently large* λ*, all iteration bounds* $T \leq 2^\lambda$*, all polynomials* $n = n(\lambda), m = m(\lambda)$*, all polynomial-sized circuits* $C : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$*, and all* $s = s(\lambda)$*-sized adversaries* $\mathcal{A}$*, for all* $t \in [T]$*,* $(C, x_0, x_t, t) \notin \mathcal{L}_{\mathsf{IVC}}$*,*

$$\Pr\left[\begin{array}{c} \mathsf{IVC}.\mathcal{V}(\mathsf{crs}, (x_0, x_t, t), \pi_t) = 1 \\ : \\ \end{array} \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{IVC.Gen}(1^\lambda, T, 1^n, 1^m, C) \\ \pi_t \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right] \leq \varepsilon(\lambda)$$

Remark 3.10. *We use the phrase* IVC *for* NP *to refer to* IVC *for* $\mathcal{L}_{\mathsf{IVC}}$*. We use the phrase* IVC *for* P *(or for deterministic computations) if the circuit* C *has no non-deterministic input.*

Definition 3.11 (k-hop IVC for NP)**.** *We say an IVC scheme is adaptively (resp. non-adaptively) sound for* k *hops if soundness only holds for* $T \leq k$*, i.e., soundness is required for statements with any number* $t \in [k]$ *of computation steps.*

4 Fully Succinct IVC for P via $i\mathcal{O}$

Ingredients.

- A $2^{\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}$-secure puncturable PRFs $(\mathsf{PPRF.Gen}, \mathsf{PPRF.Punc}, \mathsf{PPRF.Eval})$. We will use $F_K(x)$ as shorthand for $\mathsf{PPRF.Eval}(K, x)$ for readability.
- A $2^{\lambda_{\mathsf{iO}}^{\delta_{\mathsf{iO}}}}$-secure indistinguishability obfuscation $i\mathcal{O}$.
- A $2^{\lambda_{\mathsf{PRG}}^{\delta_{\mathsf{PRG}}}}$-secure injective PRG $G : \{0,1\}^{n_\pi} \to \{0,1\}^{m_{\mathsf{PRG}}}$ constructed via Theorem 3.3. While this PRG will not appear in the construction, we will crucially rely on it in our security proof.

We are now ready to describe the algorithm. We will choose the specific parameters after first describing the algorithm. Throughout this construction and its hybrid argument, we pad all verifier circuits to a common size before obfuscation, and likewise all prover circuits, including their embedded obfuscated verifiers. These sizes are polynomial in $n, \lambda, \log T$.

- $\mathsf{IVC.Gen}(1^\lambda, T, 1^n, \mathcal{M})$:
  1. Sample puncturable PRF keys $\mathsf{sk}_0, \mathsf{sk}_1, \mathsf{sk}_{\mathsf{sel}} \leftarrow \mathsf{PPRF.Gen}(1^\lambda_{\mathsf{PPRF}})$. We will treat the output of $\mathsf{PPRF.Eval}(\mathsf{sk}_b, \cdot)$ to be a string in $\{0,1\}^{n_\pi}$, and the output of $\mathsf{PPRF.Eval}(\mathsf{sk}_{\mathsf{sel}}, \cdot)$ to be a single bit.
  2. Construct the following verification algorithm $V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}}}$:

     $V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}}}(x_0 \in \{0,1\}^n, x_i \in \{0,1\}^n, i \in [T], (b, \pi))$:
     - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

  3. Let $\widetilde{V} \leftarrow i\mathcal{O}(1^{\lambda_{\mathsf{iO}}}, V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}}})$.
  4. Construct the following prover algorithm $P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}}}$:

     $P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}}}(x_0 \in \{0,1\}^n, x_i \in \{0,1\}^n, i \in \{0,1,\ldots,T-1\}, (b_i, \pi_i))$:
     - If $i = 0$, set $x_i = x_0$.
     - **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
     - **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i)$.
     - **Compute next proof:**
       * Let $b_{i+1} = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
       * Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
     - Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

  5. Let $\widetilde{P} \leftarrow i\mathcal{O}(1^{\lambda_{\mathsf{iO}}}, P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}}})$.
  6. Output $\mathsf{crs} = (\widetilde{P}, \widetilde{V})$.
- $\mathsf{IVC.Prove}(\mathsf{crs} = (\widetilde{P}, \widetilde{V}), (x_0, x_i, i), (b_i, \pi_i))$: Output $\widetilde{P}(x_0, x_i, i, (b_i, \pi_i))$.
- $\mathsf{IVC.Verify}(\mathsf{crs} = (\widetilde{P}, \widetilde{V}), (x_0, x_i, i), (b_i, \pi_i))$: Output $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i))$.

We choose the following parameters:

- Choose $\Gamma = \mathsf{poly}(\lambda, T)$ sufficiently large to bound the distinguishers used below. Since $T \leq 2^\lambda$, we have $\Gamma \leq 2^{O(\lambda)}$.
- Choose $\lambda_{\mathsf{PPRF}}, \lambda_{\mathsf{iO}}$ so that $O(2^{3n} \cdot T^2) \cdot (2^{-\lambda_{\mathsf{iO}}^{\delta_{\mathsf{iO}}}} + 2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}) \leq \mathsf{negl}(\lambda)$.
- Choose λ_{PRG} so that $\mathsf{poly}(\lambda, n, T) \cdot 2^{-\lambda_{\mathsf{PRG}}^{\delta_{\mathsf{PRG}}}} \leq \mathsf{negl}(\lambda)$.
- Additionally, require $\mathsf{poly}(\Gamma) \leq \min\{2^{\lambda_{\mathsf{iO}}^{\delta_{\mathsf{iO}}}}, 2^{\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}, 2^{\lambda_{\mathsf{PRG}}^{\delta_{\mathsf{PRG}}}}\}$, where the polynomial accounts for the reduction overhead.
- Choose the PRG G so that $n_\pi = \mathsf{poly}(\lambda, \log T)$ with $n_\pi > \lambda$, and $m_{\mathsf{PRG}} = \mathsf{poly}(n_\pi)$ with $m_{\mathsf{PRG}} > n_\pi + \lambda + 2\log_2 T$.

Theorem 4.1. *Assuming subexponentially secure $i\mathcal{O}$ and one-way functions, there exists an adaptively sound IVC scheme for deterministic computations with:*

- *CRS size* $\mathsf{poly}(n, \lambda, \log T)$,
- *prover and verifier runtime per update is* $\mathsf{poly}(n, \lambda, \log T)$,
- *and proof size* $\mathsf{poly}(\lambda, \log T)$.

Proof. By Theorem 3.3, note that an injective PRG G exists.

Suppose the scheme is not adaptively sound. Then, there exists some $T \leq 2^\lambda$ and polynomial $n(\lambda)$ such that there exists a $\mathsf{poly}(\lambda)$ adversary Adv which provides cheating proofs $(x_0, x_i, i, (b, \pi))$ for $(x_0, x_i, i) \notin \mathcal{L}_{\mathcal{M}}$ with probability p_{cheat}.

Then, there exists some $\tau \in [T]$ such that

$$\Pr\left[\begin{array}{c} \mathsf{IVC}.\mathcal{V}(\mathsf{crs}, (x_0, x_i, i), (b, \pi)) = 1 \\ \wedge\ (x_0, x_i, i) \notin \mathcal{L} \\ \wedge\ i = \tau \end{array} : \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{IVC.Gen}(1^\lambda, T, 1^n, \mathcal{M}) \\ (x_0, x_i, i, (b, \pi)) \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right] \geq \frac{p_{\mathsf{cheat}}}{T}$$

We fix this τ for the rest of the proof. We will now show that $p_{\mathsf{cheat}} = \mathsf{negl}(\lambda)$.

$\boxed{\mathsf{H}_1}$ In this experiment, generate the crs as is done in the real IVC scheme.

$\boxed{\mathsf{H}_2}$ In this experiment,

- Additionally sample a key $k_{\mathsf{sel}} \leftarrow \mathsf{PPRF.Gen}(1^{\lambda_{\mathsf{PPRF}}})$.
- Construct the machine $V_{\mathsf{sk}_0, \mathsf{sk}_1, \mathsf{sk}_{\mathsf{sel}}, k_{\mathsf{sel}}, \tau}$:

  $V_{\mathsf{sk}_0, \mathsf{sk}_1, \mathsf{sk}_{\mathsf{sel}}, k_{\mathsf{sel}}, \tau}(x_0, x_i, i, (b, \pi))$:
  - If $i \leq \tau$,
    - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept.
  - If $i > \tau$,
    - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

  Let $\widetilde{V} \leftarrow i\mathcal{O}(1^{\lambda_{\mathsf{iO}}}, V_{\mathsf{sk}_0, \mathsf{sk}_1, \mathsf{sk}_{\mathsf{sel}}, k_{\mathsf{sel}}, \tau})$.
- Construct the following prover algorithm $P_{\mathsf{sk}_0, \mathsf{sk}_1, \mathsf{sk}_{\mathsf{sel}}, k_{\mathsf{sel}}, \tau}$:

$P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau}(x_0, x_i, i, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i)$.
- **Compute next proof:**
  - If $i < \tau$, let $b_{i+1} = F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$
  - If $i \geq \tau$, let $b_{i+1} = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
  - Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

Lemma 4.2. *For all adversaries $\mathcal{B}$ of size at most Γ, we have that*

$$\left|\Pr\left[\begin{array}{c} b = 1 \\ : \\ \end{array}\begin{array}{c} \mathsf{crs} \leftarrow H_1 \\ b \leftarrow \mathcal{B}(\mathsf{crs}) \end{array}\right] - \Pr\left[\begin{array}{c} b = 1 \\ : \\ \end{array}\begin{array}{c} \mathsf{crs} \leftarrow H_2 \\ b \leftarrow \mathcal{B}(\mathsf{crs}) \end{array}\right]\right| \leq O(2^{2n} \cdot T) \cdot (2^{-\lambda_{iO}^{\delta_{iO}}} + 2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}).$$

We defer the proof of this claim to Section 4.1.

H_3 In this hybrid, we instead sample the following prover algorithm. $P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau}$:

$P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau}(x_0, x_i, i, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$ and $b_i = 0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i)$.
- **Compute next proof:**
  - If $i < \tau$, let $b_{i+1} = b_i \oplus F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}}(x_0, x_i, i)$
  - If $i \geq \tau$, let $b_{i+1} = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
  - Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

Sample the rest of the CRS as in the previous hybrid. Indistinguishability follows from functionality equivalence. Recall that V only accepts $b_i = F_{k_{\mathsf{sel}}}(x_0, x_0, 0) \oplus F_{k_{\mathsf{sel}}}(x_0, x_i, i)$. Therefore, the updated program computes b_{i+1} identically.

Therefore, by alluding to the indistinguishability of H_1 and H_3, we have that

$$\left|\Pr\left[\begin{array}{c} \mathsf{IVC}.\mathcal{V}(\mathsf{crs}, (x_0, x_i, i), (b, \pi)) = 1 \\ \wedge\ (x_0, x_i, i) \notin \mathcal{L} \\ \wedge\ i = \tau \end{array} : \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{H}_1 \\ (x_0, x_i, i, (b, \pi)) \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right] - \Pr\left[\begin{array}{c} \mathsf{IVC}.\mathcal{V}(\mathsf{crs}, (x_0, x_i, i), (b, \pi)) = 1 \\ \wedge\ (x_0, x_i, i) \notin \mathcal{L} \\ \wedge\ i = \tau \end{array} : \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{H}_3 \\ (x_0, x_i, i, (b, \pi)) \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right]\right| \leq O(2^{2n} \cdot T) \cdot (2^{-\lambda_{\mathsf{iO}}^{\delta_{\mathsf{iO}}}} + 2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}). \quad (1)$$

Now, we will define a new sequence of hybrids, where every hybrid outputs a puncturable PRF key along with the IVC CRS.

$\boxed{\mathsf{H}_4}$ In this hybrid, sample k_{sel} and crs as in H_3, and output $(k_{\mathsf{sel}}, \mathsf{crs})$.

$\boxed{\mathsf{H}_5}$ We modify the verifier algorithm as follows:

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau}(x_0, x_i, i, (b, \pi))$:

- If $i \leq \tau$,
  - If $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept.
- If $i > \tau$,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

Output $(k_{\mathsf{sel}}, \mathsf{crs})$.

Lemma 4.3. *For all adversaries $\mathcal{B}$ of size at most Γ, we have that*

$$\left|\Pr\left[\begin{array}{c} b = 1 \end{array} : \begin{array}{c} (k_{\mathsf{sel}}, \mathsf{crs}) \leftarrow \mathsf{H}_4 \\ b \leftarrow \mathcal{B}(k_{\mathsf{sel}}, \mathsf{crs}) \end{array}\right] - \Pr\left[\begin{array}{c} b = 1 \end{array} : \begin{array}{c} (k_{\mathsf{sel}}, \mathsf{crs}) \leftarrow \mathsf{H}_5 \\ b \leftarrow \mathcal{B}(k_{\mathsf{sel}}, \mathsf{crs}) \end{array}\right]\right|$$
$$\leq O(2^{2n} \cdot T) \cdot (2^{-\lambda_{iO}^{\delta_{iO}}} + 2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}) + T \cdot (2^{-\lambda_{\mathsf{PRG}}^{\delta_{\mathsf{PRG}}}} + \frac{2^{n_\pi}}{2^{m_{\mathsf{PRG}}}})$$

We defer the proof of this claim to Section 4.2.

By the indistinguishability of H_4 and H_5, we therefore have the following:

$$\left|\Pr\left[\begin{array}{c} \mathsf{IVC}.\mathcal{V}(\mathsf{crs}, (x_0, x_i, i), (b, \pi)) = 1 \\ \wedge\ (x_0, x_i, i) \notin \mathcal{L} \\ \wedge\ i = \tau \\ \wedge\ b \neq F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0) \end{array} : \begin{array}{c} (k_{\mathsf{sel}}, \mathsf{crs}) \leftarrow \mathsf{H}_4 \\ (x_0, x_i, i, (b, \pi)) \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right]\right.$$
$$\left. - \Pr\left[\begin{array}{c} \mathsf{IVC}.\mathcal{V}(\mathsf{crs}, (x_0, x_i, i), (b, \pi)) = 1 \\ \wedge\ (x_0, x_i, i) \notin \mathcal{L} \\ \wedge\ i = \tau \\ \wedge\ b \neq F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0) \end{array} : \begin{array}{c} (k_{\mathsf{sel}}, \mathsf{crs}) \leftarrow \mathsf{H}_5 \\ (x_0, x_i, i, (b, \pi)) \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right]\right|$$
$$\leq O(2^{2n} \cdot T) \cdot (2^{-\lambda_{\mathsf{iO}}^{\delta_{\mathsf{iO}}}} + 2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}) + T \cdot (2^{-\lambda_{\mathsf{PRG}}^{\delta_{\mathsf{PRG}}}} + \frac{2^{n_\pi}}{2^{m_{\mathsf{PRG}}}})$$

Moreover, the first term is 0 since $\mathcal{V}$ does not accept any proof where the bit b does not satisfy the check. Therefore, we can rearrange the equation to obtain that:

$$p^{\mathsf{NEQ}} = \Pr\left[\begin{array}{c} \mathsf{IVC}.\mathcal{V}(\mathsf{crs}, (x_0, x_i, i), (b, \pi)) = 1 \\ \wedge\ (x_0, x_i, i) \notin \mathcal{L} \\ \wedge\ i = \tau \\ \wedge\ b \neq F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0) \end{array} : \begin{array}{c} (k_{\mathsf{sel}}, \mathsf{crs}) \leftarrow \mathsf{H}_5 \\ (x_0, x_i, i, (b, \pi)) \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right]$$
$$\leq O(2^{2n} \cdot T) \cdot (2^{-\lambda_{\mathsf{iO}}^{\delta_{\mathsf{iO}}}} + 2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}) + T \cdot (2^{-\lambda_{\mathsf{PRG}}^{\delta_{\mathsf{PRG}}}} + \frac{2^{n_\pi}}{2^{m_{\mathsf{PRG}}}}) \quad (2)$$

By a similar argument, by invoking (1) with the indistinguishability of H_4 and H_5, we also have that

$$p^{\mathsf{EQ}} = \Pr\left[\begin{array}{c} \mathsf{IVC}.\mathcal{V}(\mathsf{crs},(x_0,x_i,i),(b,\pi))=1 \\ \wedge\ (x_0,x_i,i)\notin\mathcal{L} \\ \wedge\ i=\tau \\ \wedge\ b \boxed{=} F_{k_{\mathsf{sel}}}(x_0,x_i,i)\oplus F_{k_{\mathsf{sel}}}(x_0,x_0,0) \end{array} : \begin{array}{c} (k_{\mathsf{sel}},\mathsf{crs})\leftarrow \mathsf{H}_5 \\ (x_0,x_i,i,(b,\pi))\leftarrow\mathcal{A}(\mathsf{crs}) \end{array}\right]$$
$$\geq \frac{p_{\mathsf{cheat}}}{T} - O(2^{2n}\cdot T)\cdot(2^{-\lambda_{\mathsf{iO}}^{\delta_{\mathsf{iO}}}}+2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}) - T\cdot(2^{-\lambda_{\mathsf{PRG}}^{\delta_{\mathsf{PRG}}}}+\frac{2^{n_\pi}}{2^{m_{\mathsf{PRG}}}}). \tag{3}$$

For every $x_0^*, x_\tau^* \in \{0,1\}^n$ such that $(x_0^*, x_\tau^*, \tau) \notin \mathcal{L}$, we now define:

$$p^{\mathsf{NEQ}}_{(x_0^*,x_\tau^*)} = \Pr\left[\begin{array}{c} \mathsf{IVC}.\mathcal{V}(\mathsf{crs},(x_0,x_i,i),(b,\pi))=1 \\ \wedge\ (x_0,x_i,i)=(x_0^*,x_\tau^*,\tau) \\ \wedge\ b\neq F_{k_{\mathsf{sel}}}(x_0,x_i,i)\oplus F_{k_{\mathsf{sel}}}(x_0,x_0,0) \end{array} : \begin{array}{c} (k_{\mathsf{sel}},\mathsf{crs})\leftarrow \mathsf{H}_5 \\ (x_0,x_i,i,(b,\pi))\leftarrow\mathcal{A}(\mathsf{crs}) \end{array}\right]$$
$$p^{\mathsf{EQ}}_{(x_0^*,x_\tau^*)} = \Pr\left[\begin{array}{c} \mathsf{IVC}.\mathcal{V}(\mathsf{crs},(x_0,x_i,i),(b,\pi))=1 \\ \wedge\ (x_0,x_i,i)=(x_0^*,x_\tau^*,\tau) \\ \wedge\ b= F_{k_{\mathsf{sel}}}(x_0,x_i,i)\oplus F_{k_{\mathsf{sel}}}(x_0,x_0,0) \end{array} : \begin{array}{c} (k_{\mathsf{sel}},\mathsf{crs})\leftarrow \mathsf{H}_5 \\ (x_0,x_i,i,(b,\pi))\leftarrow\mathcal{A}(\mathsf{crs}) \end{array}\right]$$

By definition,

$$p^{\mathsf{EQ}} = \sum_{(x_0^*,x_\tau^*):(x_0^*,x_\tau^*,\tau)\notin\mathcal{L}} p^{\mathsf{EQ}}_{(x_0^*,x_\tau^*)}$$
$$p^{\mathsf{NEQ}} = \sum_{(x_0^*,x_\tau^*):(x_0^*,x_\tau^*,\tau)\notin\mathcal{L}} p^{\mathsf{NEQ}}_{(x_0^*,x_\tau^*)}.$$

We now claim that for all $(x_0^*, x_\tau^*, \tau) \notin \mathcal{L}$, we have that

Lemma 4.4. *We have that:*

$$\left|p^{\mathsf{EQ}}_{(x_0^*,x_\tau^*)} - p^{\mathsf{NEQ}}_{(x_0^*,x_\tau^*)}\right| = O(2^n\cdot T)\cdot(2^{-\lambda_{iO}^{\delta_{iO}}}+2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}).$$

Proof. In the rest of the proof, let $\mathsf{cf}_0 = x_0^*$, and define $\mathsf{cf}_i \in \{0,1\}^n$ to denote the true configuration reached from x_0^* after i steps. For readability, we will only specify the functionality of the obfuscated algorithm P on time steps $i \leq \tau - 1$ (because the functionality of P for inputs $i \geq \tau$ are not changed).

$\boxed{\mathsf{H}_6}$ Sample k_{sel} and crs as in H_5. Let $\beta \leftarrow F_{k_{\mathsf{sel}}}(\mathsf{cf}_0, x_\tau^*, \tau) \oplus F_{k_{\mathsf{sel}}}(\mathsf{cf}_0, x_0^*, 0)$. Output (β, crs).

$\boxed{\mathsf{H}_7}$ In this hybrid, sample k_{sel} as in H_5, but additionally sample a key k_{new}. We obfuscate the following prover algorithm:

> $P_{\mathsf{sk}_0,\mathsf{sk}_1,k_{\mathsf{sel}},k_{\mathsf{new}},x_0^*}(x_0, x_i, i \in \{0,1,\ldots,\tau-1\}, (b_i,\pi_i))$:
>
> - If $i = 0$, set $x_i = x_0$ and $b_i = 0$.
> - **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
> - **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i)$.
> - **Compute next proof:**
>   - If $x_0 = x_0^*$, let $b_{i+1} = b_i \oplus F_{k_{\mathsf{new}}}(x_i, x_{i+1}, i+1)$.

- Otherwise, let $b_{i+1} = b_i \oplus F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
- Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.

• Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

Sample crs as the obfuscation of the verifier program and above prover program. Sample

$$\beta \leftarrow \left(\bigoplus_{j=0}^{\tau-1} F_{k_{\mathsf{new}}}(\mathsf{cf}_j, \mathsf{cf}_{j+1}, j+1)\right) \oplus F_{k_{\mathsf{sel}}}(\mathsf{cf}_0, \mathsf{cf}_\tau, \tau) \oplus F_{k_{\mathsf{sel}}}(\mathsf{cf}_0, x^*_\tau, \tau).$$

Output (β, crs).

Lemma 4.5. *For all adversaries $\mathcal{B}$ of size at most Γ,*

$$\left|\Pr\left[\begin{array}{c} b = 1 \end{array} : \begin{array}{c} (\beta, \mathsf{crs}) \leftarrow H_7 \\ b \leftarrow \mathcal{B}(\beta, \mathsf{crs}) \end{array}\right] - \Pr\left[\begin{array}{c} b = 1 \end{array} : \begin{array}{c} (\beta, \mathsf{crs}) \leftarrow H_6 \\ b \leftarrow \mathcal{B}(\beta, \mathsf{crs}) \end{array}\right]\right|$$
$$= O(2^n \cdot T) \cdot (2^{-\lambda_{i\mathcal{O}}^{\delta_{i\mathcal{O}}}} + 2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}).$$

We defer this proof to Section 4.3.

H_8 In this hybrid, puncture k_{sel} on $(\mathsf{cf}_0, x^*_\tau, \tau)$ to get $k_{\mathsf{sel}}{}^* = k_{\mathsf{sel}}\{(\mathsf{cf}_0, x^*_\tau, \tau)\}$. Let $\gamma \leftarrow F_{k_{\mathsf{sel}}}(\mathsf{cf}_0, x^*_\tau, \tau)$. Replace k_{sel} with $k_{\mathsf{sel}}{}^*$ in the prover and verifier, and sample

$$\beta \leftarrow \left(\bigoplus_{j=0}^{\tau-1} F_{k_{\mathsf{new}}}(\mathsf{cf}_j, \mathsf{cf}_{j+1}, j+1)\right) \oplus F_{k_{\mathsf{sel}}{}^*}(\mathsf{cf}_0, \mathsf{cf}_\tau, \tau) \oplus \gamma.$$

Output (β, crs). Since $\mathsf{cf}_\tau \neq x^*_\tau$, puncturing correctness implies that the computation of β is unchanged. The verifier does not evaluate $F_{k_{\mathsf{sel}}}$, and the prover evaluates it only on inputs whose first coordinate is $x_0 \neq x^*_0 = \mathsf{cf}_0$. Thus, neither program evaluates the punctured point, so both programs are functionally equivalent to those in the previous hybrid. Therefore, indistinguishability follows from $i\mathcal{O}$ security.

H_9 Sample $\gamma \leftarrow \{0,1\}$, and sample β and crs using γ as in the previous hybrid. Indistinguishability follows from puncturable PRF security.

H_{10} Sample β as a random bit, and crs as in the previous hybrid. Output (β, crs). In the previous hybrid, γ is uniform and independent of the keys used to construct crs, so β is uniform and independent of crs. Therefore, the joint distributions of (β, crs) in both hybrids are identical.

It is easy to see that:

$$\Pr\left[\begin{array}{c} \mathsf{IVC}.\mathcal{V}(\mathsf{crs}, (x_0, x_i, i), (b, \pi)) = 1 \\ \wedge\ (x_0, x_i, i) = (x^*_0, x^*_\tau, \tau) \\ \wedge\ b = \beta \end{array} : \begin{array}{c} (\beta, \mathsf{crs}) \leftarrow \mathsf{H}_{10} \\ (x_0, x_i, i, (b, \pi)) \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right]$$
$$= \Pr\left[\begin{array}{c} \mathsf{IVC}.\mathcal{V}(\mathsf{crs}, (x_0, x_i, i), (b, \pi)) = 1 \\ \wedge\ (x_0, x_i, i) = (x^*_0, x^*_\tau, \tau) \\ \wedge\ b \neq \beta \end{array} : \begin{array}{c} (\beta, \mathsf{crs}) \leftarrow \mathsf{H}_{10} \\ (x_0, x_i, i, (b, \pi)) \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right] \tag{4}$$

since β is uniform and independent of crs. Moreover, by the indistinguishability of H_6 and H_{10}, we have that for $\rho \in \{0,1\}$:

$$\left| \Pr\left[\begin{array}{c} \mathsf{IVC}.\mathcal{V}(\mathsf{crs},(x_0,x_i,i),(b,\pi))=1 \\ \wedge\ (x_0,x_i,i)=(x_0^*,x_\tau^*,\tau) \\ \wedge\ b=\beta\oplus\rho \end{array} : \begin{array}{c} (\beta,\mathsf{crs})\leftarrow \mathsf{H}_6 \\ (x_0,x_i,i,(b,\pi))\leftarrow \mathcal{A}(\mathsf{crs}) \end{array} \right] \right.$$
$$\left. - \Pr\left[\begin{array}{c} \mathsf{IVC}.\mathcal{V}(\mathsf{crs},(x_0,x_i,i),(b,\pi))=1 \\ \wedge\ (x_0,x_i,i)=(x_0^*,x_\tau^*,\tau) \\ \wedge\ b=\beta\oplus\rho \end{array} : \begin{array}{c} (\beta,\mathsf{crs})\leftarrow \mathsf{H}_{10} \\ (x_0,x_i,i,(b,\pi))\leftarrow \mathcal{A}(\mathsf{crs}) \end{array} \right] \right|$$
$$\leq O(2^n\cdot T)\cdot(2^{-\lambda_{\mathsf{iO}}^{\delta_{\mathsf{iO}}}}+2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}). \tag{5}$$

Therefore, by combining (4) and (5), we have via a hybrid argument that:

$$\left| p^{\mathsf{EQ}}_{(x_0^*,x_\tau^*)} - p^{\mathsf{NEQ}}_{(x_0^*,x_\tau^*)} \right| = O(2^n\cdot T)\cdot(2^{-\lambda_{\mathsf{iO}}^{\delta_{\mathsf{iO}}}}+2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}),$$

giving us our claim. □

Therefore, we have that

$$\begin{aligned} \left| p^{\mathsf{EQ}} - p^{\mathsf{NEQ}} \right| &= \left| \sum_{(x_0^*,x_\tau^*):(x_0^*,x_\tau^*,\tau)\notin\mathcal{L}} \left(p^{\mathsf{EQ}}_{(x_0^*,x_\tau^*)} - p^{\mathsf{NEQ}}_{(x_0^*,x_\tau^*)} \right) \right| \\ &\leq \sum_{(x_0^*,x_\tau^*):(x_0^*,x_\tau^*,\tau)\notin\mathcal{L}} \left| p^{\mathsf{EQ}}_{(x_0^*,x_\tau^*)} - p^{\mathsf{NEQ}}_{(x_0^*,x_\tau^*)} \right| \\ &\leq O(2^{3n}\cdot T)\cdot(2^{-\lambda_{\mathsf{iO}}^{\delta_{\mathsf{iO}}}}+2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}). \end{aligned} \tag{6}$$

By combining (2) and (3) with (6), we have that:

$$p_{\mathsf{cheat}} \leq T\cdot\left(O(2^{3n}\cdot T)\cdot(2^{-\lambda_{\mathsf{iO}}^{\delta_{\mathsf{iO}}}}+2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}) + 2T\cdot\left(2^{-\lambda_{\mathsf{PRG}}^{\delta_{\mathsf{PRG}}}}+\frac{2^{n_\pi}}{2^{m_{\mathsf{PRG}}}}\right)\right) = \mathsf{negl}(\lambda).$$

by our choice of parameters. This completes the proof of security. □

4.1 Proof of Theorem 4.2

We order pairs $(x,y)\in\{0,1\}^n\times\{0,1\}^n$ lexicographically. We write $(i_1,x_1,y_1) < (i_2,x_2,y_2)$ if $i_1<i_2$, or if $i_1=i_2$ and $(x_1,y_1)<(x_2,y_2)$. We use $\leq$ for the corresponding order with equality of the triples included.

We index the hybrids by $(j,y_0,y_j)\in[\tau]\times\{0,1\}^n\times\{0,1\}^n$. Since $j\geq 1$, the punctured point (y_0,y_j,j) is always distinct from the root-mask point $(y_0,y_0,0)$.

$\mathsf{H}_{1,1}{}^{(j,y_0,y_j)}$ In this hybrid, the CRS contains obfuscations of the following programs.

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}}}(x_0,x_i,i,(b,\pi))$:

- If $(i,x_0,x_i)<(j,y_0,y_j)$,
  - If $b=F_{k_{\mathsf{sel}}}(x_0,x_i,i)\oplus F_{k_{\mathsf{sel}}}(x_0,x_0,0)$ and $\pi=F_{\mathsf{sk}_b}(x_0,x_i,i)$, accept.
- Else,
  - If $b=F_{\mathsf{sk}_{\mathsf{sel}}}(x_0,x_i,i)$ and $\pi=F_{\mathsf{sk}_b}(x_0,x_i,i)$, accept. Else, reject.

$P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}}}(x_0, x_i, i, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i)$.
- **Compute next proof:**
  - If $(i+1, x_0, x_{i+1}) < (j, y_0, y_j)$, let $b_{i+1} = F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$
  - Else, let $b_{i+1} = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
  - Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

Output crs.

$\mathsf{H}_{1,2}{}^{(j,y_0,y_j)}$ Sample

- $\gamma_1 \leftarrow F_{\mathsf{sk}_{\mathsf{sel}}}(y_0, y_j, j)$, and let $\mathsf{sk}_{\mathsf{sel}}{}^* := \mathsf{sk}_{\mathsf{sel}}\{(y_0, y_j, j)\}$ be the corresponding punctured point.
- $\gamma_2 \leftarrow F_{k_{\mathsf{sel}}}(y_0, y_j, j)$ and $k_{\mathsf{sel}}{}^* := k_{\mathsf{sel}}\{(y_0, y_j, j)\}$.

We now define the programs:

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}}{}^*,k_{\mathsf{sel}}{}^*,\tau}(x_0, x_i, i, (b, \pi))$:

- If $(i, x_0, x_i) < (j, y_0, y_j)$,
  - If $b \neq F_{k_{\mathsf{sel}}{}^*}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}{}^*}(x_0, x_0, 0)$, output 0.
- Else, if $(i, x_0, x_i) = (j, y_0, y_j)$, if $b \neq \gamma_1$, output 0.
- Else,
  - If $b \neq F_{\mathsf{sk}_{\mathsf{sel}}{}^*}(x_0, x_i, i)$, output 0.
- If $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

$P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}}{}^*,k_{\mathsf{sel}}{}^*}(x_0, x_i, i, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i)$.
- **Compute next proof:**
  - If $(i+1, x_0, x_{i+1}) < (j, y_0, y_j)$, let $b_{i+1} = F_{k_{\mathsf{sel}}{}^*}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}{}^*}(x_0, x_0, 0)$.
  - Else, if $(i+1, x_0, x_{i+1}) = (j, y_0, y_j)$, let $b_{i+1} = \gamma_1$.
  - Else, let $b_{i+1} = F_{\mathsf{sk}_{\mathsf{sel}}{}^*}(x_0, x_{i+1}, i+1)$.
  - Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

$\mathsf{H}_{1,3}^{(j,y_0,y_j)}$ Replace γ_1 and γ_2 with random bits, and sample the rest of the CRS as in the previous hybrid using γ_1. Output crs. Indistinguishability follows from the puncturable PRF security.

$\mathsf{H}_{1,4}^{(j,y_0,y_j)}$ Replace $\gamma_1 \leftarrow \gamma_2 \oplus F_{k_{\mathsf{sel}}}(y_0, y_0, 0)$. Sample the CRS as in the previous hybrid using γ_1. Output crs. It is easy to see that γ_1 in this hybrid has an identical distribution to the previous hybrid. Therefore, the hybrids are indistinguishable.

$\mathsf{H}_{1,5}^{(j,y_0,y_j)}$ Replace $\gamma_1 \leftarrow F_{k_{\mathsf{sel}}}(y_0, y_j, j) \oplus F_{k_{\mathsf{sel}}}(y_0, y_0, 0)$. Sample the CRS as in the previous hybrid using γ_1. Output crs. Indistinguishability follows from punctured PRF security.

$\mathsf{H}_{1,6}^{(j,y_0,y_j)}$ Unpuncture the keys $\mathsf{sk}_{\mathsf{sel}}$ and k_{sel}, and substitute and define the programs as follows:

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau}(x_0, x_i, i, (b, \pi))$:

- If $(i, x_0, x_i) < (j, y_0, y_j)$,
  - If $b \neq F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, output 0.
- Else, if $(i, x_0, x_i) = (j, y_0, y_j)$, if $b \neq F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, output 0.
- Else,
  - If $b \neq F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$, output 0.
- If $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

$P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}}}(x_0, x_i, i, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:**
  Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i)$.
- **Compute next proof:**
  - If $(i+1, x_0, x_{i+1}) < (j, y_0, y_j)$, let $b_{i+1} = F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$.
  - Else, if $(i+1, x_0, x_{i+1}) = (j, y_0, y_j)$, let $b_{i+1} = F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$.
  - Else, let $b_{i+1} = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
  - Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

Output the obfuscations of the above programs as the CRS. Indistinguishability follows from the functionality equivalence of the programs in both hybrids.

$\mathsf{H}_{1,7}^{(j,y_0,y_j)}$ Construct the following programs:

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau}(x_0, x_i, i, (b, \pi))$:

- If $(i, x_0, x_i) \boxed{\leq} (j, y_0, y_j)$,
  - If $b \neq F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, output 0.
- Else,
  - If $b \neq F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$, output 0.
- If $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

$P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}}}(x_0, x_i, i, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i)$.
- **Compute next proof:**
  - If $(i+1, x_0, x_{i+1}) \boxed{\leq} (j, y_0, y_j)$, let $b_{i+1} = F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$.
  - Else, let $b_{i+1} = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
  - Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

Output the obfuscations of the above programs as the CRS. Indistinguishability follows from the functionality equivalence and the $i\mathcal{O}$ security.

It is easy to see that $\mathsf{H}_1 \approx_c \mathsf{H}_{1,1}{}^{(1,0^n,0^n)}$. Indeed, the verifier input time i and the prover output time $i+1$ are both at least 1, so the strict cutoff in the first hybrid never applies. Moreover, suppose (ℓ, z_0, z_ℓ) directly follows (j, y_0, y_j) in lexicographic order. Then, $\mathsf{H}_{1,7}{}^{(j,y_0,y_j)} \approx_c \mathsf{H}_{1,1}{}^{(\ell,z_0,z_\ell)}$. Finally, we have that $\mathsf{H}_2 \approx_c \mathsf{H}_{1,7}{}^{(\tau,1^n,1^n)}$. Therefore, we can deduce Theorem 4.2 by a hybrid argument. These transitions follow from functional equivalence and $i\mathcal{O}$ security.

4.2 Proof of Theorem 4.3

We order pairs $(x, y) \in \{0,1\}^n \times \{0,1\}^n$ lexicographically. We write $(i_1, x_1, y_1) < (i_2, x_2, y_2)$ if $i_1 < i_2$, or if $i_1 = i_2$ and $(x_1, y_1) < (x_2, y_2)$. We use $\leq$ for the corresponding order with equality of the triples included.

In all of the following hybrids, we specify only the verifier algorithm, since the prover algorithm and k_{sel} remain unchanged.

$\boxed{\mathsf{H}_{4,1}{}^{(j)}}$ For $j \in [\tau]$, define the verifier algorithm as follows:

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau,j}(x_0, x_i, i, (b, \pi))$:

- If $i \leq j$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept.
- If $j < i \leq \tau$,
  - If $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

- If $i > \tau$,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

$\mathsf{H}_{4,2}^{(j)}$ Sample a string $z \leftarrow \{0,1\}^{m_{\mathsf{PRG}}}$. Sample the verifier algorithm as:

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau,j,z}(x_0, x_i, i, (b, \pi))$:

- If $i < j$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept.
- If $i = j$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
  - Else, accept if and only if $z = G(\pi \oplus F_{\mathsf{sk}_b}(x_0, x_i, i))$.
- If $j < i \leq \tau$,
  - If $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.
- If $i > \tau$,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

With probability $1 - \frac{2^{n_\pi}}{2^{m_{\mathsf{PRG}}}}$, we have that z is not in the image of G. In this case, the program has identical functionality, so indistinguishability follows from $i\mathcal{O}$ security.

$\mathsf{H}_{4,3}^{(j)}$ Sample $y \leftarrow \{0,1\}^{n_\pi}$, and $z = G(y)$. Sample the rest of the CRS as in the previous hybrid. Indistinguishability follows from PRG security.

$\mathsf{H}_{4,4}^{(j)}$ Sample the verifier algorithm as:

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau,j,y}(x_0, x_i, i, (b, \pi))$:

- If $i < j$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept.
- If $i = j$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
  - Else, accept if and only if $y = \pi \oplus F_{\mathsf{sk}_b}(x_0, x_i, i)$.
- If $j < i \leq \tau$,
  - If $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.
- If $i > \tau$,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

This program has identical functionality as the program in the previous hybrid since G is injective. Therefore, indistinguishability follows from $i\mathcal{O}$ security.

$\mathsf{H}_{4,5}^{(j)}$ In this hybrid, the verifier program is sampled as

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau,j,z}(x_0,x_i,i,(b,\pi))$:

- If $i<j$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0,x_i,i) \oplus F_{k_{\mathsf{sel}}}(x_0,x_0,0)$ and $\pi = F_{\mathsf{sk}_b}(x_0,x_i,i)$, accept.
- If $i=j$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0,x_i,i) \oplus F_{k_{\mathsf{sel}}}(x_0,x_0,0)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0,x_i,i)$.
  - Else, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0,x_i,i)$.
- If $j<i\le\tau$,
  - If $\pi = F_{\mathsf{sk}_b}(x_0,x_i,i)$, accept. Else, reject.
- If $i>\tau$,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0,x_i,i)$ and $\pi = F_{\mathsf{sk}_b}(x_0,x_i,i)$, accept. Else, reject.

We claim that this is indistinguishable from the previous hybrid via $O(2^{2n})$ invocations of $i\mathcal{O}$ and puncturable PRF security. We defer the proof of indistinguishability of this hybrid for now.

$\mathsf{H}_{4,6}{}^{(j)}$ In this hybrid, we argue that

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau,j}(x_0,x_i,i,(b,\pi))$:

- If $i<j$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0,x_i,i) \oplus F_{k_{\mathsf{sel}}}(x_0,x_0,0)$ and $\pi = F_{\mathsf{sk}_b}(x_0,x_i,i)$, accept.
- If $i=j$,
  - If $\pi = F_{\mathsf{sk}_b}(x_0,x_i,i)$, accept. Else, reject.
- If $j<i\le\tau$,
  - If $\pi = F_{\mathsf{sk}_b}(x_0,x_i,i)$, accept. Else, reject.
- If $i>\tau$,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0,x_i,i)$ and $\pi = F_{\mathsf{sk}_b}(x_0,x_i,i)$, accept. Else, reject.

Indistinguishability follows from $i\mathcal{O}$ security.

By construction, the programs in $\mathsf{H}_{4,1}{}^{(\tau)}$ are functionally equivalent to those in H_4. Moreover, for every $j\in\{2,\ldots,\tau\}$, the programs in $\mathsf{H}_{4,6}{}^{(j)}$ are functionally equivalent to those in $\mathsf{H}_{4,1}{}^{(j-1)}$. Finally, the programs in $\mathsf{H}_{4,6}{}^{(1)}$ are functionally equivalent to those in H_5. Therefore, by $i\mathcal{O}$ security, applying the above sequence for $j=\tau,\tau-1,\ldots,1$ gives us our claim. Now, it suffices to argue indistinguishability of $\mathsf{H}_{4,4}$ and $\mathsf{H}_{4,5}$.

$\mathsf{G}_1{}^{(j,y_0,y_j)}$ We define the verifier program in this hybrid to be:

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau,j,y,y_0,y_j}(x_0,x_i,i,(b,\pi))$:

- If $i<j$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0,x_i,i) \oplus F_{k_{\mathsf{sel}}}(x_0,x_0,0)$ and $\pi = F_{\mathsf{sk}_b}(x_0,x_i,i)$, accept.
- If $i=j$,

- If $(x_0, x_j) < (y_0, y_j)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
- Else,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
  - Else, accept if and only if $y = \pi \oplus F_{\mathsf{sk}_b}(x_0, x_i, i)$.

- If $j < i \leq \tau$,
  - If $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.
- If $i > \tau$,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

$\mathsf{G}_2^{(j,y_0,y_j)}$ Let $\beta \leftarrow F_{k_{\mathsf{sel}}}(y_0, y_j, j) \oplus F_{k_{\mathsf{sel}}}(y_0, y_0, 0)$. For $j \geq 2$, any accepted proof (b, π) for a predecessor statement $(y_0, y_{j-1}, j-1)$ must satisfy that $b = F_{k_{\mathsf{sel}}}(y_0, y_{j-1}, j-1) \oplus F_{k_{\mathsf{sel}}}(y_0, y_0, 0)$. For $j = 1$, the prover starts at y_0 with $b_0 = 0$, so its selector for any output $(y_0, y_1, 1)$ is $F_{k_{\mathsf{sel}}}(y_0, y_1, 1) \oplus F_{k_{\mathsf{sel}}}(y_0, y_0, 0) = \beta$. Thus, in both cases, the only proofs produced by the prover algorithm for (y_0, y_j, j) correspond to β, and the prover algorithm never uses the value $F_{\mathsf{sk}_{1-\beta}}(y_0, y_j, j)$. Puncture $\mathsf{sk}_{1-\beta}$ on (y_0, y_j, j) to get key $\mathsf{sk}^*_{1-\beta}$, and let γ be the corresponding value. For convenience, write $\mathsf{sk}^*_\beta := \mathsf{sk}_\beta$. Use $\mathsf{sk}^*_{1-\beta}$ in the prover algorithm, and construct the following verifier algorithm:

$V_{\mathsf{sk}_0{}^*, \mathsf{sk}_1{}^*, \mathsf{sk}_{\mathsf{sel}}, k_{\mathsf{sel}}, \tau, j, y, y_0, y_j}(x_0, x_i, i, (b, \pi))$:

- If $i < j$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$ and $\pi = F_{\mathsf{sk}_b{}^*}(x_0, x_i, i)$, accept.
- If $i = j$,
  - If $(x_0, x_j) < (y_0, y_j)$, accept if and only if $\pi = F_{\mathsf{sk}_b{}^*}(x_0, x_i, i)$.
  - If $(x_0, x_j) = (y_0, y_j)$,
    - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b{}^*}(x_0, x_i, i)$.
    - Else, accept if and only if $y = \pi \oplus \gamma$.
  - Else,
    - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b{}^*}(x_0, x_i, i)$.
    - Else, accept if and only if $y = \pi \oplus F_{\mathsf{sk}_b{}^*}(x_0, x_i, i)$.
- If $j < i \leq \tau$,
  - If $\pi = F_{\mathsf{sk}_b{}^*}(x_0, x_i, i)$, accept. Else, reject.
- If $i > \tau$,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b{}^*}(x_0, x_i, i)$, accept. Else, reject.

Indistinguishability follows via $i\mathcal{O}$ security.

$\mathsf{G}_3^{(j,y_0,y_j)}$ Sample $\rho \leftarrow \{0,1\}^{n_\pi}$, and use $\gamma = \rho$ to sample the CRS as in the previous hybrid. Indistinguishability follows from puncturable PRF security.

$\mathsf{G}_4^{(j,y_0,y_j)}$ Sample $\rho \leftarrow \{0,1\}^{n_\pi}$, and use $\gamma = \rho \oplus y$ to sample the CRS as in the previous hybrid. The distribution in this hybrid is identical to the previous.

$\mathsf{G}_5^{(j,y_0,y_j)}$ Sample $\rho \leftarrow F_{\mathsf{sk}_{1-\beta}}(y_0, y_j, j)$, and sample the rest of the hybrid using ρ as in the previous hybrid. Indistinguishability follows from puncturable PRF security.

$\mathsf{G}_6^{(j,y_0,y_j)}$ Unpuncture $\mathsf{sk}_{1-\beta}$ in both programs, and sample the verifier algorithm as follows:

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau,j,y,y_0,y_j}(x_0, x_i, i, (b, \pi))$:

- If $i < j$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept.
- If $i = j$,
  - If $(x_0, x_j) < (y_0, y_j)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
  - If $(x_0, x_j) = (y_0, y_j)$,
    - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
    - Else, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
  - Else,
    - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
    - Else, accept if and only if $y = \pi \oplus F_{\mathsf{sk}_b}(x_0, x_i, i)$.
- If $j < i \leq \tau$,
  - If $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.
- If $i > \tau$,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

Indistinguishability follows via $i\mathcal{O}$ security and by substituting $\gamma = y \oplus F_{\mathsf{sk}_{1-\beta}}(y_0, y_j, j)$.

$\mathsf{G}_7^{(j,y_0,y_j)}$ In this hybrid, sample the verifier algorithm as follows:

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau,j,y,y_0,y_j}(x_0, x_i, i, (b, \pi))$:

- If $i < j$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept.
- If $i = j$,
  - If $(x_0, x_j) \boxed{\leq} (y_0, y_j)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
  - Else,
    - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
    - Else, accept if and only if $y = \pi \oplus F_{\mathsf{sk}_b}(x_0, x_i, i)$.
- If $j < i \leq \tau$,
  - If $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

- If $i > \tau$,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

Indistinguishability follows via $i\mathcal{O}$ security.

Since $\mathsf{G}_1^{(j,0^n,0^n)} \approx_c \mathsf{H}_{4,4}$, and $\mathsf{G}_7^{(j,1^n,1^n)} \approx_c \mathsf{H}_{4,5}$, this completes the proof. These transitions follow from functional equivalence and $i\mathcal{O}$ security.

4.3 Proof of Theorem 4.5

Fix $y_0 = x_0^*$. We consider hybrids indexed by $(j, y_j) \in \{0, 1, \ldots, \tau - 1\} \times \{0,1\}^n$, ordered lexicographically. Thus, $(i, x) < (j, y)$ if $i < j$, or if $i = j$ and $x < y$ in lexicographic order. We write $(i, x) \leq (j, y)$ if $(i, x) < (j, y)$ or $(i, x) = (j, y)$.

$\mathsf{H}_{6,1}^{(j,y_j)}$ In this hybrid, the CRS obfuscates the same verifier program as in H_6, but the following prover algorithm:

$P_{\mathsf{sk}_0,\mathsf{sk}_1,k_{\mathsf{sel}},k_{\mathsf{new}},x_0^*}(x_0, x_i, i \in \{0, 1, \ldots, \tau - 1\}, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$ and $b_i = 0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i)$.
- **Compute next proof:**
  - If $x_0 = x_0^*$ and $(i, x_i) < (j, y_j)$: Let $b_{i+1} = b_i \oplus F_{k_{\mathsf{new}}}(x_i, x_{i+1}, i+1)$.
  - Else, $b_{i+1} \leftarrow b_i \oplus F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
  - Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

We sample β depending on whether $y_j \leq \mathsf{cf}_j$. If yes, we sample (β, crs) as follows:

$$\beta = \left(\bigoplus_{0 \leq \alpha \leq j-1} F_{k_{\mathsf{new}}}(\mathsf{cf}_\alpha, \mathsf{cf}_{\alpha+1}, \alpha+1) \right) \oplus F_{k_{\mathsf{sel}}}(\mathsf{cf}_0, \mathsf{cf}_j, j) \oplus F_{k_{\mathsf{sel}}}(\mathsf{cf}_0, x_\tau^*, \tau).$$

Else, we define

$$\beta = \left(\bigoplus_{0 \leq \alpha \leq j} F_{k_{\mathsf{new}}}(\mathsf{cf}_\alpha, \mathsf{cf}_{\alpha+1}, \alpha+1) \right) \oplus F_{k_{\mathsf{sel}}}(\mathsf{cf}_0, \mathsf{cf}_{j+1}, j+1) \oplus F_{k_{\mathsf{sel}}}(\mathsf{cf}_0, x_\tau^*, \tau).$$

Output (β, crs).

$\mathsf{H}_{6,2}^{(j,y_j)}$ In this hybrid, let y_{j+1} denote the configuration after y_j. We will puncture the keys:

- k_{sel} on the point (x_0^*, y_j, j) to get ${k_{\mathsf{sel}}}^* = k_{\mathsf{sel}}\{x_0^*, y_j, j\}$, and let $\gamma_1 \leftarrow F_{k_{\mathsf{sel}}}(x_0^*, y_j, j)$.

- k_{new} on the point $(y_j, y_{j+1}, j+1)$ to get ${k_{\mathsf{new}}}^* = k_{\mathsf{new}}\{y_j, y_{j+1}, j+1\}$. Let $\gamma_2 \leftarrow F_{k_{\mathsf{new}}}(y_j, y_{j+1}, j+1)$.

Replace the keys in the prover algorithm with the punctured keys, and hardcode the bit γ_1 where necessary.

$P_{\mathsf{sk}_0,\mathsf{sk}_1,{k_{\mathsf{sel}}}^*,{k_{\mathsf{new}}}^*,x_0^*}(x_0, x_i, i \in \{0,1,\ldots,\tau-1\}, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$ and $b_i = 0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i)$.
- **Compute next proof:**
  - If $x_0 = x_0^*$ and $(i, x_i) < (j, y_j)$: Let $b_{i+1} = b_i \oplus F_{{k_{\mathsf{new}}}^*}(x_i, x_{i+1}, i+1)$.
  - Else if $x_0 = x_0^*$ and $(i, x_i) = (j, y_j)$: Let $b_{i+1} = b_i \oplus \gamma_1 \oplus F_{{k_{\mathsf{sel}}}^*}(x_0^*, y_{j+1}, j+1)$.
  - Else, $b_{i+1} \leftarrow b_i \oplus F_{{k_{\mathsf{sel}}}^*}(x_0, x_i, i) \oplus F_{{k_{\mathsf{sel}}}^*}(x_0, x_{i+1}, i+1)$.
  - Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

Note that γ_2 is not yet used in the algorithm.

If $y_j \neq \mathsf{cf}_j$, then we sample β as in the previous hybrid. Otherwise, replace $F_{k_{\mathsf{sel}}}(x_0^*, y_j, j)$ in the computation of β with γ_1. Output (β, crs). Indistinguishability follows from puncturable PRF correctness and $i\mathcal{O}$ security.

$\mathsf{H}_{6,3}{}^{(j,y_j)}$ In this hybrid, sample $\gamma_1, \gamma_2 \leftarrow \{0,1\}$. Sample the β and crs using γ_1 just as in the previous hybrid. Output (β, crs). Indistinguishability follows from puncturable PRF security.

$\mathsf{H}_{6,4}{}^{(j,y_j)}$ Redefine $\gamma_1 = \gamma_2 \oplus F_{k_{\mathsf{sel}}}(x_0^*, y_{j+1}, j+1)$. Use γ_1 to sample β and crs as in the previous hybrid. Output (β, crs). Indistinguishability follows from the fact that the distributions of γ_1 in both hybrids are identical.

$\mathsf{H}_{6,5}{}^{(j,y_j)}$ Sample $\gamma_2 \leftarrow F_{k_{\mathsf{new}}}(y_j, y_{j+1}, j+1)$. Use γ_2 to sample β and crs as in the previous hybrid. Indistinguishability follows from the puncturable PRF security of k_{new}.

$\mathsf{H}_{6,6}{}^{(j,y_j)}$ Unpuncture the keys k_{sel} and k_{new}. Let the prover algorithm be the following:

$P_{\mathsf{sk}_0,\mathsf{sk}_1,{k_{\mathsf{sel}}}^*,{k_{\mathsf{new}}}^*,x_0^*}(x_0, x_i, i \in \{0,1,\ldots,\tau-1\}, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$ and $b_i = 0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i)$.
- **Compute next proof:**
  - If $x_0 = x_0^*$ and $(i, x_i) < (j, y_j)$: Let $b_{i+1} = b_i \oplus F_{k_{\mathsf{new}}}(x_i, x_{i+1}, i+1)$.
  - Else if $x_0 = x_0^*$ and $(i, x_i) = (j, y_j)$: Let $b_{i+1} = b_i \oplus F_{k_{\mathsf{new}}}(y_j, y_{j+1}, j+1)$.
  - Else, $b_{i+1} \leftarrow b_i \oplus F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.

– Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.

- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

Sample β as in the previous hybrid. Indistinguishability follows from substituting $\gamma_1 \oplus F_{k_{\mathsf{sel}}}(x_0^*, y_{j+1}, j+1) = F_{k_{\mathsf{new}}}(y_j, y_{j+1}, j+1)$.

$\mathsf{H}_{6,7}{}^{(j,y_j)}$ Sample the prover algorithm be the following:

$P_{\mathsf{sk}_0,\mathsf{sk}_1,k_{\mathsf{sel}},k_{\mathsf{new}},x_0^*}(x_0, x_i, i \in \{0,1,\ldots,\tau-1\}, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$ and $b_i = 0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i)$.
- **Compute next proof:**
  - If $x_0 = x_0^*$ and $(i, x_i) \leq (j, y_j)$: Let $b_{i+1} = b_i \oplus F_{k_{\mathsf{new}}}(x_i, x_{i+1}, i+1)$.
  - Else, $b_{i+1} \leftarrow b_i \oplus F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
  - Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

Sample β as in the previous hybrid. Indistinguishability follows from $i\mathcal{O}$ security.

It is easy to see that $\mathsf{H}_6 \approx_c \mathsf{H}_{6,1}{}^{(0,0^n)}$. Moreover, suppose (ℓ, z_ℓ) directly follows (j, y_j) in lexicographic order. Then, $\mathsf{H}_{6,7}{}^{(j,y_j)} \approx_c \mathsf{H}_{6,1}{}^{(\ell,z_\ell)}$. Finally, we have that $\mathsf{H}_7 \approx_c \mathsf{H}_{6,7}{}^{(\tau-1,1^n)}$. There are $\tau \cdot 2^n$ choices of (j, y_j) and constantly many transitions for each choice. Therefore, we can deduce Theorem 4.5 by a hybrid argument with total loss $O(2^n \cdot T) \cdot (2^{-\lambda_{\mathsf{iO}}^{\delta_{\mathsf{iO}}}} + 2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}})$. These transitions follow from functional equivalence and $i\mathcal{O}$ security.

5 Two-Hop IVC for NP with Non-Adaptive Soundness

Ingredients.

- A $2^{\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}$-secure puncturable PRFs $(\mathsf{PPRF.Gen}, \mathsf{PPRF.Punc}, \mathsf{PPRF.Eval})$. We will use $F_K(x)$ as shorthand for $\mathsf{PPRF.Eval}(K, x)$ for readability.
- A $2^{\lambda_{\mathsf{iO}}^{\delta_{\mathsf{iO}}}}$-secure indistinguishability obfuscation $i\mathcal{O}$.
- A $2^{\lambda_{\mathsf{PRG}}^{\delta_{\mathsf{PRG}}}}$-secure injective PRG $G : \{0,1\}^{n_\pi} \to \{0,1\}^{m_{\mathsf{PRG}}}$ constructed via Theorem 3.3. While this PRG will not appear in the construction, we will crucially rely on it in our security analysis.

We are now ready to describe the algorithm. We will choose the specific parameters after first describing the algorithm. Throughout this construction and its hybrid argument, we pad all verifier circuits to a common size before obfuscation, and likewise all prover circuits, including their embedded obfuscated verifiers. These sizes are polynomial in n, m, λ.

- $\mathsf{IVC.Gen}(1^\lambda, 1^n, 1^m, \mathcal{M})$:
  1. Sample puncturable PRF keys $\mathsf{sk}_0, \mathsf{sk}_1, \mathsf{sk}_{\mathsf{sel}} \leftarrow \mathsf{PPRF.Gen}(1^\lambda_{\mathsf{PPRF}})$. We will treat the output of $\mathsf{PPRF.Eval}(\mathsf{sk}_b, \cdot)$ to be a string in $\{0,1\}^{n_\pi}$, and the output of $\mathsf{PPRF.Eval}(\mathsf{sk}_{\mathsf{sel}}, \cdot)$ to be a single bit.
  2. Construct the following verification algorithm $V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}}}$:

     $V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}}}(x_0 \in \{0,1\}^n, x_i \in \{0,1\}^n, i \in [2], (b, \pi))$:
     - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

  3. Let $\widetilde{V} \leftarrow i\mathcal{O}(1^{\lambda_{\mathsf{iO}}}, V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}}})$.
  4. Construct the following prover algorithm $P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}}}$:

     $P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}}}(x_0 \in \{0,1\}^n, x_i \in \{0,1\}^n, w_{i+1} \in \{0,1\}^m, i \in \{0,1\}, (b_i, \pi_i))$:
     - If $i = 0$, set $x_i = x_0$.
     - **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
     - **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i, w_{i+1})$.
     - **Compute next proof:**
       - Let $b_{i+1} = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
       - Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
     - Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

  5. Let $\widetilde{P} \leftarrow i\mathcal{O}(1^{\lambda_{\mathsf{iO}}}, P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}}})$.
  6. Output $\mathsf{crs} = (\widetilde{P}, \widetilde{V})$.
- $\mathsf{IVC.Prove}(\mathsf{crs} = (\widetilde{P}, \widetilde{V}), (x_0, x_i, i), (b_i, \pi_i), w_{i+1})$: Output $\widetilde{P}(x_0, x_i, w_{i+1}, i, (b_i, \pi_i))$.
- $\mathsf{IVC.Verify}(\mathsf{crs} = (\widetilde{P}, \widetilde{V}), (x_0, x_i, i), (b_i, \pi_i))$: Output $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i))$.

We choose the following parameters:

- Choose $\lambda_{\mathsf{PPRF}}, \lambda_{\mathsf{iO}}$ so that $O(2^{2n}) \cdot (2^{-\lambda_{\mathsf{iO}}^{\delta_{\mathsf{iO}}}} + 2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}) \leq \mathsf{negl}(\lambda)$.
- Choose λ_{PRG} so that $2^{-\lambda_{\mathsf{PRG}}^{\delta_{\mathsf{PRG}}}} \leq \mathsf{negl}(\lambda)$.
- Choose the PRG G so that $n_\pi = \mathsf{poly}(\lambda)$ with $n_\pi > \lambda$, and $m_{\mathsf{PRG}} = \mathsf{poly}(n_\pi)$ with $m_{\mathsf{PRG}} > n_\pi + \lambda$.

Theorem 5.1. *Assuming subexponentially secure $i\mathcal{O}$ and one-way functions, there exists a non-adaptively sound IVC scheme for two-hop* NP *with:*

- *CRS size* $\mathsf{poly}(n, m, \lambda)$,
- *prover and verifier runtime per update is* $\mathsf{poly}(n, m, \lambda)$,
- *and proof size* $\mathsf{poly}(\lambda)$.

Proof. Suppose there exists a polynomial-sized adversary $\mathcal{A}$ which produces cheating proofs for some $(x_0^*, x_i^*, i) \notin \mathcal{L}$ with probability p_{cheat}. If $i = 1$, one can do a straightforward puncturing argument as in [SW14] to argue that $p_{\mathsf{cheat}} \leq \mathsf{negl}(\lambda)$. Therefore, we will assume that $i = 2$. We write the instance as $(x_0^*, x_2^*, 2)$.

H_1 In this experiment, generate the crs as is done in the real IVC scheme.

H_2 In this experiment, sample $k_{\mathsf{sel}} \leftarrow \mathsf{PPRF.Gen}(1^{\lambda_{\mathsf{PPRF}}})$. Construct the machine $V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}}}$:

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}}}(x_0, x_i, i \in [2], (b, \pi))$:

- If $i = 1$ and $x_0 = x_0^*$ or $(x_0, x_i, i) = (x_0^*, x_2^*, 2)$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept.
- Else,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

Let $\widetilde{V} \leftarrow i\mathcal{O}(1^{\lambda_{\mathsf{iO}}}, V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau})$. Construct the following prover algorithm $P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau}$:

$P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau}(x_0, x_i, w_{i+1} \in \{0,1\}^m, i \in \{0,1\}, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i, w_{i+1})$.
- **Compute next proof:**
  - If $i = 0$ and $x_0 = x_0^*$, let $b_{i+1} = F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$.
  - Else if $i = 1$ and $(x_0, x_2) = (x_0^*, x_2^*)$, let $b_{i+1} = F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$.
  - Else, let $b_{i+1} = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
  - Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

Sample the CRS as the obfuscations of the above programs.

Lemma 5.2. *For all polynomial-sized adversaries $\mathcal{B}$, we have that*

$$\left|\Pr\left[\begin{array}{c} b = 1 \\ : \\ \end{array}\begin{array}{c} \mathsf{crs} \leftarrow \mathsf{H}_2 \\ b \leftarrow \mathcal{B}(\mathsf{crs}) \end{array}\right] - \Pr\left[\begin{array}{c} b = 1 \\ : \\ \end{array}\begin{array}{c} \mathsf{crs} \leftarrow \mathsf{H}_1 \\ b \leftarrow \mathcal{B}(\mathsf{crs}) \end{array}\right]\right| = O(2^n) \cdot (2^{-\lambda_{\mathsf{iO}}^{\delta_{\mathsf{iO}}}} + 2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}).$$

We defer the proof to Section 5.1.

H_3 In this experiment,

- Construct the verification algorithm as in the previous hybrid.
- Construct the following prover algorithm $P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau}$:

$P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}}}(x_0, x_i, w_{i+1} \in \{0,1\}^m, i \in \{0,1\}, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i, w_{i+1})$.
- **Compute next proof:**
  - If $i = 0$ and $x_0 = x_0^*$, let $b_{i+1} = F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$.
  - Else if $i = 1$ and $(x_0, x_2) = (x_0^*, x_2^*)$, let $b_{i+1} = b_i \oplus F_{k_{\mathsf{sel}}}(x_0, x_1, 1) \oplus F_{k_{\mathsf{sel}}}(x_0, x_2, 2)$.
  - Else, let $b_{i+1} = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
  - Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

Sample the CRS as the obfuscations of the above programs.

Recall that for all x_1, any proof (b_1, π_1) corresponding to $(x_0^*, x_1, 1)$ must satisfy that $b_1 = F_{k_{\mathsf{sel}}}(x_0^*, x_0^*, 0) \oplus F_{k_{\mathsf{sel}}}(x_0^*, x_1, 1)$. Therefore, by setting $b_2 = b_1 \oplus F_{k_{\mathsf{sel}}}(x_0^*, x_1, 1) \oplus F_{k_{\mathsf{sel}}}(x_0^*, x_2^*, 2)$ when $(x_0, x_2) = (x_0^*, x_2^*)$, the functionality has not changed. Therefore, we can invoke $i\mathcal{O}$ security.

H_4 In this hybrid, we sample some $y_1, y_2 \leftarrow \{0,1\}^{m_{\mathsf{PRG}}}$. We now sample the following new verifier algorithm:

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},y_1,y_2}(x_0, x_i, i \in [2], (b, \pi))$:

- If $i = 1$ and $x_0 = x_0^*$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
  - Else, accept if and only if $y_1 = G(\pi \oplus F_{\mathsf{sk}_b}(x_0, x_i, i))$.
- If $(x_0, x_i, i) = (x_0^*, x_2^*, 2)$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
  - Else, accept if and only if $y_2 = G(\pi \oplus F_{\mathsf{sk}_b}(x_0, x_i, i))$.
- Else,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

Sample the prover algorithm as in the previous hybrid. With probability at least $1 - 2 \cdot \frac{2^{n_\pi}}{2^{m_{\mathsf{PRG}}}}$, neither y_1 nor y_2 has a preimage under G. In this case, the verifier is functionally equivalent to that in the previous hybrid, and indistinguishability follows from $i\mathcal{O}$ security.

By the above indistinguishability, we therefore have that:

$$\Pr\left[\mathcal{V}(\mathsf{crs}, (x_0^*, x_2^*, 2), (b, \pi)) \;:\; \begin{matrix}\mathsf{crs} \leftarrow \mathsf{H}_4 \\ (b, \pi) \leftarrow \mathcal{A}(\mathsf{crs})\end{matrix}\right] \tag{7}$$

$$\geq p_{\mathsf{cheat}} - O(2^n) \cdot (2^{-\lambda_{\mathsf{iO}}^{\delta_{\mathsf{iO}}}} + 2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}) - 2 \cdot \frac{2^{n_\pi}}{2^{m_{\mathsf{PRG}}}}. \tag{8}$$

Now, we consider a sequence of hybrids where the experiment outputs not only the crs, but also the special bit corresponding to the instance $(x_0^*, x_2^*, 2)$.

H_5 In this hybrid, sample the CRS as in H_4, and let $\beta \leftarrow F_{k_{\mathsf{sel}}}(x_0^*, x_0^*, 0) \oplus F_{k_{\mathsf{sel}}}(x_0^*, x_2^*, 2)$. Output (β, crs).

H_6 In this hybrid, sample $z_2 \leftarrow \{0,1\}^{n_\pi}$, and let $y_2 = G(z_2)$. Sample β and CRS as in the previous hybrid using y_2. Output (β, crs). Indistinguishability follows from PRG security.

H_7 In this hybrid, we sample the verifier algorithm as follows:

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},y_1,z_2}(x_0, x_i, i \in [2], (b, \pi))$:

- If $i = 1$ and $x_0 = x_0^*$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
  - Else, accept if and only if $y_1 = G(\pi \oplus F_{\mathsf{sk}_b}(x_0, x_i, i))$.
- If $(x_0, x_i, i) = (x_0^*, x_2^*, 2)$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
  - Else, accept if and only if $\pi = z_2 \oplus F_{\mathsf{sk}_b}(x_0, x_i, i)$.
- Else,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

Indistinguishability follows from PRG injectivity and $i\mathcal{O}$ security.

H_8 In this hybrid, puncture $\mathsf{sk}_{1-\beta}$ on $(x_0^*, x_2^*, 2)$ to get $\mathsf{sk}_{1-\beta}^* := \mathsf{sk}_{1-\beta}\{(x_0^*, x_2^*, 2)\}$. For convenience, we also write $\mathsf{sk}_\beta^* := \mathsf{sk}_\beta$. Let $\gamma \leftarrow F_{\mathsf{sk}_{1-\beta}}(x_0^*, x_2^*, 2)$. Sample the verifier algorithm as follows:

$V_{\mathsf{sk}_0{}^*,\mathsf{sk}_1{}^*,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},y_1,z_2}(x_0, x_i, i \in [2], (b, \pi))$:

- If $i = 1$ and $x_0 = x_0^*$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b{}^*}(x_0, x_i, i)$.
  - Else, accept if and only if $y_1 = G(\pi \oplus F_{\mathsf{sk}_b{}^*}(x_0, x_i, i))$.
- If $(x_0, x_i, i) = (x_0^*, x_2^*, 2)$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b{}^*}(x_0, x_i, i)$.
  - Else, accept if and only if $z_2 = \pi \oplus \gamma$.
- Else,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b{}^*}(x_0, x_i, i)$, accept. Else, reject.

Sample the prover algorithm with $\mathsf{sk}_{1-\beta}^*$ in place of $\mathsf{sk}_{1-\beta}$. Sample the rest of the CRS and output (β, crs).

With high probability, we have that y_1 is not in the range of the PRG.

In this case, all accepting proofs for $(x_0^*, x_1, 1)$ must correspond to the bit $b_1 = F_{k_{\mathsf{sel}}}(x_0^*, x_0^*, 0) \oplus F_{k_{\mathsf{sel}}}(x_0^*, x_1, 1)$. Therefore, the prover algorithm only ever creates proofs for the statement $(x_0^*, x_2^*, 2)$ corresponding to the selector bit β, i.e., $F_{\mathsf{sk}_{1-\beta}}(x_0^*, x_2^*, 2)$ is never called by the prover algorithm. Hence, indistinguishability follows from $i\mathcal{O}$ security.

H_9 In this hybrid, we sample $\gamma \leftarrow \{0,1\}^{n_\pi}$. The rest of the hybrid is sampled as in the previous hybrid using γ. Indistinguishability follows from the puncturable PRF security.

H_{10} In this hybrid, we sample $\gamma \leftarrow \rho \oplus z_2$, where $\rho \leftarrow \{0,1\}^{n_\pi}$. The rest of the hybrid is sampled as in the previous hybrid using γ. This hybrid has an identical distribution to the previous.

H_{11} In this hybrid, we sample $\rho \leftarrow F_{\mathsf{sk}_{1-\beta}}(x_0^*, x_2^*, 2)$. Sample $\gamma \leftarrow \rho \oplus z_2$, and sample the rest of the experiment using γ just as was done in the previous hybrid. Indistinguishability follows from the puncturable PRF security.

H_{12} In this hybrid, unpuncture $\mathsf{sk}_{1-\beta}$, and write the verifier algorithm as follows.

> $V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},y_1,z_2}(x_0, x_i, i \in [2], (b, \pi))$:
>
> - If $i = 1$ and $x_0 = x_0^*$,
> - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
> - Else, accept if and only if $y_1 = G(\pi \oplus F_{\mathsf{sk}_b}(x_0, x_i, i))$.
> - Else if $(x_0, x_i, i) = (x_0^*, x_2^*, 2)$,
> - If $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept.
> - Else,
> - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

Sample the prover algorithm with the unpunctured key as well. Sample the rest of the CRS as in the previous hybrid. Indistinguishability follows from $i\mathcal{O}$ security.

H_{13} In this hybrid, we sample $z_1 \leftarrow \{0,1\}^{n_\pi}$ and set $y_1 \leftarrow G(z_1)$, and sample (β, crs) using y_1 as was done in the previous hybrid. Indistinguishability follows from PRG security.

H_{14} In this hybrid, we sample the verification algorithm instead as:

> $V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},z_1}(x_0, x_i, i \in [2], (b, \pi))$:
>
> - If $i = 1$ and $x_0 = x_0^*$,
> - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
> - Else, accept if and only if $\pi = z_1 \oplus F_{\mathsf{sk}_b}(x_0, x_i, i)$.
> - Else if $(x_0, x_i, i) = (x_0^*, x_2^*, 2)$,
> - If $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept.
> - Else,
> - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

Sample the prover algorithm as in the previous hybrid. Output (β, crs). Indistinguishability follows from the injectivity of the PRG and $i\mathcal{O}$ security.

H_{15} In this hybrid, we sample the verification algorithm instead as:

> $V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}}}(x_0, x_i, i \in [2], (b, \pi))$:
>
> - If $i = 1$ and $x_0 = x_0^*$,
> - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
> - Else, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.

- Else if $(x_0, x_i, i) = (x_0^*, x_2^*, 2)$,
  - If $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept.
- Else,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

Compute the prover algorithm and β just as in the previous hybrid, and output (β, crs).

Lemma 5.3. *For all polynomial-sized adversaries $\mathcal{B}$, we have that*

$$\left|\Pr\left[\begin{array}{c} b = 1 \\ \vdots \\ b \leftarrow \mathcal{B}(\beta, \mathsf{crs}) \end{array} \begin{array}{c} \\ : \\ \end{array} \begin{array}{c} (\beta, \mathsf{crs}) \leftarrow H_{15} \\ \\ \end{array}\right] - \Pr\left[\begin{array}{c} b = 1 \\ \vdots \\ b \leftarrow \mathcal{B}(\beta, \mathsf{crs}) \end{array} \begin{array}{c} \\ : \\ \end{array} \begin{array}{c} (\beta, \mathsf{crs}) \leftarrow H_{14} \\ \\ \end{array}\right]\right| = O(2^n) \cdot (2^{-\lambda_{i\mathcal{O}}^{\delta_{i\mathcal{O}}}} + 2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}).$$

We defer the proof to Section 5.2.

H_{16} In this hybrid, we sample the verification algorithm instead as:

$V_{\mathsf{sk}_0, \mathsf{sk}_1, \mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i \in [2], (b, \pi))$:

- If $i = 1$ and $x_0 = x_0^*$ or $(x_0, x_i, i) = (x_0^*, x_2^*, 2)$,
  - If $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept.
- Else,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

Compute the prover algorithm and β just as in the previous hybrid, and output (β, crs). Indistinguishability follows from $i\mathcal{O}$ security.

H_{17} In this hybrid, we sample the verification algorithm as in the previous hybrid, but we modify the prover algorithm as follows. We sample an additional PRF key k_{new}.

$P_{\mathsf{sk}_0, \mathsf{sk}_1, \mathsf{sk}_{\mathsf{sel}}, k_{\mathsf{new}}}(x_0, x_i, w_{i+1} \in \{0,1\}^m, i \in \{0,1\}, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i, w_{i+1})$.
- **Compute next proof:**
  - If $i = 0$ and $x_0 = x_0^*$, let $b_{i+1} = F_{k_{\mathsf{new}}}(x_0, x_{i+1}, 1)$.
  - Else if $i = 1$ and $(x_0, x_{i+1}) = (x_0^*, x_2^*)$, let $b_{i+1} = b_i \oplus F_{k_{\mathsf{new}}}(x_1, x_{i+1}, i+1)$.
  - Else, let $b_{i+1} = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
  - Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

We sample β as in the previous hybrid.

Lemma 5.4. *For all polynomial-sized adversaries $\mathcal{B}$, we have that*

$$\left|\Pr\left[\begin{array}{c} b=1 \\ \end{array} : \begin{array}{c}(\beta,\mathsf{crs}) \leftarrow H_{17} \\ b \leftarrow \mathcal{B}(\beta,\mathsf{crs})\end{array}\right] - \Pr\left[\begin{array}{c} b=1 \\ \end{array} : \begin{array}{c}(\beta,\mathsf{crs}) \leftarrow H_{16} \\ b \leftarrow \mathcal{B}(\beta,\mathsf{crs})\end{array}\right]\right| = O(2^n)\cdot(2^{-\lambda_{iO}^{\delta_{iO}}} + 2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}).$$

We defer the proof to Section 5.3.

H_{18} Finally, sample the CRS as in the previous hybrid, and sample $\beta \leftarrow \{0,1\}$ as a truly random value. Indistinguishability follows from PRF security since k_{sel} is no longer in the view of the adversary.

By definition of H_5, we have that

$$\Pr\left[\begin{array}{c}\mathcal{V}(\mathsf{crs},(x_0^*,x_2^*,2),(b,\pi)) \\ \wedge\, b \neq \beta\end{array} : \begin{array}{c}(\beta,\mathsf{crs}) \leftarrow \mathsf{H}_5 \\ (b,\pi) \leftarrow \mathcal{A}(\mathsf{crs})\end{array}\right] \leq \frac{2^{n_\pi}}{2^{m_{\mathsf{PRG}}}}. \tag{9}$$

By the indistinguishability of the above hybrids, we have that for $\rho \in \{0,1\}$,

$$\begin{aligned}&\left|\Pr\left[\begin{array}{c}\mathcal{V}(\mathsf{crs},(x_0^*,x_2^*,2),(b,\pi)) \\ \wedge\, b = \beta \oplus \rho\end{array} : \begin{array}{c}(\beta,\mathsf{crs}) \leftarrow \mathsf{H}_5 \\ (b,\pi) \leftarrow \mathcal{A}(\mathsf{crs})\end{array}\right]\right. \\ &\left. - \Pr\left[\begin{array}{c}\mathcal{V}(\mathsf{crs},(x_0^*,x_2^*,2),(b,\pi)) \\ \wedge\, b = \beta \oplus \rho\end{array} : \begin{array}{c}(\beta,\mathsf{crs}) \leftarrow \mathsf{H}_{18} \\ (b,\pi) \leftarrow \mathcal{A}(\mathsf{crs})\end{array}\right]\right| \\ &\leq O(2^n)\cdot(2^{-\lambda_{\mathsf{iO}}^{\delta_{\mathsf{iO}}}} + 2^{-\lambda_{\mathsf{PPRF}}^{\delta_{\mathsf{PPRF}}}}) + O\left(2^{-\lambda_{\mathsf{PRG}}^{\delta_{\mathsf{PRG}}}} + \frac{2^{n_\pi}}{2^{m_{\mathsf{PRG}}}}\right) \leq \mathsf{negl}(\lambda)\end{aligned} \tag{10}$$

In the above ρ indicates whether b "matches" β. Moreover, by the independence of β from the crs in H_{18}, we have that

$$\begin{aligned}&\Pr\left[\begin{array}{c}\mathcal{V}(\mathsf{crs},(x_0^*,x_2^*,2),(b,\pi)) \\ \wedge\, b = \beta\end{array} : \begin{array}{c}(\beta,\mathsf{crs}) \leftarrow \mathsf{H}_{18} \\ (b,\pi) \leftarrow \mathcal{A}(\mathsf{crs})\end{array}\right] \\ &= \Pr\left[\begin{array}{c}\mathcal{V}(\mathsf{crs},(x_0^*,x_2^*,2),(b,\pi)) \\ \wedge\, b \neq \beta\end{array} : \begin{array}{c}(\beta,\mathsf{crs}) \leftarrow \mathsf{H}_{18} \\ (b,\pi) \leftarrow \mathcal{A}(\mathsf{crs})\end{array}\right]\end{aligned} \tag{11}$$

By combining (9), (10) and (11), we have that

$$\Pr\left[\begin{array}{c}\mathcal{V}(\mathsf{crs},(x_0^*,x_2^*,2),(b,\pi)) \\ \wedge\, b = \beta\end{array} : \begin{array}{c}(\beta,\mathsf{crs}) \leftarrow \mathsf{H}_5 \\ (b,\pi) \leftarrow \mathcal{A}(\mathsf{crs})\end{array}\right] = \mathsf{negl}(\lambda).$$

Combining the above with (7), we have that $p_{\mathsf{cheat}} = \mathsf{negl}(\lambda)$, therefore completing the proof. □

5.1 Proof of Theorem 5.2

We define the set $S = (\{1\} \times \{0,1\}^n) \cup \{(2, x_2^*)\}$. We define the following hybrids for $(j,y) \in S$. For elements in S we say that $(j_1,y_1) < (j_2,y_2)$ if either:

- $j_1 = 1$ and $(j_2,y_2) = (2,x_2^*)$, or
- $j_1 = j_2 = 1$ and $y_1 < y_2$.

We say that (j_2,y_2) comes faster (j_1,y_1) if either:

- $j_1 = j_2 = 1$ and $y_2 = y_1 + 1$,
- or $(j_1, y_1) = (1, 1^n)$ and $(j_2, y_2) = (2, x_2^*)$.

$\mathsf{H}_{1,1}^{(j,y)}$ In this experiment, construct the verifier algorithm as follows:

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}}}(x_0, x_i, i \in [2], (b, \pi))$:

- If $x_0 = x_0^*$, $(i, x_i) \in S$ and $(i, x_i) < (j, y)$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept.
- Else,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

Let $\widetilde{V} \leftarrow i\mathcal{O}(1^{\lambda_{\mathsf{iO}}}, V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau})$.
Construct the following prover algorithm $P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau}$:

$P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}}}(x_0, x_i, w_{i+1} \in \{0,1\}^m, i \in \{0,1\}, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i, w_{i+1})$.
- **Compute next proof:**
  - If $x_0 = x_0^*$, $(i+1, x_{i+1}) \in S$ and $\boxed{(i+1, x_{i+1}) < (j, y)}$, let $b_{i+1} = F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$.
  - Else, let $b_{i+1} = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
  - Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

Sample the CRS as the obfuscation of the above programs.

$\mathsf{H}_{1,2}^{(j,y)}$ In this hybrid, puncture both of $\mathsf{sk}_{\mathsf{sel}}$ and k_{sel} on (x_0^*, y, j) to obtain the keys $\mathsf{sk}_{\mathsf{sel}}{}^*$ and $k_{\mathsf{sel}}{}^*$ respectively. Let $\gamma_1 \leftarrow F_{\mathsf{sk}_{\mathsf{sel}}}(x_0^*, y, j)$ and $\gamma_2 \leftarrow F_{k_{\mathsf{sel}}}(x_0^*, y, j)$. construct the verifier algorithm as follows:

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}}{}^*,k_{\mathsf{sel}}{}^*}(x_0, x_i, i \in [2], (b, \pi))$:

- If $x_0 = x_0^*$ and $(i, x_i) \in S$ and $(i, x_i) < (j, y)$,
  - If $b = F_{k_{\mathsf{sel}}{}^*}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}{}^*}(x_0, x_0, 0)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.
- Else if $(x_0, x_i, i) = (x_0^*, y, j)$,
  - If $b = \gamma_1$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.
- Else,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}{}^*}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

Construct the following prover algorithm.

$P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}}{}^*,k_{\mathsf{sel}}{}^*}(x_0, x_i, w_{i+1} \in \{0,1\}^m, i \in \{0,1\}, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i, w_{i+1})$.
- **Compute next proof:**
  - If $x_0 = x_0^*$, $(i+1, x_{i+1}) \in S$ and $(i+1, x_{i+1}) < (j, y)$, let $b_{i+1} = F_{k_{\mathsf{sel}}{}^*}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}{}^*}(x_0, x_0, 0)$.
  - Else if $(x_0, x_{i+1}, i+1) = (x_0^*, y, j)$, let $b_{i+1} = \gamma_1$.
  - Else, let $b_{i+1} = F_{\mathsf{sk}_{\mathsf{sel}}{}^*}(x_0, x_{i+1}, i+1)$.
  - Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

Sample the CRS as the obfuscation of the above programs. Indistinguishability follows from $i\mathcal{O}$ security and puncturable PRF correctness.

$\mathsf{H}_{1,3}{}^{(j,y)}$ In this hybrid, sample both γ_1 and γ_2 as uniformly random strings, and sample the CRS using γ_1 as in the previous hybrid. Indistinguishability follows from puncturable PRF security.

$\mathsf{H}_{1,4}{}^{(j,y)}$ In this hybrid, sample $\gamma_1 = \gamma_2 \oplus F_{k_{\mathsf{sel}}}(x_0^*, x_0^*, 0)$. Sample the CRS using γ_1 as in the previous hybrid. The distribution in this hybrid is identical to the distribution in the previous hybrid.

$\mathsf{H}_{1,5}{}^{(j,y)}$ In this hybrid, sample $\gamma_2 \leftarrow F_{k_{\mathsf{sel}}}(x_0^*, y, j)$. Sample the CRS using γ_1 as in the previous hybrid. Indistinguishability follows from puncturable PRF security.

$\mathsf{H}_{1,6}{}^{(j,y)}$ In this hybrid, we unpuncture the keys $\mathsf{sk}_{\mathsf{sel}}$ and k_{sel}, and sample the verifier and prover algorithm as follows.

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}}}(x_0, x_i, i \in [2], (b, \pi))$:

- If $x_0 = x_0^*$ and $(i, x_i) \in S$ and $(i, x_i) < (j, y)$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.
- Else if $(x_0, x_i, i) = (x_0^*, y, j)$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, y, j) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.
- Else,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

Let $\widetilde{V} \leftarrow i\mathcal{O}(1^{\lambda_{\mathsf{iO}}}, V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau})$.

Construct the following prover algorithm $P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},\tau}$:

$P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}}}(x_0, x_i, w_{i+1} \in \{0,1\}^m, i \in \{0,1\}, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i, w_{i+1})$.
- **Compute next proof:**
  - If $x_0 = x_0^*$, $(i+1, x_{i+1}) \in S$ and $(i+1, x_{i+1}) < (j, y)$, let $b_{i+1} = F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$.
  - Else if $(x_0, x_{i+1}, i+1) = (x_0^*, y, j)$, let $b_{i+1} = F_{k_{\mathsf{sel}}}(x_0, y, j) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$.
  - Else, let $b_{i+1} = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
  - Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

Indistinguishability follows from $i\mathcal{O}$ security.

The programs in H_1 and $\mathsf{H}_{1,1}{}^{(1,0^n)}$ are functionally equivalent. Let $(j', y') \in S$ be the element immediately after (j, y) in S. Then, the programs in $\mathsf{H}_{1,6}{}^{(j,y)}$ and $\mathsf{H}_{1,1}{}^{(j',y')}$ are also functionally equivalent. Finally, the programs in $\mathsf{H}_{1,6}{}^{(2,x_2^*)}$ are functionally equivalent to those in H_2. These transitions therefore follow from $i\mathcal{O}$ security. By $O(2^n)$ hybrids invoking $i\mathcal{O}$ and puncturable PRF security, we prove our claim.

5.2 Proof of Theorem 5.3

We define the following hybrids for $y \in \{0,1\}^n$.

$\mathsf{H}_{14,1}{}^{(y)}$ In this hybrid, we sample the verification algorithm instead as:

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},z_1}(x_0, x_i, i \in [2], (b, \pi))$:

- If $i = 1$ and $x_0 = x_0^*$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
  - Else,
    - if $x_1 < y$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
    - else, accept if and only if $\pi = z_1 \oplus F_{\mathsf{sk}_b}(x_0, x_i, i)$.
- Else if $(x_0, x_i, i) = (x_0^*, x_2^*, 2)$,
  - If $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept.
- Else,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

Sample the rest of the CRS and β as in H_{14}. Output (β, crs).

$\mathsf{H}_{14,2}{}^{(y)}$ Let $b^* = F_{k_{\mathsf{sel}}}(x_0^*, x_0^*, 0) \oplus F_{k_{\mathsf{sel}}}(x_0^*, y, 1)$. Without loss of generality, suppose that $b^* = 0$ (a symmetric argument applies if $b^* = 1$). Puncture sk_1 on $(x_0^*, y, 1)$ to get key ${\mathsf{sk}_1}^*$, and let $\gamma \leftarrow F_{\mathsf{sk}_1}(x_0^*, y, 1)$.

For convenience, write ${\mathsf{sk}_0}^* := \mathsf{sk}_0$. We construct the verifier algorithm using γ as follows:

$V_{\mathsf{sk}_0{}^*,\mathsf{sk}_1{}^*,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},z_1}(x_0, x_i, i \in [2], (b, \pi))$:

- If $i = 1$ and $x_0 = x_0^*$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b{}^*}(x_0, x_i, i)$.
  - Else,
    - if $x_1 < y$, accept if and only if $\pi = F_{\mathsf{sk}_b{}^*}(x_0, x_i, i)$.
    - else if $x_1 = y$, accept if and only if $\pi = z_1 \oplus \gamma$.
    - else, accept if and only if $\pi = z_1 \oplus F_{\mathsf{sk}_b{}^*}(x_0, x_i, i)$.
- Else if $(x_0, x_i, i) = (x_0^*, x_2^*, 2)$,
  - If $\pi = F_{\mathsf{sk}_b{}^*}(x_0, x_i, i)$, accept.
- Else,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b{}^*}(x_0, x_i, i)$, accept. Else, reject.

Sample the prover algorithm with $\mathsf{sk}_1{}^*$ in place of sk_1. Note that by construction, $F_{\mathsf{sk}_1}(x_0^*, y, 1)$ is never called. Sample the rest of the CRS and β as in the previous hybrid. Indistinguishability follows from $i\mathcal{O}$ security and punctured PRF correctness.

$\mathsf{H}_{14,3}{}^{(y)}$ Sample γ as a random string $\rho \in \{0,1\}^{n_\pi}$. Sample the rest of the CRS and β as in the previous hybrid using γ. Output (β, crs). Indistinguishability follows from puncturable PRF security.

$\mathsf{H}_{14,4}{}^{(y)}$ Sample $\gamma \leftarrow \rho \oplus z_1$, where $\rho \leftarrow \{0,1\}^{n_\pi}$. Sample the rest of the CRS and β as in the previous hybrid using γ. Output (β, crs). The distribution of this hybrid is identical to the previous.

$\mathsf{H}_{14,5}{}^{(y)}$ Sample $\rho \leftarrow F_{\mathsf{sk}_1}(x_0^*, y, 1)$, and $\gamma \leftarrow \rho \oplus z_1$. Sample the rest of the hybrid using γ as was done in the previous hybrid. Output (β, crs). Indistinguishability follows punctured PRF security.

$\mathsf{H}_{14,6}{}^{(y)}$ In this hybrid, unpuncture sk_1, and sample the verifier program as follows:

$V_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},z_1}(x_0, x_i, i \in [2], (b, \pi))$:

- If $i = 1$ and $x_0 = x_0^*$,
  - If $b = F_{k_{\mathsf{sel}}}(x_0, x_i, i) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
  - Else,
    - if $x_1 \leq y$, accept if and only if $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$.
    - else, accept if and only if $\pi = z_1 \oplus F_{\mathsf{sk}_b}(x_0, x_i, i)$.
- Else if $(x_0, x_i, i) = (x_0^*, x_2^*, 2)$,
  - If $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept.
- Else,
  - If $b = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_i, i)$ and $\pi = F_{\mathsf{sk}_b}(x_0, x_i, i)$, accept. Else, reject.

Sample the prover algorithm with the unpunctured key as well. Sample β as in the previous hybrid, and output (β, crs). Indistinguishability follows from $i\mathcal{O}$ security.

Clearly, we have that $\mathsf{H}_{14} \approx_c \mathsf{H}_{14,1}{}^{(0^n)}$. Moreover, for $y < 1^n$, $\mathsf{H}_{14,6}{}^{(y)} \approx_c \mathsf{H}_{14,1}{}^{(y+1)}$. Finally, at $y = 1^n$, the last hybrid is computationally indistinguishable from H_{15}. These transitions follow from functional equivalence and $i\mathcal{O}$ security. Since we invoke $i\mathcal{O}$ and puncturable PRF security $O(2^n)$ times, we derive our theorem.

5.3 Proof of Theorem 5.4

We define the following hybrids for $y \in \{0,1\}^n$.

$\mathsf{H}_{16,1}{}^{(y)}$ We sample the prover algorithm as follows:

$P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}},k_{\mathsf{new}}}(x_0, x_i, w_{i+1} \in \{0,1\}^m, i \in \{0,1\}, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$ and $b_i = 0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i, w_{i+1})$.
- **Compute next proof:**
  - If $i = 0$ and $x_0 = x_0^*$:
    - if $x_1 < y$, let $b_{i+1} = F_{k_{\mathsf{new}}}(x_0, x_{i+1}, i+1)$.
    - else, let $b_{i+1} = b_i \oplus F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}}(x_0, x_0, 0)$.
  - Else if $i = 1$ and $(x_0, x_{i+1}) = (x_0^*, x_2^*)$:
    - if $x_1 < y$, let $b_{i+1} = b_i \oplus F_{k_{\mathsf{new}}}(x_1, x_2, 2)$.
    - else, let $b_{i+1} = b_i \oplus F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}}(x_0, x_1, 1)$.
  - Else, let $b_{i+1} = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
  - Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.
- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

The verifier algorithm and β are as defined in H_{16}. Output (β, crs).

$\mathsf{H}_{16,2}{}^{(y)}$ Let $\gamma \leftarrow F_{k_{\mathsf{sel}}}(x_0^*, y, 1)$. Let $\gamma_1 \leftarrow F_{k_{\mathsf{new}}}(x_0^*, y, 1)$ and $\gamma_2 \leftarrow F_{k_{\mathsf{new}}}(y, x_2^*, 2)$.

- Puncture k_{sel} on $(x_0^*, y, 1)$ to get $k_{\mathsf{sel}}{}^*$.
- Puncture k_{new} on $(x_0^*, y, 1)$ and $(y, x_2^*, 2)$ to get $k_{\mathsf{new}}{}^*$. Note that both of these points are not used in the prover algorithm in the previous hybrid.

$P_{\mathsf{sk}_0,\mathsf{sk}_1,\mathsf{sk}_{\mathsf{sel}},k_{\mathsf{sel}}{}^*,k_{\mathsf{new}}{}^*}(x_0, x_i, w_{i+1} \in \{0,1\}^m, i \in \{0,1\}, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$ and $b_i = 0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i, w_{i+1})$.
- **Compute next proof:**
  - If $i = 0$ and $x_0 = x_0^*$:

* if $x_1 < y$, let $b_{i+1} = F_{k_{\mathsf{new}}^*}(x_0, x_{i+1}, i+1)$.
* else if $x_1 = y$, let $b_{i+1} = b_i \oplus \gamma \oplus F_{k_{\mathsf{sel}}^*}(x_0, x_0, 0)$.
* else, let $b_{i+1} = b_i \oplus F_{k_{\mathsf{sel}}^*}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}^*}(x_0, x_i, i)$.

– Else if $i = 1$ and $(x_0, x_{i+1}) = (x_0^*, x_2^*)$:
* if $x_1 < y$, let $b_{i+1} = b_i \oplus F_{k_{\mathsf{new}}^*}(x_1, x_2, 2)$.
* else if $x_1 = y$, let $b_{i+1} = b_i \oplus F_{k_{\mathsf{sel}}^*}(x_0, x_{i+1}, i+1) \oplus \gamma$.
* else, let $b_{i+1} = b_i \oplus F_{k_{\mathsf{sel}}^*}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}^*}(x_0, x_i, i)$.

– Else, let $b_{i+1} = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
– Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.

- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

Sample the rest of β and the CRS. Output (β, crs). Indistinguishability follows from $i\mathcal{O}$ security and puncturable PRF correctness.

$\mathsf{H}_{16,3}^{(y)}$ Sample $\gamma, \gamma_1, \gamma_2 \leftarrow \{0,1\}$, and sample the experiment using γ just as in the previous hybrid. Output (β, crs).

Indistinguishability follows from puncturable PRF security.

$\mathsf{H}_{16,4}^{(y)}$ By assumption that $(x_0^*, x_2^*, 2) \notin \mathcal{L}$, we have that either x_0^* does not reach y, and/or y does not reach x_2^*.

- If the former holds, sample $\gamma \leftarrow \gamma_2 \oplus F_{k_{\mathsf{sel}}^*}(x_0^*, x_2^*, 2)$.
- Otherwise, sample $\gamma \leftarrow \gamma_1 \oplus F_{k_{\mathsf{sel}}^*}(x_0^*, x_0^*, 0)$.

Sample the experiment using γ just as was done in the previous experiment. The distribution in this hybrid is identical to the previous.

$\mathsf{H}_{16,5}^{(y)}$ Let $\gamma_1 \leftarrow F_{k_{\mathsf{new}}}(x_0^*, y, 1)$ and $\gamma_2 \leftarrow F_{k_{\mathsf{new}}}(y, x_2^*, 2)$. Sample γ and the rest of the CRS and β using γ_1 and γ_2 using the previous hybrid. Indistinguishability follows from puncturable PRF security.

$\mathsf{H}_{16,6}^{(y)}$ Unpuncture the keys $k_{\mathsf{sel}}, k_{\mathsf{new}}$. Sample the prover algorithm as follows:

$P_{\mathsf{sk}_0, \mathsf{sk}_1, \mathsf{sk}_{\mathsf{sel}}, k_{\mathsf{sel}}, k_{\mathsf{new}}}(x_0, x_i, w_{i+1} \in \{0,1\}^m, i \in \{0,1\}, (b_i, \pi_i))$:

- If $i = 0$, set $x_i = x_0$ and $b_i = 0$.
- **Verify previous proof:** If $i \geq 1$, if $\widetilde{V}(x_0, x_i, i, (b_i, \pi_i)) = 0$, reject.
- **Compute next configuration:** Let $x_{i+1} = \mathsf{next\text{-}config}(\mathcal{M}, x_i, w_{i+1})$.
- **Compute next proof:**
  – If $i = 0$ and $x_0 = x_0^*$:
    * if $x_1 \boxed{\leq} y$, let $b_{i+1} = F_{k_{\mathsf{new}}}(x_0, x_{i+1}, i+1)$.
    * else, let $b_{i+1} = b_i \oplus F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}}(x_0, x_i, i)$.
  – Else if $i = 1$ and $(x_0, x_{i+1}) = (x_0^*, x_2^*)$:
    * if $x_1 \boxed{\leq} y$, let $b_{i+1} = b_i \oplus F_{k_{\mathsf{new}}}(x_1, x_2, 2)$.
    * else, let $b_{i+1} = b_i \oplus F_{k_{\mathsf{sel}}}(x_0, x_{i+1}, i+1) \oplus F_{k_{\mathsf{sel}}}(x_0, x_i, i)$.

- Else, let $b_{i+1} = F_{\mathsf{sk}_{\mathsf{sel}}}(x_0, x_{i+1}, i+1)$.
- Let $\pi_{i+1} = F_{\mathsf{sk}_{b_{i+1}}}(x_0, x_{i+1}, i+1)$.

- Output $x_{i+1}, (b_{i+1}, \pi_{i+1})$.

The rest of the crs and β are sampled as in the previous hybrid.

It suffices to argue that the prover algorithm is functionally equivalent to that in the previous hybrid. Then, we can invoke $i\mathcal{O}$ security.

If x_0^* does not reach y and y does not reach x_2^*, functionality equivalence follows trivially.

Suppose without loss of generality that x_0^* reaches y (a symmetric argument applies when y reaches x_2^*). Then, the prover algorithm will never be in the case where $(x_0, x_i, i) = (x_0^*, y, 1)$, and $x_{i+1} = x_2^*$. On the other hand, $\gamma = F_{k_{\mathsf{new}}}(x_0^*, y, 1) \oplus F_{k_{\mathsf{sel}}}(x_0^*, x_0^*, 0)$. When $(x_0, x_i, i) = (x_0^*, x_0^*, 0)$ and $x_{i+1} = y$, we have functionality equivalence.

Clearly, we have that $\mathsf{H}_{16} \approx_c {\mathsf{H}_{16,1}}^{(0^n)}$. Moreover, for $y < 1^n$, ${\mathsf{H}_{16,6}}^{(y)} \approx_c {\mathsf{H}_{16,1}}^{(y+1)}$. Finally, at $y = 1^n$, the last hybrid is computationally indistinguishable from H_{17}. These transitions follow from functional equivalence and $i\mathcal{O}$ security. Since we invoke $i\mathcal{O}$ and puncturable PRF security $O(2^n)$ times, we derive our claim.

6 References

[AS15] Gilad Asharov and Gil Segev. Limits on the power of indistinguishability obfuscation and functional encryption. In Venkatesan Guruswami, editor, *56th FOCS*, pages 191–209. IEEE Computer Society Press, October 2015.

[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.

[BCG25] Annalisa Barbara, Alessandro Chiesa, and Ziyi Guan. Relativized succinct arguments in the ROM do not exist. In *TCC 2025, Part I*, LNCS, pages 400–416. Springer, Cham, November 2025.

[BCL+21] Benedikt Bünz, Alessandro Chiesa, William Lin, Pratyush Mishra, and Nicholas Spooner. Proof-carrying data without succinct arguments. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 681–710, Virtual Event, August 2021. Springer, Cham.

[BCMS20] Benedikt Bünz, Alessandro Chiesa, Pratyush Mishra, and Nicholas Spooner. Recursive proof composition from accumulation schemes. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part II*, volume 12551 of *LNCS*, pages 1–18. Springer, Cham, November 2020.

[BCTV14] Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. Scalable zero knowledge via cycles of elliptic curves. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 276–294. Springer, Berlin, Heidelberg, August 2014.

[BDFG21] Dan Boneh, Justin Drake, Ben Fisch, and Ariel Gabizon. Halo infinite: Proof-carrying data from additive polynomial commitments. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 649–680, Virtual Event, August 2021. Springer, Cham.

[BDV17] Nir Bitansky, Akshay Degwekar, and Vinod Vaikuntanathan. Structure vs. hardness through the obfuscation lens. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 696–723. Springer, Cham, August 2017.

[Bei25] Amos Beimel. Secret-sharing schemes for general access structures: An introduction. Cryptology ePrint Archive, Paper 2025/518, 2025.

[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.

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In Hugo Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 501–519. Springer, Berlin, Heidelberg, March 2014.

[BMRS20] Joseph Bonneau, Izaak Meckler, Vanishree Rao, and Evan Shapiro. Coda: Decentralized cryptocurrency at scale. Cryptology ePrint Archive, Report 2020/352, 2020.

[BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part II*, volume 8270 of *LNCS*, pages 280–300. Springer, Berlin, Heidelberg, December 2013.

[CCDW20] Weikeng Chen, Alessandro Chiesa, Emma Dauterman, and Nicholas P. Ward. Reducing participation costs via incremental verification for ledger systems. Cryptology ePrint Archive, Report 2020/1522, 2020.

[CCG+23] Megan Chen, Alessandro Chiesa, Tom Gur, Jack O'Connor, and Nicholas Spooner. Proof-carrying data from arithmetized random oracles. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part II*, volume 14005 of *LNCS*, pages 379–404. Springer, Cham, April 2023.

[CGKS23] Matteo Campanelli, Chaya Ganesh, Hamidreza Khoshakhlagh, and Janno Siim. Impossibilities in succinct arguments: Black-box extraction and more. In Nadia El Mrabet, Luca De Feo, and Sylvain Duquesne, editors, *AFRICACRYPT 23*, volume 14064 of *LNCS*, pages 465–489. Springer, Cham, July 2023.

[COS20] Alessandro Chiesa, Dev Ojha, and Nicholas Spooner. Fractal: Post-quantum and transparent recursive proofs from holography. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 769–793. Springer, Cham, May 2020.

[CTV13] Stephen Chong, Eran Tromer, and Jeffrey A. Vaughan. Enforcing language semantics using proof-carrying data. Cryptology ePrint Archive, Report 2013/513, 2013.

[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.

[DJJ+25] Pratish Datta, Abhishek Jain, Zhengzhong Jin, Alexis Korb, Surya Mathialagan, and Amit Sahai. Incrementally verifiable computation for NP from standard assumptions. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part VII*, volume 16006 of *LNCS*, pages 611–642. Springer, Cham, August 2025.

[DWW24] Lalita Devadas, Brent Waters, and David J. Wu. Batching adaptively-sound SNARGs for NP. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024, Part II*, volume 15365 of *LNCS*, pages 339–370. Springer, Cham, December 2024.

[GGM84] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. On the cryptographic applications of random functions. In G. R. Blakley and David Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 276–288. Springer, Berlin, Heidelberg, August 1984.

[GL89] Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *21st ACM STOC*, pages 25–32. ACM Press, May 1989.

[GSWW22] Rachit Garg, Kristin Sheridan, Brent Waters, and David J. Wu. Fully succinct batch arguments for np from indistinguishability obfuscation. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 526–555. Springer, Cham, November 2022.

[HAN23] Mathias Hall-Andersen and Jesper Buus Nielsen. On valiant's conjecture: Impossibility of incrementally verifiable computation from random oracles. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part II*, volume 14005 of *LNCS*, pages 438–469. Springer, Cham, April 2023.

[KPTZ13] Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 2013*, pages 669–684. ACM Press, November 2013.

[KS22] Abhiram Kothapalli and Srinath Setty. SuperNova: Proving universal machine executions without universal circuits. Cryptology ePrint Archive, Report 2022/1758, 2022.

[KST22] Abhiram Kothapalli, Srinath Setty, and Ioanna Tzialla. Nova: Recursive zero-knowledge arguments from folding schemes. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part IV*, volume 13510 of *LNCS*, pages 359–388. Springer, Cham, August 2022.

[Mic94] Silvio Micali. CS proofs (extended abstracts). In *35th FOCS*, pages 436–453. IEEE Computer Society Press, November 1994.

[NT16] Assa Naveh and Eran Tromer. PhotoProof: Cryptographic image authentication for any set of permissible transformations. In *2016 IEEE Symposium on Security and Privacy*, pages 255–271. IEEE Computer Society Press, May 2016.

[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.

[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.

[SZ25] Omri Shmueli and Mark Zhandry. On one-shot signatures, quantum vs. classical binding, and obfuscating permutations. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part II*, volume 16001 of *LNCS*, pages 350–383. Springer, Cham, August 2025.

[Val08] Paul Valiant. Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In Ran Canetti, editor, *TCC 2008*, volume 4948 of *LNCS*, pages 1–18. Springer, Berlin, Heidelberg, March 2008.

[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.

[WW25] Brent Waters and David J. Wu. A pure indistinguishability obfuscation approach to adaptively-sound SNARGs for NP. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part VII*, volume 16006 of *LNCS*, pages 292–326. Springer, Cham, August 2025.

[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.