Round-optimal Black-box Commit-and-prove with Succinct Communication

Susumu Kiyoshima

NTT Research
susumu.kiyoshima@ntt-research.com

Abstract. We give a four-round black-box construction of a commit-and-prove protocol with succinct communication. Our construction is WI and has constant soundness error, and it can be upgraded into a one that is ZK and has negligible soundness error by relying on a round-preserving transformation of Khurana et al. (TCC 2018). Our construction is obtained by combining the MPC-in-the-head technique of Ishai et al. (SICOMP 2009) with the two-round succinct argument of Kalai et al. (STOC 2014), and the main technical novelty lies in the analysis of the soundness—we show that, although the succinct argument of Kalai et al. does not necessarily provide soundness for $\mathcal{NP}$ statements, it can be used in the MPC-in-the-head technique for proving the consistency of committed MPC views. Our construction is based on sub-exponentially hard collision-resistant hash functions, two-round PIRs, and two-round OTs.

1 Introduction

In this paper, we obtain a new *commit-and-prove protocol* by relying on techniques in the area of *succinct arguments*. We start by giving some backgrounds.

Succinct arguments. Informally speaking, a *succinct argument* is an argument system with small communication complexity and fast verification time—typically, when a statement about T-time deterministic or non-deterministic computation is proven, the communication complexity and the verification time are required to be polylogarithmic in T. (The security requirements are, as usual, completeness and computational soundness.) Succinct arguments are useful when resources for communication and verification are limited; for example, a direct application of succinct arguments is *delegating computation* [GKR15] (or *verifiable computation* [GGP10]), where a computationally weak client delegates heavy computations to a powerful server and the client uses succinct arguments to verify the correctness of the server's computation efficiently. It was shown that a four-round succinct argument for all statements in $\mathcal{NP}$ can be obtained from collision-resistance hash functions [Kil92]. Since then, succinct arguments have been actively studied, and protocols with various properties have been proposed.

Among existing succinct arguments, the most relevant to this work is the one by Kalai et al. [KRR14] (KRR succinct argument in short), which has several

desirable properties such as (1) being *doubly efficient* [GKR15] (i.e., not only the verifier but also the prover is efficient), (2) being a two-round protocol (i.e., the scheme consists of a single query message from the verifier and a single answer message from the prover), and (3) being proven secure under standard assumptions, especially without relying on unfalsifiable assumptions and random oracles. More concretely, when the statement is about the correctness of a T-time computation, the communication complexity and the verifier running time is polylogarithmic in T while the prover running time is polynomial in T, and the security is proven assuming the existence of private information retrieval (PIR) or fully homomorphic encryption (FHE).

Given the powerful properties of KRR succinct argument, it is natural to expect that it has many cryptographic applications. For example, since argument systems have been extensively used in the design of cryptographic protocols, one might expect that the efficiency of such cryptographic protocols can be improved by simply plugging in KRR succinct argument.

However, using KRR succinct argument in cryptographic applications is actually non-trivial. One difficulty is that the soundness of KRR succinct argument is currently proven only for some specific types of $\mathcal{NP}$ statements [KP16, BHK17, BKK+18] (originally, its soundness was proven for statements in $\mathcal{P}$ [KRR14]). Another difficulty is that it does not provide any privacy on witnesses when it is used for $\mathcal{NP}$ statements.

Nonetheless, recent works showed that KRR succinct argument can be used in some cryptographic applications. For example, by cleverly combining KRR succinct argument with other cryptographic primitives, Bitansky et al. [BBK+16] obtained a three-round zero-knowledge argument against uniform cheating provers, Brakerski and Kalai [BK20] obtained a succinct private access control protocol for the access structures that can be expressed by monotone formulas, and Morgan et al. [MPP20] obtained a succinct non-interactive secure two-party computation protocol.

The number of applications is, however, still limited. A potential reason for this limitation is that the current techniques inherently use cryptographic primitives in non-black-box ways. Concretely, to hide the prover's witness, the current techniques use KRR succinct argument under other cryptographic protocols (such as garbling schemes) and thus require non-black-box accesses to the codes of the cryptographic primitives that underlies KRR succinct argument. Consequently, the current techniques cannot be used for applications where black-box uses of cryptographic protocols are desirable, such as the application to commit-and-prove protocols, which we discuss next.

Commit-and-prove protocols. Informally speaking, a commit-and-prove protocol is a commitment scheme in which the committer can prove a statement about the committed value without opening the commitment. Proofs by the committer are required to be *zero-knowledge* (ZK) or *witness-indistinguishable* (WI), where the former requires that the views of the receiver in the commit and prove phases can be simulated in polynomial time without knowing the committed value, and the latter requires that for any two messages and any statement such

that both of the messages satisfy the statement, the receiver cannot tell which of the messages is committed even after receiving a proof on the statement. Commit-and-prove protocols were implicitly used by Goldreich et al. [GMW87] for obtaining a secure multi-party computation protocol with malicious security, and later formalized by Canetti et al. [CLOS02].

A desirable property of commit-and-prove protocols is that they are constructed in a black-box way, i.e., in a way that uses the underlying cryptographic primitives as black-box by accessing them only through their input/output interfaces. Indeed, this black-box construction property is essential when commit-and-prove protocols are used as a tool for enforcing honest behaviors on malicious parties without relying on non-black-box uses of the underlying cryptographic primitives (see, e.g., [GLOV12, LP12, GOSV14]).

Very recently, Hazay and Venkitasubramaniam [HV18] and Khurana et al. [KOS18] gave four-round black-box constructions of ZK commit-and-prove protocols, where the round complexity of a commit-and-prove protocol is defined as the sum of that of the commit phase and that of the prove phase. Their protocols are *round optimal* since the commit and prove phases of their commit-and-prove protocols can be thought of as black-box ZK arguments (where the prover first commits to a witness and then proves the validity of the committed witness) and black-box ZK arguments are known to require at least four rounds [GK96]. Their protocols also have the *delayed-input property*, i.e., the property that statements to be proven on committed values can be chosen adaptively in the last round of the prove phase.

The commit-and-prove protocols by Hazay and Venkitasubramaniam [HV18] and Khurana et al. [KOS18] are not succinct in the sense that when the statement is expressed as a T-time predicate on the committed value, the communication complexity depends at least linearly on T. This is because both of their protocols were obtained via transformations from the three-round constant-sound commit-and-prove protocol of Hazay and Venkitasubramaniam [HV16], which is not succinct in the above sense.

1.1 Our Result

Our main result is a four-round black-box construction of a constant-sound WI commit-and-prove protocol with succinct communication complexity.

Theorem 1. *Assume the existence of sub-exponentially hard versions of the following cryptographic primitives: a collision-resistant hash function family, a two-round oblivious transfer protocol, and a two-round private information retrieval protocol. Then, there exists a constant-sound WI commit-and-prove protocol with the following properties.*

1. *The round complexity is* 4*, and the protocol satisfies the delayed-input property and uses the above cryptographic primitives in a black-box way.*
2. *When the length of the committed value is n and the statement to be proven on the committed value is a T-time predicate, the communication complexity depends polynomially on $\log n$, $\log T$, and the security parameter.*

Our commit-and-prove protocol uses a variant of KRR succinct argument (which is obtained from the private information retrieval protocol), and succinctness of our commit-and-prove protocol is inherited from that of KRR succinct argument. We assume sub-exponential hardness on the cryptographic primitives since we use complexity leveraging.

ZK and negligible soundness error. Given our constant-sound WI commit-and-prove protocol, we can use (a minor variant of) a transformation of Khurana et al. [KOS18] to transform it into a 4-round ZK commit-and-prove protocol with negligible soundness error. The resultant commit-and-prove protocol still satisfies the delayed-input property, the black-box uses of the underlying primitives, and the succinct communication complexity. (See the full version of this paper for details.)

Verification time. The verification of our commit-and-prove protocol is not succinct, i.e., the verifier running time depends polynomially on T. Although we might be able to make it succinct by appropriately modifying our protocol (see Appendix A for details), we do not explore this possibility in this work so that we can focus on our main purpose, i.e., on showing how to use KRR succinct argument in black-box constructions of commit-and-prove protocols.

Complexity leveraging. As mentioned above, we use complexity leveraging in the proof of Theorem 1. Although we might be able to avoid the use of complexity leveraging by using known techniques (e.g., by relying on extractable commitments [PW09]), we do not explore this possibility in this work for the same reason as above.

Comparison with existing schemes. As explained above, Hazay and Venkitasubramaniam [HV18] and Khurana et al. [KOS18] gave four-round black-box ZK commit-and-prove protocols with the delayed-input property. Their schemes rely on a weak primitive (injective one-way functions) but do not have succinct communication.

Goyal et al. [GOSV14] and Ishai and Weiss [IW14] studied black-box commit-and-prove protocols with succinct communication under slightly different definitions than ours.[1] If their techniques are used to obtain schemes under our definitions, the resultant schemes will rely on a weak primitive (collision-resistant hash functions) but have round complexity larger than 4.[2]

[1] For example, the definition in [IW14] considers non-deterministic statements on committed values but the statements are assumed to be fixed in the commit phase, whereas our definition considers deterministic statements but the statements are allowed to be chosen after the commit phase is completed.

[2] Roughly speaking, this is because in a setting where statements to be proven are chosen after the commit phase (e.g., the delayed-input setting), techniques in [GOSV14, IW14] require that (1) the commit phase has 2 rounds as it needs to be succinct and (2) the prove phase has 3 rounds as it has a commit-challenge-response structure.

Kalai and Paneth [KP16] observed that when messages are committed by using Merkle tree-hash, KRR succinct argument can be used for proving statements on the committed messages. The resultant scheme is succinct in terms of both communication complexity and verification time, but uses the underlying hash function in a non-black-box way and does not have privacy properties (which are not needed for the purpose of [KP16]).

1.2 Overview of Our Commit-and-prove Protocol

The overall approach is to combine KRR succinct argument with the *MPC-in-the-head technique* [IKOS09].

Let us first recall how we can obtain a non-succinct WI commit-and-prove protocol by using the MPC-in-the-head technique. Let $M \in \mathbb{N}$ be an arbitrary constant, Π be any 2-private semi-honest secure M-party computation protocol with perfect completeness,[3] OT be any two-round 1-out-of-M^2 oblivious transfer (OT) protocol, SBCom be any statistically binding commitment scheme, and SHCom be any statistically hiding commitment scheme. We assume that the hiding property of SBCom can be broken in a quasi-polynomial time T_{SB}, and the security of the other primitives holds against $\mathsf{poly}(T_{\mathrm{SB}})$-time adversaries.

Commit phase. To commit to a message x_{COM}, the committer (1) chooses random $x^1_{\mathrm{MPC}}, \ldots, x^M_{\mathrm{MPC}}$ such that $x^1_{\mathrm{MPC}} \oplus \cdots \oplus x^M_{\mathrm{MPC}} = x_{\mathrm{COM}}$, (2) chooses randomness $r^1_{\mathrm{MPC}}, \ldots, r^M_{\mathrm{MPC}}$ for the M parties of Π, and (3) commits to $\mathsf{st}^\mu_0 := (x^\mu_{\mathrm{MPC}}, r^\mu_{\mathrm{MPC}})$ for each $\mu \in [M]$ by using SHCom. (Note that each st^μ_0 can be thought of as an initial state of a party of Π.) For each $\mu \in [M]$, let $\mathsf{dec}^\mu_{\mathrm{SH}}$ denote the decommitment of SHCom for revealing st^μ_0.

Prove phase. In the first round, the receiver computes a receiver message of OT by using random $(\alpha, \beta) \in [M] \times [M]$ as the input,[4] and sends it to the committer.

In the second round, to prove $f(x_{\mathrm{COM}}) = 1$ for a predicate f, the committer does the following. (1) Execute Π in the head by using $\mathsf{st}^1_0, \ldots, \mathsf{st}^M_0$ as the initial states of the M parties and using $f' : (y^1, \ldots, y^M) \mapsto f(y^1 \oplus \cdots \oplus y^M)$ as the functionality to be computed. Let $\mathsf{view}^1, \ldots, \mathsf{view}^M$ be the views of the parties in this execution of Π. (2) For each $\mu \in [M]$, compute a commitment to $(\mathsf{dec}^\mu_{\mathrm{SH}}, \mathsf{view}^\mu)$ by using SBCom. Let $\mathsf{dec}^\mu_{\mathrm{SB}}$ be the decommitment of SBCom for revealing $(\mathsf{dec}^\mu_{\mathrm{SH}}, \mathsf{view}^\mu)$. (3) Compute a sender message of OT by using $\{(\mathsf{dec}^\mu_{\mathrm{SB}}, \mathsf{dec}^\nu_{\mathrm{SB}})\}_{\mu,\nu\in[M]}$ as the input. (4) Send the commitments and the OT message to the receiver.

In the verification, the receiver (1) recovers $\mathsf{dec}^\alpha_{\mathrm{SB}}, \mathsf{dec}^\beta_{\mathrm{SB}}$ from the OT message, (2) checks that they are valid decommitments of SBCom for revealing $\mathsf{dec}^\alpha_{\mathrm{SH}}, \mathsf{view}^\alpha, \mathsf{dec}^\beta_{\mathrm{SH}}, \mathsf{view}^\beta$ and that $\mathsf{dec}^\alpha_{\mathrm{SH}}, \mathsf{dec}^\beta_{\mathrm{SH}}$ are valid decommitments of SHCom for revealing $\mathsf{st}^\alpha_0, \mathsf{st}^\beta_0$, and (3) checks the following two conditions on $\mathsf{st}^\alpha_0, \mathsf{view}^\alpha, \mathsf{st}^\beta_0, \mathsf{view}^\beta$.

[3] Such an MPC protocol can be obtained unconditionally (e.g., the 2-private five-party protocol by Ben-Or et al. [BGW88, AL17]).

[4] We assume that the set $[M] \times [M]$ is identified with the set $[M^2]$ in a canonical way.

1. The views $\mathsf{view}^\alpha, \mathsf{view}^\beta$ are *consistent* in the sense that the messages that the party P^α receives from the party P^β in view^α is equal to the messages that P^β sends to P^α in view^β and vice versa.
2. For each $\xi \in \{\alpha, \beta\}$, the view view^ξ indicates that the initial state of P^ξ is st_0^ξ and the output is 1.

First, the constant soundness follows from the receiver security of OT and the perfect completeness of Π. Roughly speaking, this is because (1) the receiver security of OT guarantees that the committer can convince the verifier with high probability only when it commits to initial states and views that satisfy the above two conditions for every $\alpha, \beta \in [M]$,[5] and (2) when the committed initial states and views satisfy the above two conditions for every $\alpha, \beta \in [M]$, the perfect completeness of Π guarantees $f(x_{\mathrm{COM}}) = 1$, where x_{COM} is derived from the committed initial states. Next, the witness-indistinguishability follows from the receiver security of OT and the 2-privacy of Π. This is because the former guarantees that the receiver only learns the committed initial states and views of two parties and the latter guarantees the committed initial states and views of any two parties do not reveal any information about x_{COM}. Finally, this scheme is not succinct since the committer sends the initial states and views of Π (or more precisely the decommitments to them) via OT.

Now, to make the above scheme succinct, we combine it with KRR succinct argument. The idea is to let the committer send succinct arguments about the initial states and views (instead of the initial states and views themselves) via OT. That is, we let the committer prove that the above two conditions hold on the committed initial states and views of each pair of the parties, where a separate instance of KRR succinct argument is used for each pair of the parties, and let it send the resultant M^2 succinct arguments via OT. (Note that KRR succinct argument can naturally be combined with OT since it is a two-round protocol.) As a minor modification, we also let the committer use a succinct commitment scheme to commit to the initial states and the views.

Unfortunately, although the modifications are intuitive, proving the soundness of the resultant scheme is non-trivial. (In contrast, the WI property can be proven similarly to the WI property of the original scheme. The key point is that, although KRR succinct argument does not provide any witness privacy, we can still prove WI of the whole scheme since in each instance of KRR succinct argument, the witness—initial states and views of a pair of the parties—does not reveal any secret information anyway.)

A natural approach for proving the soundness would be to first prove the soundness of each instance of KRR succinct argument and then derive the soundness of the whole scheme from it. Indeed, if we can show that each of the M^2 instances of KRR succinct argument provides an argument-of-knowledge property (which allows us to extract the committed initial states and views from the cheating committer), we can easily prove the soundness of the whole scheme.

[5] Formally, complexity leveraging is required in this argument since the receiver security of OT needs to hold even against adversaries that extract the committed initial states and views by brute force.

The problem of this approach is that KRR succinct argument is not known to provide soundness for all statements in $\mathcal{NP}$, and hence, does not necessarily provide soundness when it is used as above.

Our actual approach is to show that, while each of the instances of KRR succinct argument does not necessarily provide soundness, they as a whole provide a meaningful notion of the soundness, which can be used to prove the soundness of the whole scheme. Specifically, by getting into the security proof of the soundness of KRR succinct argument, we show that when M^2 instances of KRR argument are used in parallel for proving the consistency of each pair of the committed views etc. as above, then they as a whole guarantee that the committed views are mutually consistent etc.

We give more detailed overviews of our approach from Section 3 to Section 6 after giving necessary definitions in Section 2.

2 Preliminaries

2.1 Notations and Conventions

We denote the security parameter by λ. We assume that every algorithm takes the security parameter as input, and often do not write it explicitly.

We identify a bit-string with a function in the following manner: a bit-string $x = (x_1, \ldots, x_n)$ is thought of as a function $x : [n] \to \{0,1\}$ such that $x(i) = x_i$. More generally, for any finite field $\boldsymbol{F}$, we identify a string over $\boldsymbol{F}$ with a function in the same manner. For a vector $\boldsymbol{v} = (v_1, \ldots, v_n)$ and a set $S \subseteq [n]$, we define $\boldsymbol{v}|_S$ by $\boldsymbol{v}|_S \coloneqq \{v_i\}_{i \in S}$. Similarly, for a function $f : D \to R$ and a set $S \subseteq D$, we define $f|_S$ by $f|_S \coloneqq \{f(i)\}_{i \in S}$.

For any two probabilistic interactive Turing machines A and B and any input x_A to A and x_B to B, we denote by $(\mathsf{out}_A, \mathsf{out}_B) \leftarrow \langle A(x_A), B(x_B)\rangle$ that the output of an interaction between $A(x_A)$ and $B(x_B)$ is $(\mathsf{out}_A, \mathsf{out}_B)$, where out_A is the output from A and out_B is the output from B.

2.2 Witness-indistinguishable Commit-and-prove Protocols

We give the definition of witness-indistinguishable commit-and-prove protocols. Our definition is based on the definition by Khurana et al. [KOS18] but is slightly different from it; see Appendix B for the differences.

A *witness-indistinguishable (WI) commit-and-prove protocol* $\langle C, R\rangle$ is a protocol between a committer $C = (\mathsf{C.Com}, \mathsf{C.Dec}, \mathsf{C.Prv})$ and a receiver $R = (\mathsf{R.Com}, \mathsf{R.Dec}, \mathsf{R.Prv})$, and it consists of three phases.

1. In the commit phase, $\mathsf{C.Com}$ takes a message $x \in \{0,1\}^n$ as input and interacts with $\mathsf{R.Com}$ to commit to x.[6] At the end of the interaction, $\mathsf{C.Com}$ outputs its internal state st_C and $\mathsf{R.Com}$ outputs the commitment com, which is the transcript of the commit phase.

[6] We assume that the length of the message to be committed, n, is implicitly given to the receiver as input.

2. In the prove phase, $\mathsf{C.Prv}$ takes a predicate f as input along with st_C, and interacts with $\mathsf{R.Prv}$ to prove that $f(x) = 1$ holds, where $\mathsf{R.Prv}$ takes (com, f) as input, At the end of the interaction, $\mathsf{R.Prv}$ outputs either 1 (accept) or 0 (reject).
3. In the open phase, $\mathsf{C.Dec}$ takes an index $i \in [n]$ as input along with st_C, and interacts with $\mathsf{R.Dec}$ to reveal the i-th bit of x, where $\mathsf{R.Dec}$ takes (com, i) as input. At the end of the interaction, $\mathsf{R.Dec}$ outputs either a bit x_i as the decommitted bit, or $\bot$ (reject).

In this paper, we focus on a WI commit-and-prove protocol such that (1) both the prove phase and the open phase consist of two rounds, (2) the first round of the prove phase does not depend on the commitment com and the predicate f,[7] and (3) the first round of the open phase does not depend on the commitment com. Because of (1) and (2), $\mathsf{R.Prv}$ can be split into two algorithms, $\mathsf{R.Prv.Q}$ and $\mathsf{R.Prv.D}$, such that the prove phase proceeds as follows: $(Q, \mathsf{st}_R) \leftarrow \mathsf{R.Prv.Q}$; $\pi \leftarrow \mathsf{C.Prv}(\mathsf{st}_C, f, Q)$; $b \leftarrow \mathsf{R.Prv.D}(\mathsf{st}_R, \mathsf{com}, f, \pi)$. Similarly, because of (1) and (3), $\mathsf{R.Dec}$ can be split into two algorithms, $\mathsf{R.Dec.Q}$ and $\mathsf{R.Dec.D}$, such that the open phase proceeds as follows: $(Q, \mathsf{st}_R) \leftarrow \mathsf{R.Dec.Q}(i)$; $\mathsf{dec} \leftarrow \mathsf{C.Dec}(\mathsf{st}_C, i, Q)$; $b \leftarrow \mathsf{R.Dec.D}(\mathsf{st}_R, \mathsf{com}, \mathsf{dec})$.

WI commit-and-prove protocols need to satisfy the following security notions.

Definition 1 (Completeness). *A commit-and-prove protocol $\langle C, R\rangle$ is* complete *if for any polynomial $n : \mathbb{N} \to \mathbb{N}$ and any $\lambda \in \mathbb{N}$, $x \in \{0,1\}^{n(\lambda)}$, and $i \in [n(\lambda)]$,*

$$\Pr\left[x_i = \tilde{x}_i \;\middle|\; \begin{array}{l}(\mathsf{st}_C, \mathsf{com}) \leftarrow \langle \mathsf{C.Com}(x), \mathsf{R.Com}\rangle \\ (\bot, \tilde{x}_i) \leftarrow \langle \mathsf{C.Dec}(\mathsf{st}_C, i), \mathsf{R.Dec}(\mathsf{com}, i)\rangle\end{array}\right] = 1 \ .$$

Definition 2 (Binding). *A commit-and-prove protocol $\langle C, R\rangle$ is* (computationally) binding *if for any polynomial $n : \mathbb{N} \to \mathbb{N}$, any* PPT *cheating committer $C^* = (\mathsf{C.Com}^*, \mathsf{C.Dec}^*)$, and any $\lambda \in \mathbb{N}$, the following binding condition holds with overwhelming probability over the choice of $(\mathsf{st}_C, \mathsf{com}) \leftarrow \langle \mathsf{C.Com}^*, \mathsf{R.Com}\rangle$.*

- ***Binding Condition:*** *For every $i \in [n(\lambda)]$, it holds $\Pr[b_{\mathrm{BAD}} = 1] \leq \mathsf{negl}(\lambda)$ in the following probabilistic experiment $\mathrm{EXP}^{\mathrm{bind}}(\mathsf{C.Dec}^*, \mathsf{st}_C, \mathsf{com}, i)$.*
  1. *For each $b \in \{0,1\}$, sample Q_b by $(Q_b, \mathsf{st}_b) \leftarrow \mathsf{R.Dec.Q}(i)$.*
  2. *Run $\{\mathsf{dec}_b\}_{b\in\{0,1\}} \leftarrow \mathsf{C.Dec}^*(\mathsf{st}_C, i, \{Q_b\}_{b\in\{0,1\}})$.*
  3. *For each $b \in \{0,1\}$, let $x_b^* \leftarrow \mathsf{R.Dec.D}(\mathsf{st}_b, \mathsf{com}, \mathsf{dec}_b)$.*
  4. *Output $b_{\mathrm{BAD}} := 1$ if and only if $x_0^* \neq \bot \wedge x_1^* \neq \bot \wedge x_0^* \neq x_1^*$ holds.*

Definition 3 (Soundness). *Let $\epsilon : \mathbb{N} \to [0,1]$ be a function. A commit-and-prove protocol $\langle C, R\rangle$ is* (computationally) ϵ-sound *if for any constant $c \in \mathbb{N}$, there exists a* PPT *oracle Turing machine E (called an* extractor*) such that for any polynomial $n : \mathbb{N} \to \mathbb{N}$, any* PPT *cheating committer $C^* = (\mathsf{C.Com}^*, \mathsf{C.Prv}^*)$, and any sufficiently large $\lambda \in \mathbb{N}$, the following soundness condition holds with overwhelming probability over the choice of $(\mathsf{st}_C, \mathsf{com}) \leftarrow \langle \mathsf{C.Com}^*, \mathsf{R.Com}\rangle$.*

[7] We assume that $\mathsf{Time}(f)$ is known to the both parties in advance (where f is expressed as, e.g., a Turing machine).

– ***Soundness Condition***[8]***:*** *If it holds*

$$\Pr\left[b=1 \;\middle|\; \begin{array}{l}(Q,\mathsf{st}_R)\leftarrow \mathsf{R.Prv.Q};\ (f,\pi)\leftarrow \mathsf{C.Prv}^*(\mathsf{st}_C,Q);\\ b\leftarrow \mathsf{R.Prv.D}(\mathsf{st}_R,\mathsf{com},f,\pi)\end{array}\right]\geq \epsilon(\lambda)+\frac{1}{\lambda^c}\ ,$$

then there exists $x^*=(x_1^*,\ldots,x_n^*)\in\{0,1\}^{n(\lambda)}$ *such that*

$$\forall i\in[n(\lambda)],\Pr\left[x_i=x_i^* \;\middle|\; (\bot,x_i)\leftarrow\langle E^{\mathsf{C.Prv}^*(\mathsf{st}_C,\cdot)}(\mathsf{com},i),\mathsf{R.Dec}(\mathsf{com},i)\rangle\right]\geq 1-\mathsf{negl}(\lambda)$$

and

$$\Pr\left[\begin{array}{l}b=1\\ \wedge\ f(x^*)=0\end{array} \;\middle|\; \begin{array}{l}(Q,\mathsf{st}_R)\leftarrow \mathsf{R.Prv.Q};\ (f,\pi)\leftarrow \mathsf{C.Prv}^*(\mathsf{st}_C,Q);\\ b\leftarrow \mathsf{R.Prv.D}(\mathsf{st}_R,\mathsf{com},f,\pi)\end{array}\right]\leq \epsilon(\lambda)+\mathsf{negl}(\lambda)\ .$$

$\langle C,R\rangle$ *is said to be* sound *if it is* ϵ*-sound for a negligible function* ϵ*.*

Definition 4 (Witness Indistinguishability). $\langle C,R\rangle$ *is* witness-indistinguishable *if for any polynomial* $n:\mathbb{N}\to\mathbb{N}$*, any two sequences* $\{x_\lambda^0\}_{\lambda\in\mathbb{N}}$ *and* $\{x_\lambda^1\}_{\lambda\in\mathbb{N}}$ *such that* $x_\lambda^0,x_\lambda^1\in\{0,1\}^{n(\lambda)}$*, any* PPT *cheating receiver* $R^*=(\mathsf{R.Com}^*,\mathsf{R.Prv.Q}^*)$*, the outputs of Experiment* 0 *and Experiment* 1 *are computationally indistinguishable.*

– *Experiment* b *(*$b\in\{0,1\}$*).*
  1. $(\mathsf{st}_C,\mathsf{st}_R)\leftarrow\langle\mathsf{C.Com}(x_\lambda^b),\mathsf{R.Com}^*(x_\lambda^0,x_\lambda^1)\rangle$.
  2. $(f,Q,\mathsf{st}'_R)\leftarrow\mathsf{R.Prv.Q}^*(\mathsf{st}_R)$*. If* $f(x_\lambda^0)\neq 1$ *or* $f(x_\lambda^1)\neq 1$*, abort.*
  3. $\pi\leftarrow\mathsf{C.Prv}(\mathsf{st}_C,f,Q)$.
  4. *Output* (st'_R,π).

2.3 Secure Multi-party Computation

We recall the definition of secure multi-party computation (MPC) protocols based on the description by Ishai et al. [IKOS09]. (We assume that the readers are familiar with the concept of secure MPC protocols.)

The basic model that is used in this paper is the following. The number of parties is denoted by M. We focus on MPC protocols that realize any deterministic M-party functionality that outputs a single bit (which is obtained by all the parties), given the synchronous communication over secure point-to-point channels. We assume that every party implicitly takes as input the M-party functionality to be computed.

Recall that the *view* of a party in an execution of an MPC protocol consists of its input, its randomness, and all the incoming messages that it received from the other parties during the execution of the protocol. The consistency between a pair of views is defined as follows.

[8] Roughly speaking, the soundness condition requires that if a cheating prover convinces the verifier with sufficiently high probability, then there exists a value x^* such that (1) the extractor can decommit com to x^* and (2) the cheating prover cannot prove false statements about x^*.

Definition 5 (Consistent Views). *A pair of views* $\mathsf{view}^i, \mathsf{view}^j$ *is* consistent *(w.r.t. an MPC protocol* Π *for a functionality* f*) if the outgoing messages that are implicitly reported in* view^i *are identical to the incoming messages that are reported in* view^j *and vice versa.*

We consider security against semi-honest adversaries. Concretely, we use the following two security notions.

Definition 6 (Perfect correctness). *We say that an MPC protocol* Π *satisfies* perfect correctness *if for any deterministic* M*-party functionality* f *and for any private inputs to the parties, the probability that the output of some party in an honest execution of* Π *is different from the output of* f *is* 0*.*

Definition 7 (2-privacy). *We say that an MPC protocol* Π *satisfies* perfect 2-privacy *if for any deterministic* M*-party functionality* f*, there exists a* PPT *simulator* $\mathcal{S}_{\text{MPC}}$ *such that for any private inputs* $x_1, \ldots, x_M$ *to the parties and every pair of corrupted parties,* $T \subset [M]$ *such that* $|T| = 2$*, the joint view* $\mathsf{View}_T(x_1, \ldots, x_M)$ *of the parties in* T *is identically distributed with* $\mathcal{S}_{\text{MPC}}(T, \{x_i\}_{i \in T}, f(x_1, \ldots, x_M))$*.*

2.4 Probabilistically Checkable Proofs (PCPs)

We recall the definition of probabilistically checkable proofs (PCPs) based on the description by Brakerski et al. [BHK17]. Roughly speaking, PCPs are proof systems with which one can probabilistically verify the correctness of statements by reading only a few bits or symbols of the proof strings. A formal definition is given below.

Definition 8. *A* κ*-query PCP system* (P, V) *for an NP language* L*, where* $\mathsf{V} = (\mathsf{Q}, \mathsf{D})$*, satisfies the following.*

- ***(Completeness)*** *For all* $\lambda \in \mathbb{N}$ *and* $x \in L$ *(with witness* w*) such that* $|x| \leq 2^\lambda$*,*

$$\Pr\left[\mathsf{D}(\mathsf{st}, x, \pi|_Q) = 1 \;\middle|\; \begin{array}{l}(Q, \mathsf{st}) \leftarrow \mathsf{Q}(1^\lambda) \\ \pi \leftarrow \mathsf{P}(1^\lambda, x, w)\end{array}\right] = 1 \ .$$

  The PCP proof π *is a string of characters over some alphabet* Σ*, and it can be thought that this string is indexed by a set* Γ *(by identifying* Γ *with* $[N]$ *in a canonical way, where* N *is the length of the string) and* $Q \subseteq \Gamma$*. Alternatively,* π *can be thought of as a function from* Γ *to* Σ*.*
- ***(Soundness)*** *For all* $\lambda \in \mathbb{N}$*, all* $x \notin L$ *such that* $|x| \leq 2^\lambda$*, and all proof string* π^**,*

$$\Pr\left[\mathsf{D}(\mathsf{st}, x, \pi^*|_Q) \;\middle|\; (Q, \mathsf{st}) \leftarrow \mathsf{Q}(1^\lambda)\right] \leq \frac{1}{2} \ .$$

- ***(Query Efficiency)*** *If* $(Q, \mathsf{st}) \leftarrow \mathsf{Q}(1^\lambda)$*, then* $|Q| \leq \kappa(\lambda)$ *and the combined run-time of* Q *and* D *is* $\mathsf{poly}(\lambda)$*.*
- ***(Prover Efficiency)*** *The prover* P *runs in polynomial time, where its input is* $(1^\lambda, x, w)$*.*

2.5 Definitions from Kalai et al. [KRR14] and Subsequent Works

Computational no-signaling (CNS). We recall the definition of *adaptive (computational) no-signaling* [KRR14, BHK17].

Definition 9. *Fix any alphabet $\{\Sigma_\lambda\}_{\lambda\in\mathbb{N}}$, any $\{N_\lambda\}_{\lambda\in\mathbb{N}}$ such that $N_\lambda \in \mathbb{N}$, any function $\kappa_{\max} : \mathbb{N} \to \mathbb{N}$ such that $\kappa_{\max}(\lambda) \leq N_\lambda$, and any algorithm* Algo *such that for any $\lambda \in \mathbb{N}$, on input a subset $Q \subset [N_\lambda]$ of size at most $\kappa_{\max}(\lambda)$,* Algo *outputs (the truth table of) a function $A : Q \to \Sigma \cup \{\perp\}$ with an auxiliary output* out.

Then, the algorithm Algo *is* adaptive $\kappa_{\max}$-computational no-signaling *(CNS) if for any* PPT *distinguisher $\mathcal{D}$, any sufficiently large $\lambda \in \mathbb{N}$, any $Q, S \subset [N_\lambda]$ such that $Q \subseteq S$ and $|S| \leq \kappa_{\max}(\lambda)$, and any $z \in \{0,1\}^{\mathsf{poly}(\lambda)}$,*

$$\left| \begin{array}{l} \Pr\left[\mathcal{D}(\mathsf{out}, A, z) = 1 \mid (\mathsf{out}, A) \leftarrow \mathsf{Algo}(Q)\right] \\ \quad - \Pr\left[\mathcal{D}(\mathsf{out}, A|_Q, z) = 1 \mid (\mathsf{out}, A) \leftarrow \mathsf{Algo}(S)\right] \end{array} \right| \leq \mathsf{negl}(\lambda) \ .$$

We remark that the above definition can be naturally extended for the case that Algo takes auxiliary inputs, as well as for the case that Algo takes multiple subsets as input and then outputs multiple functions (see the full version of this paper) .

Adaptive local assignment generator. We recall the definition of *adaptive local assignment generators* [PR14, BHK17].

Definition 10. *For any function $\kappa_{\max} : \mathbb{N} \to \mathbb{N}$, an* adaptive $\kappa_{\max}$-local assignment generator Assign *on variables $\{V_\lambda\}_{\lambda\in\mathbb{N}}$ is an algorithm that takes as input a security parameter 1^λ and a set of at most $\kappa_{\max}(\lambda)$ queries $W \subseteq \{1, \ldots, |V_\lambda|\}$, and outputs a 3CNF formula φ on variables V_λ and assignments $A : W \to \{0,1\}$ such that the following two properties hold.*

- ***Everywhere Local Consistency.*** *For every $\lambda \in \mathbb{N}$ and every set $W \subseteq \{1, \ldots, |V_\lambda|\}$ such that $|W| \leq \kappa_{\max}(\lambda)$, with probability at least $1 - \mathsf{negl}(\lambda)$ over sampling $(\varphi, A) \leftarrow \mathsf{Assign}(1^\lambda, W)$, the assignment A is "locally consistent" with the formula φ. That is, for any $i_1, i_2, i_3 \in W$, if φ has a clause whose variables are $v_{i_1}, v_{i_2}, v_{i_3}$, then this clause is satisfied with the assignment $A(i_1), A(i_2), A(i_3)$ with probability at least $1 - \mathsf{negl}(\lambda)$.*
- ***Computational No-signaling.*** Assign *is adaptive $\kappa_{\max}$-CNS.*

No-signaling PCPs. We recall the definition of *(computational) no-signaling PCPs* [KRR14, BHK17]. Essentially, no-signaling PCPs are PCP systems that are sound against no-signaling cheating provers. Specifically, for any function $\kappa_{\max} : \mathbb{N} \to \mathbb{N}$, a PCP system (P, V) for a language L, where $\mathsf{V} = (\mathsf{Q}, \mathsf{D})$, is *adaptive $\kappa_{\max}$-no-signaling sound* with negligible soundness error if it satisfies the following.

– **(No-signaling Soundness)** For any adaptive $\kappa_{\max}$-CNS cheating prover P^* and any $\lambda \in \mathbb{N}$,

$$\Pr\left[x^* \notin L \wedge \mathsf{D}(\mathsf{st}, x^*, \pi^*) = 1 \;\middle|\; \begin{array}{l}(Q, \mathsf{st}) \leftarrow \mathsf{Q}(1^\lambda) \\ (x^*, \pi^*) \leftarrow P^*(1^\lambda, Q)\end{array}\right] \leq \mathsf{negl}(\lambda) \ .$$

3 Outline of Proof of Theorem 1

As mentioned in Section 1.2, our commit-and-prove protocol uses the succinct argument of Kalai et al. [KRR14] (KRR succinct argument in short). Unfortunately, we do not use it modularly—we slightly modify a building block of KRR succinct argument (namely, their no-signaling PCP system) when constructing our protocol, and we see low-level parts of the analysis of KRR succinct argument when analyzing our protocol.

At a high level, KRR succinct argument is obtained in three steps, starting from a scheme with a weak soundness notion.

1. Obtain a PCP system such that no CNS adversary can break the soundness with overwhelming success probability.
2. Obtain a PCP system such that no CNS adversary can break the soundness with non-negligible success probability.
3. Obtain a succinct argument such that no adversary can break the soundness with non-negligible success probability.

Somewhat similarly, our commit-and-prove protocol is obtained in five steps, starting from a non-WI scheme with a weak soundness notion.

1. Obtain a non-WI scheme, $\langle C_1, R_1 \rangle$, such that no CNS "well-behaving" adversary can break the soundness with overwhelming success probability. (Well-behaving adversaries is the class of adversaries that we introduce later.)
2. Obtain a non-WI scheme, $\langle C_2, R_2 \rangle$, such that no CNS adversary can break the soundness with overwhelming success probability.
3. Obtain a non-WI scheme, $\langle C_3, R_3 \rangle$, such that no CNS adversary can break the soundness with non-negligible success probability.
4. Obtain a non-WI scheme, $\langle C_4, R_4 \rangle$, such that no adversary can break the soundness with non-negligible success probability.
5. Obtain a WI scheme, $\langle C_5, R_5 \rangle$, such that no adversary can break the soundness with constant success probability.

The most technically interesting step is the first step, and an extensive overview of this step is given in Section 4. Overviews of the other steps are given in Section 5 and Section 6. The formal proof is given in the full version of this paper.

3.1 Building Block: Perfect 2-private MPC protocol Π

In addition to the cryptographic primitives that are listed in Theorem 1, we use a 2-private semi-honest secure M-party computation protocol Π with perfect completeness, where M is an arbitrary constant. (Note that such an MPC protocol can be obtained unconditionally; cf. Footnote 3.) We denote the parties of Π by $P^1, \dots P^M$.

For editorial simplicity, we make several simplifying assumptions on Π.

- The length of the initial state of each party is denoted by $n_{\mathrm{st}} = n + n_{\mathrm{MPC}}$, where n is the input length and n_{MPC} is the randomness length, and each party has n_{st}-bit internal state at the beginning of each round.
- Every party uses the same next-message function in every round.[9]
- Every party sends a 1-bit message to each party at the end of each round.
- Every party receives dummy incoming messages from all the parties at the beginning of the first round, and every party sends a dummy outgoing message to itself at the end of each round. (This assumption is made so that the next-message function always takes an $(n_{\mathrm{st}} + M)$-bit input, where the last M bits are the concatenation of the incoming messages.)
- The first bit of the final state of each party denotes the output of that party.

4 Overview of Step 1 (Non-WI Scheme with Soundness against CNS Well-behaving Provers)

We give an extensive overview of our non-WI commit-and-prove protocol $\langle C_1, R_1 \rangle$, which is $(1 - \mathsf{negl})$-sound against CNS "well-behaving" provers. At a high level, we follow the approach that we outline in Section 1.2. That is, we implement the MPC-in-the-head technique with the MPC protocol Π and a succinct argument. However, instead of using KRR succinct argument, we use a variant of the no-signaling PCP system $(\mathsf{PCP.P}_{\mathsf{KRR}}, \mathsf{PCP.V}_{\mathsf{KRR}})$ of Kalai et al. [KRR14] (which is the main building block of KRR succinct argument and is referred to as KRR no-signaling PCP in what follows), and we do not use any cryptographic primitives in this step so that we can focus on information theoretical arguments in the analysis. As a result, we can prove soundness only against very restricted provers, which we define as CNS well-behaving provers.

For simplicity, in this overview, we focus on static soundness, where the statement to be proven by the cheating prover is fixed at the beginning of the prove phase. We will also make several implicit oversimplifications in this overview.

[9] The next-message function takes as input an internal state and incoming messages of a round, and it outputs the internal state and outgoing messages of the round. (We assume that the internal state implicitly includes all the incoming messages of the previous rounds.)

4.1 Preliminary: Overview of Analysis of KRR No-signaling PCP

We start by briefly recalling the analysis of KRR no-signaling PCP (i.e., the analysis of its no-signaling soundness for statements in $\mathcal{P}$), focusing on the parts that are relevant to this work.[10]

We first remark that KRR no-signaling PCP is a PCP system for 3SAT, so at the beginning the statement to be proven is converted into a 3SAT instance. Specifically, given any statement in $\mathcal{P}$ of the form "(f, x) satisfies $f(x) = 1$" for some public function f and input x, first the function f is converted into a carefully designed Boolean circuit C that computes f, and next the statement is converted into a 3SAT instance φ that has the following properties.

1. φ has a variable for each of the wires in C, and the values that are assigned to these variables are interpreted as an assignment to the corresponding wires in C.
2. The clauses of φ checks that (1) for each gate in C, the assignment to its input and output wires is consistent with the computation of the gate, (2) the assignment to the input wires of C is equal to x, and (3) the assignment to the output wire of C is equal to 1.

Now, the analysis of KRR no-signaling PCP roughly consists of three parts.

The first part of the analysis shows that any successful CNS cheating prover for a statement (f, x) can be converted into a local assignment generator for the 3SAT instance φ that is obtained from (f, x) as above. That is, it shows that any successful CNS cheating prover can be converted into a probabilistic algorithm Assign such that (1) Assign takes as input a small-size subset of the variables of φ and it outputs an assignment to these variables, and (2) Assign is guaranteed to satisfy the following everywhere local consistency.

Everywhere local consistency. Assign does not make an assignment that violates any clause of φ. Specifically, when Assign is asked to make an assignment to the three variables that appear in a clause of φ, it makes an assignment that satisfies this clause.

(Actually, Assign is also guaranteed to be CNS, but we ignore it in this overview for simplicity.[11]) We note that Assign does not necessarily comply with a single global assignment, that is, Assign can assign different values to the same variable depending on the randomness and the input. We also note that this part of the analysis holds even for statements in $\mathcal{NP}$. For simplicity, in this overview we assume that Assign does not err (i.e., the everywhere local consistency holds with probability 1).

The second part of the analysis shows that the local assignment generator Assign that is obtained in the first part is guaranteed to comply with a single global "correct" assignment. A bit more precisely, this part shows the following.

[10] We follow the modularization by Paneth and Rothblum [PR14].

[11] Concretely, in this overview we assume that Assign is perfect no-signaling, i.e., that the RHS of the equation in Definition 10 is 0 even against computationally unbounded distinguishers.

Let *the correct assignment* to a wire in C (or, equivalently, to a variable in φ) be defined as the assignment that is obtained by evaluating C on x, and let Assign be called *correct* on a wire in C (or variable in φ) if Assign makes the correct assignment to it whenever Assign is asked to make an assignment to it. Then, Assign is correct on any wire in C (or variable in φ), and in particular correct on the output wire of C.

Roughly speaking, the above is shown in two steps.

1. First, it is shown, by relying on a specific structure of C, that Assign is correct on any wire in C if Assign is correct on each input wire of C.
2. Next, it is observed that Assign is indeed correct on each input wire of C due to the everywhere local consistency and the definition of φ (which has clauses that check that the assignment to the input wires of C is equal to x).

Finally, the last part of the analysis obtains the soundness by combining what are shown by the preceding two parts. In particular, it is observed that the existence of Assign as above implies $f(x) = 1$ since (1) on the one hand, Assign always assigns 1 to the output wire of C due to the everywhere local consistency and the definition of φ (which has clauses that check that the assignment to the output wire of C is 1), and (2) on the other hand, Assign always assigns $f(x)$ to the output wire of C since what is shown by the second part implies that Assign is correct on the output wire of C.

Remark 1 (Difficulty in the case of $\mathcal{NP}$ statements). The above analysis does not work in general for statements in $\mathcal{NP}$. A difficulty is that when the statement is in $\mathcal{NP}$, it is unclear how we should define the correct assignment in the second part of the analysis. Indeed, on the one hand, the correct assignment can be naturally defined in the case of statements in $\mathcal{P}$ since there exists a unique assignment that any successful prover is supposed to use (namely the assignment that is derived from x); on the other hand, in the case of statements in $\mathcal{NP}$, there does not exist a single such assignment. Jumping ahead, below we define well-behaving provers so that we can define the correct assignment naturally (while at the same time so that we can use cryptographic primitives later to force any prover to be well-behaving). $\Diamond$

4.2 Protocol Description

In this overview, we consider the following protocol $\langle C_1, R_1 \rangle = (\mathsf{C.Com}_1, \mathsf{C.Prv}_1, \mathsf{R.Com}_1, \mathsf{R.Prv.Q}_1, \mathsf{R.Prv.D}_1)$, which is slightly oversimplified from the actual protocol (see the full version of this paper for the actual protocol). (At this point, we temporarily ignore the open phase.) We warn that $\langle C_1, R_1 \rangle$ is not biding at all in the standard sense since the committer sends no message in the commit phase.

Commit Phase:

Round 1: Given x_{COM} as the value to be committed, $\mathsf{C.Com}_1$ does the following.

1. Sample random $x^1_{\mathrm{MPC}}, \ldots, x^M_{\mathrm{MPC}}$ such that $x^1_{\mathrm{MPC}} \oplus \cdots \oplus x^M_{\mathrm{MPC}} = x_{\mathrm{COM}}$.
2. For each $\mu \in [M]$, define $x^\mu_{1,\mathsf{in}}$ as follows: sample random $r^\mu_{\mathrm{MPC}} \in \{0,1\}^{n_{\mathrm{MPC}}}$ and let $\mathsf{st}^\mu_0 := x^\mu_{\mathrm{MPC}} \| r^\mu_{\mathrm{MPC}}$, $\mathsf{i\text{-}msgs}^\mu_1 := 0^M$, $x^\mu_{1,\mathsf{in}} := \mathsf{st}^\mu_0 \| \mathsf{i\text{-}msgs}^\mu_1$.
3. Output an empty string as the commitment and store $\{x^\mu_{1,\mathsf{in}}\}_{\mu\in[M]}$ as the internal state.

Prove Phase:

Round 1 $\mathsf{R.Prv.Q}_1$ does the following.

1. For each $\mu, \nu \in [M]$, obtain a set of queries $Q^{\mu,\nu}$ by running the verifier of KRR no-signaling PCP.
2. Output $\{Q^{\mu,\nu}\}_{\mu,\nu\in[M]}$ as the query.

Round 2: Given the statement f and the query $\{Q^{\mu,\nu}\}_{\mu,\nu\in[M]}$ as input, $\mathsf{C.Prv}_1$ does the following.

1. Run the MPC protocol Π in the head for functionality f' and initial states $\{(\mathsf{st}^\mu_0, \mathsf{i\text{-}msgs}^\mu_1)\}_{\mu\in[M]}$,[12] where f' is defined as $f' : (y^1, \ldots, y^M) \mapsto f(y^1 \oplus \cdots \oplus y^M)$ and each $(\mathsf{st}^\mu_0, \mathsf{i\text{-}msgs}^\mu_1)$ is recovered from the internal state of the commit phase. Let $\{\mathsf{view}^\mu\}_{\mu\in[M]}$ be the view of the parties in this execution.
2. For each $\mu, \nu \in [M]$, obtain a PCP proof $\pi^{\mu:\nu}$ by running the prover of KRR no-signaling PCP on the 3SAT instance $\varphi^{\mu:\nu}$ that we will carefully design later—roughly speaking, $\varphi^{\mu:\nu}$ takes views of the parties P^μ, P^ν of Π as input, and checks that the views are consistent and that P^μ and P^ν output 1 in the views. (In an honest execution, $\mathsf{C.Prv}_1$ uses $(\mathsf{view}^\mu, \mathsf{view}^\nu)$ to obtain a satisfying assignment to $\varphi^{\mu:\nu}$ and then uses it to obtain $\pi^{\mu:\nu}$.)
3. Output $\{\pi^{\mu:\nu}|_{Q^{\mu:\nu}}\}_{\mu,\nu\in[M]}$ as the proof.

Verification: Given the statement f and the proof $\{\pi^{*\mu:\nu}\}_{\mu,\nu\in[M]}$ as input, $\mathsf{R.Prv.D}_1$ does the following.

1. Verify each $\pi^{*\mu:\nu}$ by running the verifier of KRR no-signaling PCP, and let $b^{\mu:\nu}$ be the verification result.
2. Output 1 if and only if $b^{\mu:\nu} = 1$ for every $\mu, \nu \in [M]$.

4.3 Proof of Soundness

We give an overview of the proof of the soundness. To focus on the main technical idea, in this overview we consider a weak version of the soundness where the extractor is only required to extract a committed value (rather than decommit the commitment as required in Definition 3). Thus, for any successful cheating prover, the extractor is required to extract a value such that the cheating prover cannot prove false statements on it.

[12] Each $\mathsf{i\text{-}msgs}^\mu_1$ is the dummy incoming messages of the first round (cf. Section 3.1).

Overall approach. At a very high level, the proof consists of two parts.

The first part is to obtain an extractor. Toward this end, we first observe that, by borrowing analyses from Kalai et al. [KRR14], we can convert any successful CNS cheating prover against $\langle C_1, R_1 \rangle$ into a *parallel local assignment generator* p-Assign, which gives M^2 local assignments to the 3SAT instances $\{\varphi^{\mu:\nu}\}_{\mu,\nu\in[M]}$ in parallel when it is given M^2 subsets of the variables as input. (To see this, observe that the prove phase of $\langle C_1, R_1 \rangle$ consists of M^2 parallel executions of KRR no-signaling PCP.) Then, we obtain an extractor by using p-Assign as follows.

- Note that since each $\varphi^{\mu:\nu}$ is a 3SAT instance that takes views of P^μ, P^ν as input, for any particular parts of P^μ and P^ν's views, $\varphi^{\mu:\nu}$ has variables that are supposed to be assigned with these parts. In the following, when we say that p-Assign makes an assignment to particular parts of P^μ and P^ν's views in $\varphi^{\mu:\nu}$, we mean that p-Assign makes an assignment to the variables that are supposed to be assigned with these parts in $\varphi^{\mu:\nu}$.
- Now, to extract the i-th bit of the committed value, the extractor obtains the i-th bit of each party's MPC input by asking p-Assign to make an assignment to the i-th bit of P^μ's input in $\varphi^{\mu:\mu}$ for every $\mu \in [M]$, and then takes XOR of the obtained bits.

The second part is to show that any cheating prover cannot prove false statements on the extracted value. In this part, the analysis proceeds similarly to the analysis of KRR no-signaling PCP. That is, we first define the correct assignment for each of $\varphi^{\mu:\nu}$, and next show that p-Assign always makes the correct assignment to any variable in any of $\varphi^{\mu:\nu}$.

Unfortunately, we do not know how to prove the second part against CNS cheating provers in general, and thus, we further restrict the provers to be "well-behaving."

Well-behaving provers. Roughly speaking, we define well-behaving provers as follows. Recall that the extractor is obtained by converting the cheating prover into a parallel local assignment generator. Now, we define well-behaving provers so that when we convert a successful CNS well-behaving prover into a parallel local assignment generator p-Assign, it satisfies the following two consistency properties.

Consistency on the initial states: Once the commit phase is completed, there exists a unique set of MPC initial states $\{(\mathsf{st}_0^\mu, \mathsf{i\text{-}msgs}_1^\mu)\}_{\mu\in[M]}$ such that p-Assign always makes assignments that are consistent with it (i.e., for any $\mu, \nu \in [M]$, when p-Assign is asked to make an assignment to any bit of the initial state of P^μ or P^ν in $\varphi^{\mu:\nu}$, then p-Assign always assigns the corresponding bit of $(\mathsf{st}_0^\mu, \mathsf{i\text{-}msgs}_1^\mu)$ or $(\mathsf{st}_0^\nu, \mathsf{i\text{-}msgs}_1^\nu)$).

Consistency on the views: For every $\mu, \nu, \xi \in [M]$, when p-Assign is asked to make an assignment to any bit of P^μ's view in both $\varphi^{\mu:\nu}$ and $\varphi^{\mu:\xi}$, then the value that p-Assign assigns to it in $\varphi^{\mu:\nu}$ is identical with the value that

p-Assign assigns to it in $\varphi^{\mu:\xi}$. (The same holds for $\varphi^{\nu:\mu}$ and $\varphi^{\xi:\mu}$ and for $\varphi^{\mu:\nu}$ and $\varphi^{\xi:\mu}$ etc.)

Remark 2 (Intuition of the two consistency properties of p-Assign*).* Essentially, the above two consistency properties guarantee that p-Assign behaves as if it were obtained from an honest prover. This is because when p-Assign is indeed obtained from an honest prover, we can show that p-Assign always assigns the same MPC initial states once the commit phase is fixed, and assigns the same P^μ's view in any $\varphi^{\mu:\nu}$ and $\varphi^{\mu:\xi}$. (Roughly speaking, this is because in an honest execution of $\langle C_1, R_1\rangle$, a set of MPC initial states are fixed in the commit phase, and the same P^μ's view is used for computing PCPs on any $\varphi^{\mu:\nu}$ and $\varphi^{\mu:\xi}$ in the prove phase.) $\Diamond$

Before giving more details on the definition of well-behaving provers, we show that by restricting the provers to be well-behaving, we can complete the second part of the above overall approach, where our goal is to show that any cheating prover cannot prove false statements on the extracted value.

Showing that cheating prover cannot prove false statements. As stated earlier, the analysis proceeds similarly to the analysis of KRR no-signaling PCP. That is, we first define the correct assignment for each of $\varphi^{\mu:\nu}$, and next show that p-Assign always makes the correct assignment to any variable in any of $\varphi^{\mu:\nu}$.

Step 1: Defining the correct assignments. We define the correct assignments for $\{\varphi^{\mu:\nu}\}_{\mu,\nu\in[M]}$ by relying on that p-Assign satisfies the consistency on the initial states. Recall that it guarantees that once the commit phase is completed, there exists a unique set of MPC initial states $\{(\mathsf{st}_0^\mu, \mathsf{i\text{-}msgs}_1^\mu)\}_{\mu\in[M]}$ such that p-Assign always makes local assignments that are consistent with it. Then, we first define *the correct views* $\{\mathsf{view}^\mu\}_{\mu\in[M]}$ as the views that are obtained by executing Π on these unique initial states $\{(\mathsf{st}_0^\mu, \mathsf{i\text{-}msgs}_1^\mu)\}_{\mu\in[M]}$, and then define *the correct assignment* for $\varphi^{\mu:\nu}$ $(\mu, \nu \in [M])$ as the assignment that is derived from the correct views $(\mathsf{view}^\mu, \mathsf{view}^\nu)$ of P^μ, P^ν. (Recall that $\varphi^{\mu:\nu}$ is a 3SAT instance that takes views of P^μ, P^ν as input.)

From the definition, it is clear that p-Assign is correct on the initial states in every $\varphi^{\mu:\nu}$ (i.e., p-Assign always assigns the correct assignment to any bit of the initial states of P^μ, P^ν in $\varphi^{\mu:\nu}$ for every $\mu, \nu \in [M]$). Also, since the extractor extracts the committed value by taking XOR of the MPC inputs that are obtained from p-Assign, p-Assign's correctness on the initial states implies that the value that the extractor extracts is unique and is equal to the XOR of the MPC inputs that are used in the correct views.

Step 2: Showing that p-Assign *is correct on every variable.* At a high level, our approach is to apply the second part of the analysis of KRR no-signaling PCP (Section 4.1) on each party's next-message computation in a "round-by-round" manner. More concretely, our approach is to first show that p-Assign is correct on each of the variables that correspond to the internal states and

incoming/outgoing messages of Round 1 of Π in every $\varphi^{\mu:\nu}$, next show it on each of the variables that correspond to those of Round 2 of Π in every $\varphi^{\mu:\nu}$, and so on.

Toward this end, we first remark that we design each 3SAT instance $\varphi^{\mu:\nu}$ carefully so that it has the following specific structure.

1. Let N_{round} be the round complexity of Π. Then, $\varphi^{\mu:\nu}$ has variables that can be partitioned into $4N_{\mathrm{round}}$ sequences of variables, $\boldsymbol{w}^{\xi}_{1,\mathsf{in}}, \boldsymbol{w}^{\xi}_{1,\mathsf{out}}, \ldots, \boldsymbol{w}^{\xi}_{N_{\mathrm{round}},\mathsf{in}}, \boldsymbol{w}^{\xi}_{N_{\mathrm{round}},\mathsf{out}}$ for $\xi \in \{\mu, \nu\}$, such that for each $\ell \in [N_{\mathrm{round}}]$:
   - $\boldsymbol{w}^{\xi}_{\ell,\mathsf{in}}$ is a sequence of variables such that the values that are assigned to them are interpreted as an internal state and incoming messages of P^{ξ} at the beginning of Round ℓ.[13]
   - $\boldsymbol{w}^{\xi}_{\ell,\mathsf{out}}$ is a sequence of variables such that the values that are assigned to them are interpreted as an internal state and outgoing messages of P^{ξ} at the end of Round ℓ.
2. $\varphi^{\mu:\nu}$ has clauses that check the following.
   - In each round, for each of P^{μ} and P^{ν}, its end state (i.e., its internal state at the end of the round) and outgoing messages are correctly derived from its start state (i.e., its internal state at the beginning of the round) and incoming messages.
   - In each round, for each of P^{μ} and P^{ν}, its start state is equal to its end state of the previous round.
   - In each round, P^{μ}'s incoming message from P^{ν} at the beginning of the round is equal to P^{ν}'s outgoing message to P^{μ} at the end of the previous round, and vise versa.
   - Both P^{μ} and P^{ν} output 1 in the last round.

We note that given consistent views of P^{μ}, P^{ν} in which they output 1, we can compute a satisfying assignment to the variables in $\varphi^{\mu:\nu}$ efficiently by obtaining each party's end state and outgoing messages of each round through the next-message function.

Now, we first show that if in every $\varphi^{\mu:\nu}$, $\mathsf{p\text{-}Assign}$ is correct on P^{μ} and P^{ν}'s start states and incoming messages in Round 1, then in every $\varphi^{\mu:\nu}$, $\mathsf{p\text{-}Assign}$ is also correct on P^{μ} and P^{ν}'s end states and outgoing messages in Round 1. A key observation on this step is that, essentially, what we need to show is that in every $\varphi^{\mu:\nu}$, for each $\xi \in \{\mu, \nu\}$, if $\mathsf{p\text{-}Assign}$ is correct on the input of P^{ξ}'s next-message computation of Round 1, then $\mathsf{p\text{-}Assign}$ is also correct on the output of it. Given this observation (and by designing the details of $\varphi^{\mu:\nu}$ appropriately), we can complete this step by just reusing the second part of the analysis of KRR no-signaling PCP, where it is shown that if Assign is correct on the input, then Assign is also correct on the output.

We next show that in every $\varphi^{\mu:\nu}$, if $\mathsf{p\text{-}Assign}$ is correct on P^{μ} and P^{ν}'s end states and outgoing messages in Round 1, then in every $\varphi^{\mu:\nu}$, $\mathsf{p\text{-}Assign}$ is also

[13] We think that each round of Π starts when each party receives incoming messages from the other parties, and ends when each party sends outgoing messages to the other parties.

correct on P^μ and P^ν's start states and incoming messages in Round 2. In this step, we consider three cases for each $\varphi^{\mu:\nu}$.

Case 1. We first consider the correctness on P^ξ's start state of Round 2 ($\xi \in \{\mu, \nu\}$). This case is easy and we just need to use the everywhere local consistency of p-Assign and the definition of $\varphi^{\mu:\nu}$. Specifically, since $\varphi^{\mu:\nu}$ has clauses that check that P^ξ's start state of Round 2 is equal to its end state of Round 1, the everywhere local consistency of p-Assign guarantees that p-Assign assigns the same value on P^ξ's start state of Round 2 and on P^ξ's end state of Round 1, and thus, if p-Assign is correct on the latter, it is also correct on the former.

Case 2. We next consider the correctness on P^μ's incoming message from P^ν and P^ν's incoming message from P^μ at the beginning of Round 2. Again, this case is easy and we just need to use the everywhere local consistency of p-Assign and the definition of $\varphi^{\mu:\nu}$ (which has clauses that check that the message that P^μ receives from P^ν at the beginning of Round 2 is equal to the one that P^ν sends to P^μ at the end of Round 1, and vise versa).

Case 3. We finally consider the correctness on P^μ and P^ν's incoming messages from the parties other than P^μ and P^ν at the beginning of Round 2. This case is not straightforward, and we rely on that p-Assign satisfies the consistency on the views, which is guaranteed since p-Assign is obtained from a well-behaving prover. Let us consider, for example, P^μ's incoming message from P^ξ ($\xi \notin \{\mu, \nu\}$). Then, since the consistency on the views guarantees that p-Assign assigns the same value in $\varphi^{\mu:\nu}$ and $\varphi^{\mu:\xi}$ as P^μ's incoming message from P^ξ, if p-Assign is correct on it in $\varphi^{\mu:\xi}$, then p-Assign is also correct on it in $\varphi^{\mu:\nu}$. Then, since we showed in Case 2 that p-Assign is indeed correct on it in $\varphi^{\mu:\xi}$, we conclude that p-Assign is correct on it in $\varphi^{\mu:\nu}$.[14]

By proceeding identically (and observing that, by definition, p-Assign is correct on P^μ and P^ν's start states and incoming messages in Round 1 in every $\varphi^{\mu:\nu}$), we conclude that p-Assign is correct on any variable, and in particular correct on P^μ and P^ν's final states in every $\varphi^{\mu:\nu}$.

Step 3: Obtaining soundness. On the one hand, the value that p-Assign assigns as the output of any party P^μ is always 1 due to the everywhere local consistency of p-Assign (recall that $\varphi^{\mu:\nu}$ has a clause that checks that P^μ's output is 1). On the other hand, since p-Assign is correct on the output of P^μ, it is also equal to the value that P^μ outputs in the correct views. Thus, P^μ outputs 1 in the correct view, which means that the statement proven by the prover is true on the XOR of the MPC inputs of the correct views. From the definition of the extractor, it follows that the prover cannot prove false statements on the extracted value.

[14] Note that we cannot use this argument if we try to reuse the analysis of Kalai et al. [KRR14] for each $\varphi^{\mu:\nu}$ individually (rather than in the round-by-round manner) since we show the correctness in $\varphi^{\mu:\nu}$ by using the correctness in $\varphi^{\mu:\xi}$.

More details of well-behaving provers. It remains to give an overview of the concrete definition of well-behaving provers. As we mentioned earlier, we define well-behaving provers so that when we convert a CNS well-behaving prover into a parallel local assignment generator $\mathsf{p\text{-}Assign}$, then $\mathsf{p\text{-}Assign}$ has the aforementioned two consistency properties.

Before giving the definition of well-behaving provers, we give a few details about the construction of KRR no-signaling PCP.

- When a PCP proof π for a 3SAT instance φ is created by using a satisfying assignment $\boldsymbol{x}$ to φ, the PCP proof π contains an encoding of $\boldsymbol{x}$,[15] i.e., there is a set of queries $D(X)$ such that $\pi|_{D(X)}$ is an encoding of $\boldsymbol{x}$.
- Furthermore, we can make sure that in our protocol, each PCP proof $\pi^{\mu:\nu}$ for $\varphi^{\mu:\nu}$ (where $\pi^{\mu:\nu}$ is created by using $(\mathsf{view}^\mu, \mathsf{view}^\nu)$) contains encodings of $x^\mu_{1,\mathsf{in}}$, $x^\nu_{1,\mathsf{in}}$, view^μ, and view^ν, i.e., there are sets of queries $D(X^\mu_{1,\mathsf{in}}), D(X^\nu_{1,\mathsf{in}})$, $D(X^\mu), D(X^\nu)$ such that:
  - $\pi^{\mu:\nu}|_{D(X^\mu_{1,\mathsf{in}})}$ and $\pi^{\mu:\nu}|_{D(X^\nu_{1,\mathsf{in}})}$ are encodings of $x^\mu_{1,\mathsf{in}}$ and $x^\nu_{1,\mathsf{in}}$, respectively.
  - $\pi^{\mu:\nu}|_{D(X^\mu)}$ and $\pi^{\mu:\nu}|_{D(X^\nu)}$ are encodings of view^μ and view^ν, respectively.

  (Recall that $x^\mu_{1,\mathsf{in}} := \mathsf{st}^\mu_0 \parallel \mathsf{i\text{-}msgs}^\mu_1$ and $x^\nu_{1,\mathsf{in}} := \mathsf{st}^\nu_0 \parallel \mathsf{i\text{-}msgs}^\nu_1$ are the initial states and dummy incoming messages that are computed in the commit phase.)

Then, informally speaking, a CNS prover is said to be *well-behaving* if it satisfies the following two consistency properties.

Consistency on $D(X^\mu_{1,\mathsf{in}})$. Once the commit phase is completed, the prover gives the same response to a query in $D(X^\mu_{1,\mathsf{in}})$ ($\mu \in [M]$) in different invocations. More concretely, for any queries $\{Q_0^{\mu:\nu}\}_{\mu,\nu\in[M]}, \{Q_1^{\mu:\nu}\}_{\mu,\nu\in[M]}$, any $\alpha,\beta,\gamma,\delta \in [M]$ such that $\exists\xi \in \{\alpha,\beta\} \cap \{\gamma,\delta\}$, and any $q \in Q_0^{\alpha:\beta} \cap Q_1^{\gamma:\delta} \cap D(X^\xi_{1,\mathsf{in}})$, we have ${\pi^*}_0^{\alpha:\beta}(q) = {\pi^*}_1^{\gamma:\delta}(q)$, where ${\pi^*}_0^{\alpha:\beta}$ and ${\pi^*}_1^{\gamma:\delta}$ are generated as follows.

1. $(\mathsf{st}_C, \mathsf{com}) \leftarrow \langle \mathsf{C.Com}^*_1, \mathsf{R.Com}_1 \rangle$
2. $(f_0, \{{\pi^*}_0^{\mu:\nu}\}_{\mu,\nu\in[M]}) \leftarrow \mathsf{C.Prv}^*_1(\mathsf{st}_C, \{Q_0^{\mu:\nu}\}_{\mu,\nu\in[M]})$
3. $(f_1, \{{\pi^*}_1^{\mu:\nu}\}_{\mu,\nu\in[M]}) \leftarrow \mathsf{C.Prv}^*_1(\mathsf{st}_C, \{Q_1^{\mu:\nu}\}_{\mu,\nu\in[M]})$

Consistency on $D(X^\mu)$. The prover gives the same responses to a query in $D(X^\mu)$ ($\mu \in [M]$) in a single invocation. More concretely, for any queries $\{Q^{\mu:\nu}\}_{\mu,\nu\in[M]}$, any $\alpha,\beta,\gamma,\delta \in [M]$ such that $\exists\xi \in \{\alpha,\beta\} \cap \{\gamma,\delta\}$, and any $q \in Q^{\alpha:\beta} \cap Q^{\gamma:\delta} \cap D(X^\xi)$, we have ${\pi^*}^{\alpha:\beta}(q) = {\pi^*}^{\gamma:\delta}(q)$, where π^* is generated as follows.

1. $(\mathsf{st}_C, \mathsf{com}) \leftarrow \langle \mathsf{C.Com}^*_1, \mathsf{R.Com}_1 \rangle$
2. $(f, \{{\pi^*}^{\mu:\nu}\}_{\mu,\nu\in[M]}) \leftarrow \mathsf{C.Prv}^*_1(\mathsf{st}_C, \{Q^{\mu:\nu}\}_{\mu,\nu\in[M]})$

To show that the above definition indeed implies the aforementioned two consistency properties of $\mathsf{p\text{-}Assign}$, we need to see the details of $\mathsf{p\text{-}Assign}$. Specifically, we rely on that $\mathsf{p\text{-}Assign}$ obtains local assignments by applying a procedure called *self-correction* on the cheating prover. In this overview, we do not give the

[15] Concretely, a *low-degree extension* of $\boldsymbol{x}$.

details of self-correction, and we just note that p-Assign obtains local assignments in the following manner: p-Assign first creates some queries $Q^{\mu:\nu}$ for each $\mu, \nu \in [M]$ based on its input, next queries $\{Q^{\mu:\nu}\}_{\mu,\nu\in[M]}$ to the prover, and finally obtains the local assignments based on the prover's responses.

Now, at first sight, it seems trivial to show that the above definition of well-behaving provers implies the two consistency properties of p-Assign. Consider, for example, showing that the above definition implies that p-Assign has the consistency on the initial states. Then, since p-Assign obtains local assignments based on the prover's responses, and well-behaving provers are guaranteed to give unique responses to any queries on the initial states (i.e., any queries in $D(X^{\mu}_{1,\mathsf{in}})$ ($\mu \in [M]$)), it seems trivial to show that p-Assign makes unique assignments on the initial states.

However, this intuition is wrong. For example, in the case of showing the consistency on the initial states, the problem is that even when making assignments on the initial states, p-Assign's queries to the prover includes those that are not in $D(X^{\mu}_{1,\mathsf{in}})$ ($\mu \in [M]$), and well-behaving provers' responses to such queries are not necessarily unique.

Fortunately, this problem can be solved relatively easily by using a technique in a previous work [HR18]. Specifically, by letting the verifier of KRR no-signaling PCP do several additional tests on the prover, we can show that it suffices to consider a modified version of p-Assign, which obtains local assignments on the initial states (resp., the views) based solely on the prover's responses to the queries in $D(X^{\mu}_{1,\mathsf{in}})$ (resp., in $D(X^{\mu})$).[16] On this modified version of p-Assign, it is indeed easy to show that the two consistency properties of well-behaving provers imply the two consistency properties of p-Assign by relying on analyses given in [KRR14].

Towards formal proof. Finally, we discuss what modifications are needed to turn the above proof idea into a formal proof.

First, we need to modify the extractor so that it can open the commitment (instead of just extracting a committed value) as required in Definition 3; along the way, we also need to define the open phase of the protocol appropriately. Recall that in the above, the extractor uses the parallel local assignment generator p-Assign to extract a committed value. Motivated by this construction of the extractor, we follow the following overall approach: we define the open phase so that running p-Assign jointly with the receiver is sufficient for the committer to succeed in the open phase. To implement this approach, we rely on that, as mentioned above, p-Assign obtains local assignments in the following manner: p-Assign first creates some queries $Q^{\mu:\nu}$ for each $\mu, \nu \in [M]$ based on its input, next queries $\{Q^{\mu:\nu}\}_{\mu,\nu\in[M]}$ to the prover, and finally obtains the local

[16] Concretely, we use *layer-parallel low-degree tests* [HR18] to guarantee that the initial states (resp., the views) that are recovered through self-correction in p-Assign do not change when the queries are sampled from $D(X^{\mu}_{1,\mathsf{in}})$ (resp., from $D(X^{\mu})$) rather than from $D(X)$.

assignments based on the prover's responses. Given this structure of p-Assign, we define the open phase as follows.

1. In the first round, the receiver computes queries as in p-Assign and sends them to the committer.
2. In the second round, the committer gives responses to the queries.
3. Finally, the receiver computes the local assignments from the responses as in p-Assign and then uses them to extract a committed value as in the extractor.

Then, we modify the extractor so that it simply forwards the queries from the receiver to the cheating prover and next forwards the responses from the cheating prover to the receiver. Since the extracted value is computed from the output of p-Assign just as before (the only difference is that now p-Assign is executed jointly between the extractor and the receiver), we can still prove that any CNS well-behaving cheating prover cannot prove false statements on the extracted value. Furthermore, we can show that the above open phase is strong enough to guarantee a meaningful binding property. Specifically, by letting the receiver make additional queries in the open phase,[17] we can prove the binding property against CNS *well-behaving decommitters*, which are defined similarly to CNS well-behaving provers. (The proof of the binding property proceeds essentially in the same way as we show that p-Assign satisfies the consistency on the initial states in the proof of the soundness against well-behaving provers, where we show that once the commit phase is completed, the assignments by p-Assign on the MPC initial states—which define the committed value—are unique.)

Second, we need to consider the case that p-Assign can err (i.e., the everywhere local consistency does not necessarily hold with probability 1). Fortunately, this case is already handled in Kalai et al. [KRR14], and we can handle it identically. (Concretely, when showing that p-Assign is correct on every variable in the round-by-round way, we only show that p-Assign is correct *on average*, i.e., instead of showing that p-Assign is correct on any variables that correspond to, say, the start state and incoming message of a round, we only show that p-Assign is correct on randomly chosen $\omega(\log \lambda)$ such variables. It is shown in [KRR14] that showing such average-case correctness is sufficient to prove the soundness.)

Third, we need to consider adaptive soundness, where the cheating prover chooses the statement to prove at the last round of the prove phase. Fortunately, adaptive soundness is already considered in previous works (e.g., [BHK17]), and we can handle it identically.

[17] Specifically, the receiver make queries for a low-degree test (just like the verifier of KRR succinct argument does) so that we can reuse analyses of Kalai et al. [KRR14] as in the proof of soundness.

5 Overview of Step 2 (Non-WI Scheme with Soundness against CNS Provers)

We give an overview of our non-WI commit-and-prove protocol $\langle C_2, R_2 \rangle$, which is $(1 - \mathsf{negl})$-sound against CNS provers.

Our high-level approach is to upgrade the protocol $\langle C_1, R_1 \rangle$ that we give in Step 1 so that the soundness holds against any (not necessarily well-behaving) CNS provers. Recall that, roughly speaking, an adversary is well-behaving if for every $\mu \in [M]$,

1. it does not give different responses to a query in $D(X^{\mu}_{1,\mathsf{in}})$ in different invocations, and
2. it does not give different responses to a query in $D(X^{\mu})$ in a single invocation,

where $D(X^{\mu}_{1,\mathsf{in}})$ and $D(X^{\mu})$ are sets of queries such that in $\langle C_1, R_1 \rangle$, the prover is supposed to create PCPs $\{\pi^{\mu:\nu}\}_{\mu,\nu\in[M]}$ such that $\pi^{\mu:\nu}|_{D(X^{\mu}_{1,\mathsf{in}})}$ is an encoding of $x^{\mu}_{1,\mathsf{in}}$ and $\pi^{\mu:\nu}|_{D(X^{\mu})}$ is an encoding of view^{μ} for every $\nu \in [M]$, where $x^{\mu}_{1,\mathsf{in}}$ is the value that is fixed in the commit phase and view^{μ} is the view that is fixed in the prove phase. Naturally, we enforce this behavior on the prover by relying on collision-resistant hash functions: we require the prover to publish the roots of the tree-hash of the encodings of $\{x^{\mu}_{1,\mathsf{in}}\}_{\mu\in[M]}$ and $\{\mathsf{view}^{\mu}\}_{\mu\in[M]}$, and also require it to give responses along with appropriate certificates when it is queried on these values.

More concretely, we consider the following protocol (which is slightly over-simplified from the actual protocol). In the following, for a hash function hf, we denote by $\mathsf{TreeHash}_{\mathsf{hf}}$ an algorithm that computes the Merkle tree-hash of the input.

Commit Phase

Round 1: $\mathsf{R.Com}_2$ sends a hash function $\mathsf{hf} \in \mathcal{H}$ to $\mathsf{C.Com}_2$.

Round 2: Given $(x_{\mathrm{COM}}, \mathsf{hf})$ as input, $\mathsf{C.Com}_2$ obtains $\{x^{\mu}_{1,\mathsf{in}}\}_{\mu\in[M]}$ by running $\mathsf{C.Com}_1(x_{\mathrm{COM}})$, computes encodings $\{X^{\mu}_{1,\mathsf{in}}\}_{\mu\in[M]}$ of them, and then outputs $\{\mathsf{rt}^{\mu}_{1,\mathsf{in}} := \mathsf{TreeHash}_{\mathsf{hf}}(X^{\mu}_{1,\mathsf{in}})\}_{\mu\in[M]}$ as the commitment and store $(\mathsf{hf}, \{X^{\mu}_{1,\mathsf{in}}\}_{\mu\in[M]})$ as the internal state.

Prove Phase

Round 1: $\mathsf{R.Prv.Q}_2$ works identically with $\mathsf{R.Prv.Q}_1$. That is, $\mathsf{R.Prv.Q}_2$ obtains $\{Q^{\mu,\nu}\}_{\mu,\nu\in[M]}$ just like $\mathsf{R.Prv.Q}_1$ does, and outputs $\{Q^{\mu,\nu}\}_{\mu,\nu\in[M]}$ as the query.

Round 2: Given the statement f and the query $\{Q^{\mu,\nu}\}_{\mu,\nu\in[M]}$ as input, $\mathsf{C.Prv}_2$ does the following.

1. Obtain $\{\mathsf{view}^{\mu}\}_{\mu\in[M]}$ and $\{\pi^{\mu:\nu}\}_{\mu,\nu\in[M]}$ just like $\mathsf{C.Prv}_1$ does.
2. Compute encodings $\{X^{\mu}\}_{\mu\in[M]}$ of $\{\mathsf{view}^{\mu}\}_{\mu\in[M]}$, and compute $\{\mathsf{rt}^{\mu} := \mathsf{TreeHash}_{\mathsf{hf}}(X^{\mu})\}_{\mu\in[M]}$.
3. Augment each $\pi^{\mu:\nu}$ as follows.
   - Augment each symbol in $\pi^{\mu:\nu}|_{D(X^{\xi}_{1,\mathsf{in}})}$ $(\xi \in \{\mu, \nu\})$ with a certificate for opening $\mathsf{rt}^{\xi}_{1,\mathsf{in}}$ to it.

– Augment each symbol in $\pi^{\mu:\nu}|_{D(X^\xi)\setminus D(X^\xi_{1,\mathsf{in}})}$ ($\xi \in \{\mu,\nu\}$) with a certificate for opening rt^ξ to it.

4. Output $(\{\mathsf{rt}^\mu\}_{\mu\in[M]}, \{\pi^{\mu:\nu}|_{Q^{\mu:\nu}}\}_{\mu,\nu\in[M]})$ as the proof.

Verification: Given the commitment $\{\mathsf{rt}^\mu_{1,\mathsf{in}}\}_{\mu\in[M]}$, the statement f, and the proof $(\{\mathsf{rt}^\mu\}_{\mu\in[M]}, \{\pi^{*\mu:\nu}\}_{\mu,\nu\in[M]})$ as input, $\mathsf{R.Prv.D}_2$ works identically with $\mathsf{R.Prv.D}_1$ except that before the verification, each $\pi^{*\mu:\nu}$ is "filtered" as follows.

– Replace each symbol (x, cert) in $\pi^{*\mu:\nu}|_{D(X^\xi_{1,\mathsf{in}})}$ ($\xi \in \{\mu,\nu\}$) with x if cert is a valid certificate for opening $\mathsf{rt}^\xi_{1,\mathsf{in}}$ to x, and replace it with $\bot$ otherwise.

– Replace each symbol (x, cert) in $\pi^{*\mu:\nu}|_{D(X^\xi)\setminus D(X^\xi_{1,\mathsf{in}})}$ ($\xi \in \{\mu,\nu\}$) with x if cert is a valid certificate for opening rt^ξ to x, and replace it with $\bot$ otherwise.

We prove the soundness of $\langle C_2, R_2\rangle$ by relying on the soundness of $\langle C_1, R_1\rangle$. Specifically, for any cheating committer-prover $C^*_2 = (\mathsf{C.Com}^*_2, \mathsf{C.Prv}^*_2)$ against $\langle C_2, R_2\rangle$, we consider the following cheating committer-prover $C^*_1 = (\mathsf{C.Com}^*_1, \mathsf{C.Prv}^*_1)$ against $\langle C_1, R_1\rangle$.

– **Committer.** $\mathsf{C.Com}^*_1$ runs $(\mathsf{st}_C, \mathsf{com}) \leftarrow \langle \mathsf{C.Com}^*_2, \mathsf{R.Com}_2\rangle$ internally, sends an empty string to $\mathsf{R.Com}_1$ as the commitment, and stores $(\mathsf{com}, \mathsf{st}_C)$ as the internal state.

– **Prover.** Given $(\mathsf{com}, \mathsf{st}_C)$ and $\{Q^{\mu:\nu}\}_{\mu,\nu\in[M]}$ as input, $\mathsf{C.Prv}^*_1$ first runs $(f, \{\mathsf{rt}^\mu\}_{\mu\in[M]}, \{\pi^{*\mu:\nu}\}_{\mu,\nu\in[M]}) \leftarrow \mathsf{C.Prv}^*_2(\mathsf{st}_C, \{Q^{\mu:\nu}\}_{\mu,\nu\in[M]})$. Then, $\mathsf{C.Prv}^*_1$ filters each $\pi^{*\mu:\nu}$ as in the verification of $\langle C_2, R_2\rangle$, and sends $(f, \{\pi^{*\mu:\nu}\}_{\mu,\nu\in[M]})$ to $\mathsf{R.Prv}_1$ as the proof.

It is straightforward to show that (1) C^*_1 is successful if C^*_2 is successful and (2) C^*_1 is well-behaving CNS. (The latter follows from the the binding property of $\mathsf{TreeHash}_{\mathsf{hf}}$.)

6 Overview of Subsequent Steps of Proof of Theorem 1

In Step 3, we upgrade the soundness to the one with negligible soundness error. Fortunately, this type of soundness amplification is already studied by Kalai et al. [KRR14] as mentioned in Section 3, and it suffices to apply their soundness amplification on the protocol $\langle C_2, R_2\rangle$ that we obtained in Step 2. Concretely, in this step, we just borrow a soundness amplification technique from [KRR14, BHK17], which amplifies soundness by letting the verifier use a smaller threshold parameter for the PCP decision algorithm (i.e., letting the verifier tolerate a smaller number of failures on the tests that it applies on the prover).

In Step 4, we upgrade the soundness to the one against any (not necessarily CNS) adversaries. Again, this type of soundness amplification is already studied by Kalai et al. [KRR14] as mentioned in Section 3, and it suffices to apply their soundness amplification on the protocol $\langle C_3, R_3\rangle$ that we obtained in Step 3.

Concretely, in this step, we just borrow a transformation from [KRR14], which enforces CNS behavior on the committer by encrypting the verifier queries by PIR. (Intuitively, encrypting the verifier queries by PIR is helpful to enforce CNS behavior since it forces the prover to answer each query independently of the other queries.)

In Step 5, we add the WI property while tolerating that the soundness error increases to a constant. Toward this end, we augment the protocol $\langle C_4, R_4 \rangle$ that we obtained in Step 4 with commitment schemes and OT by using these two primitives as in the non-succinct protocol that we sketched in Section 1.2. The soundness and WI of the resultant protocol $\langle C_5, R_5 \rangle$ can be shown similarly to those of the non-succinct protocol in Section 1.2. That is, the soundness follows from the security of OT and the soundness of $\langle C_4, R_4 \rangle$,[18] and the WI property follows from the 2-*privacy* of $\langle C_4, R_4 \rangle$, which roughly guarantees that the verifier does not learn any secret information if it only obtains one of the M^2 KRR no-signaling PCP strings. (The 2-privacy of $\langle C_4, R_4 \rangle$, in turn, follows immediately from the 2-privacy of the underlying MPC protocol Π.)

References

AL17. Gilad Asharov and Yehuda Lindell. A full proof of the BGW protocol for perfectly secure multiparty computation. *Journal of Cryptology*, 30(1):58–151, January 2017.

BBK+16. Nir Bitansky, Zvika Brakerski, Yael Tauman Kalai, Omer Paneth, and Vinod Vaikuntanathan. 3-message zero knowledge against human ignorance. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part I*, volume 9985 of *LNCS*, pages 57–83. Springer, Heidelberg, October / November 2016.

BGW88. Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *20th ACM STOC*, pages 1–10. ACM Press, May 1988.

BHK17. Zvika Brakerski, Justin Holmgren, and Yael Tauman Kalai. Non-interactive delegation and batch NP verification from standard computational assumptions. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *49th ACM STOC*, pages 474–482. ACM Press, June 2017.

BK20. Zvika Brakerski and Yael Kalai. Witness indistinguishability for any single-round argument with applications to access control. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part II*, volume 12111 of *LNCS*, pages 97–123. Springer, Heidelberg, May 2020.

BKK+18. Saikrishna Badrinarayanan, Yael Tauman Kalai, Dakshita Khurana, Amit Sahai, and Daniel Wichs. Succinct delegation for low-space non-deterministic computation. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *50th ACM STOC*, pages 709–721. ACM Press, June 2018.

CLOS02. Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *34th ACM STOC*, pages 494–503. ACM Press, May 2002.

[18] Formally, as in the case of the non-succinct protocol in Section 1.2, complexity leveraging is required.

GGP10. Rosario Gennaro, Craig Gentry, and Bryan Parno. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 465–482. Springer, Heidelberg, August 2010.

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

GKR15. Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: Interactive proofs for muggles. *Journal of the ACM*, 62(4):27:1–27:64, 2015.

GLOV12. Vipul Goyal, Chen-Kuei Lee, Rafail Ostrovsky, and Ivan Visconti. Constructing non-malleable commitments: A black-box approach. In *53rd FOCS*, pages 51–60. IEEE Computer Society Press, October 2012.

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

GMW87. Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987.

GOSV14. Vipul Goyal, Rafail Ostrovsky, Alessandra Scafuro, and Ivan Visconti. Black-box non-black-box zero knowledge. In David B. Shmoys, editor, *46th ACM STOC*, pages 515–524. ACM Press, May / June 2014.

HR18. Justin Holmgren and Ron Rothblum. Delegating computations with (almost) minimal time and space overhead. In Mikkel Thorup, editor, *59th FOCS*, pages 124–135. IEEE Computer Society Press, October 2018.

HV16. Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. On the power of secure two-party computation. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 397–429. Springer, Heidelberg, August 2016.

HV18. Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. Round-optimal fully black-box zero-knowledge arguments from one-way permutations. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 263–285. Springer, Heidelberg, November 2018.

IKOS09. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge proofs from secure multiparty computation. *SIAM Journal on Computing*, 39(3):1121–1152, 2009.

IW14. Yuval Ishai and Mor Weiss. Probabilistically checkable proofs of proximity with zero-knowledge. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 121–145. Springer, Heidelberg, February 2014.

Kil92. Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *24th ACM STOC*, pages 723–732. ACM Press, May 1992.

KOS18. Dakshita Khurana, Rafail Ostrovsky, and Akshayaram Srinivasan. Round optimal black-box "commit-and-prove". In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 286–313. Springer, Heidelberg, November 2018.

KP16. Yael Tauman Kalai and Omer Paneth. Delegating RAM computations. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 91–118. Springer, Heidelberg, October / November 2016.

KRR14. Yael Tauman Kalai, Ran Raz, and Ron D. Rothblum. How to delegate computations: the power of no-signaling proofs. In David B. Shmoys, editor, *46th ACM STOC*, pages 485–494. ACM Press, May / June 2014.

LP12. Huijia Lin and Rafael Pass. Black-box constructions of composable protocols without set-up. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 461–478. Springer, Heidelberg, August 2012.

MPP20. Andrew Morgan, Rafael Pass, and Antigoni Polychroniadou. Succinct non-interactive secure computation. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 216–245. Springer, Heidelberg, May 2020.

PR14. Omer Paneth and Guy N. Rothblum. Publicly verifiable non-interactive arguments for delegating computation. Cryptology ePrint Archive, Report 2014/981, 2014. http://eprint.iacr.org/2014/981.

PW09. Rafael Pass and Hoeteck Wee. Black-box constructions of two-party protocols from one-way functions. In Omer Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 403–418. Springer, Heidelberg, March 2009.

A On Verification Time

The verification of our protocol is not succinct since we use a simpler version of KRR succinct argument where the verifier naively evaluates a low-degree extension (LDE) of the indicator function of a 3CNF formula whose size is polynomially related to the complexity of the statement. In Kalai et al. [KRR14] and subsequent works [BHK17, HR18], the verification is made succinct by observing that when the statement to be proven satisfies some conditions, the evaluation of the LDE can be either recursively delegated to the prover succinctly or locally performed by the verifier efficiently. In our protocol, KRR succinct argument is used for proving statements that are related to the next-message function of the underlying perfect 2-privacy MPC protocol (cf. Section 1.2 and Section 4). Thus, if we can show that the above-mentioned conditions are satisfied for a specific perfect 2-privacy MPC protocol, the verification of our protocol can be made succinct.

B On Definition of Commit-and-prove Protocols

B.1 Differences from Definition in Khurana et al. [KOS18]

Our definition of commit-and-prove protocols in Section 2.2 has several differences from the definition in Khurana et al. [KOS18]. First, our definition has several syntactical differences.

- Instead of thinking the prove phase as a part of the commit phase, we separate the prove phase from the commit phase.
- We focus on the case that each of the prove phase and the open phase consists of two rounds.

Next, our definition is stronger than the definition of Khurana et al. [KOS18] in the following points.

- We explicitly define the soundness and witness indistinguishability in the delayed-input setting, where the statement to be proven is chosen at the last round of the prove phase.
- In the definition of the soundness, we require the extractor to decommit the commitment to a value on which any committer cannot prove false statements. (In the definition in [KOS18], the extractor just outputs such a value without decommitment, with the guarantee that any committer cannot decommit the commitment to a value other than the extracted one.)
  We think that requiring the extractor to decommit the commitment is important, as otherwise the definition would not prevent an attack where the committer gives an accepting proof on an invalid commitment (i.e., a commitment that cannot be opened to any value).[19] (This is because even if such an attack is possible, we can still show that any committer cannot decommit the commitment to a value other than the extracted one, since an invalid commitment cannot be opened to any value.) We remark that such an attack is possible if a commit-and-prove protocol is naively executed in parallel multiple times.

Finally, our definition is weaker than the definition of Khurana et al. [KOS18] in the following points.

- In the definitions of the binding and the soundness, the extractor succeeds only on an overwhelming fraction of the executions of the commit phase, rather than on any execution of the commit phase.
- In the definition of the soundness, the extractor is allowed to depend on the success probability of the cheating committer.

B.2 Rationale behind Our Definition

Since our definition of commit-and-prove protocols in Section 2.2 might look too cumbersome, we explain the rationale behind it.

First, binding and witness-indistinguishability are defined naturally, and the only complication is that we allow the open phase to be interactive in the definition of binding. To guarantee a stronger notion of binding, our definition considers an adversary that obtains two sets of receiver decommitment queries simultaneously (rather than obtain each of them separately).

Next, soundness is defined similarly to proof-of-knowledge of interactive proofs [GMR89]. A complication is that we define it so that it guarantees the adaptive delayed-input property, i.e., it holds against an adversary that chooses

[19] We remark that the schemes by Khurana et al. [KOS18] are designed to prevent such an attack. What we claim is that the definition by Khurana et al. [KOS18] does not prevent such an attack.

the statement to prove at the last round of the prove phase.[20] To guarantee proof-of-knowledge with the adaptive delayed-input property, our definition requires that if a cheating prover convinces the verifier for a commitment com with sufficiently high probability, then there exists a value x^* such that (1) the extractor can decommit com to x^* and (2) the cheating prover cannot prove false statements about x^*. (We remark that the extractor is required to succeed in the decommitment of each bit of x^* with overwhelming probability so that we can obtain the whole x^* by repeatedly using the extractor.)

[20] The adaptive delayed-input property is required to, e.g., upgrade our WI commit-and-prove protocol to a ZK one by using the transformation of Khurana et al. [KOS18].