J Cryptol (2019) 32:178–238
https://doi.org/10.1007/s00145-018-9276-1

Journal of
CRYPTOLOGY

Round-Efficient Black-Box Construction of Composable Multi-Party Computation*

Susumu Kiyoshima
NTT Secure Platform Laboratories, Tokyo, Japan
kiyoshima.susumu@lab.ntt.co.jp

Communicated by Rafail Ostrovsky.

Received 4 October 2015 / Revised 18 November 2017
Online publication 5 February 2018

Abstract. We present a round-efficient black-box construction of a general multi-party computation (MPC) protocol that satisfies composability in the plain model. The security of our protocol is proven in the angel-based UC framework [Prabhakaran and Sahai, STOC'04] under the minimal assumption of the existence of semi-honest oblivious transfer protocols. The round complexity of our protocol is $\max(\widetilde{O}(\log^2 n), O(R_{\mathsf{OT}}))$ when the round complexity of the underlying oblivious transfer protocol is R_{OT}. Since constant-round semi-honest oblivious transfer protocols can be constructed under standard assumptions (such as the existence of enhanced trapdoor permutations), our result gives a $\widetilde{O}(\log^2 n)$-round protocol under these assumptions. Previously, only an $O(\max(n^{\epsilon}, R_{\mathsf{OT}}))$-round protocol was shown, where $\epsilon > 0$ is an arbitrary constant. We obtain our MPC protocol by constructing a $\widetilde{O}(\log^2 n)$-round CCA-secure commitment scheme in a black-box way under the assumption of the existence of one-way functions.

Keywords. Multi-party computation, Composability, Angel-based UC security, Black-box construction, CCA-secure commitment.

1. Introduction

Secure multi-party computation (MPC) protocols enable mutually distrustful parties to compute a functionality without compromising the correctness of the outputs and the privacy of their inputs. In the seminal work of Goldreich et al. [16], it was shown that *general MPC protocols*—MPC protocols that can be used to securely compute any functionality—can be constructed even in the model with malicious adversaries and a dishonest majority.[1]

[1] In the following, we consider only such a model.

*This article is based on an earlier article: Round-Efficient Black-Box Construction of Composable Multi-party Computation, in Proceedings of CRYPTO 2014, ©IACR 2014, https://doi.org/10.1007/978-3-662-44381-1_20.

In this paper, we consider a *black-box construction* of a general MPC protocol that guarantees *composable security*. Before stating our result, we explain black-box constructions and composable security.

Black-Box Constructions

A construction of a cryptographic protocol is *black-box* if it uses the underlying cryptographic primitives only in a black-box way (i.e., only through their input/output interfaces). If a construction uses the codes of the underlying primitives, it is *non-black-box*.

As argued by Ishai et al. [21], constructing black-box constructions is important for both theoretical and practical reasons. Theoretically, it is important because understanding whether non-black-box use of cryptographic primitives is necessary for a cryptographic task is of great interest. Practically, it is important because black-box constructions are typically more efficient than non-black-box ones in terms of both communication and computational complexity. (In fact, most non-black-box constructions of general MPC protocols are highly inefficient and hard to implement because they use general NP reductions when executing zero-knowledge proofs.)

Recently, a number of works have studied black-box constructions of general MPC protocols. Ishai et al. [21] showed the first construction of a general MPC protocol that uses the underlying low-level primitives (such as enhanced trapdoor permutations or homomorphic public-key encryption schemes) in a black-box way. Combined with the subsequent work by Haitner [18], who showed a black-box construction of a (maliciously secure) oblivious transfer protocol based on a semi-honest oblivious transfer protocol, their work gave a black-box construction of a general MPC protocol based on a semi-honest oblivious transfer protocol [19]. Subsequently, Wee [36] reduced the round complexity to $O(\log^* n)$, and Goyal [17] further reduced the round complexity to $O(1)$.

The security of these black-box protocols are proven in the *stand-alone setting*. Hence, these protocols are secure when a single instance of the protocol is executed at a time.

Composable Security

A setting that is more general and realistic than the stand-alone setting is the *concurrent setting*, in which many instances of many different protocols are concurrently executed in an arbitrary schedule. A notable difference from the stand-alone setting is that adversaries can now perform a coordinated attack by choosing their messages in an instance based on the executions of the other instances.

As a strong and realistic security notion in the concurrent setting, Canetti [2] proposed *universally composable (UC) security*. The main advantage of UC security is *composability*, which guarantees that UC-secure protocols can be composed in such a way that the security of the resultant protocol can be deduced from the security of its components (in other words, UC security enables modular constructions of secure protocols). Composability also guarantees that a protocol remains secure even when it is concurrently executed with any other protocols in any schedule (that is, UC security implies security in the concurrent setting). A UC-secure general MPC protocol was constructed by Canetti et al. [8] in the *common reference string (CRS) model* (i.e., in a model in which all parties are given a common public string that is chosen by a trusted third party). A

black-box construction of a UC-secure general MPC protocol was constructed by Ishai et al. [22] in the $\mathcal{F}_{\mathrm{OT}}$-hybrid model (i.e., in model with the ideal oblivious transfer functionality) and by Choi et al. [4] in the $\mathcal{F}_{\mathrm{COM}}$-hybrid model (i.e., in the model with the ideal commitment functionality).

UC security, however, turned out to be too strong to achieve in the *plain model*. That is, it was shown that even with non-black-box use of cryptographic primitives, we cannot construct UC-secure general MPC protocols in the model with no trusted setup [6,7].

To achieve composable security in the plain model, Prabhakaran and Sahai [32] proposed a variant of UC security called *angel-based UC security*. Roughly speaking, angel-based UC security is the same as UC security except that the adversary and the simulator have access to an additional entity—an *angel*—that allows some judicious use of super-polynomial-time resources. Angel-based UC security is weaker than UC security but guarantees meaningful security in many settings. (For example, angel-based UC security implies *super-polynomial-time simulation (SPS) security* [1,13,29,30], in which the simulator is allowed to run in super-polynomial time. Hence, angel-based UC security guarantees that whatever an adversary can do in the real world can also be done in the ideal world *in super-polynomial time*.) Furthermore, it was proven that, like UC security, angel-based UC security guarantees composability. (In contrast, SPS security does not guarantee composability.), Prabhakaran and Sahai [32] presented a general MPC protocol that satisfies angel-based UC security in the plain model under new assumptions. Subsequently, Malkin et al. [26] constructed another general MPC protocol that satisfies angel-based UC security in the plain model under a new number-theoretic assumption.

Several works have constructed general MPC protocols with angel-based UC security under standard assumptions. Canetti et al. [9,10] constructed a polynomial-round general MPC protocol in angel-based UC security assuming the existence of enhanced trapdoor permutations. Subsequently, Goyal et al. [15] reduced the round complexity to $\widetilde{O}(\log n)$ under the same assumption. They also showed that by using enhanced trapdoor permutations that are secure against quasi-polynomial-time adversaries, the round complexity of their protocols can be reduced to $O(1)$.

The constructions of these MPC protocols are non-black-box, so they use underlying primitives in a non-black-box way.

Black-Box Constructions of Composable Protocols

Recently, Lin and Pass [24] showed the first black-box construction of a general MPC protocol that guarantees composable security in the plain model. The security of their protocol is proven under angel-based UC security and based on the minimal assumption of the existence of semi-honest oblivious transfer (OT) protocols. The round complexity of their protocol is $O(\max(n^{\epsilon}, R_{\mathrm{OT}}))$, where $\epsilon > 0$ is an arbitrary constant and R_{OT} is the round complexity of the underlying semi-honest OT protocols. Thus, with enhanced trapdoor permutations (from which we can construct constant-round semi-honest OT protocols), their result gives an $O(n^{\epsilon})$-round protocol. Subsequently, a constant-round protocol was constructed by Kiyoshima et al. [23] from constant-round semi-honest OT protocols that are secure against quasi-polynomial-time adversaries and one-way functions that are secure against subexponential-time adversaries.

Summarizing the state of the art, for composable protocols in the plain model, we have

- $\widetilde{O}(\log n)$-round non-black-box constructions under a standard polynomial-time hardness assumption [15],
- a $O(n^{\epsilon})$-round black-box construction under a standard polynomial-time hardness assumption [24], and
- $O(1)$-round black-box or non-black-box constructions under standard super-polynomial-time hardness assumptions [15,23].

Thus, for composable protocols based on standard polynomial-time hardness assumptions, there exists a gap between the round complexity of the non-black-box protocols ($\widetilde{O}(\log n)$ rounds [15]) and that of the black-box protocols ($O(n^{\epsilon})$ rounds [24]). The following is therefore an interesting open question.

> *Does there exists a* ***round-efficient*** *black-box construction of a general MPC protocol that guarantees composability in the plain model under polynomial-time hardness assumptions?*

1.1. *Our Result*

In this paper, we narrow the gap between the round complexity of black-box composable general MPC protocols and that of non-black-box ones.

Main Theorem. *Assume the existence of R_{OT}-round semi-honest oblivious transfer protocols. Then, there exists a $\max(\widetilde{O}(\log^2 n), O(R_{\mathsf{OT}}))$-round black-box construction of a general MPC protocol that satisfies angel-based UC security in the plain model.*

Recall that, assuming the existence of enhanced trapdoor permutations, we have a constant-round semi-honest OT protocol. Thus, under this assumption, our main theorem gives a $\widetilde{O}(\log^2 n)$-round protocol.

CCA-secure commitment scheme. To prove our main theorem, we construct a $\widetilde{O}(\log^2 n)$-round black-box construction of a *CCA-secure commitment scheme* [9,10,15,23,24] from one-way functions.

Theorem. *Assume the existence of one-way functions. Then, there exists a $\widetilde{O}(\log^2 n)$-round black-box construction of a CCA-secure commitment scheme.*

Roughly speaking, a CCA-secure commitment scheme is a tag-based commitment scheme (i.e., a commitment scheme that takes an n-bit string, a *tag*, as an additional input) such that the hiding property holds even against adversaries that interact with the *committed-value oracle* during the interaction with the challenger. The committed-value oracle interacts with the adversary as an honest receiver in many concurrent sessions of the commit phase. At the end of each session, if the commitment of this session is invalid or has multiple committed values, the oracle returns $\perp$ to the adversary. Otherwise, the oracle returns the unique committed value to the adversary.

Lin and Pass [24] showed that in angel-based UC security, an $O(\max(R_{\mathsf{CCA}}, R_{\mathsf{OT}}))$-round general MPC protocol can be obtained in a black-box way from a R_{CCA}-round

CCA-secure commitment scheme and a R_{OT}-round semi-honest OT protocol. Thus, we can prove our main theorem by combining the above theorem with the result of Lin and Pass [24].

1.2. *Outline*

In Sect. 2, we give an overview of our CCA-secure commitment scheme. In Sect. 3, we give definitions that are used throughout the paper. In Sect. 4, we show the building blocks that are used in our CCA-secure commitment scheme. In Sect. 5, we show our CCA-secure commitment scheme and prove its security. In Sect. 6, we show our main theorem.

2. Overview of Our CCA-Secure Commitment Scheme

In the previous work on CCA-secure commitment schemes [9,10,15,23,24], *extractability* and *non-malleability* play fundamental roles in the proof of CCA security. Roughly speaking, the CCA security of the existing CCA-secure commitment schemes is proven by reducing it to the hiding property [9,10,24] or by showing that the proof of the hiding property goes though even in the presence of the committed-value oracle [15,23]. During the security proofs, extractability is used to show that the committed-value oracle can be emulated in polynomial time by extracting the committed values from the adversary, and non-malleability is used to show that the emulation of the oracle can be performed without "disturbing" the hiding property [9,10,24] or each step of the proof of the hiding property [15,23].

In this work, we use stronger notions of extractability and non-malleability called *strong extractability* and *one-one CCA security*. In the following, we explain how we construct commitment schemes that satisfy these two notions and how we construct our CCA-secure commitment scheme by using them as building blocks.

2.1. *Building Block 1: Strongly Extractable Commitment Scheme*

A commitment scheme is *strongly extractable* if a rewinding extractor can extract the committed value of a commitment in such a way that the extractor outputs $\perp$ when the commitment is invalid.[2] Strong extractability differs from basic extractability in that it requires the extractor to output $\perp$ when the commitment is invalid; basic extractability, in contrast, allows the extractor to output an arbitrary value when the commitment is invalid. (This is called *over-extraction*.) A constant-round extractable commitment scheme ExtCom can be constructed in a black-box way from one-way functions [34]; however, no black-box construction of a strong extractable commitment scheme has been constructed.

To construct a strongly extractable commitment scheme, we start from the following scheme, in which the cut-and-choose technique is used in the same way as in the previous work on black-box protocols [3–5,23,24,36].

[2] A commitment is *valid* if there exists a valid decommitment of this commitment; otherwise, it is *invalid*.

1. Let v be the value to be committed. Then, the committer computes an $(n+1)$-out-of-$10n$ Shamir's secret sharing $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of value v and commits to each s_j in parallel by using ExtCom.
2. The receiver sends a random subset $\Gamma \subset [10n]$ of size n.
3. For every $j \in \Gamma$, the committer decommits the jth ExtCom commitment to s_j.
4. The receiver accepts the commitment if and only if the decommitments of ExtCom are valid for every $j \in \Gamma$.

For $j \in [10n]$, let us call the jth ExtCom commitment *the jth column*. In this scheme, the ExtCom commitments are valid in most columns when the receiver accepts the commitment in Step 4; this is because when the ExtCom commitments are invalid in, say, n columns, at least one of them is chosen by Γ, and the receiver rejects the commitment in Step 4 except with exponentially small probability. Since the committed value of a ExtCom commitment can be extracted when it is valid, this implies that the committed shares can be extracted in most columns when the receiver accepts the commitment in Step 4; therefore, when the commitment is valid, the committed value v can be recovered by extracting the committed shares from the ExtCom commitments and then using the error-correcting property of Shamir's secret sharing scheme.[3] Furthermore, by carefully designing the decommit phase as in [3–5,23,24,36], we can make sure that the extractor outputs $\perp$ when the commitment is invalid.

The problem of this scheme is that we do not know how to prove its hiding property. In particular, since the receiver requests the committer to open adaptively chosen ExtCom commitments, it can perform *selective opening attacks* [12], and therefore the hiding property of this scheme cannot be reduced to the hiding property of ExtCom easily.

We therefore modify the scheme and let the receiver commit to Γ at the beginning by using a statistically binding commitment scheme Com. Now, since the receiver no longer chooses the subset Γ adaptively, we can prove the hiding property by using a standard technique. Furthermore, at first sight, the hiding property of Com seems to guarantee that the scheme remains strongly extractable.

In the modified scheme, however, we cannot prove the strong extractability. This is because we can no longer show that most of the ExtCom commitments are valid in an accepting commitment. Consider, for example, that there exists a cheating committer C^* such that after receiving a Com commitment to Γ at the beginning, C^* somehow generates an invalid ExtCom commitment in the jth column for every $j \notin \Gamma$ and commits to 0^n in the jth column for every $j \in \Gamma$. Intuitively, it seems that C^* breaks the hiding property of Com. However, we do not know how to use C^* to break the hiding property of Com. To see this, observe the following. Recall that since ExtCom is extractable *with* over-extraction, the extractor of ExtCom may output an arbitrary value when the ExtCom commitment is invalid. Hence, when we extract the committed values of the ExtCom commitments from C^*, the extracted value may be 0^n in every column. Therefore, although C^* behaves differently in ExtCom based on the value of Γ, we do not know how to detect it.

To overcome this problem, we use the commitment scheme wExtCom that was introduced by Goyal et al. [14]. Roughly speaking, wExtCom is a scheme that is

[3] Recall that Shamir's secret sharing is also a codeword of Reed-Solomon code.

extractable only in a weak sense—extractions may fail with probability at most $1/2$—but is extractable without over-extraction. That is, the extractor may output $\perp$ with probability $1/2$, but when the extractor outputs $v \neq \perp$, the commitment is valid and its committed value is v. Concretely, the commit phase of wExtCom consists of three stages.

1. `commit stage`. The committer commits to random $a_0, a_1 \in \{0,1\}^n$ such that $a_0 \oplus a_1 = v$.
2. `challenge stage`. The receiver sends a random bit $ch \in \{0,1\}$.
3. `reply stage`. The committer reveals a_{ch} and decommits the corresponding commitment.

It is easy to see that wExtCom satisfies the following property: For a fixed transcript of the `commit stage`, if a cheating committer returns a valid reply with probability $1/\mathsf{poly}(n)$ for both $ch = 0$ and $ch = 1$, then the committed value can be extracted with probability 1 in expected polynomial time by rewinding the cheating committer.

Using wExtCom, we modify our scheme as follows. After committing to $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ with ExtCom, the committer commits to (s_j, d_j) for each $j \in [10n]$ in parallel by using wExtCom, where (s_j, d_j) is a decommitment of the ExtCom commitment in the jth column. We then show that most columns are *consistent* in an accepted commitment except with negligible probability, meaning that in most columns on an accepted commitment, the wExtCom commitment is valid and its committed value is a valid decommitment of the corresponding ExtCom commitment except with negligible probability. Toward this end, we observe the following.

- If a cheating committer generates an accepting commitment with non-negligible probability, in wExtCom of more than $9n$ columns the cheating committer returns a valid reply with non-negligible probability for both $ch = 0$ and $ch = 1$. This is because if the cheating committer returns a valid reply with non-negligible probability for both $ch = 0$ and $ch = 1$ in wExtCom of at most $9n$ columns, there are n columns in which the wExtCom commitment is accepted with probability at most $1/2 + \mathsf{negl}(n)$, so the probability that all wExtCom commitments are accepted is negligible.[4]
- Then, from the property of wExtCom, we can extract the committed values of the wExtCom commitments without over-extraction in more than $9n$ columns.
- Then, from the property of the cut-and-choose technique, we can show that in most columns of an accepting commitment, the wExtCom commitment is valid and its committed value is a valid decommitment of the corresponding ExtCom commitment. Note that since the committed values of wExtCom commitments can be extracted without over-extraction, we can show that the cheating committer cannot give invalid wExtCom commitments in many columns.

Then, since the ExtCom commitments are valid in consistent rows, we have that most of the ExtCom commitments are valid whenever the commitment is accepted. We can thus extract the committed value of the scheme without over-extraction as before, i.e., by extracting the committed values of ExtCom commitments and then using the error-correcting property of Shamir's secret sharing scheme.

[4] The formal proof is more complicated because the wExtCom commitments are executed in parallel and thus the columns are not independent of each other.

2.2. *Building Block 2: One-One CCA-Secure Commitment Scheme*

A *one-one CCA-secure commitment scheme*, which is closely related to a *non-malleable commitment scheme*, is one that is CCA secure w.r.t. a restricted class of adversaries that execute only a single session with the committed-value oracle and obtain its committed value from the oracle at the end of the session.[5]

We construct a black-box $O(\log n)$-round one-one CCA-secure commitment scheme by simplifying the CCA-secure commitment scheme of Lin and Pass [24] and then applying the "DDN $\log n$ trick" [11,25] on it, where the DDN $\log n$ trick is a transformation by Dolev, Dwork, and Naor (DDN) [11] and has been used to transform a concurrent non-malleable commitment scheme for tags of length $O(\log n)$ to a non-malleable commitment scheme for tags of length $O(n)$ without increasing the round complexity. Roughly speaking, the scheme of [24] consists of polynomially many *rows*—each row is a parallel execution of (a part of) the trapdoor commitment scheme of [34]—and a cut-and-choose phase, which forces the committer to give valid and consistent trapdoor commitments in every row. Our idea is to reduce the number of rows from $\mathsf{poly}(n)$ to $\ell(n)$ in the scheme of [24], where $\ell(n)$ is the length of the tags. The resultant scheme is no longer CCA secure, but can be shown to be *parallel CCA secure*, i.e., CCA secure w.r.t. a restricted class of adversaries that give only a single parallel queries to the oracle. Then, we set $\ell(n) := O(\log n)$ and apply the DDN $\log n$ trick to the above parallel CCA-secure commitment scheme. It is not hard to show that the resultant scheme is one-one CCA secure.

2.3. *CCA-Secure Commitment Scheme from the Building Blocks*

Now, we explain how we obtain our CCA-secure commitment scheme, CCACom, using a constant-round strongly extractable commitment scheme $\mathsf{sExtCom}$ and a $O(\log n)$-round one-one CCA-secure commitment scheme $\mathsf{CCACom}^{1:1}$ as building blocks.

In addition to $\mathsf{sExtCom}$ and $\mathsf{CCACom}^{1:1}$, we use the *concurrently extractable commitment scheme* of Micciancio et al. [27] in our CCA-secure commitment scheme. Roughly speaking, concurrent extractability guarantees that a rewinding extractor can extract committed values even from polynomially many commitments that are concurrently generated by an adversarial committer. The concurrently extractable commitment scheme of Micciancio et al. [27], which we denote by CECom, is an abstraction of the preamble stage of the concurrent zero-knowledge protocol of Prabhakaran et al. [31] and is constructed in a black-box way from one-way functions. CECom satisfies even a stronger notion of concurrent extractability called *robust concurrent extractability* [15], which roughly guarantees that the extractor works even against adversarial committers that additionally participate in an arbitrary external protocol, and furthermore, even though the extractor rewinds the adversarial committers, the external protocol is not rewound during the extraction. CECom satisfies robust concurrent extractability for a k-round external protocol if a parameter ℓ of CECom (often called "the number of slots" in CECom) satisfies $\ell = \omega(k \log n)$. The round complexity of CECom is $O(\ell)$.

[5] In contrast, a non-malleable commitment scheme is one that is CCA secure w.r.t. a restricted class of adversaries that execute a single session with the oracle and obtain its committed value after completing the session with the oracle *and the session with the challenger*.

Using $\mathsf{sExtCom}$, $\mathsf{CCACom}^{1:1}$, and CECom as building blocks, we construct CCACom roughly as follows. Let v be the value to be committed to and tag be the tag.

1. The receiver commits to a random subset $\Gamma \subset [10n]$ of size n by using $\mathsf{CCACom}^{1:1}$, where the tag of $\mathsf{CCACom}^{1:1}$ is tag.
2. The committer computes an $(n+1)$-out-of-$10n$ Shamir's secret sharing $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of value v and commits to each s_j in parallel by using a two-round statistically binding commitment scheme Com. Let $\phi_1, \ldots, \phi_{10n}$ be the commitments and $d_1, \ldots, d_{10n}$ be their decommitments.
3. The committer commits to s_j by using CECom for every $j \in [10n]$ in parallel. Let $\psi_1, \ldots, \psi_{10n}$ be the commitments and $e_1, \ldots, e_{10n}$ be their decommitments. The parameter ℓ of CECom is set as $\ell := O(\log^2 n \log\log n)$ so that we have $\ell = \omega(\log^2 n)$.
4. The committer commits to $u_j \stackrel{\text{def}}{=} (s_j, d_j, e_j)$ by using $\mathsf{sExtCom}$ for every $j \in [10n]$ in parallel.
5. The receiver decommits the $\mathsf{CCACom}^{1:1}$ commitment in the first step to Γ.
6. For every $j \in \Gamma$, the committer decommits the jth $\mathsf{sExtCom}$ commitment to $u_j = (s_j, d_j, e_j)$. The receiver verifies whether (s_j, d_j) and (s_j, e_j) are valid decommitments of ϕ_j and $\psi_{\eta,j}$ for every $j \in \Gamma$.

The committed value of a CCACom commitment is defined by the shares that are committed to in the Com commitments (i.e., the committed value is the value that can be reconstructed from these shares).

We prove the CCA security using a hybrid argument. Recall that CCA security requires that the hiding property holds even against adversaries that interact with the committed-value oracle. Toward proving the CCA security of CCACom, we design a series of hybrid experiments in which the CCACom commitment that the adversary receives in the *left session* (the session between the adversary and the challenger) is gradually changed as follows.

- In Hybrid H_0, the CCA-security experiment is executed honestly.
- In Hybrid H_1, the values that are committed to by $\mathsf{sExtCom}$ are switched from u_j to $0^{|u_j|}$ for every $j \notin \Gamma$, where Γ is the subset that is committed to by the adversary in the first step.
- In Hybrid H_2, the values that are committed to by CECom are switched from s_j to $0^{|s_j|}$ for every $j \notin \Gamma$.
- In Hybrid H_3, the values that are committed to by Com are switched from s_j to $0^{|s_j|}$ for every $j \notin \Gamma$.

From the security of Shamir's secret sharing, the adversary receives no information about v in H_3. Hence, from a hybrid argument, we can prove CCA security by showing indistinguishability between neighboring hybrid experiments.

Since neighboring hybrids differ only in the values that are committed to in the *row* of $\mathsf{sExtCom}$, CECom, or Com (i.e., the parallel commitments of $\mathsf{sExtCom}$, CECom, or Com), our overall strategy for proving the indistinguishability is to use the hiding property of $\mathsf{sExtCom}$, CECom, and Com. A problem is that the adversary interacts with the committed-value oracle, which extracts the committed values of the *right sessions*

(the sessions between the adversary and the committed-value oracle) in super-polynomial time; because of the super-polynomial power of the oracle, the indistinguishability does not follow directly from the hiding property of sExtCom, CECom, and Com. We overcome this problem by showing that the committed-value oracle can be emulated in polynomial time. Specifically, we show that the oracle can be emulated by extracting the committed shares from the rows of CECom using its concurrent extractability and then computing the committed value of each right session from the extracted shares. Roughly speaking, this emulation works because in an accepting right session, the shares committed to in the row of CECom must be "close" to the shares that are committed to in the row of Com (recall that the committed value of a CCACom commitment is defined based on the shares that are committed to in the row of Com); in fact, if they disagree in many locations, the session will be rejected in the last step of the scheme.

In more detail, we prove the indistinguishability between, say, the first and second hybrids in two steps.

Step 1 Prove the indistinguishability assuming that the adversary does not "cheat" in each right session, where, roughly speaking, we say that the adversary cheats in a right session if the adversary commits to $u_j = (s_j, d_j, e_j)$ in the row of sExtCom as specified by the scheme in at most $9n$ locations in an accepting session.

Step 2 Prove that the adversary does not cheat in the right sessions except with negligible probability.

Each step is explained in more detail below.

Step 1: Proving the indistinguishability assuming that the adversary does not cheat. Recall that H_0 and H_1 differ only in the values that are committed to in the row of sExtCom in the left session. For proving indistinguishability between them, we consider new hybrid experiments, G_0 and G_1, such that G_h ($h \in \{0, 1\}$) is the same as H_h except that the committed-value oracle computes the committed value of each right session from the shares that are extracted from the row of CECom (rather than from the row of Com), and those shares are extracted using the robust concurrent extractability of CECom so that the row of sExtCom in the left session is not rewound during the extraction. We then prove the indistinguishability between H_0 and H_1 in two steps.

1. First, we show the indistinguishability between H_h and G_h. Since we assume that the adversary does not cheat in the right sessions, the shares that are committed to in the row of Com and those that are committed to in the row of CECom are 0.9-close. Combined with an error-correcting property of Shamir's secret sharing, their closeness guarantees that the correct committed values of the right seasons are computable even from the shares that are committed to in the row of CECom; hence, the committed-value oracle computes the same value in H_h and G_h, so these two hybrids are indistinguishable.
2. Second, we show the indistinguishability between G_0 and G_1 by using the hiding property of sExtCom. Since these two hybrids run in polynomial time while the adversary is receiving the row of sExtCom in the left session, and the row of sExtCom in the left session is not rewound thanks to the robust concurrent

extractability of CECom, we can easily design a (non-uniform) reduction from the indistinguishability between G_0 and G_1 to the hiding property of $\mathsf{sExtCom}$.

Combining these two, we obtain the indistinguishability between H_0 and H_1 under the assumption that the adversary does not cheat in the right sessions.

Step 2: Proving that the adversary cannot cheat. Intuitively, the adversary cannot cheat in a right session because the subset that is committed to in the $\mathsf{CCACom}^{1:1}$ commitment of that right session is hidden from the adversary. In fact, if the subsets are hidden from the adversary, we can argue that a right session will be rejected in the last step of the scheme when the adversary tries to cheat in that session. However, to formalize this intuition, we need to overcome two obstacles.

Obstacle 1 The adversary interacts with the committed-value oracle, which runs in super-polynomial time. We overcome this obstacle by, again, considering a hybrid in which the oracle is emulated in polynomial time.

Obstacle 2 The challenger cheats in the left session in H_1, H_2, H_3 (recall that in these hybrids, the challenger commits to $0^{|u_j|}$ rather than u_j for every $j \notin \Gamma$ in the row of $\mathsf{sExtCom}$), and thus, the adversary may be able to cheat in a right session by using the messages in the left session. We overcome this obstacle by using the *simulation-soundness* of the cut-and-choose phase. Specifically, since the cheating challenger can be emulated in polynomial time by making a single query to the committed-value oracle of $\mathsf{CCACom}^{1:1}$ (that is, the left session can be emulated in polynomial time if the subset that is committed in to $\mathsf{CCACom}^{1:1}$ is given), the one-one CCA security of $\mathsf{CCACom}^{1:1}$ guarantees that the subset in each right session is hidden even though the challenger cheats in the left session.

More formally, the proof proceeds as follows. Assume for contradiction that the adversary cheats in a right session with non-negligible probability in, say, H_1. Then, there exists a right session such that the adversary cheats with non-negligible probability in this right session but does not cheat except with negligible probability in any right session that completed before this right session; we call this right session the *target right session*. Then, we consider a hybrid experiment that is the same as H_1 except for the following.

- The execution of H_1 is terminated just before the committed-value oracle returns the committed value in the target right session.
- The oracle computes the committed value of each right session from the shares that are extracted from the row of CECom, and those shares are extracted using the robust concurrent extractability of CECom so that the row of $\mathsf{CCACom}^{1:1}$ in the left session, the row of $\mathsf{CCACom}^{1:1}$ in the target right session, and the row of $\mathsf{sExtCom}$ in the target right session are not rewound during the extraction. (Such robust concurrent extraction is possible since the total round complexity of these rows is $O(\log n)$ and the parameter ℓ of CECom satisfies $\ell = \omega(\log^2 n)$.)

Since the oracle returns the committed values only in the right session that terminates before the target right session, and it is assumed that the adversary does not cheat in such right sessions, we can show, as before, that the oracle is correctly emulated in

this hybrid. Thus, the adversary cheats in the target right session with non-negligible probability even in this hybrid. Now, since this hybrid runs in polynomial time except when extracting the subset from the $\mathsf{CCACom}^{1:1}$ commitment in the left session, we can break the one-one CCA security of $\mathsf{CCACom}^{1:1}$ in the target right session by extracting the shares committed to in the row of $\mathsf{sExtCom}$ in the target right session and checking the locations where the adversary does not commit to $u_j = (s_j, d_j, e_j)$ in the row of $\mathsf{sExtCom}$ as specified by the scheme, while simulating the left session using the committed-value oracle of $\mathsf{CCACom}^{1:1}$. Hence, we conclude that the adversary does not cheat in the right sessions except with negligible probability.

Remark 1. In the above explanation, we assume that $\mathsf{sExtCom}$ has a robust extractability property such that the extraction from the row of $\mathsf{sExtCom}$ is possible even while the $\mathsf{CCACom}^{1:1}$ commitment in the left session is forwarded to the committed-value oracle of $\mathsf{CCACom}^{1:1}$. In the actual proof, we remove the necessity of robust extractability by increasing the number of rows of $\mathsf{sExtCom}$ to $R_{\mathsf{CCA}^{1:1}} + 1$, where $R_{\mathsf{CCA}^{1:1}}$ is the round complexity of $\mathsf{CCACom}^{1:1}$. With $R_{\mathsf{CCA}^{1:1}} + 1$ rows of $\mathsf{sExtCom}$, we can argue that one of the rows of $\mathsf{sExtCom}$ in the target right session does not "interleave" with the $\mathsf{CCACom}^{1:1}$ commitment of the left session, so we extract the values that are committed to in this row of $\mathsf{sExtCom}$. □

Remark 2. We note that in the above argument, $\mathsf{CCACom}^{1:1}$ need to be one-one CCA secure (rather than just non-malleable) since we need to obtain the committed subset from the oracle immediately after completing the query to the oracle (and possibly before completing the challenge commitment). We also note that $\mathsf{sExtCom}$ must be strongly extractable since otherwise the adversary may give invalid commitments in more than n locations without being detected in the cut-and-choose phase. (As explained in Sect. 2.1, the existence of such an adversary does not contradict the one-one CCA security of $\mathsf{CCACom}^{1:1}$ if over-extraction can occur.) □

Combining Steps 1 and 2, we conclude that H_0 and H_1 are indistinguishable. The indistinguishability between other neighboring hybrids can be shown similarly.

3. Preliminaries

Throughout the paper, we use n to denote the security parameter, $\mathbb{N}$ to denote the set of all natural numbers, and PPT as an abbreviation of "probabilistic polynomial time." For any $k \in \mathbb{N}$, we use $[k]$ to denote the set $\{1, 2, \ldots, k\}$. For any two ensembles of random variables, $\{X_n\}_{n\in\mathbb{N}}$ and $\{Y_n\}_{n\in\mathbb{N}}$, we use $\{X_n\}_{n\in\mathbb{N}} \overset{c}{\approx} \{Y_n\}_{n\in\mathbb{N}}$ to denote that $\{X_n\}_{n\in\mathbb{N}}$ and $\{Y_n\}_{n\in\mathbb{N}}$ are computationally indistinguishable and $\{X_n\}_{n\in\mathbb{N}} \overset{s}{\approx} \{Y_n\}_{n\in\mathbb{N}}$ to denote that $\{X_n\}_{n\in\mathbb{N}}$ and $\{Y_n\}_{n\in\mathbb{N}}$ are statistically indistinguishable. We assume familiarity with the notion of cryptographic protocols, which are formalized as interactions between interactive Turing machines (ITMs). We remind the reader that the *view* of a party in the execution of a cryptographic protocol consists of the input of the party, randomness of the party, and all the messages received by the party.

Reconstruction procedure $\mathsf{Value}_\Gamma(\boldsymbol{s})$**.** For $\boldsymbol{s} = (s_1, \ldots, s_{10n})$, the output of $\mathsf{Value}_\Gamma(\boldsymbol{s})$ is computed as follows. If $\boldsymbol{s}$ is 0.9-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ that satisfies $w_i = s_i$ for every $i \in \Gamma$, then $\mathsf{Value}(\boldsymbol{s})$ is the value that is decoded from $\boldsymbol{w}$, i.e., $\mathsf{Value}_\Gamma(\boldsymbol{s}) \stackrel{\text{def}}{=} \mathsf{Decode}(\boldsymbol{w})$. Otherwise, $\mathsf{Value}_\Gamma(\boldsymbol{s}) \stackrel{\text{def}}{=} \perp$.

Fig. 1. Function $\mathsf{Value}_\Gamma(\cdot)$.

3.1. *Shamir's Secret Sharing*

We first recall Shamir's secret sharing scheme. (In this paper, we use only the $(n+1)$-out-of-$10n$ version of it.) To compute a $(n+1)$-out-of-$10n$ secret sharing $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of a value $v \in GF(2^n)$, we choose random $a_1, \ldots, a_n \in GF(2^n)$, let $p(z) \stackrel{\text{def}}{=} v + a_1 z + \cdots + a_n z^n$, and set $s_i := p(i)$ for each $i \in [10n]$. Given $\boldsymbol{s}$, we can recover v by obtaining polynomial $p(\cdot)$ thorough interpolation and then computing $p(0)$. We use $\mathsf{Decode}(\cdot)$ to denote a function that recovers v from $\boldsymbol{s}$ as above.

For any positive real number $x \leq 1$ and any $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ and $\boldsymbol{s'} = (s'_1, \ldots, s'_{10n})$, we say that $\boldsymbol{s}$ and $\boldsymbol{s'}$ are *x-close* if $|\{i \in [10n] \text{ s.t. } s_i = s'_i\}| \geq x \cdot 10n$. If $\boldsymbol{s}$ and $\boldsymbol{s'}$ are not x-close, we say that they are *$(1-x)$-far*. Since the shares generated by $(n+1)$-out-of-$10n$ Shamir's secret sharing scheme are actually a codeword of the Reed-Solomon code with minimum relative distance 0.9, if a (possibly incorrectly generated) sharing $\boldsymbol{s}$ is 0.55-close to a valid codeword $\boldsymbol{w}$, we can recover $\boldsymbol{w}$ from $\boldsymbol{s}$ efficiently by using, for example, the Berlekamp–Welch algorithm.

The following technical lemma will be used in the analyses of our commitment schemes in Sects. 4.1 and 5.

Lemma 1. *Let $\boldsymbol{x} = (x_1, \ldots, x_{10n})$ and $\boldsymbol{y} = (y_1, \ldots, y_{10n})$ be any (possibly incorrectly generated) shares of $(n+1)$-out-of-$10n$ Shamir's secret sharing scheme, where some of these shares may be equal to a special error symbol $\perp$. For any set $\Gamma \subset [10n]$ of size n, let $\mathsf{Value}_\Gamma(\cdot)$ be the function that is defined in Fig. 1.*

Then, we have $\mathsf{Value}_\Gamma(\boldsymbol{x}) = \mathsf{Value}_\Gamma(\boldsymbol{y})$ if the following three conditions hold.

1. *For every $i \in [10n]$, if $x_i \neq \perp$, it holds $x_i = y_i$.*
2. *$|\{i \in [10n] \text{ s.t. } x_i = \perp\}| < n \bigwedge \{i \in [10n] \text{ s.t. } x_i = \perp\} \cap \Gamma = \emptyset$.*
3. *$\boldsymbol{x}$ is either 0.9-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ that satisfies $w_i = x_i$ for every $i \in \Gamma$ or 0.2-far from any such valid codeword.*

Proof. We consider two cases.

Case 1 $\boldsymbol{x}$ **is** 0.9**-close to a valid codeword** $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ **that satisfies** $w_i = x_i$ **for every** $i \in \Gamma$: First, we observe that $\boldsymbol{y}$ is also 0.9-close to $\boldsymbol{w}$. Since $\boldsymbol{w}$ is a valid codeword, we have $w_i \neq \perp$ for every $i \in [10n]$; thus, we have $x_i \neq \perp$ for every i such that $x_i = w_i$. Also, from the first assumed condition, we have $x_i = y_i$ for every i such that $x_i \neq \perp$. Therefore, we have $y_i = w_i$ for every i such that $x_i = w_i$. Then, since $\boldsymbol{x}$ is 0.9-close to $\boldsymbol{w}$ from the assumption of this case, we have that $\boldsymbol{y}$ is 0.9-close to $\boldsymbol{w}$.

Next, we observe that $\boldsymbol{w}$ satisfies $w_i = y_i$ for every $i \in \Gamma$. From the second assumed condition, we have $x_i \neq \perp$ for every $i \in \Gamma$. Also, from the first assumed condition, we have $x_i = y_i$ for every i such that $x_i \neq \perp$. Thus, we have $x_i = y_i$ for every $i \in \Gamma$. Then, since we have $w_i = x_i$ for every $i \in \Gamma$ from the assumption of this case, we have $w_i = y_i$ for every $i \in \Gamma$.
Now, since $\boldsymbol{y}$ is 0.9-close to $\boldsymbol{w}$, and $\boldsymbol{w}$ satisfies $w_i = y_i$ for every $i \in \Gamma$, we have $\mathsf{Value}_\Gamma(\boldsymbol{x}) = \mathsf{Value}_\Gamma(\boldsymbol{y}) = \mathsf{Decode}(\boldsymbol{w})$ from the definition of $\mathsf{Value}_\Gamma(\cdot)$.

Case 2 $\boldsymbol{x}$ **is** 0.2**-far from any valid codeword** $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ **that satisfies** $w_i = x_i$ **for every** $i \in \Gamma$: For any valid codeword $\boldsymbol{w'} = (w'_1, \ldots, w'_{10n})$ that satisfies $w'_i = y_i$ for every $i \in \Gamma$, we observe that $\boldsymbol{y}$ is 0.1-far from $\boldsymbol{w'}$. Since we have $x_i \neq \perp$ for every $i \in \Gamma$ (the second assumed condition) and $x_i = y_i$ for every i such that $x_i \neq \perp$ (the first assumed condition), we have $x_i = y_i$ for every $i \in \Gamma$. Then, since we have $w'_i = y_i$ for every $i \in \Gamma$, we have $w'_i = x_i$ for every $i \in \Gamma$. Thus, $\boldsymbol{x}$ is 0.2-far from $\boldsymbol{w'}$ from the assumption of this case.
Now, since $\boldsymbol{x}$ and $\boldsymbol{y}$ are 0.9-close from the first and second assumed conditions, it follows that $\boldsymbol{y}$ is 0.1-far from $\boldsymbol{w'}$.
Now, from the definition of $\mathsf{Value}_\Gamma(\cdot)$, we conclude that $\mathsf{Value}_\Gamma(\boldsymbol{x}) = \mathsf{Value}_\Gamma(\boldsymbol{y}) = \perp$.

Notice that from the third assumed condition, either Case 1 or 2 is true. This concludes the proof of Lemma 1. □

3.2. *Commitment Schemes*

We next recall the definition of commitment schemes. Commitment schemes, often described as a digital equivalent of sealed envelopes, are two-party protocols between a *committer* and a *receiver*. Commitment schemes have two phases: the *commit phase* and the *decommit phase*. In the commit phase, the committer *commits* to a secret input $v \in \{0, 1\}^n$ by interacting with the receiver; the transcript of the commit phase is called the *commitment*. In the decommit phase, the committer *decommits* the commitment to v by sending the receiver a message called the *decommitment*; the receiver then outputs either 1 (accept) or 0 (reject). It is required that the receiver accepts the decommitment with probability 1 when both the committer and the receiver behave honestly. Additionally, it is required that the committer cannot decommit a commitment to two different values and that the committed value is hidden from the receiver in the commit phase; the former is called the *binding* property and the latter is called the *hiding* property. Formal definitions of the (statistically) binding and (computationally) hiding properties are given below.

Definition 1. (*Statistical binding property*) For a commitment scheme $\langle C, R\rangle$ and any (not necessarily PPT) adversarial committer C^*, consider the following probabilistic experiment $\mathrm{Exp}^{\mathrm{bind}}(\langle C, R\rangle, C^*, n, z)$ for any $n \in \mathbb{N}$ and $z \in \{0, 1\}^*$.

> On input 1^n and auxiliary input z, the adversary C^* interacts with an honest receiver in the commit phase of $\langle C, R\rangle$ and then outputs two decommitments, (v_0, d_0) and (v_1, d_1). Then, C^* is said to win the experiment if $v_0 \neq v_1$ but the receiver accepts both (v_0, d_0) and (v_1, d_1) in the decommit phase.

Then, $\langle C, R\rangle$ is **statistically binding** if for any sequence of auxiliary inputs $\{z_n\}_{n\in\mathbb{N}}$, the probability that C^* wins the experiment $\mathrm{Exp}^{\mathrm{bind}}(\langle C, R\rangle, C^*, n, z_n)$ is negligible. □

Definition 2. (*Computational hiding property*) For a commitment scheme $\langle C, R\rangle$ and any PPT adversarial receiver R^*, consider the following probabilistic experiment $\mathrm{Exp}_b^{\mathsf{hide}}(\langle C, R\rangle, R^*, n, z)$ for any $b \in \{0,1\}$, $n \in \mathbb{N}$, and $z \in \{0,1\}^*$.

> On input 1^n and auxiliary input z, the adversary R^* chooses a pair of challenge values $v_0, v_1 \in \{0,1\}^n$ and then interacts with an honest committer in the commit phase of $\langle C, R\rangle$, where the committer commits to v_b. The output of the experiment is the view of R^*

Let $\mathsf{Exp}_b^{\mathsf{hide}}(\langle C, R\rangle, R^*, n, z)$ denote the output of experiment $\mathrm{Exp}_b^{\mathsf{hide}}(\langle C, R\rangle, R^*, n, z)$. Then, $\langle C, R\rangle$ is **computationally hiding** if the following are computationally indistinguishable.

- $\left\{\mathsf{Exp}_0^{\mathsf{hide}}(\langle C, R\rangle, R^*, n, z)\right\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$
- $\left\{\mathsf{Exp}_1^{\mathsf{hide}}(\langle C, R\rangle, R^*, n, z)\right\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$ □

Unless stated otherwise, all the commitment schemes in this paper are statistically binding and computationally hiding. We say that a commitment is *accepting* if the receiver does not abort in the commit phase, and *valid* if there exists a value to which the commitment can be decommitted (i.e., if there exists a decommitment that the verifier accepts in the decommit phase). The *committed value* of a commitment is the value to which the commitment can be decommitted; we define the committed value of an invalid commitment as $\perp$.

There exists a two-round statistically binding commitment scheme Com based on one-way functions [20,28], and it uses the underlying one-way function in a black-box way.

Strong Computational Binding Property. We say that a commitment scheme $\langle C, R\rangle$ satisfies *strong computational binding property* if any PPT committer C^* can generate a commitment that has more than one committed value with at most negligible probability.[6] A formal definition of the strong computational binding property is given below.

Definition 3. (*Strong computational binding property*) For a commitment scheme $\langle C, R\rangle$ and any PPT adversarial committer C^*, consider the following probabilistic experiment $\mathrm{Exp}^{\mathrm{bind2}}(\langle C, R\rangle, C^*, n, z)$ for any $n \in \mathbb{N}$ and $z \in \{0,1\}^*$.

> On input 1^n and auxiliary input z, the adversary C^* interacts with an honest receiver in the commit phase of $\langle C, R\rangle$. Then, C^* is said to win the experiment if there exists two decommitments, (v_0, d_0) and (v_1, d_1), such that $v_0 \neq v_1$, but the receiver accepts both (v_0, d_0) and (v_1, d_1) in the decommit phase.

Then, $\langle C, R\rangle$ is **strongly computationally binding** if for any sequence of auxiliary inputs $\{z_n\}_{n\in\mathbb{N}}$, the probability that C^* wins the experiment $\mathrm{Exp}^{\mathrm{bind2}}(\langle C, R\rangle, C^*, n, z_n)$ is negligible. □

[6] The standard computational binding property guarantees that for any PPT committer C^*, the commitment that C^* generates cannot be decommitted to more than one value *in polynomial time*. Thus, the commitment that C^* generates is allowed to have more than one committed value.

Let Com be the two-round statistically binding commitment scheme that is mentioned in Section 3.2.

Commit Phase

The committer C and the receiver R take common input 1^n, and C additionally takes private input $v \in \{0,1\}^n$. To commit to v, the committer C does the following with the receiver R.

commit stage. C chooses n independent random pairs $\{(a_0^i, a_1^i)\}_{i\in[n]}$ such that $a_0^i \oplus a_1^i = v$ for every $i \in [n]$. Then, C commits to a_0^i and a_1^i for every $i \in [n]$ by using Com. For each $i \in [n]$ and $b \in \{0,1\}$, let c_b^i be the commitment to a_b^i.

challenge stage. R sends uniformly random bits $\{e_i\}_{i\in[n]}$ to C.

reply stage. C decommits $c_{e_i}^i$ to $a_{e_i}^i$ for every $i \in [n]$.

Decommit Phase

C sends v to R and decommits c_0^i and c_1^i to a_0^i and a_1^i for every $i \in [n]$. Then, R checks whether $a_0^0 \oplus a_1^0 = \cdots = a_0^n \oplus a_1^n = v$.

Fig. 2. Extractable commitment scheme ExtCom[34].

3.3. *Extractable Commitment Schemes*

We next recall the definition of *extractable commitment schemes* from [34]. Roughly speaking, a commitment scheme is *extractable* if there exists an expected polynomial-time oracle machine, called *extractor* E, such that for any adversarial committer C^* that gives a commitment to honest receiver, E^{C^*} extracts the committed value of the commitment from C^* as long as the commitment is valid. We note that when the commitment is invalid, E can output an arbitrary garbage value; this is called *over-extraction*.

Formally, extractable commitment schemes are defined as follows. A commitment scheme $\langle C, R\rangle$ is *extractable* if there exists an expected polynomial-time extractor E such that for any PPT committer C^*, the extractor E^{C^*} outputs a pair (τ, σ) that satisfies the following properties.

- τ is identically distributed with the view of C^* that interacts with an honest receiver R in the commit phase of $\langle C, R\rangle$. Let c_τ be the commitment that C^* gives in τ.
- If c_τ is accepting, then $\sigma \neq \perp$ except with negligible probability.
- If $\sigma \neq \perp$, then it is statistically impossible to decommit c_τ to any value other than σ.

There exists a four-round extractable commitment scheme ExtCom based on one-way functions [34], and it uses the underlying one-way function in a black-box way. Furthermore, ExtCom satisfies extractability in a stronger sense: It is extractable even against adversarial committers that give polynomially many ExtCom commitments *in parallel*. (The extractor outputs $(\tau, \sigma_1, \sigma_2, \ldots)$ for such committers.) ExtCom is shown in Fig. 2.

Strongly Extractable Commitment Schemes. We also use a stronger notion of extractability called *strong extractability*. Roughly speaking, an extractable commit-

Let Com be the two-round statistically binding commitment scheme that is mentioned in Section 3.2.

Commit Phase

The committer C and the receiver R take common input 1^n, and C additionally takes private input $v \in \{0,1\}^n$. To commit to v, the committer C does the following with the receiver R.

`commit stage`. C chooses a pair of random n-bit strings (a_0, a_1) such that $a_0 \oplus a_1 = v$. Then, C commits to a_0 and a_1 by using Com. For each $b \in \{0,1\}$, let c_b be the commitment to a_b.

`challenge stage`. The receiver R sends a random bit $e \in \{0,1\}$ to C.

`reply stage`. C decommits c_e to a_e.

Decommit Phase

C sends v to R and decommits c_0 and c_1 to a_0 and a_1. Then, R checks whether $a_0 \oplus a_1 = v$.

Fig. 3. Weakly extractable commitment scheme wExtCom [14].

ment scheme is strongly extractable if no over-extraction occurs during the extraction. Formally, a statistically binding commitment scheme $\langle C, R \rangle$ is *strongly extractable* if there exists an expected polynomial-time extractor E such that for any PPT committer C^*, the extractor E^{C^*} outputs a pair (τ, σ) that satisfies the following properties.

- τ is identically distributed with the view of C^* that interacts with an honest receiver R in the commit phase of $\langle C, R \rangle$. Let c_τ be the commitment that C^* gives in τ.
- If c_τ is invalid, then $\sigma = \bot$ except with negligible probability.
- If c_τ is valid, then it is statistically impossible to decommit c_τ to any value other that σ.

Weakly Extractable Commitment Schemes. We also use a weaker notion of extractability called *weak extractability*. A commitment scheme $\langle C, R \rangle$ is *weakly extractable* if there exists an expected polynomial-time extractor E such that for any PPT committer C^*, the extractor E^{C^*} outputs a pair (τ, σ) that satisfies the following properties.

- τ is identically distributed with the view of C^* that interacts with an honest receiver R in the commit phase of $\langle C, R \rangle$. Let c_τ be the commitment that C^* gives in τ.
- The probability that c_τ is accepting and $\sigma = \bot$ is at most $1/2$.
- If $\sigma \neq \bot$, then c_τ is valid and it is statistically impossible to decommit c_τ to any value other than σ.

There exists a four-round weakly extractable commitment scheme wExtCom based on one-way functions [14], and it uses the underlying one-way function in a black-box way. wExtCom is shown in Fig. 3. We note that given two accepted transcripts of wExtCom such that `commit stage` is identical but `challenge stage` is different, we can extract the committed value.

CECom is based on the extractable commitment scheme ExtCom in Figure 2, which consists of three stages—`commit`, `challenge`, and `reply`.

Commit Phase

The committer C and the receiver R take common input 1^n and parameter ℓ. (In [MOSV06], $\ell = \omega(\log n)$.) To commit to $v \in \{0,1\}^n$, the committer C commits to v concurrently ℓ times by using ExtCom as follows.

1. C and R execute `commit` stage of ExtCom ℓ times in parallel.
2. C and R do the following for each $j \in [\ell]$ in sequence.
   (a) R sends the `challenge` message of ExtCom for the j-th session.
   (b) C sends the `reply` message of ExtCom for the j-th session.

Decommit Phase

C sends v to R and decommits all the ExtCom commitments.

Fig. 4. Concurrently extractable commitment CECom [27].

3.4. *Concurrently Extractable Commitment Schemes*

We next recall the notion of *concurrently extractable commitment schemes*. Roughly speaking, a commitment scheme is *concurrently extractable* if there exists a polynomial-time extractor such that for any adversarial committer that commits to polynomially many values concurrently, the extractor can extract the committed values of all the valid commitments from the committer.

There exists a $\widetilde{O}(\log n)$-round concurrently extractable commitment CECom based on one-way functions [27], and it uses the underlying one-way function in a black-box way. CECom is an abstraction of the preamble stage of the concurrent zero-knowledge protocol of Prabhakaran et al. [31], and the extractor of CECom performs the extraction by rewinding the adversarial committer according to the carefully designed rewinding strategy of [31,33]. CECom is described in Fig. 4. We remark that CECom has a parameter ℓ, which is the number of ExtCom commitments that are generated in a CECom commitment. (In [27], $\ell = \omega(\log n)$.)

3.4.1. *Robust Concurrent Extraction Lemma [15]*

On the concurrently extractable commitment scheme CECom of Micciancio et al. [27], we will use the *robust concurrent extraction lemma*, which is a useful lemma shown by Goyal et al. [15]. Roughly speaking, the robust concurrent extraction lemma states that when the adversarial committer additionally participates in an external protocol, the values that are committed to by the adversarial committer can be extracted *without rewinding the external protocol*. More precisely, consider any PPT adversarial committer $\mathcal{A}$ that commits to multiple values in concurrent sessions of CECom—these sessions are denoted as the *right sessions*—and simultaneously participates in an execution of an arbitrary protocol $\Pi := \langle B, A \rangle$ with an honest B—this session is denoted as the *left session*. The robust concurrent extraction lemma states that for every $\mathcal{A}$, there exists an

extractor E that extracts the committed values from $\mathcal{A}$ in every valid right session without rewinding the external party B in the left session. The extractor E fails with probability that is exponentially small in $\ell - O(k \log n)$, where ℓ is the parameter of CECom and k is the round complexity of Π. Hence, E fails only with negligible probability if we set $\ell := \omega(k \log n)$.

A formal description of the robust concurrent extraction lemma is given below. (Large parts of the text below are taken from [15].)

The external protocol Π. Let $\Pi := \langle B, A \rangle$ be an arbitrary two-party protocol. Let $\mathrm{dom}_B(n)$ denote the domain of the input for B and $k := k(n)$ denote the round complexity of Π.

The robust-concurrent attack. Let $x \in \mathrm{dom}_B(n)$. In the *robust-concurrent attack*, the adversary $\mathcal{A}$ interacts with a special (possibly super-polynomial-time) party $\mathcal{E}$ called the *online extractor*. The online extractor $\mathcal{E}$ simultaneously participates in one execution of Π and several executions of CECom, where $\mathcal{E}$ interacts with $\mathcal{A}$ as an honest $B(1^n, x)$ in the execution of Π and interacts with $\mathcal{A}$ as an honest receiver in each execution of CECom. The scheduling of all messages in all sessions—Π as well as CECom—is controlled by $\mathcal{A}$. When $\mathcal{A}$ successfully completes a CECom commitment s, the online extractor $\mathcal{E}$ sends a value α_s to $\mathcal{A}$.

For $n \in \mathbb{N}, x \in \mathrm{dom}_B(n), z \in \{0,1\}^*$, let $\mathrm{REAL}^{\mathcal{A}}_{\mathcal{E},\Pi}(n, x, z)$ denote the following probabilistic experiment: On inputs 1^n, x, z, the experiment starts an execution of $\mathcal{A}(1^n, z)$, which launches the robust-concurrent attack by interacting with $\mathcal{E}(1^n, x, z)$; the output of the experiment is the view of $\mathcal{A}$ and the output of B (who was emulated by $\mathcal{E}$). Let $\mathsf{Real}^{\mathcal{A}}_{\mathcal{E},\Pi}(n, x, z)$ denote the output of $\mathrm{REAL}^{\mathcal{A}}_{\mathcal{E},\Pi}(n, x, z)$.

The robust concurrent extraction lemma. Roughly speaking, the lemma states that there exists an interactive Turing machine, called the *robust simulator*, that statistically simulates $\mathsf{Real}^{\mathcal{A}}_{\mathcal{E},\Pi}(n, x, z)$ even if the value that the online extractor $\mathcal{E}$ returns to $\mathcal{A}$ at the end of each successful CECom commitment is the committed value of this commitment. Furthermore, the robust simulator does not "rewind" B and runs in time polynomial in the number of the sessions opened by $\mathcal{A}$. A formal statement of the lemma is given below.

Lemma 2. *(*Robust Concurrent Extraction Lemma [15]*) There exists an interactive Turing machine $\mathcal{S}$ called a **robust simulator** such that for every adversary $\mathcal{A}$ and every two-party protocol $\Pi := \langle B, A \rangle$, there exists a party $\mathcal{E}$ called an **online extractor** such that for every $n \in \mathbb{N}$, $x \in dom_B(n)$, and $z \in \{0,1\}^*$, the following conditions hold:*

1. ***Validity constraint.*** *For every view ρ of $\mathcal{A}$ in $\mathsf{Real}^{\mathcal{A}}_{\mathcal{E},\Pi}(n, x, z)$ and for every CECom commitment s appearing in ρ, if there exists a unique value $v \in \{0,1\}^n$ to which the commitment s can be decommitted, then*

$$\alpha_s = v,$$

   where α_s is the value that $\mathcal{E}$ sends to $\mathcal{A}$ at the end of s.
2. ***Statistical simulation.*** *Let $k = k(n)$ be the round complexity of Π. Then the statistical distance between $\mathsf{Real}^{\mathcal{A}}_{\mathcal{E},\Pi}(n, x, z)$ and $\mathsf{output}_{B,\mathcal{S}}\left[B(1^n, x) \leftrightarrow \mathcal{S}^{\mathcal{A}}(1^n, z)\right]$ is given by*

$$\Delta(n) \leq 2^{-\Omega(\ell - k \cdot \log T(n))},$$

where $\mathsf{output}_{B,\mathcal{S}}\left[B(1^n, x) \leftrightarrow \mathcal{S}^{\mathcal{A}}(1^n, z)\right]$ *denotes the joint outputs of* $B(1^n, x)$ *and* $\mathcal{S}(1^n, z)$ *after an interaction between them,* $\ell := \ell(n)$ *is the parameter of* CECom*, and* $T(n)$ *is the number of the* CECom *commitments between* $\mathcal{A}$ *and* $\mathcal{E}$*. Furthermore, the running time of* $\mathcal{S}$ *is* $\mathsf{poly}(n) \cdot T(n)^2$.

3.5. *Trapdoor Commitment Schemes*

We next recall *trapdoor commitment schemes* [34]. Roughly speaking, trapdoor commitment schemes are commitment schemes such that there exists a simulator that can generate a simulated commitment and can later decommit it to any value. Pass and Wee [34] showed that the black-box scheme TrapCom in Fig. 5 is a trapdoor bit commitment. TrapCom is not statistically binding, but it satisfies the strong computational binding property. (The strong computational binding property holds since if an adversarial committer C^* generates a TrapCom commitment that can be decommitted to both 0 and 1, we can break the hiding property of Com using C^* by extracting the committed values of the ExtCom commitments from C^* and then computing the committed value e of Com from them.) Pass and Wee also showed that by running TrapCom in parallel, we can obtain a black-box trapdoor commitment scheme PTrapCom for multiple bits. PTrapCom also satisfies the strong computational binding property.

Commit Phase

To commit to $\sigma \in \{0,1\}$ on common input 1^n, the committer C does the following with the receiver R:

Step 1. R chooses a random n-bit string $e = (e_1, \ldots, e_n)$ and commits to e by using Com.

Step 2. For each $i \in [n]$, the committer C chooses a random $\eta_i \in \{0,1\}$ and then sets

$$v_i := \begin{pmatrix} v_i^{00} & v_i^{01} \\ v_i^{10} & v_i^{11} \end{pmatrix} = \begin{pmatrix} \eta_i & \eta_i \\ \sigma \oplus \eta_i & \sigma \oplus \eta_i \end{pmatrix} .$$

Then, for each $i \in [n]$, $\alpha \in \{0,1\}$, and $\beta \in \{0,1\}$ in parallel, C commits to $v_i^{\alpha\beta}$ by using ExtCom; let $(v_i^{\alpha\beta}, d_i^{\alpha\beta})$ be the corresponding decommitment.

Step 3. R decommits the commitment in Step 1 to e.

Step 4. For each $i \in [n]$, C sends $(v_i^{e_i 0}, d_i^{e_i 0})$ and $(v_i^{e_i 1}, d_i^{e_i 1})$ to R. Then, R checks whether these are valid decommitments and whether $v_i^{e_i 0} = v_i^{e_i 1}$.

Decommit Phase

C sends σ and random $\gamma \in \{0,1\}$ to R. In addition, for every $i \in [n]$, C sends $(v_i^{0\gamma}, d_i^{0\gamma})$ and $(v_i^{1\gamma}, d_i^{1\gamma})$ to R. Then, R checks whether $(v_i^{0\gamma}, d_i^{0\gamma})$ and $(v_i^{1\gamma}, d_i^{1\gamma})$ are valid decommitments and whether $v_0^{0\gamma} \oplus v_0^{1\gamma} = \cdots = v_n^{0\gamma} \oplus v_n^{1\gamma} = \sigma$.

Fig. 5. Black-box trapdoor bit commitment scheme TrapCom.

3.6. *CCA-Secure Commitment Schemes*

We next recall the definitions of CCA-secure commitment schemes and their κ-robustness [9,10,24].

CCA Security (w.r.t. the Committed-Value Oracle)

Roughly speaking, a tag-based commitment scheme (i.e., a commitment scheme that takes an n-bit string, a *tag*, as an additional input) $\langle C, R\rangle$ is *CCA-secure* if its hiding property holds even against any adversary $\mathcal{A}$ that interacts with the *committed-value oracle* during the interaction with the committer. The committed-value oracle $\mathcal{O}$ interacts with $\mathcal{A}$ as an honest receiver in many concurrent sessions of the commit phase of $\langle C, R\rangle$ using tags chosen adaptively by $\mathcal{A}$. At the end of each session, if the commitment of this session is invalid or has multiple committed values, $\mathcal{O}$ returns $\perp$ to $\mathcal{A}$. Otherwise, $\mathcal{O}$ returns the unique committed value to $\mathcal{A}$.

More precisely, let us consider the following probabilistic experiment $\mathrm{IND}_b(\langle C, R\rangle, \mathcal{A}, n, z)$ for each $b \in \{0,1\}$. On input 1^n and auxiliary input z, the adversary $\mathcal{A}^{\mathcal{O}}$ adaptively chooses a pair of challenge values $v_0, v_1 \in \{0,1\}^n$ and an n-bit tag $\mathsf{tag} \in \{0,1\}^n$. Then, $\mathcal{A}^{\mathcal{O}}$ receives a commitment to v_b with tag tag from the challenger. Let y be the output of $\mathcal{A}$. The output of the experiment is $\perp$ if during the experiment, $\mathcal{A}$ sends $\mathcal{O}$ any commitment using tag tag. Otherwise, the output of the experiment is y. Let $\mathsf{IND}_b(\langle C, R\rangle, \mathcal{A}, n, z)$ denote the output of experiment $\mathrm{IND}_b(\langle C, R\rangle, \mathcal{A}, n, z)$.

Definition 4. Let $\langle C, R\rangle$ be a tag-based commitment scheme and $\mathcal{O}$ be the committed-value oracle of $\langle C, R\rangle$. Then, $\langle C, R\rangle$ is **CCA-secure (w.r.t the committed-value oracle)** if for any PPT adversary $\mathcal{A}$, the following are computationally indistinguishable:

- $\{\mathsf{IND}_0(\langle C, R\rangle, \mathcal{A}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$
- $\{\mathsf{IND}_1(\langle C, R\rangle, \mathcal{A}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$

The **left session** is the session between the challenger and $\mathcal{A}$, and **right sessions** are the sessions between $\mathcal{A}$ and $\mathcal{O}$. □

We say a commitment scheme is *one-one CCA-secure* if it is CCA secure w.r.t. a restricted class of adversaries that start only a single right session.

κ-Robustness (w.r.t. the Committed-Value Oracle)

Roughly speaking, a tag-based commitment scheme is *κ-robust* if for any adversary $\mathcal{A}$ and any ITM B, a PPT simulator can simulate the joint output of a κ-round interaction between $\mathcal{A}^{\mathcal{O}}$ and B. Thus, the κ-robustness guarantees that the committed-value oracle is useless for attacking any κ-round protocol.

Definition 5. Let $\langle C, R\rangle$ be a tag-based commitment scheme and $\mathcal{O}$ be the committed-value oracle of $\langle C, R\rangle$. For any constant $\kappa \in \mathbb{N}$, we say that $\langle C, R\rangle$ is **κ-robust** (w.r.t. **the committed-value oracle)** if for any PPT adversary $\mathcal{A}$, there exists a PPT machine $\mathcal{S}$ called a **simulator** such that for any κ-round PPT ITM B, the following are computationally indistinguishable:

- $\left\{\mathsf{output}_{B,\mathcal{A}^{\mathcal{O}}}\left[B(1^n, y) \leftrightarrow \mathcal{A}^{\mathcal{O}}(1^n, z)\right]\right\}_{n\in\mathbb{N},y,z\in\{0,1\}^n}$
- $\left\{\mathsf{output}_{B,\mathcal{S}}\left[B(1^n, y) \leftrightarrow \mathcal{S}(1^n, z)\right]\right\}_{n\in\mathbb{N},y,z\in\{0,1\}^n}$

Here, for any ITMs A and B, we use $\mathsf{output}_{A,B}\left[A(1^n, y) \leftrightarrow B(1^n, z)\right]$ to denote the joint output of A and B in an interaction between them on inputs $(1^n, y)$ to A and $(1^n, z)$ to B, respectively. If $\langle C, R\rangle$ is κ-robust for any constant κ, we say that $\langle C, R\rangle$ is **robust**. □

4. Building Blocks

In this section, we construct a constant-round strongly extractable commitment scheme and a $O(\log n)$-round one-one CCA-secure commitment scheme. Both schemes are used in our $\widetilde{O}(\log^2 n)$-round CCA-secure commitment scheme in Sect. 5.

4.1. *Strongly Extractable Commitment Scheme*

Using one-way functions in a black-box way, we construct a constant-round strongly extractable commitment scheme sExtCom. Recall that a commitment scheme is strongly extractable if a rewinding extractor outputs a correct committed value when the commitment is valid and outputs $\perp$ when the commitment is invalid.

Lemma 3. *Assume the existence of one-way functions. Then, there exists a constant-round strongly extractable commitment scheme* sExtCom *that uses the underlying one-way function only in a black-box way.*

Proof. The scheme sExtCom is shown in Fig. 6, in which we use the following tools (all of which can be constructed from one-way functions in a black-box way).

- A two-round statistically binding commitment scheme Com. (See Sect. 3.2.)
- A constant-round extractable commitment scheme ExtCom. (See Sect. 3.3.)
- The constant-round weakly extractable commitment scheme wExtCom of Goyal et al. [14]. (See Sect. 3.3.)

We prove the binding property and the hiding property in Sect. 4.1.1 and the strong extractability in Sect. 4.1.2.

4.1.1. *Proofs of Binding and Hiding*

First, we show that sExtCom is statistically binding and computationally hiding. The binding property follows directly from that of ExtCom. To show the hiding property, we consider the following hybrid experiments for any PPT cheating receiver R^* and each $b \in \{0, 1\}$.

> **Hybrid** $H_0^b(n, z)$ is an experiment in which R^* takes input 1^n and auxiliary input z and receives a sExtCom commitment to σ_b from an honest committer, where (σ_0, σ_1) is the challenge values that R^* chooses at the beginning. The output of $H_0^b(n, z)$ is that of R^*.

Commit Phase

The committer C and the receiver R take common input 1^n, and C additionally takes private input $\sigma \in \{0,1\}^n$. To commit to σ, the committer C does the following with the receiver R.

Step 1. R commits to a random subset $\Gamma \subset [10n]$ of size n by using Com.

Step 2. C computes an $(n+1)$-out-of-$10n$ Shamir's secret sharing $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of value σ. Then, for each $j \in [10n]$ in parallel, C commits to s_j by using ExtCom. Let $\phi_1, \ldots, \phi_{10n}$ be the commitments and $d_1, \ldots, d_{10n}$ be the decommitments.

Step 3. For each $j \in [10n]$ in parallel, C commits to (s_j, d_j) by using $\mathsf{wExtCom}$. Let $\psi_0, \ldots, \psi_{10n}$ be the commitments.

Step 4. R decommits the commitment in Step 1 to Γ.

Step 5. C decommits the j-th $\mathsf{wExtCom}$ commitment ψ_j in Step 3 to (s_j, d_j) for each $j \in \Gamma$. R checks whether (s_j, d_j) is a valid decommitment of the j-th ExtCom commitment ϕ_j in Step 2 for every $j \in \Gamma$.

Decommit Phase

- C sends σ and decommits the ExtCom commitments $\phi_1, \ldots \phi_{10n}$ in Step 2 to $s_1, \ldots, s_{10n}$.
- R accepts if and only if the following holds w.r.t. $\boldsymbol{s} = (s_1, \ldots, s_{10n})$, where s_j is defined to be $\perp$ if the decommitment of ϕ_j is invalid.
  - $\boldsymbol{s}$ is 0.9-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ that satisfies $w_j = s_j$ for every $j \in \Gamma$, and $\boldsymbol{w}$ is a codeword of σ.

Fig. 6. Strongly extractable commitment scheme $\mathsf{sExtCom}$.

Hybrid $H_1^b(n, z)$ is the same as $H_0^b(n, z)$ except that the $\mathsf{sExtCom}$ commitment from the committer is modified as follows.

- In Step 1, the committed value Γ is extracted by brute force.
- In Step 2, the committer commits to $0^{|s_j|}$ instead of s_j for every $j \notin \Gamma$.
- In Step 3, the committer commits to $(0^{|s_j|}, 0^{|d_j|})$ instead of (s_j, d_j) for every $j \notin \Gamma$.

Let $\mathsf{H}_i^b(n, z)$ be the random variable representing the output of $H_i^b(n, z)$ for $i \in \{0, 1\}$ and $b \in \{0, 1\}$. From the construction, R^* receives no information about b in $H_1^b(n, z)$ for each $b \in \{0, 1\}$, so the distributions of $\mathsf{H}_1^0(n, z)$ and $\mathsf{H}_1^1(n, z)$ are identical. Hence, from a hybrid argument, we can show the hiding property by showing that $\mathsf{H}_0^b(n, z)$ and $\mathsf{H}_1^b(n, z)$ are indistinguishable for each $b \in \{0, 1\}$. Assume for contradiction that there exists $b \in \{0, 1\}$ such that for infinitely many n, there exists $z \in \{0, 1\}^*$ such that $\mathsf{H}_0^b(n, z)$ and $\mathsf{H}_1^b(n, z)$ are distinguishable with advantage $1/\mathsf{poly}(n)$. Fix any such b, n, and z. From an average argument, there exists a transcript ρ of Step 1 such that under the condition that the transcript of Step 1 is ρ, $\mathsf{H}_0^b(n, z)$ and $\mathsf{H}_1^b(n, z)$ are distinguishable with advantage $1/\mathsf{poly}(n)$. Let Γ be the subset that is committed to in ρ. Since we can execute $H_1^b(n, z)$ from ρ in polynomial time given ρ and Γ, by using a standard technique, we can break the hiding property of either ExtCom or $\mathsf{wExtCom}$ by using ρ and Γ as auxiliary input. Thus, we reach a contradiction.

4.1.2. *Proof of Strong Extractability*

Next, we show that sExtCom is strongly extractable. That is, we show that an extractor extracts a correct committed value from a valid sExtCom commitment and extracts $\perp$ from an invalid one except with negligible probability.

We first remark that from the construction of the decommit phase of sExtCom, the committed value of sExtCom is defined as follows.

Definition 6. (*Committed value of* sExtCom) If the shares $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ that are committed to in Step 2 are 0.9-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ that satisfies $w_j = s_j$ for every $j \in \Gamma$, the committed value of a sExtCom commitment is $\mathsf{Decode}(\boldsymbol{w})$. Otherwise, the committed value is $\perp$ (i.e., the commitment is invalid). □

We notice that the function $\mathsf{Value}_\Gamma(\cdot)$ in Fig. 1 (Sect. 3.1) computes the committed value of a sExtCom commitment as above on input the shares $\boldsymbol{s}$ that are committed to in Step 2.

Our extractor E extracts the committed value of a sExtCom commitment by extracting the committed values of the ExtCom commitments in Step 2. Formally, for any PPT cheating committer C^*, the extractor E does the following.

- E internally invokes C^* and interacts with C^* as a receiver honestly except that E extracts the committed values of the ExtCom commitments in Step 2 by using their extractability. Let τ be the view of internal C^*. If the sExtCom commitment in τ is rejecting or E fails to extract the committed values of the ExtCom commitments in Step 2, E sets $\widetilde{\sigma} := \perp$. Otherwise, E sets $\widetilde{\sigma} := \mathsf{Value}_\Gamma(\widetilde{\boldsymbol{s}})$, where $\widetilde{\boldsymbol{s}}$ is the shares that are extracted from the ExtCom commitments and Γ is the subset that is committed to in Step 1. E then outputs $(\tau, \widetilde{\sigma})$.

From the extractability of ExtCom, the simulated view τ is identically distributed with the real view. Hence, it remains to show that $\widetilde{\sigma}$ is a committed value of τ except with negligible probability.

Fix any PPT cheating committer C^*. Without loss of generality, we assume that C^* is deterministic.

First, we show that the extracted value $\widetilde{\sigma}$ is indeed equal to a committed value of the simulated view τ as long as the ExtCom commitments in Step 2 in τ are "good."

Definition 7. (*Good* ExtCom *commitments in Step 2*) In a sExtCom commitment, we say that the ExtCom commitments in Step 2 are **good** if all of the following conditions hold.

- Their committed values $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ are uniquely determined. (That is, none of them has more than one committed value.)
- $\left|\left\{j \in [10n] \text{ s.t. } s_j = \perp\right\}\right| < 0.5n$. (That is, less than $0.5n$ of them are invalid.)
- $\boldsymbol{s}$ is either 0.9-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ that satisfies $w_j = s_j$ for every $j \in \Gamma$ or 0.2-far from any such valid codeword. □

Claim 1. *Assume that in the interaction between C^* and an honest receiver, the probability that the* sExtCom *commitment from C^* is accepting but the* ExtCom *commitments*

in Step 2 are not good is negligible. Then, in the execution of E, the extracted value $\widetilde{\sigma}$ is a correct committed value of the sExtCom *commitment in τ except with negligible probability.*

Proof. When the sExtCom commitment in τ is rejecting, E sets $\widetilde{\sigma} := \bot$, which is a correct committed value of this sExtCom commitment. Hence, it remains to show that the probability that the sExtCom commitment in τ is accepting but $\widetilde{\sigma}$ is not its committed value is negligible.

Let BAD be the event that in the execution of E, the sExtCom commitment in the simulated view τ is accepting but the extracted value $\widetilde{\sigma} = \mathsf{Value}_{\Gamma}(\widetilde{\boldsymbol{s}})$ is not a committed value of it. Our goal is to show that BAD occurs only with negligible probability. Since the simulated view τ is identically distributed with the real view of C^*, from our assumption the probability that the sExtCom commitment in τ is accepting but the ExtCom commitments in Step 2 of it are not good is negligible. Hence, it suffices to show that under the condition that those ExtCom commitments are good, BAD occurs only with negligible probability. Furthermore, since the extraction from ExtCom succeeds except with negligible probability, and the values extracted from valid ExtCom commitments are the correct committed values except with negligible probability, it suffices to show that under the conditions that in the sExtCom commitment in τ,

- the ExtCom commitments in Step 2 are good, and
- the (unique) committed value of each valid ExtCom commitment is correctly extracted,

BAD occurs only with negligible probability. Then, we notice that under the above conditions, we have the following when the sExtCom commitment in τ is accepting.

1. For every $j \in [10n]$, if $s_j \neq \bot$, it holds $s_j = \widetilde{s}_j$.
   (This is because of the assumption that the correct committed value is extracted from every valid ExtCom commitment.)
2. $\left|\left\{j \text{ s.t. } s_j = \bot\right\}\right| < 0.5n \bigwedge \left\{j \text{ s.t. } s_j = \bot\right\} \cap \Gamma = \emptyset$.
   (This is because the sExtCom commitment would be rejected in Step 5 if $\{j \text{ s.t. } s_j = \bot\} \cap \Gamma \neq \emptyset$.)
3. $\boldsymbol{s}$ is either 0.9-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ that satisfies $w_j = s_j$ for every $j \in \Gamma$ or 0.2-far from any such valid codeword.

Hence, using Lemma 1 in Sect. 3.1, we conclude that under the above conditions, we have $\mathsf{Value}_{\Gamma}(\widetilde{\boldsymbol{s}}) = \mathsf{Value}_{\Gamma}(\boldsymbol{s})$ (i.e., $\mathsf{Value}_{\Gamma}(\widetilde{\boldsymbol{s}})$ is equal to the committed value) when the sExtCom commitment in τ is accepting. Thus, BAD never occurs under the above conditions. This completes the proof of Claim 1. □

It remains to show that in the interaction between C^* and an honest receiver, the probability that the sExtCom commitment from C^* is accepting but the ExtCom commitments in Step 2 are not good is negligible. Recall that the ExtCom commitments are good if their committed values $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ are uniquely determined, at least $9.5n$ of them are valid, and $\boldsymbol{s}$ is either 0.9-close to a valid codeword $\boldsymbol{w}$ that satisfies $w_j = s_j$ for every $j \in \Gamma$ or 0.2-far from any such codewords. We show the following two claims.

Claim 2. *In the interaction between C^* and an honest receiver, the probability that the* sExtCom *commitment from C^* is accepting but at least $0.5n$* ExtCom *commitments in Step 2 are invalid is negligible.*

Claim 3. *In the interaction with C^* and an honest receiver, the probability that the* sExtCom *commitment from C^* is accepting but either of the following conditions does not hold is negligible.*

- The committed values $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of the ExtCom commitments are uniquely determined.
- $\boldsymbol{s}$ is either 0.9-close to a valid codeword $\boldsymbol{w}$ that satisfies $w_j = s_j$ holds for every $j \in \Gamma$ or 0.2-far from any such codewords.

Proof of Claim 2

In this proof, we use the following notations. For $j \in [10n]$, the *j-th column* is the pair of the jth ExtCom commitment in Step 2 and the jth wExtCom commitment in Step 3. A column is *consistent* if the committed value of the wExtCom commitment is a valid decommitment of the ExtCom commitment in that column; otherwise, the column is *inconsistent*. C^* *cheats* if all of the following conditions hold: every wExtCom commitment is accepting, the jth column is consistent for every $j \in \Gamma$, and at least $0.5n$ columns are inconsistent.

In the following, we show that C^* cheats only with negligible probability. This suffices to prove the claim because from the definition of the cheating, C^* cheats whenever the sExtCom commitment from C^* is accepting but at least $0.5n$ ExtCom commitments in Step 2 are invalid.

Assume for contradiction that there exists a constant c such that C^* cheats with probability at least $1/n^c$ for infinitely many n. Fix any such c and n.

We derive a contradiction by constructing an adversary $\mathcal{B}$ that breaks the hiding property of Com. For random subsets $\Gamma_0, \Gamma_1 \subset [10n]$ of size n, $\mathcal{B}$ tries to distinguish a Com commitment to Γ_0 from a Com commitment to Γ_1 as follows. $\mathcal{B}$ internally invokes C^* and interacts with it as a receiver of sExtCom honestly except for the following.

- In Step 1, $\mathcal{B}$ receives a Com commitment from the external committer (who commits to either Γ_0 or Γ_1) and forwards the commitment to C^* as the commitment in Step 1.
- If Step 3 is accepting (i.e., all of the wExtCom commitments are accepting), $\mathcal{B}$ does the following repeatedly: $\mathcal{B}$ rewinds C^* to the point just before $\mathcal{B}$ sends the challenge bits of the wExtCom commitments to C^*; then, $\mathcal{B}$ sends new random challenge bits to C^* and receives the replies from C^*. $\mathcal{B}$ repeats this rewinding until it obtains other n^{c+3} accepted transcripts of Step 3. If the number of the rewinding exceeds n^{3c+4}, $\mathcal{B}$ terminates and outputs fail. Otherwise, $\mathcal{B}$ outputs 1 if and only if all of the following conditions hold.
  1. From the $n^{c+3}+1$ accepted transcripts of Step 3 (the first one and the subsequent n^{c+3} ones), $\mathcal{B}$ can extract the committed values of the wExtCom commitments in at least $9.9n$ columns.

2. In at least $0.4n$ columns of these $9.9n$ columns, the extracted values are not valid decommitments of the ExtCom commitments.
3. For every $j \in \Gamma_1$, either the extraction from the jth column fails or the value extracted from the jth column is a valid decommitment of the ExtCom commitment of the jth column.

In the following, the first transcript that $\mathcal{B}$ generates in Step 3 is called the *main thread* and the other n^{c+3} accepted transcripts are called the *look-ahead threads*.

First, we analyze the adversary $\mathcal{B}'$ that is the same as $\mathcal{B}$ except that $\mathcal{B}'$ does not terminate even after rewinding C^* more than n^{3c+4} times. When $\mathcal{B}'$ receives a commitment to Γ_0, the internal C^* receives no information about Γ_1, so the probability that the extracted values are not valid decommitments of the ExtCom commitments in at least $0.4n$ columns but are valid decommitments in all the columns selected by Γ_1 is exponentially small. Hence, when $\mathcal{B}'$ receives a commitment to Γ_0, $\mathcal{B}'$ outputs 1 only with exponentially small probability. In the following, we show that when $\mathcal{B}'$ receives a commitment to Γ_1, $\mathcal{B}'$ outputs 1 with probability $1/\mathsf{poly}(n)$. Let CHEAT be the event that C^* cheats on the main thread, and $\mathsf{EXTRACT}$ be the event that $\mathcal{B}'$ succeeds in extracting the committed values of the $\mathsf{wExtCom}$ commitments from at least $9.9n$ columns. Since over-extraction never occurs in the extraction from $\mathsf{wExtCom}$, $\mathcal{B}'$ outputs 1 whenever CHEAT and $\mathsf{EXTRACT}$ occur. Hence, to show that $\mathcal{B}'$ outputs 1 with probability at least $1/\mathsf{poly}(n)$, it suffices to show that we have

$$\Pr\left[\mathsf{CHEAT} \wedge \mathsf{EXTRACT}\right] \geq \frac{1}{\mathsf{poly}(n)}. \tag{1}$$

For any prefix ρ of the transcript between C^* and an honest receiver up until the challenge bits of $\mathsf{wExtCom}$ (exclusive), let PREFIX_ρ be the event that ρ is a prefix of the main thread. Since C^* cheats with probability at least $1/n^c$, from an average argument we have $\Pr\left[\mathsf{CHEAT} \mid \mathsf{PREFIX}_\rho\right] \geq 1/2n^c$ with probability at least $1/2n^c$ over the choice of ρ (i.e., over the distribution of ρ in the interaction between C^* and an honest receiver). Let Δ be the set of prefixes with which $\Pr\left[\mathsf{CHEAT} \mid \mathsf{PREFIX}_\rho\right] \geq 1/2n^c$ holds. As noted above, we have $\sum_{\rho\in\Delta} \Pr\left[\mathsf{PREFIX}_\rho\right] \geq 1/2n^c$. Hence, we have

$$\begin{aligned}
\Pr\left[\mathsf{CHEAT} \wedge \mathsf{EXTRACT}\right] &\geq \sum_{\rho\in\Delta} \Pr\left[\mathsf{CHEAT} \wedge \mathsf{EXTRACT} \mid \mathsf{PREFIX}_\rho\right] \cdot \Pr\left[\mathsf{PREFIX}_\rho\right] \\
&\geq \min_{\rho\in\Delta}\left(\Pr\left[\mathsf{CHEAT} \wedge \mathsf{EXTRACT} \mid \mathsf{PREFIX}_\rho\right]\right) \cdot \sum_{\rho\in\Delta} \Pr\left[\mathsf{PREFIX}_\rho\right] \\
&\geq \frac{1}{2n^c} \min_{\rho\in\Delta}\left(\Pr\left[\mathsf{CHEAT} \wedge \mathsf{EXTRACT} \mid \mathsf{PREFIX}_\rho\right]\right). \qquad (2)
\end{aligned}$$

Thus, to show Eq. (1), it suffices to show that for any $\rho \in \Delta$, we have

$$\Pr\left[\mathsf{CHEAT} \wedge \mathsf{EXTRACT} \mid \mathsf{PREFIX}_\rho\right] \geq \frac{1}{\mathsf{poly}(n)}. \tag{3}$$

Fix any $\rho^* \in \Delta$. From the definition of Δ, we have

$$\Pr\left[\mathsf{CHEAT} \mid \mathsf{PREFIX}_{\rho^*}\right] \geq \frac{1}{2n^c}. \tag{4}$$

Thus, we have

$$\begin{aligned}\Pr\left[\mathsf{CHEAT} \wedge \mathsf{EXTRACT} \mid \mathsf{PREFIX}_{\rho^*}\right] &= \Pr\left[\mathsf{CHEAT} \mid \mathsf{PREFIX}_{\rho^*}\right] \\ &\quad \cdot \Pr\left[\mathsf{EXTRACT} \mid \mathsf{PREFIX}_{\rho^*} \wedge \mathsf{CHEAT}\right] \\ &\geq \frac{1}{2n^c} \Pr\left[\mathsf{EXTRACT} \mid \mathsf{PREFIX}_{\rho^*} \wedge \mathsf{CHEAT}\right] \end{aligned} \tag{5}$$

Thus, to show Eq. (3), it suffices to show that

$$\Pr\left[\mathsf{EXTRACT} \mid \mathsf{PREFIX}_{\rho^*} \wedge \mathsf{CHEAT}\right] \geq \frac{1}{\mathsf{poly}(n)}. \tag{6}$$

Recall that EXTRACT is the event that $\mathcal{B}'$ succeeds in extracting the committed values of the wExtCom commitments from at least $9.9n$ columns. From the construction of wExtCom, EXTRACT occurs if in at least $9.9n$ columns, the challenge bit of the wExtCom commitment on a look-ahead thread is different from the challenge bit on the main thread. Hence, to show Eq. (6), it suffices to show that in at least $9.9n$ columns, the probability that the challenge bit of wExtCom is b is "high" for both $b = 0$ and $b = 1$ on each look-ahead thread. Furthermore, since each look-ahead thread is generated by repeatedly executing the main thread from ρ^* until a new accepting transcript of Step 3 is obtained, it suffices to show that under the condition that PREFIX_{ρ^*} occurs and Step 3 is accepted, the probability that the challenge bit of the wExtCom commitment is b is "high" for both $b = 0$ and $b = 1$ in at least $9.9n$ columns. Based on these observations, we show the following subclaim.

Subclaim 1. *Let ch_j be the random variable representing the challenge bit of* wExtCom *in the jth column on the main thread, and let* ACCEPT *be the event that every* wExtCom *commitment is accepting on the main thread. Then, there exists a subset $\mathcal{J}_{\mathrm{good}} \subset [10n]$ such that:*

- $|\mathcal{J}_{\mathrm{good}}| \geq 9.9n$
- *For every $j \in \mathcal{J}_{\mathrm{good}}$ and $b \in \{0, 1\}$,*

$$\Pr\left[ch_j = b \mid \mathit{PREFIX}_{\rho^*} \wedge \mathit{ACCEPT}\right] \geq \frac{1}{40n^{c+1}}.$$

Proof. For any $j \in [10n]$ and $b \in \{0, 1\}$, we have

$$\begin{aligned}\Pr\left[ch_j = b \mid \mathsf{PREFIX}_{\rho^*} \wedge \mathsf{ACCEPT}\right] &= \frac{\Pr\left[\mathsf{ACCEPT} \wedge ch_j = b \;\middle|\; \mathsf{PREFIX}_{\rho^*}\right]}{\Pr\left[\mathsf{ACCEPT} \;\middle|\; \mathsf{PREFIX}_{\rho^*}\right]} \\ &\geq \Pr\left[\mathsf{ACCEPT} \wedge ch_j = b \;\middle|\; \mathsf{PREFIX}_{\rho^*}\right]. \end{aligned} \tag{7}$$

Hence, we show that in at least $9.9n$ columns, for any $b \in \{0,1\}$ we have

$$\Pr\left[\mathsf{ACCEPT} \wedge ch_j = b \;\middle|\; \mathsf{PREFIX}_{\rho^*}\right] \geq \frac{1}{40n^{c+1}}. \tag{8}$$

Let

$$\mathcal{J}_{\mathrm{bad}} \stackrel{\mathrm{def}}{=} \left\{ j \in [10n] \;\middle|\; \exists b_j^* \in \{0,1\} \text{ s.t. } \Pr\left[\mathsf{ACCEPT} \wedge ch_j = b_j^* \;\middle|\; \mathsf{PREFIX}_{\rho^*}\right] < \frac{1}{40n^{c+1}} \right\}.$$

We have

$$\begin{aligned}
\Pr\left[\mathsf{ACCEPT} \;\middle|\; \mathsf{PREFIX}_{\rho^*}\right] &\leq \Pr\left[\bigwedge_{j \in \mathcal{J}_{\mathrm{bad}}} ch_j = 1 - b_j^*\right] \\
&\quad + \Pr\left[\mathsf{ACCEPT} \bigwedge \left(\bigvee_{j \in \mathcal{J}_{\mathrm{bad}}} ch_j = b_j^*\right) \;\middle|\; \mathsf{PREFIX}_{\rho^*}\right] \\
&\leq 2^{-|\mathcal{J}_{\mathrm{bad}}|} + \sum_{j \in \mathcal{J}_{\mathrm{bad}}} \Pr\left[\mathsf{ACCEPT} \wedge ch_j = b_j^* \mid \mathsf{PREFIX}_{\rho^*}\right] \\
&< 2^{-|\mathcal{J}_{\mathrm{bad}}|} + 10n \cdot \frac{1}{40n^{c+1}} \\
&= 2^{-|\mathcal{J}_{\mathrm{bad}}|} + \frac{1}{4n^c}.
\end{aligned} \tag{9}$$

On the other hand, since ACCEPT occurs whenever CHEAT occurs, from Eq. (4) we have

$$\Pr\left[\mathsf{ACCEPT} \;\middle|\; \mathsf{PREFIX}_{\rho^*}\right] \geq \Pr\left[\mathsf{CHEAT} \;\middle|\; \mathsf{PREFIX}_{\rho^*}\right] \geq \frac{1}{2n^c}. \tag{10}$$

From Eqs. (9) and (10), we have $|\mathcal{J}_{\mathrm{bad}}| = O(\log n)$ and therefore $|\mathcal{J}_{\mathrm{bad}}| < 0.1n$. Thus, in at least $9.9n$ columns, we have Eq. (8) for any $b \in \{0,1\}$.

Define $\mathcal{J}_{\mathrm{good}} \stackrel{\mathrm{def}}{=} [10n] \setminus \mathcal{J}_{\mathrm{bad}}$. Since $|\mathcal{J}_{\mathrm{bad}}| < 0.1n$, we have $|\mathcal{J}_{\mathrm{good}}| \geq 9.9n$. Furthermore, from Eqs. (7) and (8), for any $j \in \mathcal{J}_{\mathrm{good}}$ and $b \in \{0,1\}$ we have

$$\Pr\left[ch_j = b \mid \mathsf{PREFIX}_{\rho^*} \wedge \mathsf{ACCEPT}\right] \geq \frac{1}{40n^{c+1}}.$$

This concludes the proof of Subclaim 1. □

As mentioned above, we can obtain Eq. (1) by using Subclaim 1. First, since the distribution of each look-ahead thread is the same as that of the main thread, Subclaim 1 implies that under the condition that PREFIX_{ρ^*} and CHEAT occur, $\mathcal{B}'$ requires $40n^{c+1}$ accepted transcripts of Step 3 on average to extract the committed value of wExtCom in the jth columns for any $j \in \mathcal{J}_{\mathrm{good}}$. Since $\mathcal{B}'$ collects n^{c+3} accepted transcripts, it

follows from Markov's inequality that for any $j \in \mathcal{J}_{\text{good}}$, $\mathcal{B}'$ extracts the committed value of $\mathsf{wExtCom}$ in the jth column except with probability $40n^{c+1}/n^{c+3} = 40/n^2$ under the condition that PREFIX_{ρ^*} and CHEAT occur. Thus, from the union bound, $\mathcal{B}'$ extracts the committed value of $\mathsf{wExtCom}$ in the jth column for every $j \in \mathcal{J}_{\text{good}}$ except with probability $9.9n \cdot 40/n^2 = 396/n$. We therefore have

$$\Pr\left[\mathsf{EXTRACT} \mid \mathsf{PREFIX}_{\rho^*} \wedge \mathsf{CHEAT}\right] \geq 1 - \frac{396}{n}. \tag{11}$$

Then, from Eqs. (5) and (11), we have

$$\Pr\left[\mathsf{CHEAT} \wedge \mathsf{EXTRACT} \mid \mathsf{PREFIX}_{\rho^*}\right] \geq \frac{1}{2n^c} \cdot \left(1 - \frac{396}{n}\right) \geq \frac{1}{4n^c}. \tag{12}$$

Since ρ^* is any prefix in Δ, from Eqs. (2) and (12) we have

$$\Pr\left[\mathsf{CHEAT} \wedge \mathsf{EXTRACT}\right] \geq \frac{1}{2n^c} \cdot \frac{1}{4n^c} = \frac{1}{8n^{2c}}.$$

Thus, we have Eq. (1). We therefore conclude that $\mathcal{B}'$ outputs 1 with probability at least $1/8n^{2c}$ when $\mathcal{B}'$ receives a commitment to Γ_1. Hence, $\mathcal{B}'$ distinguishes a commitment to Γ_1 from a commitment to Γ_0 with advantage $1/8n^{2c} - \mathsf{negl}(n)$.

Now, we are ready to show that $\mathcal{B}$ breaks the hiding property of Com. The running time of $\mathcal{B}$ is clearly at most $\mathsf{poly}(n)$. Hence, to show that $\mathcal{B}$ distinguishes a Com commitment, it suffices to show that the output of $\mathcal{B}$ is the same as that of $\mathcal{B}'$ except with probability $1/n^{2c+1}$. (This is because $\mathcal{B}'$ distinguishes a Com commitment with advantage $1/8n^{2c} - \mathsf{negl}(n)$.) Recall that the output of $\mathcal{B}$ differs from that of $\mathcal{B}'$ if and only if $\mathcal{B}'$ rewinds C^* more than n^{3c+4} times. Let $T(n)$ be a random variable for the number of rewinding in $\mathcal{B}'$. For any prefix ρ of the transcript between C^* and an honest receiver up until the challenge bits of $\mathsf{wExtCom}$ (exclusive), we have

$$\mathrm{E}\left[T(n) \mid \mathsf{PREFIX}_{\rho}\right] \leq \Pr\left[\mathsf{ACCEPT} \mid \mathsf{PREFIX}_{\rho}\right] \cdot \frac{n^{c+3}}{\Pr\left[\mathsf{ACCEPT} \mid \mathsf{PREFIX}_{\rho}\right]} = n^{c+3}.$$

Thus, we have

$$\begin{aligned}\mathrm{E}\left[T(n)\right] &= \sum_{\rho} \Pr\left[\mathsf{PREFIX}_{\rho}\right] \mathrm{E}\left[T(n) \mid \mathsf{PREFIX}_{\rho}\right] \\ &\leq n^{c+3} \sum_{\rho} \Pr\left[\mathsf{PREFIX}_{\rho}\right] \leq n^{c+3}.\end{aligned}$$

From Markov's inequality, $\mathcal{B}'$ rewinds C^* more than n^{3c+4} times with probability at most $n^{c+3}/n^{3c+4} = 1/n^{2c+1}$. Thus, the output of $\mathcal{B}$ is the same as that of $\mathcal{B}'$ except with probability $1/n^{2c+1}$, and therefore $\mathcal{B}$ distinguishes a commitment to Γ_1 from a commitment to Γ_0 with advantage at least $1/8n^{2c} - \mathsf{negl}(n) - 1/n^{2c+1} \geq 1/16n^{2c}$.

□

Proof of Claim 3

From the binding property of ExtCom, the committed values $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of the ExtCom commitments in Step 2 are uniquely determined except with negligible probability. Hence, to prove the claim, it suffices to show that the following holds in an accepting sExtCom commitment only with negligible probability.

- The committed values $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of the ExtCom commitments are uniquely determined, but
- $\boldsymbol{s}$ is 0.8-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ that satisfies $s_j = w_j$ for every $j \in \Gamma$, but $\boldsymbol{s}$ is 0.1-far from $\boldsymbol{w}$.

Assume for contradiction that for infinitely many n, the above hold in an accepting sExtCom commitment with probability at least $1/p(n)$ for a polynomial $p(\cdot)$. Then, from Claim 2, the following holds in an accepting sExtCom commitment with probability at least $1/2p(n)$ for infinitely many n.

- At least $9.5n$ of the ExtCom commitments are valid, and
- the committed values $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of the ExtCom commitments are uniquely determined, but
- $\boldsymbol{s}$ is 0.8-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ that satisfies $s_j = w_j$ for every $j \in \Gamma$, but $\boldsymbol{s}$ is 0.1-far from $\boldsymbol{w}$.

Fix any such n. We derive a contradiction by constructing an adversary $\mathcal{B}$ that breaks the hiding property of Com. For random subsets $\Gamma_0, \Gamma_1 \subset [10n]$ of size n, $\mathcal{B}$ tries to distinguish a Com commitment to Γ_0 from a Com commitment to Γ_1 as follows. $\mathcal{B}$ internally invokes C^* and interacts with it as a receiver of sExtCom honestly except for the following.

- In Step 1, $\mathcal{B}$ receives a Com commitment from the external committer (who commits to either Γ_0 or Γ_1) and forwards the commitment to C^* as the commitment in Step 1.
- In Step 2, the committed values are extracted by using the extractor of ExtCom. If the extractor runs more than $6p(n) \cdot T(n)$ steps, $\mathcal{B}$ terminates immediately with output fail, where $T(n) = \mathsf{poly}(n)$ is an expected running time of the extractor of ExtCom. Otherwise, let $\widetilde{\boldsymbol{s}} = (\widetilde{s}_1, \ldots, \widetilde{s}_{10n})$ be the extracted values.
- After Step 2 ends, $\mathcal{B}$ outputs 1 if there exists a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ such that $\widetilde{\boldsymbol{s}}$ is 0.8-close to but 0.05-far from $\boldsymbol{w}$ and that $\widetilde{s}_j = w_j$ holds for every $j \in \Gamma_1$. Otherwise, $\mathcal{B}$ outputs 0.

First, we analyze an adversary $\mathcal{B}'$ that is the same as $\mathcal{B}$ except that $\mathcal{B}'$ does not terminate even after the extractor of ExtCom runs more than $6p(n) \cdot T(n)$ steps. When $\mathcal{B}'$ receives a commitment to Γ_0, the internal C^* receives no information about Γ_1, so the probability that $\widetilde{\boldsymbol{s}}$ is 0.05-far from $\boldsymbol{w}$ but $\widetilde{s}_j = w_j$ holds for every $j \in \Gamma_1$ is exponentially small; thus, $\mathcal{B}'$ outputs 1 with exponentially small probability. We next compute the probability that $\mathcal{B}'$ outputs 1 when it receives a commitment to Γ_1. From our assumption, with probability $1/2p(n)$ it holds that $9.5n$ of the ExtCom commitments are valid and the unique committed values $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of the ExtCom commitments are 0.8-close to but 0.1-far from a valid codeword $\boldsymbol{w}$ that satisfies $s_j = w_j$ for every $j \in \Gamma_1$. Since the extractability of ExtCom guarantees that $\widetilde{s}_j = s_j$ holds except with

negligible probability when the jth ExtCom commitment is valid (and in particular when $s_j = w_j \neq \perp$), with probability at least $1/3p(n)$, $\widetilde{\boldsymbol{s}}$ is 0.8-close to but 0.05-far from a valid codeword $\boldsymbol{w}$ that satisfies $\widetilde{s}_j = w_j$ for every $j \in \Gamma_1$. Hence, when $\mathcal{B}'$ receives a commitment to Γ_1, $\mathcal{B}'$ outputs 1 with probability at least $1/3p(n)$. Therefore, $\mathcal{B}'$ distinguishes a Com commitment with advantage $1/3p(n) - \mathsf{negl}(n)$.

Now, we are ready to argue that $\mathcal{B}$ breaks the hiding property of Com. The output of $\mathcal{B}$ differs from that of $\mathcal{B}'$ if and only if the extraction from ExtCom takes more than $6p(n) \cdot T(n)$ steps. From Markov's inequality, the extraction from ExtCom takes more than $6p(n) \cdot T(n)$ steps only with probability $1/6p(n)$. Hence, $\mathcal{B}$ distinguishes a Com commitment with advantage $1/3p(n) - \mathsf{negl}(n) - 1/6p(n) \geq 1/6p(n)$. Since the running time of $\mathcal{B}$ can be bounded by $\mathsf{poly}(n)$, $\mathcal{B}$ breaks the hiding property of Com. □

Conclusion of Proof of Lemma 3

From Claims 2 and 3 , the probability that the ExtCom commitments are not good in an accepting $\mathsf{sExtCom}$ commitment is negligible. Hence, from Claim 1, the extractor E outputs a correct committed value except with negligible probability. This completes the proof of Lemma 3. □

4.2. *One-One CCA-Secure Commitment Scheme*

Using one-way functions in a black-box way, we construct a $O(\log n)$-round one-one CCA-secure commitment scheme $\mathsf{CCACom}^{1:1}$. Recall that a commitment scheme is one-one CCA secure if it is CCA secure w.r.t. a restricted class of adversaries that start only a single right session. Our scheme does not satisfy the statistically binding property but does satisfy the strong computational binding property.

Lemma 4. *Assume the existence of one-way functions. Then, there exists a $O(\log n)$-round one-one CCA-secure commitment scheme $\mathsf{CCACom}^{1:1}$ that satisfies the strong computational binding property and the computational hiding property. Furthermore, $\mathsf{CCACom}^{1:1}$ uses the underlying one-way function only in a black-box way.*

Proof. We construct $\mathsf{CCACom}^{1:1}$ by slightly modifying the black-box $O(n^\epsilon)$-round CCA-secure commitment scheme of Lin and Pass [24] and then applying the "DDN log n trick" [11,25] on it, where the DDN log n trick is a transformation by Dolev, Dwork, and Naor (DDN) [11] and has been used to transform concurrent non-malleable commitment schemes for tags of length $O(\log n)$ to non-malleable commitment schemes for tags of length $O(n)$ without increasing round complexity.

First, we recall the CCA-secure commitment scheme of [24] (see Figs. 7, 8). Roughly speaking, the commitment scheme of [24] consists of $4\ell(n)\eta(n)$ *rows*—each row is a parallel execution of a part of the trapdoor commitment scheme $\mathsf{PTrapCom}$ of [34] (see Sect. 3.5)—followed by a cut-and-choose phase, where $\ell(n)$ is the length of the tag and $\eta(n) \stackrel{\text{def}}{=} n^\epsilon$ for $\epsilon > 0$. In the analysis of [24], which is based on that of [9,10], it is shown that in any transcript of one left session and many right sessions of the scheme, each right session has $\Omega(\eta(n))$ $\mathsf{safe\text{-}points}$, from which we can rewind the right session and extract its committed value without breaking the hiding property of the left session.

Commit Phase

Let ℓ, η be two polynomials such that $\ell(n) = n^{\nu}$ and $\eta(n) = n^{\epsilon}$ for $\nu, \epsilon > 0$, and L be a polynomial such that $L(n) = 4\ell(n)\eta(n)$. To commit to a value v, the committer C and the receiver R, on common input 1^n and $\mathsf{tag} \in \{0,1\}^{\ell(n)}$, do the following.

Stage 1: R sends the Step 1 message of a commitment of PTrapCom. That is, a commitment of Com to a randomly chosen string challenge $\boldsymbol{e} = (e_1, \ldots, e_n)$.

Stage 2: C computes an $(n+1)$-out-of-$10n$ Shamir's secret sharing $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of value v, and commits to these shares using Step 2 of PTrapCom in parallel for $L(n)$ times; we call the i-th parallel commitment the i-th *row*, and all the commitments to s_j the j-th *column*. Messages in the $4\ell(n)\eta(n)$ rows are scheduled based on tag and the schedules design_0 and design_1 depicted in Fig. 8. More precisely, Stage 2 consists of $\ell(n)$ phases. In phase i, C provides $\eta(n)$ sequential $\mathsf{design}_{\mathsf{tag}_i}$ pairs of rows, followed by $\eta(n)$ sequential $\mathsf{design}_{1-\mathsf{tag}_i}$ pairs of rows.

Stage 3: R decommits the commitment in Stage 1 to $\boldsymbol{e}$. C completes the $10nL(n)$ executions of PTrapCom w.r.t. challenge $\boldsymbol{e}$ in parallel.

Stage 4: R sends a randomly chosen subset $\Gamma \subset [10n]$ of size n. For every $j \in \Gamma$, C decommits all the commitments in the j-th column of Stage 3. R checks whether all the decommitments are valid and reveal the same committed values s_j.

Fig. 7. Black-box CCA-secure commitment scheme of [24].

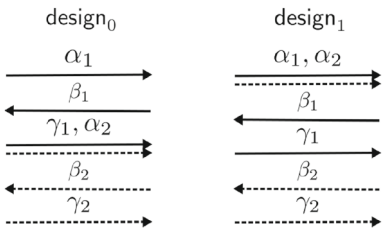

Fig. 8. Description of the schedules used in Stage 2 of the protocol of [24]. $(\alpha_1, \beta_1, \gamma_1)$ and $(\alpha_2, \beta_2, \gamma_2)$ are the transcripts of a pair of rows in Stage 2.

Then, since each right session has $\Omega(\eta(n))$ safe-points, we can extract the committed value of each right session even in the concurrent setting by using the rewinding strategy of Richardson and Kilian [35] to deal with the problem of recursive rewinding. Thus, by extracting the committed value of a row in each right session, we can emulate the committed-value oracle in polynomial time without breaking the hiding property of the left session. Thus, the CCA security follows from the hiding property of the left session.

Next, we observe that by setting $\eta(n) := 1$ in the scheme of [24], we obtain a black-box $O(\ell(n))$-round *parallel CCA-secure* commitment scheme for tags of length $\ell(n)$, where a commitment scheme is *parallel CCA secure* if it is CCA secure w.r.t. a restricted class of adversaries that start only a single parallel right session. This is because when an adversary starts only a single parallel right session, the problem of

recursive rewinding does not occur, so each right session need to have only a single safe-point as in the concurrent non-malleable commitment scheme of [25] (on which the CCA-secure commitment schemes of [9,10,24] are based). Therefore, by setting $\eta(n) := 1$ and $\ell(n) := O(\log n)$, we obtain a black-box $O(\log n)$-round commitment scheme that is parallel CCA secure for tags of length $O(\log n)$.

We then observe that the DDN $\log n$ trick [11,25] transforms any black-box parallel CCA-secure commitment scheme for tags of length $O(\log n)$ to a black-box one-one CCA-secure commitment scheme for tags of length $O(n)$. This can be proven in essentially the same way as the proof of the fact that the DDN $\log n$ trick transforms a concurrent non-malleable commitment scheme for tags of length $O(\log n)$ to a non-malleable commitment scheme for tags of length $O(n)$. For details, see Appendix A.

Combining the above, we obtain a black-box $O(\log n)$-round one-one CCA-secure commitment scheme $\mathsf{CCACom}^{1:1}$. $\mathsf{CCACom}^{1:1}$ satisfies the strong computational binding property and the computational hiding property because the CCA-secure commitment scheme of [24] satisfies both properties and the DDN $\log n$ trick preserves both properties. (The strong computational binding property of [24] follows from that of the trapdoor commitment scheme of [34].) □

5. CCA-Secure Commitment Scheme

In this section, we construct a $\widetilde{O}(\log^2 n)$-round robust CCA-secure commitment scheme by using one-way functions in a black-box way.

Theorem 1. *Assume the existence of one-way functions. Then, there exists a $\widetilde{O}(\log^2 n)$-round robust CCA-secure commitment scheme* CCACom. *Furthermore,* CCACom *uses the underlying one-way function only in a black-box way.*

Proof. CCACom is shown in Fig. 9, in which we use the following tools (all of which can be constructed from one-way functions in a black-box way).

- A two-round statistically binding commitment scheme Com. (See Sect. 3.2.)
- The concurrently extractable commitment scheme CECom of Micciancio et al. [27]. (See Sect. 3.4.) The parameter ℓ in CECom is set as $\ell := O(\log^2 n \log\log n)$ so that $\ell = \omega(\log^2 n)$.
- A constant-round strongly extractable commitment scheme sExtCom. (See Lemma 3 in Sect. 4.1.)
- A $O(\log n)$-round one-one CCA-secure commitment scheme $\mathsf{CCACom}^{1:1}$ that satisfies strong computational binding property. (See Lemma 4 in Sect. 4.2.)

The round complexity of CCACom is clearly $\widetilde{O}(\log^2 n)$. The statistical binding property of CCACom follows directly from that of Com. Hence, it remains to show that CCACom is robust CCA secure. (The hiding property follows from CCA security.) In what follows, we prove CCA security in Sect. 5.1 and robustness in Section 5.2.

Commit Phase

The committer C and the receiver R take common inputs 1^n and $\mathsf{tag} \in \{0,1\}^n$, and C additionally takes private input $v \in \{0,1\}^n$. To commit to v, the committer C does the following with the receiver R.

Stage 1. R commits to a random subset $\Gamma \subset [10n]$ of size n by using $\mathsf{CCACom}^{1:1}$ with tag tag.

Stage 2. C computes an $(n+1)$-out-of-$10n$ Shamir's secret sharing $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of value v. Next, for each $j \in [10n]$ in parallel, C commits to s_j by using Com. Let $\phi_1, \ldots, \phi_{10n}$ denote the commitments and $d_1, \ldots, d_{10n}$ denote the decommitments.

Stage 3. For each $j \in [10n]$ in parallel, C commits to s_j by using CECom, where the parameter ℓ in CECom is set as $\ell := \widetilde{O}(\log^2 n)$. Let $\psi_1, \ldots, \psi_{10n}$ denote the commitments and $e_1, \ldots, e_{10n}$ denote the decommitments.

We call these parallel CECom commitments the *row* of CECom, and the commitment ψ_j in the row the *j-th column* (of CECom).

Stage 4. Let $R_{\mathsf{CCA}^{1:1}} := R_{\mathsf{CCA}^{1:1}}(n)$ be the round complexity of $\mathsf{CCACom}^{1:1}$. Let $\eta' := R_{\mathsf{CCA}^{1:1}} + 1$. Then, for each $i \in [\eta']$ in sequence, C does the following.

- For each $j \in [10n]$ in parallel, C commits to (s_j, d_j, e_j) by using $\mathsf{sExtCom}$. Let $\theta_{i,1}, \ldots, \theta_{i,10n}$ denote the commitments.

We call the i-th parallel commitments the *i-th row* (of $\mathsf{sExtCom}$), and call all the commitments to (s_j, d_j) the *j-th column* (of $\mathsf{sExtCom}$).

Stage 5. R decommits the commitment in Stage 1 to Γ.

Stage 6. For each $j \in \Gamma$, C decommits $\theta_{i,j}$ to (s_j, d_j, e_j) for each $i \in [\eta']$. Then, for every $j \in \Gamma$, R checks whether (s_j, d_j) is a valid decommitment of ϕ_j and (s_j, e_j) is a valid decommitment of ψ_j.

Decommit Phase

- C sends v and decommits the Com commitments $\phi_1, \ldots \phi_{10n}$ in Stage 2 to $s_1, \ldots, s_{10n}$.
- R accepts if and only if the following holds w.r.t. $\boldsymbol{s} = (s_1, \ldots, s_{10n})$, where s_j is defined to be $\perp$ if the decommitment of ϕ_j is invalid.
  - $\boldsymbol{s}$ is 0.9-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ that satisfies $w_j = s_j$ for every $j \in \Gamma$, and $\boldsymbol{w}$ is a codeword of v.

Fig. 9. CCA commitment scheme CCACom.

5.1. *Proof of CCA Security*

Lemma 5. CCACom *is CCA secure.*

Proof. We first remark that from the construction of the decommit phase of CCACom, the committed value of a CCACom commitment is defined as follows.

Definition 8. (*Committed value of* CCACom) If the shares $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ that are committed to in Stage 2 are 0.9-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ that satisfies $w_j = s_j$ for every $j \in \Gamma$, the committed value of a CCACom commitment is $\mathsf{Decode}(\boldsymbol{w})$. Otherwise, the committed value is $\perp$ (i.e., the commitment is invalid). □

We notice that the function $\mathsf{Value}_\Gamma(\cdot)$ in Fig. 1 (Sect. 3.1) computes the committed value of a CCACom commitment as above on input the shares $\boldsymbol{s}$ that are committed to in Stage 2.

To prove the CCA security of CCACom, we show the following indistinguishability for any PPT adversary $\mathcal{A}_{\mathrm{cca}}$ (see Definition 4).

$$\{\mathsf{IND}_0(\mathsf{CCACom}, \mathcal{A}_{\mathrm{cca}}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*} \overset{c}{\approx} \{\mathsf{IND}_1(\mathsf{CCACom}, \mathcal{A}_{\mathrm{cca}}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}. \tag{13}$$

Fix any PPT adversary $\mathcal{A}_{\mathrm{cca}}$. We prove Indistinguishability (13) by a hybrid argument. Since the experiments $\mathrm{IND}_0(\mathsf{CCACom}, \mathcal{A}_{\mathrm{cca}}, n, z)$ and $\mathrm{IND}_1(\mathsf{CCACom}, \mathcal{A}_{\mathrm{cca}}, n, z)$ differ only in the value that is committed to in the left session, we consider a series of hybrid experiments in which the left session is gradually modified so that in the last hybrid the adversary receives no information about the value that is committed to in the left session. Formally, for each $b \in \{0, 1\}$, we consider the following hybrid experiments.

Hybrid $H_0^b(n, z)$: Hybrid $H_0^b(n, z)$ is the same as $\mathrm{IND}_b(\mathsf{CCACom}, \mathcal{A}_{\mathrm{cca}}, n, z)$.

Hybrid $H_1^b(n, z)$ **to Hybrid** $H_{\eta'}^b(n, z)$: For $k \in [\eta']$, Hybrid $H_k^b(n, z)$ is the same as $H_0^b(n, z)$ except for the following.

- In Stage 1 of the left session, the committed value Γ is extracted by brute force from the $\mathsf{CCACom}^{1:1}$ commitment. If the commitment is invalid, Γ is set to be a random subset. If the commitment has more than one committed value, $H_k^b(n, z)$ outputs fail and terminates.
- In Stage 4 of the left session, the left committer commits to $0^{|u_j|}$ instead of u_j for every $j \notin \Gamma$ in the ith row for $i \in [k]$.

Hybrid $H_{\eta'+1}^b(n, z)$: Hybrid $H_{\eta'+1}^b(n, z)$ is the same as $H_{\eta'}^b(n, z)$ except that in Stage 3 of the left session, the left committer commits to $0^{|s_j|}$ instead of s_j for every $j \notin \Gamma$.

Hybrid $H_{\eta'+2}^b(n, z)$: Hybrid $H_{\eta'+2}^b(n, z)$ is the same as $H_{\eta'+1}^b(n, z)$ except that in Stage 2 of the left session, the left committer commits to $0^{|s_j|}$ instead of s_j for every $j \notin \Gamma$.

For $k \in \{0, \ldots, \eta' + 2\}$, let $\mathsf{H}_k^b(n, z)$ be the random variable for the output of $H_k^b(n, z)$. Since $\mathcal{A}_{\mathrm{cca}}$ receives no information about b in $H_{\eta'+2}^b(n, z)$, we have

$$\left\{\mathsf{H}_{\eta'+2}^0(n, z)\right\}_{n\in\mathbb{N}, z\in\{0,1\}^*} = \left\{\mathsf{H}_{\eta'+2}^1(n, z)\right\}_{n\in\mathbb{N}, z\in\{0,1\}^*}. \tag{14}$$

Hence, from a hybrid argument, we can show Indistinguishability (13) by showing the following three claims.

Claim 4. *For every $b \in \{0,1\}$ and $k \in [\eta']$, we have the following indistinguishability.*

$$\left\{\mathsf{H}^b_{k-1}(n,z)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*} \overset{c}{\approx} \left\{\mathsf{H}^b_{k}(n,z)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*}.$$

Claim 5. *For every $b \in \{0,1\}$, we have the following indistinguishability.*

$$\left\{\mathsf{H}^b_{\eta'}(n,z)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*} \overset{c}{\approx} \left\{\mathsf{H}^b_{\eta'+1}(n,z)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*}.$$

Claim 6. *For every $b \in \{0,1\}$, we have the following indistinguishability.*

$$\left\{\mathsf{H}^b_{\eta'+1}(n,z)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*} \overset{c}{\approx} \left\{\mathsf{H}^b_{\eta'+2}(n,z)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*}.$$

We prove Claim 4 in Sect. 5.1.1 and prove Claims 5 and 6 in Sect. 5.1.4.

5.1.1. *Proof of Claim 4*

Below, we prove Claim 4 using the following subclaim.

Subclaim 2. *For every $b \in \{0,1\}$ and $k \in \{0,\ldots,\eta'+2\}$, $H^b_k(n,z)$ outputs* fail *with at most negligible probability.*

The proof of Subclaim 2 is given in Sect. 5.1.3.

Proof of Claim 4. Since $H^b_{k-1}(n,z)$ and $H^b_k(n,z)$ differ only in the values that are committed to in a row of $\mathsf{sExtCom}$ in the left session, we use the hiding property of $\mathsf{sExtCom}$ to prove the indistinguishability. A problem is that $\mathcal{A}_{\mathrm{cca}}$ interacts with the committed-value oracle $\mathcal{O}$, which runs in super-polynomial time; because of the super-polynomial-time power of $\mathcal{O}$, the indistinguishability between the two hybrids does not follow directly from the *computational* hiding property of $\mathsf{sExtCom}$. To overcome this problem, we show that the oracle $\mathcal{O}$ can be emulated in polynomial time. Specifically, we show that the oracle $\mathcal{O}$ can be emulated by extracting the shares that are committed to in the rows of CECom and then computing the committed values of the right sessions from the extracted shares. When extracting the committed shares from the row of CECom, we use the robust concurrent extraction lemma (Lemma 2) so that we can use the hiding property of the kth row of $\mathsf{sExtCom}$ even in the presence of the extraction from CECom. Formally, we consider the following hybrids $G^b_{h:1}(n,z),\ldots,G^b_{h:3}(n,z)$ for each $h \in \{k-1,k\}$.

Hybrid $G^b_{h:1}(n,z)$: Hybrid $G^b_{h:1}(n,z)$ is the same as $H^b_h(n,z)$ except that at the end of each right session, the oracle $\mathcal{O}$ returns $\mathsf{Value}_\Gamma(\boldsymbol{s}^{\mathrm{CEC}})$ to $\mathcal{A}_{\mathrm{cca}}$ rather than $\mathsf{Value}_\Gamma(\boldsymbol{s})$ as the committed value of this session, where $\boldsymbol{s}=(s_1,\ldots,s_{10n})$ is the shares that are committed to in the row of Com in Stage 2, $\boldsymbol{s}^{\mathrm{CEC}}=(s^{\mathrm{CEC}}_1,\ldots,s^{\mathrm{CEC}}_{10n})$ is the shares that are committed to in the row of CECom in Stage 3, and Γ is the subset that is committed to in the $\mathsf{CCACom}^{1:1}$ commitment in Stage 1.

Hybrid $G^b_{h:2}(n,z)$: Hybrid $G^b_{h:2}(n,z)$ is the same as $G^b_{h:1}(n,z)$ except for syntactical differences: Roughly speaking, $G^b_{h:2}(n,z)$ is an experiment in which $G^b_{h:1}(n,z)$

is executed in such a way that we can use the robust concurrent extraction lemma later. Formally, $G^b_{h:2}(n, z)$ is defined as follows. Recall that in the setting of the robust concurrent extraction lemma (Lemma 2), an adversary, $\mathcal{A}_{\text{robust}}$, launches the robust-concurrent attack by interacting with the online extractor $\mathcal{E}$; specifically $\mathcal{A}_{\text{robust}}$ interacts with $\mathcal{E}$ as a party A of an arbitrary two-party protocol $\Pi = \langle B, A\rangle$ while interacting with $\mathcal{E}$ as the committers of CECom concurrently and obtaining a value from $\mathcal{E}$ at the end of each session of CECom (where the values that are returned from $\mathcal{E}$ are supposed to be the committed values of the CECom sessions). Then, consider the following Π and $\mathcal{A}_{\text{robust}}$ (see also Fig. 10).

$\Pi = \langle B, A\rangle$: First, party A gives a $\mathsf{CCACom}^{1:1}$ commitment to party B, where the tag in the $\mathsf{CCACom}^{1:1}$ commitment is chosen by A. Then, B extracts the committed value Γ of this $\mathsf{CCACom}^{1:1}$ commitment by brute force and sends it back to A. (If the $\mathsf{CCACom}^{1:1}$ commitment is invalid, Γ is set to be a random subset, and if the $\mathsf{CCACom}^{1:1}$ commitment has more than one committed value, B outputs fail and terminates.)
Next, A sends a sequence of strings $(m_1, \ldots, m_{9n})$ to B. Then, when $h = k-1$, B commits to each m_j ($j \in [9n]$) in parallel using $\mathsf{sExtCom}$, and when $h = k$, B commits to each $0^{|m_j|}$ ($j \in [9n]$) in parallel using $\mathsf{sExtCom}$.
$\mathcal{A}_{\text{robust}}$: $\mathcal{A}_{\text{robust}}$ takes non-uniform advice z and internally executes $G^b_{h:1}(n, z)$ with the following changes. (Recall that the execution of $G^b_{h:1}(n, z)$ involves an interaction with the CCA-security adversary $\mathcal{A}_{\text{cca}}$.)

- In Stage 1 of the left session, $\mathcal{A}_{\text{robust}}$ forwards the $\mathsf{CCACom}^{1:1}$ commitment from $\mathcal{A}_{\text{cca}}$ to the online extractor $\mathcal{E}$ (who internally emulates party B of Π). Then, instead of extracting the committed subset Γ from this $\mathsf{CCACom}^{1:1}$ commitment by brute force, $\mathcal{A}_{\text{robust}}$ obtains Γ from $\mathcal{E}$.
- In the kth row of $\mathsf{sExtCom}$ of the left session, $\mathcal{A}_{\text{robust}}$ sends $\{u_j\}_{j\notin\Gamma}$ to $\mathcal{E}$ (who internally emulates party B of Π), receives $\mathsf{sExtCom}$ commitments from $\mathcal{E}$, and forwards them to $\mathcal{A}_{\text{cca}}$. (At the same time, $\mathcal{A}_{\text{robust}}$ correctly commits to $\{u_j\}_{j\in\Gamma}$ for $\mathcal{A}_{\text{cca}}$ by using $\mathsf{sExtCom}$.)
- In Stage 3 of each right session, $\mathcal{A}_{\text{robust}}$ receives a row of CECom commitments from $\mathcal{A}_{\text{cca}}$ and forwards it to $\mathcal{E}$ (who internally emulates the receivers of CECom). Let $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{10n})$ denote the responses from $\mathcal{E}$ at the end of the row of the CECom commitments.
- At the end of each right session, $\mathcal{A}_{\text{robust}}$ sends $\mathsf{Value}_\Gamma(\boldsymbol{\alpha})$ to $\mathcal{A}_{\text{cca}}$ as the committed value of this right session.

The output of $\mathcal{A}_{\text{robust}}$ is that of the internally executed $G^b_{h:1}(n, z)$.

From the robust concurrent extraction lemma, there exists a robust simulator $\mathcal{S}$ such that for the above $\mathcal{A}_{\text{robust}}$, there exists an online extractor $\mathcal{E}$ that satisfies the following.

- For any row of CECom that $\mathcal{A}_{\text{robust}}$ sends to $\mathcal{E}$, let $\boldsymbol{s}^{\text{CEC}} = (s^{\text{CEC}}_1, \ldots, s^{\text{CEC}}_{10n})$ be the shares that are committed to in this row of CECom and $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{10n})$ be the responses from $\mathcal{E}$ at the end of this row. Then, for every $j \in [10n]$, if the jth CECom commitment in this row is valid and its committed value is uniquely determined, $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{10n})$ satisfies $\alpha_j = s^{\text{CEC}}_j$.

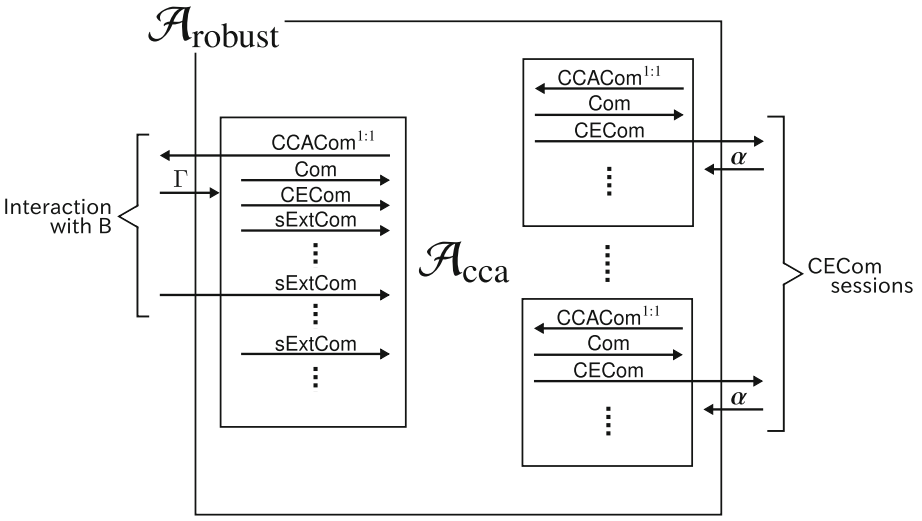

Fig. 10. Adversary $\mathcal{A}_{\text{robust}}$ in Hybrid $G^b_{h:2}(n, z)$. For simplicity, the right sessions are illustrated as if they are executed sequentially.

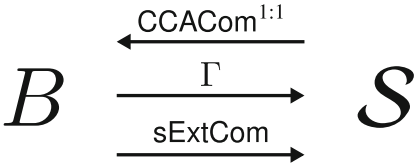

Fig. 11. Simulator $\mathcal{S}$ in Hybrid $G^b_{h:3}(n, z)$.

- $\mathcal{S}$ can simulate the robust-concurrent attack between $\mathcal{A}_{\text{robust}}$ and $\mathcal{E}$.

Hybrid $G^b_{h:2}(n, z)$ is the experiment $\text{REAL}^{\mathcal{A}_{\text{robust}}}_{\mathcal{E},\Pi}(n, \perp, z)$ of the robust concurrent extraction lemma. The output of $G^b_{h:2}(n, z)$ is that of $\mathcal{A}_{\text{robust}}$ in $\text{REAL}^{\mathcal{A}_{\text{robust}}}_{\mathcal{E},\Pi}(n, \perp, z)$.
Hybrid $G^b_{h:3}(n, z)$: Hybrid $G^b_{h:3}(n, z)$ differs from $G^b_{h:2}(n, z)$ in that the execution of $\text{REAL}^{\mathcal{A}_{\text{robust}}}_{\mathcal{E},\Pi}(n, \perp, z)$ (i.e., the robust-concurrent attack between $\mathcal{A}_{\text{robust}}$ and $\mathcal{E}$) is replaced with an interaction between party B of Π and the robust simulator $\mathcal{S}$ of the robust concurrent extraction lemma (see Fig. 11). The output of $G^b_{h:3}(n, z)$ is that of $\mathcal{A}_{\text{robust}}$ that is simulated by $\mathcal{S}$.

For $\ell \in \{1, 2, 3\}$, let $\mathsf{G}^b_{h:\ell}(n, z)$ be the random variable for the output of $G^b_{h:\ell}(n, z)$. We now prove the following four claims.

Claim 7. *For every $b \in \{0, 1\}$ and $h \in \{k - 1, k\}$, we have the following indistinguishability.*

$$\left\{\mathsf{H}^b_h(n, z)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*} \overset{s}{\approx} \left\{\mathsf{G}^b_{h:1}(n, z)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*}.$$

Claim 8. *For every $b \in \{0,1\}$ and $h \in \{k-1,k\}$, we have the following indistinguishability.*

$$\left\{\mathsf{G}^b_{h:1}(n,z)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*} \stackrel{s}{\approx} \left\{\mathsf{G}^b_{h:2}(n,z)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*}.$$

Claim 9. *For every $b \in \{0,1\}$ and $h \in \{k-1,k\}$, we have the following indistinguishability.*

$$\left\{\mathsf{G}^b_{h:2}(n,z)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*} \stackrel{s}{\approx} \left\{\mathsf{G}^b_{h:3}(n,z)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*}.$$

Claim 10. *For every $b \in \{0,1\}$, we have the following indistinguishability.*

$$\left\{\mathsf{G}^b_{k-1:3}(n,z)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*} \stackrel{c}{\approx} \left\{\mathsf{G}^b_{k:3}(n,z)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*}.$$

Claim 4 follows from these four claims.

Proof of Claim 7. Recall that $G^b_{h:1}(n,z)$ differs from $H^b_h(n,z)$ in that the committed value of a right session is computed by $\mathsf{Value}_\Gamma(\boldsymbol{s}^{\mathrm{CEC}})$ rather than by $\mathsf{Value}_\Gamma(\boldsymbol{s})$, where $\boldsymbol{s} = (s_1,\ldots,s_{10n})$ is the shares that are committed to in the row of Com in Stage 2, $\boldsymbol{s}^{\mathrm{CEC}} = (s_1^{\mathrm{CEC}},\ldots,s_{10n}^{\mathrm{CEC}})$ is the shares that are committed to in the row of CECom in Stage 3, and Γ is the subset that is committed to in the $\mathsf{CCACom}^{1:1}$ commitment in Stage 1. Roughly speaking, we prove this claim in two steps.

Step 1. Showing that $\mathsf{Value}_\Gamma(\boldsymbol{s}^{\mathrm{CEC}}) = \mathsf{Value}_\Gamma(\boldsymbol{s})$ holds in any right session if $\mathcal{A}_{\mathrm{cca}}$ does not "cheat" in that right session.

Step 2. Showing that $\mathcal{A}_{\mathrm{cca}}$ "cheats" in a right session with at most negligible probability.

Here, we say that $\mathcal{A}_{\mathrm{cca}}$ cheats in a right session if, roughly speaking, in every row of $\mathsf{sExtCom}$ in that session $\mathcal{A}_{\mathrm{cca}}$ does not commit to $u_j = (s_j,d_j,e_j)$ correctly in many columns. Hence, if $\mathcal{A}_{\mathrm{cca}}$ does not cheat, there exists a row of $\mathsf{sExtCom}$ in which $\mathcal{A}_{\mathrm{cca}}$ commits to $u_j = (s_j,d_j,e_j)$ as specified by the protocol in most columns, which guarantees that in most columns the share that is committed to by CECom is equal to the share that is committed to by Com, which in turn guarantees that the committed value of the session can be recovered from the shares that are committed to in the row of CECom instead of from those that are committed to in the row of Com. Details are given below.

First, we define the cheating behavior of $\mathcal{A}_{\mathrm{cca}}$.

Definition 9. (*Cheating by $\mathcal{A}_{\mathrm{cca}}$*) In each right session, let us say that a row of $\mathsf{sExtCom}$ in Stage 4 is **bad** if the values $\{u'_j = (s'_j,d'_j,e'_j)\}_{j\in[10n]}$ that are committed to in it satisfy the following condition.

Badness Condition. Let $\boldsymbol{s}^{\mathrm{sExt}} = (s_1^{\mathrm{sExt}},\ldots,s_{10n}^{\mathrm{sExt}})$ be the shares that are defined as follows. Let $s_j^{\mathrm{sExt}} \stackrel{\mathrm{def}}{=} s'_j$ if (s'_j,d'_j) is a valid decommitment of the jth Com commitment

in Stage 2 and (s'_j, e'_j) is a valid decommitment of the jth CECom commitment in Stage 3. Let $s_j^{\mathrm{sExt}} \stackrel{\mathrm{def}}{=} \bot$ otherwise. Then, the badness condition is defined as follows.

1. $\left|\left\{j \in [10n] \text{ s.t. } s_j^{\mathrm{sExt}} = \bot\right\}\right| \geq n \bigwedge \left\{j \in [10n] \text{ s.t. } s_j^{\mathrm{sExt}} = \bot\right\} \cap \Gamma = \emptyset$, or
2. $\boldsymbol{s}^{\mathrm{sExt}}$ is 0.8-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ that satisfies $w_j = s_j^{\mathrm{sExt}}$ for every $j \in \Gamma$, but $\boldsymbol{s}^{\mathrm{sExt}}$ is 0.1-far from $\boldsymbol{w}$.

Let us say that a row of $\mathsf{sExtCom}$ is **good** if it is not bad. Then, we say that $\mathcal{A}_{\mathrm{cca}}$ **cheats** in a right session if every row of $\mathsf{sExtCom}$ in that right session is bad. □

We then prove the following two subclaims.

Subclaim 3. *If the probability that $\mathcal{A}_{\mathrm{cca}}$ cheats in a right session in $H_h^b(n, z)$ is negligible, we have the following indistinguishability.*

$$\left\{\mathsf{H}_h^b(n, z)\right\}_{n\in\mathbb{N}, z\in\{0,1\}^*} \stackrel{s}{\approx} \left\{\mathsf{G}_{h:1}^b(n, z)\right\}_{n\in\mathbb{N}, z\in\{0,1\}^*}.$$

Subclaim 4. *The probability that $\mathcal{A}_{\mathrm{cca}}$ cheats in a right session in $H_h^b(n, z)$ is negligible.*

The proof of Subclaim 3 is given below. The proof of Subclaim 4 is given in Sect. 5.1.2.

Proof of Subclaim 3. We first show that $\mathsf{Value}_\Gamma(\boldsymbol{s}) = \mathsf{Value}_\Gamma(\boldsymbol{s}^{\mathrm{CEC}})$ holds in an accepting right session if $\mathcal{A}_{\mathrm{cca}}$ does not cheat in that right session, where, as defined in the description of Hybrid $G_{h:1}^b(n, z)$, $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ is the shares that are committed to in the row of Com in Stage 2, $\boldsymbol{s}^{\mathrm{CEC}} = (s_1^{\mathrm{CEC}}, \ldots, s_{10n}^{\mathrm{CEC}})$ is the shares that are committed to in the row of CECom in Stage 3, and Γ is the subset that is committed to in the $\mathsf{CCACom}^{1:1}$ commitment in Stage 1. Fix any right session, and assume that that right session is accepting and $\mathcal{A}$ does not cheat in it. Then, from the definition of cheating (Definition 9), that right session has a good row of $\mathsf{sExtCom}$. Let $\{u'_j = (s'_j, d'_j, e'_j)\}_{j\in[10n]}$ be the values that are committed to in that good row of $\mathsf{sExtCom}$. Let $\boldsymbol{s}^{\mathrm{sExt}} = (s_1^{\mathrm{sExt}}, \ldots, s_{10n}^{\mathrm{sExt}})$ be the shares that are derived from $\boldsymbol{u'} = (u'_1, \ldots, u'_{10n})$ as in the definition of cheating. Then, from the definitions of cheating and $\boldsymbol{s}^{\mathrm{sExt}}$, we have the following.

1. For every $j \in [10n]$, if $s_j^{\mathrm{sExt}} \neq \bot$, it holds $s_j^{\mathrm{sExt}} = s_j = s_j^{\mathrm{CEC}}$. (This follows from the definition of $\boldsymbol{s}^{\mathrm{sExt}}$.)
2. $\left|\left\{j \text{ s.t. } s_j^{\mathrm{sExt}} = \bot\right\}\right| < n \bigwedge \left\{j \text{ s.t. } s_j^{\mathrm{sExt}} = \bot\right\} \cap \Gamma = \emptyset$. (This is because the session would be rejected in Stage 6 if $\{j \text{ s.t. } s_j^{\mathrm{sExt}} = \bot\} \cap \Gamma \neq \emptyset$.)
3. $\boldsymbol{s}^{\mathrm{sExt}}$ is either 0.9-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ that satisfies $w_j = s_j^{\mathrm{sExt}}$ for every $j \in \Gamma$ or 0.2-far from any such valid codeword.

Hence, from Lemma 1 in Sect. 3.1, we have $\mathsf{Value}_\Gamma(\boldsymbol{s}) = \mathsf{Value}_\Gamma(\boldsymbol{s}^{\mathrm{CEC}}) = \mathsf{Value}_\Gamma(\boldsymbol{s}^{\mathrm{sExt}})$ in that session. Therefore, for any accepting right session, we have $\mathsf{Value}_\Gamma(\boldsymbol{s}) = \mathsf{Value}_\Gamma(\boldsymbol{s}^{\mathrm{CEC}})$ if $\mathcal{A}_{\mathrm{cca}}$ does not cheat in that session.

Since $G^b_{h:1}(n,z)$ differs from $H^b_h(n,z)$ only in that $\mathcal{O}$ returns $\mathsf{Value}_\Gamma(\boldsymbol{s}^{\mathrm{CEC}})$ to $\mathcal{A}_{\mathrm{cca}}$ rather than $\mathsf{Value}_\Gamma(\boldsymbol{s})$ in each right session, we conclude that $\mathsf{H}^b_h(n,z)$ and $\mathsf{G}^b_{h:1}(n,z)$ are statistically indistinguishable if $\mathcal{A}_{\mathrm{cca}}$ cheats in a right session with at most negligible probability. □

Now, Claim 7 follows immediately from Subclaims 3 and 4. This concludes the proof of Claim 7. □

Proof of Claim 8. From the construction of $G^b_{h:2}(n,z)$, the execution of $G^b_{h:1}(n,z)$ is perfectly emulated in $G_{h:2}(n,z)$ as long as we have $\mathsf{Value}_\Gamma(\boldsymbol{\alpha}) = \mathsf{Value}_\Gamma(\boldsymbol{s}^{\mathrm{CEC}})$ in each accepting right session.

First, we observe that if $\mathcal{A}_{\mathrm{cca}}$ does not cheat in an accepting right session, we have $\mathsf{Value}_\Gamma(\boldsymbol{\alpha}) = \mathsf{Value}_\Gamma(\boldsymbol{s}^{\mathrm{CEC}})$ in that right session except with negligible probability. Fix any right session, and assume that that right session is accepting and $\mathcal{A}$ does not cheat in it. Then, from the definition of cheating, that right session has a good row of $\mathsf{sExtCom}$. Let $\{u'_j = (s'_j, d'_j, e'_j)\}_{j\in[10n]}$ be the values that are committed to in that good row of $\mathsf{sExtCom}$. Let $\boldsymbol{s}^{\mathrm{sExt}} = (s^{\mathrm{sExt}}_1, \ldots, s^{\mathrm{sExt}}_{10n})$ be the shares that are derived from $\boldsymbol{u'} = (u'_1, \ldots, u'_{10n})$ as in the definition of the cheating. From the definition of cheating and the robust concurrent extraction lemma, we have the following in that session except with negligible probability.

1. For every $j \in [10n]$, if $s^{\mathrm{sExt}}_j \neq \perp$, it holds $s^{\mathrm{sExt}}_j = s^{\mathrm{CEC}}_j = \alpha_j$.
   (When $s^{\mathrm{sExt}}_j \neq \perp$, the jth CECom commitment in the row of CECom is valid and has a unique committed value except with negligible probability; therefore, from the robust concurrent extraction lemma, $\alpha_j = s^{\mathrm{CEC}}_j$ holds except with negligible probability.)
2. $\left|\left\{j \text{ s.t. } s^{\mathrm{sExt}}_j = \perp\right\}\right| < n \bigwedge \left\{j \text{ s.t. } s^{\mathrm{sExt}}_j = \perp\right\} \cap \Gamma = \emptyset$.
3. $\boldsymbol{s}^{\mathrm{sExt}}$ is either 0.9-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ that satisfies $w_j = s^{\mathrm{sExt}}_j$ for every $j \in \Gamma$ or 0.2-far from any such valid codeword.

Hence, from Lemma 1 in Sect. 3.1, we have $\mathsf{Value}_\Gamma(\boldsymbol{s}^{\mathrm{CEC}}) = \mathsf{Value}_\Gamma(\boldsymbol{\alpha}) = \mathsf{Value}_\Gamma(\boldsymbol{s}^{\mathrm{sExt}})$ except with negligible probability. Therefore, if $\mathcal{A}_{\mathrm{cca}}$ does not cheat in an accepting right session, we have $\mathsf{Value}_\Gamma(\boldsymbol{\alpha}) = \mathsf{Value}_\Gamma(\boldsymbol{s}^{\mathrm{CEC}})$ in that right session except with negligible probability.

Next, we observe that in $G^b_{h:1}(n,z)$, $\mathcal{A}_{\mathrm{cca}}$ cheats in a right session with at most negligible probability. This follows immediately from Subclaim 4 (which says that $\mathcal{A}_{\mathrm{cca}}$ cheats in a right session with negligible probability in $H^b_h(n,z)$) and Claim 7 (which says that the view of $\mathcal{A}_{\mathrm{cca}}$ in $G^b_{h:1}(n,z)$ is statistically indistinguishable from that in $H^b_h(n,z)$).

From what are observed in the above two paragraphs, it follows that we have $\mathsf{Value}_\Gamma(\boldsymbol{\alpha}) = \mathsf{Value}_\Gamma(\boldsymbol{s}^{\mathrm{CEC}})$ in each accepting right session except with negligible probability. □

Proof of Claim 9. Recall that $G^b_{h:3}(n,z)$ differs from $G^b_{h:2}(n,z)$ in that the execution of $\mathrm{REAL}^{\mathcal{A}_{\mathrm{robust}}}_{\mathcal{E},\Pi}(n,\perp,z)$ (i.e., the robust-concurrent attack between $\mathcal{A}_{\mathrm{robust}}$ and $\mathcal{E}$) is replaced with an interaction between party B of Π and the robust simulator $\mathcal{S}$ of the robust concurrent extraction lemma. Hence, this claim follows immediately from the robust

concurrent extraction lemma. (Notice that the round complexity of Π, denoted by R_Π, is $O(R_{\mathsf{CCA}^{1:1}}) = O(\log n)$ and thus the parameter ℓ of CECom satisfies $\ell = \omega(R_\Pi \log n)$.) □

Proof of Claim 10. We prove this claim by using the hiding property of $\mathsf{sExtCom}$. Roughly speaking, since $G^b_{k-1:3}(n, z)$ and $G^b_{k:3}(n, z)$ differ only in the shares that are committed to in the row of $\mathsf{sExtCom}$ that $\mathcal{S}$ receives in Π, and $G^b_{k-1:3}(n, z)$ and $G^b_{k:3}(n, z)$ run in polynomial time while $\mathcal{S}$ is receiving the row of $\mathsf{sExtCom}$, the indistinguishability follows directly from the hiding property of $\mathsf{sExtCom}$.

Formally, assume for contradiction that for infinitely many n, there exists $z \in \{0, 1\}^*$ such that $\mathsf{G}^b_{k-1:3}(n, z)$ and $\mathsf{G}^b_{k:3}(n, z)$ are distinguishable with advantage $1/\mathsf{poly}(n)$. Since $G^b_{k-1:3}(n, z)$ and $G^b_{k:3}(n, z)$ proceed identically until B starts sending the row of $\mathsf{sExtCom}$ to $\mathcal{S}$, there exists a prefix ρ of a transcript of $G^b_{k-1:3}(n, z)$ up until the row of $\mathsf{sExtCom}$ (exclusive) such that under the condition that ρ is a prefix of the transcript, $\mathsf{G}^b_{k-1:3}(n, z)$ and $\mathsf{G}^b_{k:3}(n, z)$ are distinguishable with advantage $1/\mathsf{poly}(n)$. Note that ρ contains the entire transcript of the $\mathsf{CCACom}^{1:1}$ commitment that $\mathcal{S}$ sends to B, and thus ρ uniquely determines the committed value Γ of this $\mathsf{CCACom}^{1:1}$ commitment. We then consider the following PPT adversary $\mathcal{B}$ against the hiding property of $\mathsf{sExtCom}$.

- Taking ρ and Γ as auxiliary inputs, $\mathcal{B}$ internally invokes $\mathcal{S}$ and emulates $G^b_{k-1:3}(n, z)$ from ρ by receiving either commitments to $\{u_j\}_{j \notin \Gamma}$ or commitments to $\{0^{|u_j|}\}_{j \notin \Gamma}$ from the external committer and then forwarding them to $\mathcal{S}$. Finally, $\mathcal{B}$ outputs whatever $\mathcal{S}$ outputs.

Since $\mathcal{B}$ perfectly emulates either $G^b_{k-1:3}(n, z)$ or $G^b_{k:3}(n, z)$ depending on the commitments it receives, our assumption implies that $\mathcal{B}$ distinguishes commitments to $\{u_j\}_{j \notin \Gamma}$ and commitments to $\{0^{|u_j|}\}_{j \notin \Gamma}$ with advantage $1/\mathsf{poly}(n)$. Thus, we reach a contradiction. □

As noted before, Claim 4 follows immediately from Claims 7– 10. This concludes the proof of Claim 4. □

5.1.2. *Proof of Subclaim 4*

We now prove Subclaim 4, which says that $\mathcal{A}_{\mathrm{cca}}$ cheats in a right session in $H^b_h(n, z)$ with at most negligible probability.

Proof of Subclaim 4. First, we introduce notations. For any $q \in \mathbb{N}$, we say that a right session has *end-index* q if this session is the qth right session that $\mathcal{A}_{\mathrm{cca}}$ completes. Similarly, we say that a right session has *start-index* q if this session is the qth right session that $\mathcal{A}_{\mathrm{cca}}$ starts. Note that the end-index of a session is undefined until the session completes, whereas the start-index is defined when the session starts. Jumping ahead, in the proof, we assume for contradiction that there exists an end-index q_{end} such that $\mathcal{A}_{\mathrm{cca}}$ cheats in the session having end-index q_{end}. Then, since we do not know which session has the end-index q_{end} until the session completes, we guess a start-index q_{start} such that the session having the start-index q_{start} has the end-index q_{end}.

We argue that $\mathcal{A}_{\mathrm{cca}}$ cannot cheat in any right session because of the hiding property of $\mathsf{CCACom}^{1:1}$.

However, there are two problems.

- Since $\mathcal{A}_{\mathrm{cca}}$ interacts with the committed-value oracle $\mathcal{O}$, which runs in super-polynomial-time, we cannot directly use the *computational* hiding property of $\mathsf{CCACom}^{1:1}$. We overcome this problem by considering a hybrid experiment in which $\mathcal{O}$ is emulated in polynomial time.
- $\mathcal{A}_{\mathrm{cca}}$ may cheat in a right session by using the messages that it receives in the left session, in which the left committer cheats. We overcome this problem by using one-one CCA security of $\mathsf{CCACom}^{1:1}$ instead of its hiding property. Since the left session can be emulated in polynomial time given the committed value Γ of the $\mathsf{CCACom}^{1:1}$ commitment in the left session, one-one CCA security of $\mathsf{CCACom}^{1:1}$ guarantees that the $\mathsf{CCACom}^{1:1}$ commitment in each right session is hiding even when the left committer cheats.

When simulating $\mathcal{O}$ in polynomial time, we use the concurrent extractability of CECom for obtaining the shares that are committed to in the row of CECom. Since we want to use the one-one CCA security of $\mathsf{CCACom}^{1:1}$, we use the robust concurrent extraction lemma so that we can use the one-one CCA security of $\mathsf{CCACom}^{1:1}$ even in the presence of the concurrent extraction from CECom.

Formally, assume for contradiction that there exists a right session in which $\mathcal{A}_{\mathrm{cca}}$ cheats with non-negligible probability. Then, there exists an end-index q_{end} such that (i) $\mathcal{A}_{\mathrm{cca}}$ cheats with at most negligible probability in any right session having an end-index less than q_{end}, but (ii) $\mathcal{A}_{\mathrm{cca}}$ cheats with non-negligible probability in the session having end-index q_{end}.

To reach a contradiction, we consider the following hybrid experiments $F^b_{h:1}(n,z),\ldots,F^b_{h:4}(n,z)$.

- Hybrid $F^b_{h:1}(n,z)$ is the same as $H^b_h(n,z)$ except that $F^b_{h:1}(n,z)$ halts immediately after $\mathcal{A}_{\mathrm{cca}}$ completes the session having end-index q_{end} (and immediately before $\mathcal{O}$ returns the committed value of this session to $\mathcal{A}_{\mathrm{cca}}$). Note that in $F^b_{h:1}(n,z)$, $\mathcal{O}$ returns the committed values to $\mathcal{A}_{\mathrm{cca}}$ only in the right sessions having the end-index less than q_{end}, and $\mathcal{A}_{\mathrm{cca}}$ cheats in those sessions only with negligible probability.
- Hybrid $F^b_{h:2}(n,z)$ is the same as $F^b_{h:1}(n,z)$ except that at the end of each right session, the oracle $\mathcal{O}$ returns $\mathsf{Value}_\Gamma(\boldsymbol{s}^{\mathrm{CEC}})$ to $\mathcal{A}_{\mathrm{cca}}$ rather than $\mathsf{Value}_\Gamma(\boldsymbol{s})$ as the committed value of this session, where $\boldsymbol{s}=(s_1,\ldots,s_{10n})$ is the shares that are committed to in the row of Com in Stage 2, $\boldsymbol{s}^{\mathrm{CEC}}=(s_1^{\mathrm{CEC}},\ldots,s_{10n}^{\mathrm{CEC}})$ is the shares that are committed to in the row of CECom in Stage 3, and Γ is the subset that is committed to in the $\mathsf{CCACom}^{1:1}$ commitment in Stage 1.
- Hybrid $F^b_{h:3}(n,z)$ is the same as $F^b_{h:2}(n,z)$ except for syntactical differences: Roughly speaking, $F^b_{h:3}(n,z)$ is an experiment in which $F^b_{h:2}(n,z)$ is executed in such a way that we can use the robust concurrent extraction lemma later. Formally, $F^b_{h:3}(n,z)$ is defined as follows. Recall that in the setting of the robust concurrent extraction lemma (Lemma 2), an adversary, $\mathcal{A}_{\mathrm{robust}}$, launches the robust-concurrent attack by interacting with the online extractor $\mathcal{E}$; specifically, $\mathcal{A}_{\mathrm{robust}}$ interacts with $\mathcal{E}$ as a party A of an arbitrary two-party protocol $\Pi=\langle B,A\rangle$ while interacting

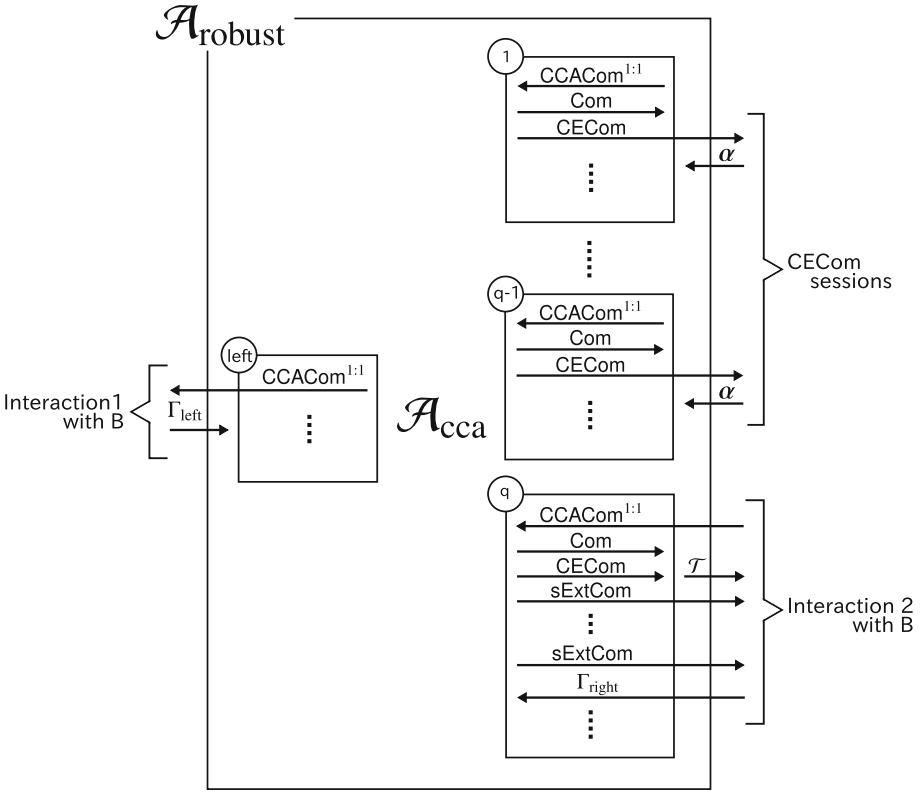

Fig. 12. Adversary $\mathcal{A}_{\text{robust}}$ in Hybrid $F^b_{h:3}(n, z)$. For simplicity, the right sessions are illustrated as if they are executed sequentially.

with $\mathcal{E}$ as the committers of CECom concurrently and obtaining a value from $\mathcal{E}$ at the end of each session of CECom (where the values that are returned from $\mathcal{E}$ are supposed to be the committed values of the CECom sessions). Then, consider the following Π and $\mathcal{A}_{\text{robust}}$ (see also Fig. 12).

$\Pi = \langle B, A \rangle$: Parties A and B do the following two interactions concurrently. (The schedule is controlled by A.)

Interaction 1. A gives a $\text{CCACom}^{1:1}$ commitment to B, where the tag is chosen by A. Then, B extracts the committed value of this $\text{CCACom}^{1:1}$ commitment, denoted by Γ_{left}, by brute force and sends it back to A. (If the $\text{CCACom}^{1:1}$ commitment is invalid, Γ_{left} is set to be a random subset, and if the $\text{CCACom}^{1:1}$ commitment has more than one committed value, B outputs fail and terminates.)

Interaction 2. First, B commits to a random subset $\Gamma_{\text{right}} \subset [10n]$ of size n using $\text{CCACom}^{1:1}$, where the tag is chosen by A. Next, A sends a transcript $\mathcal{T}$ of Stages 2 and 3 of CCACom (i.e., a row of Com followed by a row of CECom), and then gives η' rows of sExtCom to B, where each row consists of $10n$ parallel sExtCom commitments. (Recall that η' is the number of the rows of sExtCom in CCACom.) Finally, B decommits the $\text{CCACom}^{1:1}$ commitment to Γ_{right}.

$\mathcal{A}_{\text{robust}}$: $\mathcal{A}_{\text{robust}}$ takes non-uniform advice z and internally executes $F^b_{h:2}(n, z)$ as follows. (Recall that the execution of $F^b_{h:2}(n, z)$ involves an interaction with the CCA-security adversary $\mathcal{A}_{\text{cca}}$.)

- A start-index q_{start} is chosen at random at the beginning.
- In the left session, $\mathcal{A}_{\text{robust}}$ receives a $\mathsf{CCACom}^{1:1}$ commitment from $\mathcal{A}_{\text{cca}}$ in Stage 1 and forwards it to the online extractor $\mathcal{E}$ (who internally emulates party B of Π). Then, instead of extracting the committed subset Γ_{left} from this $\mathsf{CCACom}^{1:1}$ commitment by brute force, $\mathcal{A}_{\text{robust}}$ obtains Γ_{left} from $\mathcal{E}$. Subsequently, $\mathcal{A}_{\text{robust}}$ emulates the left session for $\mathcal{A}_{\text{cca}}$ honesty by using Γ_{left}.
- In the right session having start-index q_{start}, $\mathcal{A}_{\text{robust}}$ receives a $\mathsf{CCACom}^{1:1}$ commitment from $\mathcal{E}$ (who internally emulates party B of Π) and forwards it to $\mathcal{A}_{\text{cca}}$ in Stage 1. Then, $\mathcal{A}_{\text{robust}}$ emulates Stages 2 and 3 for $\mathcal{A}_{\text{cca}}$ honestly and sends the transcript $\mathcal{T}$ of these stages to $\mathcal{E}$. Then, $\mathcal{A}_{\text{robust}}$ receives η' rows of $\mathsf{sExtCom}$ from $\mathcal{A}_{\text{cca}}$ in Stage 4 and forwards them to $\mathcal{E}$. Then, $\mathcal{A}_{\text{robust}}$ receives a decommitment for the $\mathsf{CCACom}^{1:1}$ commitment from $\mathcal{E}$ and forwards it to $\mathcal{A}_{\text{cca}}$ in Stage 5. Then, $\mathcal{A}_{\text{robust}}$ emulates Stage 6 for $\mathcal{A}_{\text{cca}}$ honestly.
- In every other right session, $\mathcal{A}_{\text{robust}}$ emulates Stages 1 – 6 honestly except for forwarding the row of CECom in Stage 3 to $\mathcal{E}$ (who internally emulates the receivers of CECom). Let $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{10n})$ denote the responses from $\mathcal{E}$ at the end of the row of CECom. Then, at the end of the right session, $\mathcal{A}_{\text{robust}}$ sends $\mathsf{Value}_{\Gamma}(\boldsymbol{\alpha})$ to $\mathcal{A}_{\text{cca}}$ as the committed value of this right session.

The output of $\mathcal{A}_{\text{robust}}$ is that of the internally executed $F^b_{h:2}(n, z)$.

From the robust concurrent extraction lemma, there exists a robust simulator $\mathcal{S}$ such that for the above $\mathcal{A}_{\text{robust}}$, there exists an online extractor $\mathcal{E}$ that satisfies the following.

- For any row of CECom that $\mathcal{A}_{\text{robust}}$ sends to $\mathcal{E}$, let $\boldsymbol{s}^{\text{CEC}} = (s^{\text{CEC}}_1, \ldots, s^{\text{CEC}}_{10n})$ be the shares that are committed to in this row of CECom and $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{10n})$ be the responses from $\mathcal{E}$ at the end of this row. Then, for every $j \in [10n]$, if the jth CECom commitment in this row is valid and its committed value is uniquely determined, $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{10n})$ satisfies $\alpha_j = s^{\text{CEC}}_j$.
- $\mathcal{S}$ can simulate the robust-concurrent attack between $\mathcal{A}_{\text{robust}}$ and $\mathcal{E}$.

Hybrid $F^b_{h:3}(n, z)$ is the experiment $\text{REAL}^{\mathcal{A}_{\text{robust}}}_{\mathcal{E},\Pi}(n, \bot, z)$ of the robust concurrent extraction lemma. The output of $F^b_{h:3}(n, z)$ is that of $\mathcal{A}_{\text{robust}}$ in $\text{REAL}^{\mathcal{A}_{\text{robust}}}_{\mathcal{E},\Pi}(n, \bot, z)$.

In what follows, we say that $\mathcal{A}_{\text{robust}}$ *cheats* in $F^b_{h:3}(n)$ if in the execution of $F^b_{h:2}(n, z)$ that is emulated by $\mathcal{A}_{\text{robust}}$ in $F^b_{h:3}(n)$, $\mathcal{A}_{\text{cca}}$ cheats in the right session having start-index q_{start}. We remark that, since $\mathcal{A}_{\text{robust}}$ sends the transcript $\mathcal{T}$ of Stages 2 and 3 to $\mathcal{E}$ in Π, we can see whether $\mathcal{A}_{\text{robust}}$ cheats in $F^b_{h:3}(n)$ or not by examining the transcript of Π between $\mathcal{A}_{\text{robust}}$ and $\mathcal{E}$ (specifically, by extracting the committed values from each row of $\mathsf{sExtCom}$ by brute force and then checking whether those committed values satisfy the badness condition in Definition 9 w.r.t. Stages 2 and 3 of CCACom that appear in $\mathcal{T}$).

- Hybrid $F^b_{h:4}(n, z)$ differs from $F^b_{h:3}(n, z)$ in that the execution of $\text{REAL}^{\mathcal{A}_{\text{robust}}}_{\mathcal{E},\Pi}(n, \bot, z)$ (i.e., the robust-concurrent attack between $\mathcal{A}_{\text{robust}}$ and $\mathcal{E}$) is replaced with an inter-

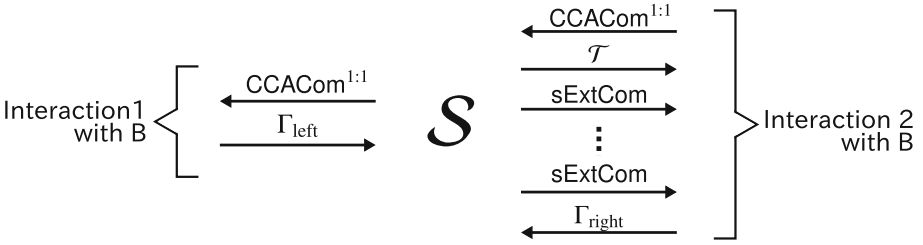

Fig. 13. Simulator $\mathcal{S}$ in Hybrid $F^b_{h:4}(n, z)$.

action between party B of Π and the robust simulator $\mathcal{S}$ of the robust concurrent extraction lemma (see Fig. 13). The output of $F^b_{h:4}(n, z)$ is that of $\mathcal{A}_{\text{robust}}$ that is simulated by $\mathcal{S}$.
In what follows, we say that $\mathcal{S}$ *cheats* in $F^b_{h:4}(n, z)$ if $\mathcal{A}_{\text{robust}}$ cheats in the view that is simulated by $\mathcal{S}$.

First, we notice that in $F^b_{h:1}(n, z)$, $\mathcal{A}_{\text{cca}}$ cheats with at most negligible probability in any right session having an end-index less than q_{end}, and $\mathcal{A}_{\text{cca}}$ cheats with non-negligible probability in the session having end-index q_{end}. This is because $F^b_{h:1}(n, z)$ proceeds identically with $H^b_h(n, z)$ until the end of the right session having end-index q_{end}.

Next, we observe that in $F^b_{h:2}(n, z)$, $\mathcal{A}_{\text{cca}}$ cheats with non-negligible probability in the session having end-index q_{end}. Recall that $F^b_{h:2}(n, z)$ differs from $F^b_{h:1}(n, z)$ in that at the end of each right session having an end-index less than q_{end}, the oracle $\mathcal{O}$ computes the committed value of the session by $\mathsf{Value}_\Gamma(\boldsymbol{s}^{\text{CEC}})$ rather than by $\mathsf{Value}_\Gamma(\boldsymbol{s})$. Then, since in $F^b_{h:1}(n, z)$ $\mathcal{A}_{\text{cca}}$ cheats with at most negligible probability in any right session having an end-index less than q_{end}, we can show that $\mathsf{Value}_\Gamma(\boldsymbol{s}^{\text{CEC}}) = \mathsf{Value}_\Gamma(\boldsymbol{s})$ holds in any such right session except with negligible probability by using the same argument as in the proof of Subclaim 3. Hence, the view of $\mathcal{A}_{\text{cca}}$ in $F^b_{h:2}(n, z)$ is statistically indistinguishable from that in $F^b_{h:1}(n, z)$, so $\mathcal{A}_{\text{cca}}$ cheats with non-negligible probability in the session having end-index q_{end} in $F^b_{h:2}(n, z)$.

Next, we observe that $\mathcal{A}_{\text{robust}}$ cheats in $F^b_{h:3}(n, z)$ with non-negligible probability. From the construction of $F^b_{h:3}(n, z)$, an execution of $F^b_{h:2}(n, z)$ is perfectly emulated in $F^b_{h:3}(n, z)$ as long as we have $\mathsf{Value}_\Gamma(\boldsymbol{\alpha}) = \mathsf{Value}_\Gamma(\boldsymbol{s}^{\text{CEC}})$ in each accepting right session that has an end-index less than q_{end}. Then, since in $F^b_{h:2}(n, z)$ $\mathcal{A}_{\text{cca}}$ cheats with at most negligible probability in any right session having an end-index less than q_{end}, we can show that $\mathsf{Value}_\Gamma(\boldsymbol{\alpha}) = \mathsf{Value}_\Gamma(\boldsymbol{s}^{\text{CEC}})$ holds in any such right session except with negligible probability by using the same argument as in the proof of Claim 8. Hence, in the execution of $F^b_{h:2}(n, z)$ that is emulated in $F^b_{h:3}(n, z)$, $\mathcal{A}_{\text{cca}}$ cheats with non-negligible probability in the session having end-index q_{end}. Now, since the number of the right sessions is polynomially bounded, we conclude that in the execution of $F^b_{h:2}(n, z)$ that is emulated in $F^b_{h:3}(n, z)$, $\mathcal{A}_{\text{cca}}$ cheats with non-negligible probability in the session having start-index q_{start}.

Next, we observe that $\mathcal{S}$ cheats in $F^b_{h:4}(n, z)$ with non-negligible probability. This follows from the robust concurrent extraction lemma, which guarantees that $\mathcal{A}_{\text{robust}}$'s

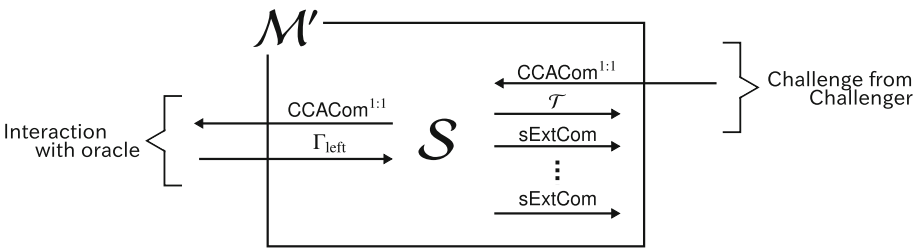

Fig. 14. Adversary $\mathcal{M}'$ against the one-one CCA security of $\mathsf{CCACom}^{1:1}$.

view is statistically simulated in $F^b_{h:4}(n, z)$. (Notice that the round complexity R_Π of Π is $O(R_{\mathsf{CCA}^{1:1}}) = O(\log n)$ and thus the parameter ℓ of CECom satisfies $\ell = \omega(R_\Pi \log n)$.)

We then derive a contradiction by showing that we can break the one-one CCA security of $\mathsf{CCACom}^{1:1}$ using $F^b_{h:4}(n, z)$.

For a warm up, we first consider the following super-polynomial-time adversary $\mathcal{M}'$ against the one-one CCA security of $\mathsf{CCACom}^{1:1}$ (see also Fig. 14).

- Externally, $\mathcal{M}'$ sends random subsets $\Gamma_0, \Gamma_1 \subset [10n]$ to a committer of $\mathsf{CCACom}^{1:1}$ and receives a $\mathsf{CCACom}^{1:1}$ commitment from it (the committed value is either Γ_0 or Γ_1). Concurrently, $\mathcal{M}'$ also interacts with the committed-value oracle of $\mathsf{CCACom}^{1:1}$ in a single session.
  Internally, $\mathcal{M}'$ invokes $\mathcal{S}$ and emulates $F^b_{h:4}(n, z)$ for $\mathcal{S}$ honestly except for the following.
  - When sending a $\mathsf{CCACom}^{1:1}$ commitment to $\mathcal{S}$ as the commitment from B in Π, $\mathcal{M}'$ obtains a $\mathsf{CCACom}^{1:1}$ commitment from the external committer and forwards it to $\mathcal{S}$.
  - When $\mathcal{S}$ starts sending a $\mathsf{CCACom}^{1:1}$ commitment to B in Π, $\mathcal{M}'$ forwards it to external $\mathcal{O}$, and then, instead of extracting its committed value Γ_{left} by brute force, $\mathcal{M}'$ obtains Γ_{left} from $\mathcal{O}$.
  - When $\mathcal{S}$ starts sending η' rows of $\mathsf{sExtCom}$ to B in Π, $\mathcal{M}'$ extracts the committed values of an arbitrarily chosen row by brute force. $\mathcal{M}'$ then stops emulating $F^b_{h:4}(n, z)$.

  Let $\{u_j = (s_j, d_j, e_j)\}_{j\in[10n]}$ be the values that are extracted from the arbitrarily chosen row of $\mathsf{sExtCom}$, and $\mathcal{T}$ be the message that $\mathcal{S}$ sends to B in Π as the transcript of Stages 2 and 3 of CCACom. Let $\boldsymbol{s}^{\text{sExt}} = (s_1^{\text{sExt}}, \ldots, s_{10n}^{\text{sExt}})$ be the shares that are derived from $\boldsymbol{u} = (u_1, \ldots, u_{10n})$ and $\mathcal{T}$ as in the definition of the cheating (Definition 9). Then, $\mathcal{M}'$ outputs 1 if and only if either of the following holds.

1. $\left|\left\{j \in [10n] \text{ s.t. } s_j^{\text{sExt}} = \perp\right\}\right| \geq n \bigwedge \left\{j \in [10n] \text{ s.t. } s_j^{\text{sExt}} = \perp\right\} \cap \Gamma_1 = \emptyset$.
2. $\boldsymbol{s}^{\text{sExt}}$ is 0.8-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ that satisfies $s_j^{\text{sExt}} = w_j$ for every $j \in \Gamma_1$, but $\boldsymbol{s}^{\text{sExt}}$ is 0.1-far from $\boldsymbol{w}$.

  When $\mathcal{M}'$ receives a commitment to Γ_0, $\mathcal{M}'$ outputs 1 only with negligible probability; this is because when $\mathcal{M}'$ receives a commitment to Γ_0, the internal $\mathcal{S}$ receives

no information about Γ_1, and thus, the probability that either of the following holds is negligible.

1. $\left|\left\{j \in [10n] \text{ s.t. } s_j^{\mathsf{sExt}} = \bot\right\}\right| \geq n$ but $\left\{j \in [10n] \text{ s.t. } s_j^{\mathsf{sExt}} = \bot\right\} \cap \Gamma_1 = \emptyset$.
2. $\boldsymbol{s}^{\mathsf{sExt}}$ is 0.1-far from a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ but we have $s_j^{\mathsf{sExt}} = w_j$ for every $j \in \Gamma_1$.

On the other hand, when $\mathcal{M}'$ receives a commitment to Γ_1, the internal $\mathcal{S}$ cheats in the emulated execution of $F_{h:4}^b(n, z)$ with non-negligible probability, so all the rows of $\mathsf{sExtCom}$ from $\mathcal{S}$ are bad (w.r.t. Stages 2 and 3 of CCACom that appear in $\mathcal{T}$) with non-negligible probability; hence, from the definition of cheating (Definition 9), $\mathcal{M}'$ outputs 1 with non-negligible probability. Thus, $\mathcal{M}'$ distinguishes a commitment to Γ_0 and a commitment to Γ_1 with non-negligible advantage.

We then consider an adversary $\mathcal{M}$ that emulates $\mathcal{M}'$ in polynomial time by extracting the committed values of a row of $\mathsf{sExtCom}$ by using the extractability of $\mathsf{sExtCom}$. To formally define $\mathcal{M}$, we first define the following machine $\widehat{\mathcal{M}}$.

- Externally, $\widehat{\mathcal{M}}$ sends random subsets $\Gamma_0, \Gamma_1 \subset [10n]$ to an external party, receives a $\mathsf{CCACom}^{1:1}$ commitment from a committer of $\mathsf{CCACom}^{1:1}$ (the committed value is either Γ_0 or Γ_1), sends a transcript $\mathcal{T}$ of Stages 2 and 3 of CCACom to an external party, and then gives a row of $\mathsf{sExtCom}$ to a receiver of $\mathsf{sExtCom}$. Concurrently, $\widehat{\mathcal{M}}$ also interacts with the committed-value oracle of $\mathsf{CCACom}^{1:1}$ in a single session. Internally, $\widehat{\mathcal{M}}$ invokes $\mathcal{S}$ and emulates $F_{h:4}^b(n, z)$ for $\mathcal{S}$ honestly except for the following.
  - When sending a $\mathsf{CCACom}^{1:1}$ commitment to $\mathcal{S}$ as the commitment from B in Π, $\widehat{\mathcal{M}}$ obtains a $\mathsf{CCACom}^{1:1}$ commitment from the external committer and forwards it to $\mathcal{S}$.
  - When $\mathcal{S}$ starts sending a $\mathsf{CCACom}^{1:1}$ commitment to B in Π, $\widehat{\mathcal{M}}$ forwards it to external $\mathcal{O}$, and then, instead of extracting its committed value Γ_{left} by brute force, $\widehat{\mathcal{M}}$ obtains Γ_{left} from $\mathcal{O}$.
  - After receiving a transcript $\mathcal{T}$ of Stages 2 and 3 of CCACom from $\mathcal{S}$, $\widehat{\mathcal{M}}$ forwards it to the external party.
  - When $\mathcal{S}$ starts sending η' rows of $\mathsf{sExtCom}$ to B in Π, $\widehat{\mathcal{M}}$ forwards a randomly chosen row among them to the external receiver of $\mathsf{sExtCom}$. If the randomly chosen row of $\mathsf{sExtCom}$ "interleaves" with any messages of the $\mathsf{CCACom}^{1:1}$ commitment that are being forwarded to $\mathcal{O}$ (namely, if $\mathcal{S}$ tries to send/receive a message of that $\mathsf{CCACom}^{1:1}$ commitment while sending that row of $\mathsf{sExtCom}$), $\widehat{\mathcal{M}}$ stops emulating $F_{h:4}^b(n, z)$ immediately and terminates. In other cases, $\widehat{\mathcal{M}}$ stops emulating $F_{h:4}^b(n, z)$ and terminates when the randomly chosen row of $\mathsf{sExtCom}$ completes.

We remark that once $\widehat{\mathcal{M}}$ starts sending a $\mathsf{sExtCom}$ commitment to the external receiver of $\mathsf{sExtCom}$, $\widehat{\mathcal{M}}$ no longer interacts with the oracle $\mathcal{O}$. (Once $\widehat{\mathcal{M}}$ starts sending a $\mathsf{sExtCom}$ commitment, either $\widehat{\mathcal{M}}$ terminates in the middle of $\mathsf{sExtCom}$ (because the internal $\mathcal{S}$ tries to send/receive a message of $\mathsf{CCACom}^{1:1}$) or $\widehat{\mathcal{M}}$ completes the $\mathsf{sExtCom}$ commitment.) Furthermore, since $\eta' = R_{\mathsf{CCA}^{1:1}} + 1$ (and thus the number of rows of $\mathsf{sExtCom}$ is bigger than the number of rounds in $\mathsf{CCACom}^{1:1}$),

a randomly chosen row of $\mathsf{sExtCom}$ does not interleave with any messages of $\mathsf{CCACom}^{1:1}$ with non-negligible probability; thus, $\widehat{\mathcal{M}}$ completes the $\mathsf{sExtCom}$ commitment with non-negligible probability.
Using $\widehat{\mathcal{M}}$, we define $\mathcal{M}$ as follows.

- Externally, $\mathcal{M}$ sends random subsets $\Gamma_0, \Gamma_1 \subset [10n]$ to a committer of $\mathsf{CCACom}^{1:1}$ and receives a $\mathsf{CCACom}^{1:1}$ commitment from it (the committed value is either Γ_0 or Γ_1). Concurrently, $\mathcal{M}$ also interacts with the committed-value oracle of $\mathsf{CCACom}^{1:1}$ in a single session.
Internally, $\mathcal{M}$ invokes $\widehat{\mathcal{M}}$ and lets it interact with the external committer of $\mathsf{CCACom}^{1:1}$ and the oracle $\mathcal{O}$. When $\widehat{\mathcal{M}}$ starts sending a row of $\mathsf{sExtCom}$, $\mathcal{M}$ invokes the extractor of $\mathsf{sExtCom}$ against $\widehat{\mathcal{M}}$ and obtains (τ, σ), where τ is the view of $\widehat{\mathcal{M}}$ as a committer of $\mathsf{sExtCom}$ and σ is a possible value that $\widehat{\mathcal{M}}$ committed to in τ.
If the $\mathsf{sExtCom}$ commitment that $\widehat{\mathcal{M}}$ gives in τ is not accepting or the extractor of $\mathsf{sExtCom}$ fails (i.e., the commitment in τ is accepting but $\sigma = \perp$ holds), $\mathcal{M}$ outputs 0. Otherwise, parse σ as $\{u_j = (s_j, d_j, e_j)\}_{j\in[10n]}$, and let $\mathcal{T}$ be the transcript that $\mathcal{M}$ obtained from $\widehat{\mathcal{M}}$ before the row of $\mathsf{sExtCom}$. Let $\boldsymbol{s}^{\mathrm{sExt}} = (s_1^{\mathrm{sExt}}, \ldots, s_{10n}^{\mathrm{sExt}})$ be the shares that are derived from $\boldsymbol{u} = (u_1, \ldots, u_{10n})$ and $\mathcal{T}$ as in the definition of the cheating (Definition 9). Then, $\mathcal{M}$ outputs 1 if and only if either of the following holds.
  1. $\left|\left\{j \in [10n] \text{ s.t. } s_j^{\mathrm{sExt}} = \perp\right\}\right| \geq n \bigwedge \left\{j \in [10n] \text{ s.t. } s_j^{\mathrm{sExt}} = \perp\right\} \cap \Gamma_1 = \emptyset.$
  2. $\boldsymbol{s}^{\mathrm{sExt}}$ is 0.8-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ that satisfies $s_j^{\mathrm{sExt}} = w_j$ for every $j \in \Gamma_1$, but $\boldsymbol{s}^{\mathrm{sExt}}$ is 0.1-far from $\boldsymbol{w}$.

  Recall that, as observed above, $\widehat{\mathcal{M}}$ gives an accepting $\mathsf{sExtCom}$ commitment with non-negligible probability. Furthermore, the extractor of $\mathsf{sExtCom}$ fails only with negligible probability, and no over-extraction occur during the extraction. Hence, from exactly the same argument as in the analysis of $\mathcal{M}'$ above, $\mathcal{M}$ distinguishes a commitment to Γ_0 and a commitment to Γ_1 with non-negligible advantage. Since $\mathcal{M}$ runs in polynomial time, this is a contradiction. □

5.1.3. *Proof of Subclaim 2*

We now prove Subclaim 2, which says that $H_k^b(n, z)$ outputs fail with at most negligible probability. Recall that $H_k^b(n, z)$ outputs fail when the $\mathsf{CCACom}^{1:1}$ commitment in Stage 1 of the left session has more than one committed value.

Proof of Subclaim 2. Since $H_k^b(n, z)$ outputs fail only if the commitment in Stage 1 has more than one committed value in the left session, we prove this claim by using the binding property of $\mathsf{CCACom}^{1:1}$. A problem is that $\mathcal{A}_{\mathrm{cca}}$ interacts with the committed-value oracle $\mathcal{O}$, which runs in super-polynomial time; because of the super-polynomial-time power of $\mathcal{O}$, the claim does not follow directly from the strong computational binding property of $\mathsf{CCACom}^{1:1}$. We overcome this problem by, again, emulating $\mathcal{O}$ in polynomial time using the robust concurrent extraction lemma on CECom. The proof is similar to that of Claim 4.

Formally, assume for contradiction that $H^b_k(n, z)$ outputs fail with non-negligible probability. Then, the $\mathsf{CCACom}^{1:1}$ commitment in Stage 1 of the left session has more than one committed value with non-negligible probability.

We consider the following hybrid experiments.

Hybrid $E^b_{k:1}(n, z)$: Hybrid $E^b_{k:1}(n, z)$ is the same as $H^b_k(n, z)$ except for the following.

- At the end of each right session, the oracle $\mathcal{O}$ returns $\mathsf{Value}_\Gamma(\boldsymbol{s}^{\mathrm{CEC}})$ to $\mathcal{A}_{\mathrm{cca}}$ rather than $\mathsf{Value}_\Gamma(\boldsymbol{s})$ as the committed value of this session, where $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ is the shares that are committed to in the row of Com in Stage 2, $\boldsymbol{s}^{\mathrm{CEC}} = (s^{\mathrm{CEC}}_1, \ldots, s^{\mathrm{CEC}}_{10n})$ is the shares that are committed to in the row of CECom in Stage 3, and Γ is the subset that is committed to in the $\mathsf{CCACom}^{1:1}$ commitment in Stage 1.
- The experiment is terminated at the end of Stage 1 of the left session.

Hybrid $E^b_{k:2}(n, z)$: Hybrid $E^b_{k:2}(n, z)$ is the same as $E^b_{k:1}(n, z)$ except for syntactical differences: Roughly speaking, $E^b_{k:2}(n, z)$ is an experiment in which $E^b_{k:1}(n, z)$ is executed in such a way that we can use the robust concurrent extraction lemma later. Formally, $E^b_{k:2}(n, z)$ is defined as follows. Recall that in the setting of the robust concurrent extraction lemma (Lemma 2), an adversary, $\mathcal{A}_{\mathrm{robust}}$, launches the robust-concurrent attack by interacting with the online extractor $\mathcal{E}$; specifically, $\mathcal{A}_{\mathrm{robust}}$ interacts with $\mathcal{E}$ as a party A of an arbitrary two-party protocol $\Pi = \langle B, A\rangle$ while interacting with $\mathcal{E}$ as the committers of CECom concurrently and obtaining a value from $\mathcal{E}$ at the end of each session of CECom (where the values that are returned from $\mathcal{E}$ are supposed to be the committed values of the CECom sessions). Then, consider the following Π and $\mathcal{A}_{\mathrm{robust}}$.

> $\Pi = \langle B, A\rangle$: Party A gives a $\mathsf{CCACom}^{1:1}$ commitment to party B, where the tag in the $\mathsf{CCACom}^{1:1}$ commitment is chosen by A.
>
> $\mathcal{A}_{\mathrm{robust}}$: $\mathcal{A}_{\mathrm{robust}}$ takes non-uniform advice z and internally executes $E^b_{k:1}(n, z)$ with the following changes. (Recall that the execution of $E^b_{k:1}(n, z)$ involves an interaction with the CCA-security adversary $\mathcal{A}_{\mathrm{cca}}$.)
>
> - In Stage 1 of the left session, $\mathcal{A}_{\mathrm{robust}}$ forwards the $\mathsf{CCACom}^{1:1}$ commitment from $\mathcal{A}_{\mathrm{cca}}$ to the online extractor $\mathcal{E}$ (who internally emulates party B of Π).
> - In Stage 3 of each right session, $\mathcal{A}_{\mathrm{robust}}$ receives a row of CECom commitments from $\mathcal{A}_{\mathrm{cca}}$ and forwards it to $\mathcal{E}$ (who internally emulates the receivers of CECom). Let $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{10n})$ denote the responses from $\mathcal{E}$ at the end of the row of the CECom commitments.
> - At the end of each right session, $\mathcal{A}_{\mathrm{robust}}$ sends $\mathsf{Value}_\Gamma(\boldsymbol{\alpha})$ to $\mathcal{A}_{\mathrm{cca}}$ as the committed value of this right session.

From the robust concurrent extraction lemma, there exists a robust simulator $\mathcal{S}$ such that for the above $\mathcal{A}_{\mathrm{robust}}$, there exists an online extractor $\mathcal{E}$ that satisfies the following.

- For any row of CECom that $\mathcal{A}_{\mathrm{robust}}$ sends to $\mathcal{E}$, let $\boldsymbol{s}^{\mathrm{CEC}} = (s^{\mathrm{CEC}}_1, \ldots, s^{\mathrm{CEC}}_{10n})$ be the shares that are committed to in this row of CECom and $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{10n})$

be the responses from $\mathcal{E}$ at the end of this row. Then, for every $j \in [10n]$, if the jth CECom commitment in this row is valid and its committed value is uniquely determined, $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{10n})$ satisfies $\alpha_j = s_j^{\mathrm{CEC}}$.

- $\mathcal{S}$ can simulate the robust-concurrent attack between $\mathcal{A}_{\mathrm{robust}}$ and $\mathcal{E}$.

Hybrid $E^b_{k:2}(n, z)$ is the experiment $\mathrm{REAL}^{\mathcal{A}_{\mathrm{robust}}}_{\mathcal{E},\Pi}(n, \bot, z)$ of the robust concurrent extraction lemma.

Hybrid $E^b_{k:3}(n, z)$: Hybrid $E^b_{k:3}(n, z)$ differs from $E^b_{k:2}(n, z)$ in that the execution of $\mathrm{REAL}^{\mathcal{A}_{\mathrm{robust}}}_{\mathcal{E},\Pi}(n, \bot, z)$ (i.e., the robust-concurrent attack by $\mathcal{A}_{\mathrm{robust}}$ against $\mathcal{E}$) is replaced with an interaction between party B of Π and the robust simulator $\mathcal{S}$ of the robust concurrent extraction lemma.

First, we notice that in $E^b_{k:1}(n, z)$, the $\mathsf{CCACom}^{1:1}$ commitment in Stage 1 of the left session has more than one committed value with non-negligible probability. This is because from the same argument as in the proof of Claim 7, we can show that the view of $\mathcal{A}_{\mathrm{cca}}$ in $E^b_{k:1}(n, z)$ is statistically close to that in $H^b_k(n, z)$.

Next, we notice that in $E^b_{k:2}(n, z)$, the $\mathsf{CCACom}^{1:1}$ commitment from $\mathcal{A}_{\mathrm{robust}}$ to $\mathcal{E}$ has more than one committed value with non-negligible probability. This is because from the same argument as in the proof of Claim 8, we can show that an execution of $E^b_{k:1}(n, z)$ is statistically simulated in $E^b_{k:2}(n, z)$.

Next, we notice that in $E^b_{k:3}(n, z)$, the $\mathsf{CCACom}^{1:1}$ commitment from $\mathcal{S}$ to B has more than one committed value with non-negligible probability. This is because from the robust concurrent extraction lemma, we can show that the $\mathsf{CCACom}^{1:1}$ commitment between $\mathcal{S}$ and B in $E^b_{k:3}(n, z)$ is statistically close to that between $\mathcal{A}_{\mathrm{robust}}$ and $\mathcal{E}$ in $E^b_{k:2}(n, z)$.

Now, since $E^b_{k:3}(n, z)$ runs in polynomial time and $\mathcal{S}$ interacts with an honest receiver of $\mathsf{CCACom}^{1:1}$ in it, we reach a contradiction to the strong computational binding property of $\mathsf{CCACom}^{1:1}$. This concludes the proof of Subclaim 2. □

5.1.4. *Proofs of Claims 5 and 6*

Claims 5 and 6 can be proven very similarly to Claim 4. For example, consider the case of Claim 5, which says that the output of $H^b_{\eta'}(n, z)$ and that of $H^b_{\eta'+1}(n, z)$ are computationally indistinguishable. Since $H^b_{\eta'}(n, z)$ and $H^b_{\eta'+1}(n, z)$ differ only in the committed values of a row of CECom, we can prove Claim 5 by modifying the proof of Claim 4 accordingly. (Recall that in the proof of Claim 4, our goal is to show the indistinguishability between the outputs of two hybrids that differ only in the values committed to in a row of $\mathsf{sExtCom}$.) The only problem is that the round complexity of CECom is $O(\ell) = \widetilde{O}(\log^2 n)$ (whereas the round complexity of $\mathsf{sExtCom}$ is $O(1)$), and thus we cannot use the robust concurrent extraction lemma in the same way as in the proof of Claim 4. However, since a CECom commitment can be decomposed into n ExtCom commitments, we can easily solve this problem by designing a sequence of sub-hybrids such that each neighboring sub-hybrids differ in the values that are committed to in a row of ExtCom, which has only $O(1)$ rounds.

Below, we give more details about the proofs of Claims 5 and 6 , which can be skipped with little loss of understanding.

Proof sketch of Claim 5. We consider the following sub-hybrids $H^b_{\eta':0}(n,z),\ldots,H^b_{\eta':k}(n,z)$. Recall that a CECom commitment consists of n ExtCom commitments (see Fig. 4).

Sub-hybrid $H^b_{\eta':0}(n,z)$: Sub-hybrid $H^b_{\eta':0}(n,z)$ is the same as $H^b_{\eta'}(n,z)$.

Sub-hybrid $H^b_{\eta':1}(n,z)$ **to Sub-hybrid** $H^b_{\eta':n}(n,z)$: For $k\in[n]$, Sub-hybrid $H^b_{\eta':k}(n,z)$ is the same as $H^b_{\eta':0}(n,z)$ except that in Stage 3 of the left session, for every $j\notin\Gamma$ the jth commitment in the row of CECom is computed as follows. Recall that a CECom commitment consist of n ExtCom commitments. Then, the left committer commits to $0^{|s_j|}$ instead of s_j in the first k ExtCom commitments and commits to s_j in the other $(n-k)$ ExtCom commitments.

Notice that $H^b_{\eta':k}(n,z)$ is identical with $H^b_{\eta'+1}(n,z)$.

We can prove Claim 5 by showing that the output of $H^b_{\eta':k-1}(n,z)$ and that of $H^b_{\eta':k}(n,z)$ are computationally indistinguishable for each $k\in[n]$, and we can prove this indistinguishability similarly to Claim 4. In more detail, we can prove this indistinguishability as follows.

1. Design hybrid experiments $G'^b_{h:1}(n,z),\ldots,G'^b_{h:3}(n,z)$ for $h\in\{k-1,k\}$ in the same way as we design $G^b_{h:1}(n,z),\ldots,G^b_{h:3}(n,z)$ in the proof of Claim 4, where the differences from $G^b_{h:1}(n,z),\ldots,G^b_{h:3}(n,z)$ are the following.
   - $G'^b_{h:1}(n,z),\ldots,G'^b_{h:3}(n,z)$ are defined by modifying $H^b_{\eta':h}(n,z)$ rather than $H^b_h(n,z)$.
   - In the definition of $G'^b_{h:2}(n,z)$, party B in the two-party protocol Π sends party A a row of ExtCom commitments rather than a row of sExtCom, and $\mathcal{A}_{\text{robust}}$ forwards the ExtCom commitments from $\mathcal{E}$ to the internally emulated $\mathcal{A}_{\text{cca}}$ as the kth ExtCom commitment of each CECom commitment in Stage 3 of the left session (rather than forwarding the sExtCom commitments in a row of sExtCom in the left session).
2. Prove, as in the proofs of Claims 7 – 10, that the outputs of $H^b_{\eta':h}(n,z)$, $G'^b_{h:1}(n,z),\ldots,G'^b_{h:3}(n,z)$ are computationally indistinguishable for each $h\in\{k-1,k\}$ and that the outputs of $G'^b_{k-1:3}(n,z)$ and $G'^b_{k:3}(n,z)$ are computationally indistinguishable. The only difference from the proofs of Claims 7 – 10 is that we use the hiding property of ExtCom (rather than that of sExtCom) when proving the indistinguishability between the outputs of $G'^b_{k-1:3}(n,z)$ and $G'^b_{k:3}(n,z)$.
   (When proving these indistinguishabilities, it is required to prove that $\mathcal{A}_{\text{cca}}$ does not cheat in $H^b_{\eta':h}(n,z)$, and this can be proven in the same way as in the proof of Subclaim 4.)
3. Use a hybrid argument to conclude that the output of $H^b_{\eta':k-1}(n,z)$ and that of $H^b_{\eta':k}(n,z)$ are computationally indistinguishable. □

Proof sketch of Claim 6. We can prove the indistinguishability between the outputs of $H^b_{\eta'+1}(n,z)$ and $H^b_{\eta'+2}(n,z)$ similarly to Claim 4. In more detail, we can prove this indistinguishability as follows.

1. Design hybrid experiments $G''^{b}_{h:1}(n,z),\ldots,G''^{b}_{h:3}(n,z)$ for $h \in \{\eta'+1,\eta'+2\}$ in the same way as we design $G^{b}_{h:1}(n,z),\ldots,G^{b}_{h:3}(n,z)$ in the proof of Claim 4, where the difference from $G^{b}_{h:1}(n,z),\ldots,G^{b}_{h:3}(n,z)$ is the following.
   - In the definition of $G''^{b}_{h:2}(n,z)$, party B in the two-party protocol Π sends party A a row of Com commitments rather than a row of $\mathsf{sExtCom}$, and $\mathcal{A}_{\mathrm{robust}}$ forwards the Com commitments from $\mathcal{E}$ to the internally emulated $\mathcal{A}_{\mathrm{cca}}$ as the row of Com in Stage 2 of the left session (rather than forwarding the $\mathsf{sExtCom}$ commitments in a row of $\mathsf{sExtCom}$ in the left session).
2. Prove, as in the proofs of Claims 7 – 10, that the outputs of $H^{b}_{h}(n,z)$, $G''^{b}_{h:1}(n,z),\ldots,G''^{b}_{h:3}(n,z)$ are computationally indistinguishable for each $h \in \{\eta'+1,\eta'+2\}$ and that the outputs of $G''^{b}_{\eta'+1:3}(n,z)$ and $G''^{b}_{\eta'+2:3}(n,z)$ are computationally indistinguishable. The only difference from the proofs of Claims 7 – 10 is that we use the hiding property of Com (rather than that of $\mathsf{sExtCom}$) when proving the indistinguishability between the outputs of $G''^{b}_{\eta'+1:3}(n,z)$ and $G''^{b}_{\eta'+2:3}(n,z)$.
3. Use a hybrid argument to conclude that the output of $H^{b}_{\eta'+1}(n,z)$ and that of $H^{b}_{\eta'+2}(n,z)$ are computationally indistinguishable. □

Combining Claims 4, 5, and 6 and Eq. (14), we obtain Lemma 5. This concludes the proof of Lemma 5. □

5.2. *Proof of Robustness*

Lemma 6. *For any constant $\kappa \in \mathbb{N}$, CCACom is κ-robust.*

Like the robustness of previous CCA-secure commitments [9,10,24], the robustness of our CCA-secure commitment can be shown by using the techniques in the proof of its CCA security.

Proof of Lemma 6. We show that there exists a PPT simulator $\mathcal{S}$ such that for any PPT adversary $\mathcal{A}$ and any κ-round PPT ITM B, the following indistinguishability holds.

$$\begin{aligned}&\left\{\mathsf{output}_{B,\mathcal{A}^{\mathcal{O}}}\left[B(1^n,y)\leftrightarrow\mathcal{A}^{\mathcal{O}}(1^n,z)\right]\right\}_{n\in\mathbb{N},y,z\in\{0,1\}^n}\\ &\overset{c}{\approx}\left\{\mathsf{output}_{B,\mathcal{S}}\left[B(1^n,y)\leftrightarrow\mathcal{S}(1^n,z)\right]\right\}_{n\in\mathbb{N},y,z\in\{0,1\}^n}\end{aligned}\tag{15}$$

First, we consider the following hybrid experiments.

Hybrid $D_0(n,y,z)$: Hybrid $D_0(n,y,z)$ is an experiment in which $\mathcal{A}^{\mathcal{O}}(1^n,x,z)$ interacts with party $B(1^n,y,z)$ as in the definition of robustness, i.e., $\mathcal{A}$ interacts with B while interacting with the committed-value oracle $\mathcal{O}$ in concurrent sessions of CCACom. The output of the experiment is the joint output of B and $\mathcal{A}$.

Hybrid $D_1(n,y,z)$: Hybrid $D_1(n,y,z)$ is the same as $D_0(n,y,z)$ except that at the end of each right session (i.e., each session between $\mathcal{A}$ and $\mathcal{O}$), the oracle $\mathcal{O}$ returns $\mathsf{Value}_{\Gamma}(\boldsymbol{s}^{\mathrm{CEC}})$ to $\mathcal{A}$ rather than $\mathsf{Value}_{\Gamma}(\boldsymbol{s})$ as the committed value of this session, where $\boldsymbol{s}=(s_1,\ldots,s_{10n})$ is the shares that are committed to in the row of Com in

Stage 2, $\boldsymbol{s}^{\mathrm{CEC}} = (s_1^{\mathrm{CEC}}, \ldots, s_{10n}^{\mathrm{CEC}})$ is the shares that are committed to in the row of CECom in Stage 3, and Γ is the subset that is committed to in the $\mathsf{CCACom}^{1:1}$ commitment in Stage 1.

Hybrid $D_2(n, y, z)$: Hybrid $D_2(n, y, z)$ is the same as $D_1(n, y, z)$ except for syntactical differences: Roughly speaking, $D_2(n, y, z)$ is an experiment in which $D_1(n, y, z)$ is executed in such a way that we can use the robust concurrent extraction lemma later. Formally, $D_2(n, y, z)$ is defined as follows. Recall that in the setting of the robust concurrent extraction lemma (Lemma 2), an adversary, $\mathcal{A}_{\mathrm{robust}}$, launches the robust-concurrent attack by interacting with the online extractor $\mathcal{E}$; specifically, $\mathcal{A}_{\mathrm{robust}}$ interacts with $\mathcal{E}$ as a party A of an arbitrary two-party protocol Π while interacting with $\mathcal{E}$ as the committers of CECom concurrently and obtaining a value from $\mathcal{E}$ at the end of each session of CECom (where the values that are returned from $\mathcal{E}$ are supposed to be the committed values of the CECom sessions). Then, consider the following Π and $\mathcal{A}_{\mathrm{robust}}$.

> Π: In Π, the κ-round PPT ITM B (for which we are proving robustness of CCACom) interacts with party A honestly.
>
> $\mathcal{A}_{\mathrm{robust}}$: $\mathcal{A}_{\mathrm{robust}}$ takes a non-uniform advice z and internally executes $D_1(n, y, z)$ with the following changes. (Recall that the execution of $D_1(n, y, z)$ involves an interaction with $\mathcal{A}$.)
>
> - In the session between $\mathcal{A}$ and B, $\mathcal{A}_{\mathrm{robust}}$ forwards all the messages from $\mathcal{A}$ to $\mathcal{E}$ (who internally emulates B) and forwards back all the messages from $\mathcal{E}$ to $\mathcal{A}$.
> - In Stage 3 of each right session, $\mathcal{A}_{\mathrm{robust}}$ receives a row of CECom commitments from $\mathcal{A}$ and forwards it to $\mathcal{E}$ (who internally emulates the receivers of CECom). Let $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{10n})$ denote the responses from $\mathcal{E}$ at the end of the row of the CECom commitments.
> - At the end of each right session, $\mathcal{A}_{\mathrm{robust}}$ sends $\mathsf{Value}_{\Gamma}(\boldsymbol{\alpha})$ to $\mathcal{A}$ as the committed value of this right session.
>
> The output of $\mathcal{A}_{\mathrm{robust}}$ is that of the internally emulated $\mathcal{A}$.

From the robust concurrent extraction lemma, there exists a robust simulator $\mathcal{S}_{\mathrm{robust}}$ such that for the above $\mathcal{A}_{\mathrm{robust}}$, there exists an online extractor $\mathcal{E}$ that satisfies the following.

- For any row of CECom that $\mathcal{A}_{\mathrm{robust}}$ sends to $\mathcal{E}$, let $\boldsymbol{s}^{\mathrm{CEC}} = (s_1^{\mathrm{CEC}}, \ldots, s_{10n}^{\mathrm{CEC}})$ be the shares that are committed to in this row of CECom and $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{10n})$ be the responses from $\mathcal{E}$ at the end of this row. Then, for every $j \in [10n]$, if the jth CECom commitment in this row is valid and its committed value is uniquely determined, $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{10n})$ satisfies $\alpha_j = s_j^{\mathrm{CEC}}$.
- $\mathcal{S}_{\mathrm{robust}}$ can simulate the robust-concurrent attack between $\mathcal{A}_{\mathrm{robust}}$ and $\mathcal{E}$.

Hybrid $D_2(n, y, z)$ is the experiment $\mathrm{REAL}_{\mathcal{E},\Pi}^{\mathcal{A}_{\mathrm{robust}}}(n, y, z)$ of the robust concurrent extraction lemma. The output of $D_2(n, y, z)$ is that of the internally emulated $\mathrm{REAL}_{\mathcal{E},\Pi}^{\mathcal{A}_{\mathrm{robust}}}(n, y, z)$.

Hybrid $D_3(n, y, z)$: Hybrid $D_3(n, y, z)$ differs from $D_2(n, y, z)$ in that the execution of $\mathrm{REAL}_{\mathcal{E},\Pi}^{\mathcal{A}_{\mathrm{robust}}}(n, y, z)$ (i.e., the robust-concurrent attack between $\mathcal{A}_{\mathrm{robust}}$ and

$\mathcal{E}$) is replaced with an interaction between party B of Π and the robust simulator $\mathcal{S}_{\text{robust}}$ of the robust concurrent extraction lemma. The output of $D_3(n, y, z)$ is the joint output of B and $\mathcal{S}_{\text{robust}}$.

For $k \in \{0, \ldots, 3\}$, let $\mathsf{D}_k(n, y, z)$ be the random variable for the output of $D_k(n, y, z)$.

Our simulator $\mathcal{S}$ is the simulator $\mathcal{S}_{\text{robust}}$ in $D_3(n, y, z)$. Notice that from the constructions of the hybrids, we have

$$\mathsf{D}_0(n, y, z) = \mathsf{output}_{B,\mathcal{A}^{\mathcal{O}}}\left[B(1^n, y) \leftrightarrow \mathcal{A}^{\mathcal{O}}(1^n, z)\right],$$
$$\mathsf{D}_3(n, y, z) = \mathsf{output}_{B,\mathcal{S}}\left[B(1^n, y) \leftrightarrow \mathcal{S}(1^n, z)\right].$$

First, we notice that we have $\{\mathsf{D}_0(n, y, z)\}_{n\in\mathbb{N},y,z\in\{0,1\}^n} \overset{s}{\approx} \{\mathsf{D}_1(n, y, z)\}_{n\in\mathbb{N},y,z\in\{0,1\}^n}$. This is because from the same argument as in the proof of Claim 7, we can show that the view of $\mathcal{A}$ in $D_1(n, y, z)$ is statistically close to that in $D_0(n, y, z)$.

Next, we notice that we have $\{\mathsf{D}_1(n, y, z)\}_{n\in\mathbb{N},y,z\in\{0,1\}^n} \overset{s}{\approx} \{\mathsf{D}_2(n, y, z)\}_{n\in\mathbb{N},y,z\in\{0,1\}^n}$. This is because from the same argument as in the proof of Claim 8, we can show that an execution of $D_1(n, y, z)$ is statistically simulated in $D_2(n, y, z)$.

Next, we notice that we have $\{\mathsf{D}_2(n, y, z)\}_{n\in\mathbb{N},y,z\in\{0,1\}^n} \overset{s}{\approx} \{\mathsf{D}_3(n, y, z)\}_{n\in\mathbb{N},y,z\in\{0,1\}^n}$. This is because from the robust concurrent extraction lemma, we can show that the interaction between $\mathcal{S}_{\text{robust}}$ and B in $D_3(n, y, z)$ is statistically close to that between $\mathcal{A}_{\text{robust}}$ and $\mathcal{E}$ in $D_2(n, y, z)$.

Now, from the hybrid argument, we obtain Indistinguishability (15). □

Combining Lemmas 5 and 6 , we obtain Theorem 1. This concludes the proof of Theorem 1. □

6. Black-Box Composable MPC Protocol

In this section, we show our black-box construction of a general MPC protocol. Our protocol is secure in the angel-based UC framework [9,10,32]. Roughly speaking, this framework (called the $\mathcal{H}$-EUC framework) is the same as the UC framework [2] except that both the adversary and the environment in the real and ideal worlds have access to a super-polynomial-time functionality $\mathcal{H}$ called an *angel* (or a *helper*). For details, see [9,10,32].

We use the results of Canetti et al. [9,10] and Lin and Pass [24]. Let $\langle C, R\rangle$ be any R_{CCA}-round robust CCA-secure commitment scheme, $\langle S, R\rangle$ be any R_{OT}-round semi-honest oblivious transfer protocol, and $\mathcal{H}$ be a helper that breaks $\langle C, R\rangle$ in essentially the same way as the committed-value oracle of $\langle C, R\rangle$ does. Then, Lin and Pass [24] showed that there exists a black-box $O(\max(R_{\mathsf{OT}}, R_{\mathsf{CCA}}))$-round protocol that securely realizes the ideal oblivious transfer functionality $\mathcal{F}_{OT}$ in the $\mathcal{H}$-EUC framework.

Theorem 2. ([24]) *Assume the existence of an R_{CCA}-round robust CCA-secure commitment scheme $\langle C, R\rangle$ and the existence of an R_{OT}-round semi-honest oblivious transfer protocol $\langle S, R\rangle$. Then, there exists an $O(\max(R_{\mathsf{CCA}}, R_{\mathsf{OT}}))$-round protocol that $\mathcal{H}$-EUC-*

realizes $\mathcal{F}_{OT}$. Furthermore, this protocol uses $\langle C, R\rangle$ and $\langle S, R\rangle$ only in a black-box way.

In [9,10], Canetti et al. showed the following.

Theorem 3. ([9,10]) *For every well-formed functionality $\mathcal{F}$, there exists a constant-round $\mathcal{F}_{OT}$-hybrid protocol that $\mathcal{H}$-EUC-realizes $\mathcal{F}$.*

Then, we obtain the following theorem by combining Theorems 1, 2, and 3.

Theorem 4. *Assume the existence of R_{OT}-round semi-honest oblivious transfer protocols. Then, there exists a super-polynomial-time helper $\mathcal{H}$ such that for every well-formed functionality $\mathcal{F}$, there exists a $\max(\widetilde{O}(\log^2 n), O(R_{\mathsf{OT}}))$-round protocol that $\mathcal{H}$-EUC-realizes $\mathcal{F}$. Furthermore, this protocol uses the underlying oblivious transfer protocol only in a black-box way.*

Appendix A: One-One CCA Commitment for Long Tags from Parallel CCA Commitment for Short Tags

Lemma 7. *Let $r(\cdot)$ and $t(\cdot)$ be arbitrary functions such that $t(n) = O(\log n)$, and let* CCACom *be an $r(n)$-round commitment scheme that satisfies strong computational binding property and parallel CCA security for tags of length $t(n)$. Then, there exists an $r(n)$-round commitment scheme* $\mathsf{CCACom}^{1:1}$ *that satisfies strong computational binding property and one-one CCA security for tags of length $2^{t(n)-1}$. Furthermore, if* CCACom *uses the underlying one-way function only in a black-box way, so does* $\mathsf{CCACom}^{1:1}$.

Proof. $\mathsf{CCACom}^{1:1}$ is shown in Fig. 15. The strong computational binding property follows from that of CCACom. Thus, it remains to show that $\mathsf{CCACom}^{1:1}$ is one-one CCA secure.

We show that for any PPT adversary $\mathcal{A}$ that interacts with $\mathcal{O}$ only in a single session, the following are computationally indistinguishable:

- $\{\mathsf{IND}_0(\mathsf{CCACom}^{1:1}, \mathcal{A}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$
- $\{\mathsf{IND}_1(\mathsf{CCACom}^{1:1}, \mathcal{A}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$

Commit Phase

The committer C and the receiver R receive common inputs 1^n and $\mathsf{tag} \in \{0,1\}^{2^{t(n)-1}}$. To commit to $v \in \{0,1\}$, the committer C chooses random $v_1, \ldots, v_{2^{t(n)-1}} \in \{0,1\}^n$ such that $v = \bigoplus_j v_j$, and for each $j \in [2^{t(n)-1}]$ in parallel, C commits to v_j by using CCACom with tag (j, tag_j), where tag_j is the j-th bit of tag.

Decommit Phase

C sends v to R and decommits all the CCACom commitments.

Fig. 15. One-one CCA-secure commitment scheme $\mathsf{CCACom}^{1:1}$.

Without loss of generality, we can assume that the tag that $\mathcal{A}$ uses in the right session is always different from the tag that $\mathcal{A}$ uses in the left session. (This is because instead of using the same tag in the left and right sessions, $\mathcal{A}$ can use different tags in the left and right sessions and then output $\perp$; recall that the output of the experiment is $\perp$ whenever $\mathcal{A}$ uses the same tag in the left and right sessions.)

Assume for contradiction that there exist a PPT distinguisher $\mathcal{D}$ and a polynomial $p(\cdot)$ such that for infinitely many n, there exists $z \in \{0,1\}^*$ such that $\mathcal{D}$ distinguishes $\mathsf{IND}_0(\mathsf{CCACom}^{1:1}, \mathcal{A}, n, z)$ and $\mathsf{IND}_1(\mathsf{CCACom}^{1:1}, \mathcal{A}, n, z)$ with advantage at least $1/p(n)$. In the following, we fix any such n and z.

Let us consider the following PPT adversary $\mathcal{B}$ against CCA security of CCACom. $\mathcal{B}$ internally invokes $\mathcal{A}$ and simulates the experiment $\mathrm{IND}_0(\mathsf{CCACom}^{1:1}, \mathcal{A}, n, z)$ for $\mathcal{A}$ as follows. First, $\mathcal{B}$ chooses random $j^* \in [2^{t(n)-1}]$, and for each $j \in [2^{t(n)-1}] \setminus \{j^*\}$, $\mathcal{B}$ chooses random $v_j \in \{0,1\}^n$. Then, in the left session, when $\mathcal{A}$ outputs challenge values $m_0, m_1 \in \{0,1\}^n$ and tag $\mathsf{tag} = (\mathsf{tag}_1, \ldots, \mathsf{tag}_{2^{t(n)-1}})$, $\mathcal{B}$ sets $v_{j^*}^{(b)} := m_b \oplus \bigoplus_{j \neq j^*} v_j$ for each $b \in \{0,1\}$ and sends challenge $v_{j^*}^{(0)}, v_{j^*}^{(1)}$ and tag $(j^*, \mathsf{tag}_{j^*}) \in \{0,1\}^{t(n)}$ to the external left committer. When $\mathcal{B}$ receives a CCACom commitment from the left committer (the committed value is either $v_{j^*}^{(0)}$ or $v_{j^*}^{(1)}$), $\mathcal{B}$ forwards it to $\mathcal{A}$. At the same time, $\mathcal{B}$ generates CCACom commitments to $(v_j)_{j \neq j^*}$ and sends them to $\mathcal{A}$. In the right session, $\mathcal{B}$ forwards a $\mathsf{CCACom}^{1:1}$ commitment from $\mathcal{A}$ to $\mathcal{O}$ as $2^{t(n)-1}$ parallel commitments of CCACom with tags $\{(j, \widetilde{\mathsf{tag}}_j)\}_{j \in [2^{t(n)-1}]}$. Then, $\mathcal{B}$ receives $(v_1, \ldots, v_{2^{t(n)-1}})$ from $\mathcal{O}$, and if $v_j \neq \perp$ for all $j \in [2^{t(n)-1}]$, $\mathcal{B}$ returns $v := \bigoplus_j v_j$ to $\mathcal{A}$; if $v_j = \perp$ for some j, $\mathcal{B}$ returns $\perp$ to $\mathcal{A}$. Let y be the output of the simulated experiment $\mathrm{IND}_0(\mathsf{CCACom}^{1:1}, \mathcal{A}, n, z)$, and let $\beta \leftarrow \mathcal{D}(y)$. Then, if $\mathsf{tag}_{j^*} = \widetilde{\mathsf{tag}}_{j^*}$, $\mathcal{B}$ outputs fail, and otherwise, $\mathcal{A}$ outputs fail with probability $(N-1)/N$ and outputs β with probability $1/N$, where $N = |\{j \text{ s.t. } \mathsf{tag}_j \neq \widetilde{\mathsf{tag}}_j\}|$ is the Hamming distance between tag and $\widetilde{\mathsf{tag}}$.

We reach a contradiction by showing that $\mathcal{B}$ breaks the CCA security of CCACom; in particular,

$$\left|\Pr\left[\mathsf{IND}_0(\mathsf{CCACom}, \mathcal{B}, n, z) = 1\right] - \Pr\left[\mathsf{IND}_1(\mathsf{CCACom}, \mathcal{B}, n, z) = 1\right]\right| \geq \frac{1}{p(n) \cdot \mathsf{poly}(n)}.$$

For $b \in \{0,1\}$, let β_b be the random variable representing the value of β in $\mathrm{IND}_b(\mathsf{CCACom}, \mathcal{B}, n, z)$ and abort_b be the event that $\mathcal{B}$ outputs fail in $\mathrm{IND}_b(\mathsf{CCACom}, \mathcal{B}, n, z)$. Since $\mathcal{B}$ internally simulates $\mathrm{IND}_0(\mathsf{CCACom}^{1:1}, \mathcal{A}, n, z)$ or $\mathrm{IND}_1(\mathsf{CCACom}^{1:1}, \mathcal{A}, n, z)$ perfectly depending on the value that is committed to in the left session, we have

$$\Pr[\beta_b = 1] = \Pr\left[\mathcal{D}(\mathsf{IND}_b(\mathsf{CCACom}^{1:1}, \mathcal{A}, n, z)) = 1\right].$$

Hence, from our assumption, we have

$$|\Pr[\beta_0 = 1] - \Pr[\beta_1 = 1]| \geq \frac{1}{p(n)}.$$

Also, since we assume that it always holds that $\mathsf{tag} \neq \widetilde{\mathsf{tag}}$, for each $b \in \{0, 1\}$ we have

$$\Pr\left[\neg\mathsf{abort}_b\right] = \frac{N}{2^{t(n)-1}} \cdot \frac{1}{N} = \frac{1}{2^{t(n)-1}}.$$

Note that when $\mathsf{tag}_{j^*} \neq \widetilde{\mathsf{tag}}_{j^*}$, a tag $(j, \widetilde{\mathsf{tag}}_j)$ in the right session is different from the tag $(j^*, \mathsf{tag}_{j^*})$ in the left session for each $j \in [2^{t(n)-1}]$. Hence, when abort_b does not occur, the output of $\mathsf{IND}_b(\mathsf{CCACom}, \mathcal{B}, n, z)$ is β_b. Thus, we have

$$\begin{aligned}
&\left|\Pr\left[\mathsf{IND}_0(\mathsf{CCACom}, \mathcal{B}, n, z) = 1\right] - \Pr\left[\mathsf{IND}_1(\mathsf{CCACom}, \mathcal{B}, n, z) = 1\right]\right| \\
&= \left|\Pr\left[\beta_0 = 1 \wedge \neg\mathsf{abort}_0\right] - \Pr\left[\beta_1 = 1 \wedge \neg\mathsf{abort}_1\right]\right| \\
&= |\Pr[\beta_0 = 1] - \Pr[\beta_1 = 1]| \cdot \frac{1}{2^{t(n)-1}} \\
&\geq \frac{1}{p(n) \cdot \mathsf{poly}(n)}.
\end{aligned}$$

In the third line, we use $\Pr\left[\beta_b = 1 \wedge \neg\mathsf{abort}_b\right] = \Pr[\beta_b = 1] \cdot \Pr\left[\neg\mathsf{abort}_b\right]$ (i.e., the independence between the event abort_b and the event that $\beta_b = 1$, which follows from the fact that abort_b always occurs with probability $1/2^{t(n)-1}$, independently of the values of tag and $\widetilde{\mathsf{tag}}$). This concludes the proof. □

References

[1] B. Barak, How to play almost any mental game over the net - Concurrent composition via super-polynomial simulation, in *46th FOCS* (IEEE Computer Society Press, October 2005), pp. 543–552

[2] R. Canetti, Universally composable security: A new paradigm for cryptographic protocols, in *42nd FOCS* (IEEE Computer Society Press, October 2001), pp. 136–145

[3] S.G. Choi, D. Dachman-Soled, T. Malkin, Hoeteck Wee, Black-box construction of a non-malleable encryption scheme from any semantically secure one, in R. Canetti, editor, *TCC 2008*, vol. 4948 of *LNCS* (Springer, Heidelberg, March 2008), pp. 427–444

[4] S.G. Choi, D. Dachman-Soled, T. Malkin, H. Wee, Simple, black-box constructions of adaptively secure protocols, in O. Reingold, editor, *TCC 2009*, vol. 5444 of *LNCS* (Springer, Heidelberg, March 2009), pp. 387–402

[5] S.G. Choi, D. Dachman-Soled, T. Malkin, H. Wee, A black-box construction of non-malleable encryption from semantically secure encryption. *J. Cryptol.* (2017)

[6] R. Canetti, M. Fischlin, Universally composable commitments, in J. Kilian, editor, *CRYPTO 2001*, vol. 2139 of *LNCS* (Springer, Heidelberg, August 2001), pp. 19–40

[7] R. Canetti, E. Kushilevitz, Y. Lindell, On the limitations of universally composable two-party computation without set-up assumptions. *J. Cryptol.* **19**(2), 135–167 (2006)

[8] R. Canetti, Y. Lindell, R. Ostrovsky, A. Sahai, Universally composable two-party and multi-party secure computation, in *34th ACM STOC* (ACM Press, May 2002), pp. 494–503

[9] R. Canetti, H. Lin, R. Pass, Adaptive hardness and composable security in the plain model from standard assumptions, in *51st FOCS* (IEEE Computer Society Press, October 2010), pp. 541–550

[10] R. Canetti, H. Lin, R. Pass, Adaptive hardness and composable security in the plain model from standard assumptions. *SIAM J. Comput.* **45**(5), 1793–1834 (2016)
[11] D. Dolev, C. Dwork, M. Naor, Nonmalleable cryptography. *SIAM J. Comput.* **30**(2), 391–437 (2000)
[12] C. Dwork, M. Naor, O. Reingold, L. Stockmeyer, Magic functions. *J. ACM* **50**(6), 852–921 (2003)
[13] S. Garg, V. Goyal, A. Jain, A. Sahai, Concurrently secure computation in constant rounds, in D. Pointcheval and T. Johansson, editors, *EUROCRYPT 2012*, vol. 7237 of *LNCS* (Springer, Heidelberg, April 2012), pp. 99–116
[14] V. Goyal, C.-K. Lee, R. Ostrovsky, I. Visconti, Constructing non-malleable commitments: A black-box approach, in *53rd FOCS* (IEEE Computer Society Press, October 2012), pp. 51–60
[15] V. Goyal, H. Lin, O. Pandey, R. Pass, A. Sahai, Round-efficient concurrently composable secure computation via a robust extraction lemma, in Y. Dodis and J.B. Nielsen, editors, *TCC 2015, Part I*, vol. 9014 of *LNCS* (Springer, Heidelberg, March 2015), pp. 260–289
[16] O. Goldreich, S. Micali, A. Wigderson, How to play any mental game or A completeness theorem for protocols with honest majority, in A. Aho, editor, *19th ACM STOC* (ACM Press, May 1987), pp. 218–229
[17] V. Goyal, Constant round non-malleable protocols using one way functions, in L. Fortnow and S.P. Vadhan, editors, *43rd ACM STOC* (ACM Press, June 2011), pp. 695–704
[18] I. Haitner, Semi-honest to malicious oblivious transfer - the black-box way, in R. Canetti, editor, *TCC 2008*, vol. 4948 of *LNCS* (Springer, Heidelberg, March 2008), pp. 412–426
[19] I. Haitner, Y. Ishai, E. Kushilevitz, Y. Lindell, E. Petrank, Black-box constructions of protocols for secure computation. *SIAM J. Comput.* **40**(2), 225–266 (2011)
[20] J. Håstad, R. Impagliazzo, L.A. Levin, M. Luby, A pseudorandom generator from any one-way function. *SIAM J. Comput.* **28**(4), 1364–1396 (1999)
[21] Y. Ishai, E. Kushilevitz, Y. Lindell, E. Petrank, Black-box constructions for secure computation, in J.M. Kleinberg, editor, *38th ACM STOC* (ACM Press, May 2006), pp. 99–108
[22] Y. Ishai, M. Prabhakaran, A. Sahai, Founding cryptography on oblivious transfer - efficiently, in D. Wagner, editor, *CRYPTO 2008*, vol. 5157 of *LNCS* (Springer, Heidelberg, August 2008), pp. 572–591
[23] S. Kiyoshima, Y. Manabe, T. Okamoto, Constant-round black-box construction of composable multi-party computation protocol, in Y. Lindell, editor, *TCC 2014*, vol. 8349 of *LNCS* (Springer, Heidelberg, February 2014), pp. 343–367
[24] H. Lin, R. Pass, Black-box constructions of composable protocols without set-up, in R. Safavi-Naini and R. Canetti, editors, *CRYPTO 2012*, vol. 7417 of *LNCS* (Springer, Heidelberg, August 2012), pp. 461–478
[25] H. Lin, R. Pass, M. Venkitasubramaniam, Concurrent non-malleable commitments from any one-way function, in R. Canetti, editor, *TCC 2008*, vol. 4948 of *LNCS* (Springer, Heidelberg, March 2008), pp. 571–588
[26] T. Malkin, R. Moriarty, N. Yakovenko, Generalized environmental security from number theoretic assumptions, in S. Halevi and T. Rabin, editors, *TCC 2006*, vol. 3876 of *LNCS* (Springer, Heidelberg, March 2006), pp. 343–359
[27] D. Micciancio, S.J. Ong, A. Sahai, S.P. Vadhan, Concurrent zero knowledge without complexity assumptions, in S. Halevi and T. Rabin, editors, *TCC 2006*, vol. 3876 of *LNCS* (Springer, Heidelberg, March 2006), pp. 1–20
[28] M. Naor, Bit commitment using pseudorandomness. *J. Cryptol.* **4**(2), 151–158 (1991)
[29] R. Pass, Simulation in quasi-polynomial time, and its application to protocol composition, in E. Biham, editor, *EUROCRYPT 2003*, vol. 2656 of *LNCS* (Springer, Heidelberg, May 2003), pp. 160–176
[30] R. Pass, H. Lin, M. Venkitasubramaniam, A unified framework for UC from only OT, in X. Wang and K. Sako, editors, *ASIACRYPT 2012*, vol. 7658 of *LNCS* (Springer, Heidelberg, December 2012), pp. 699–717
[31] M. Prabhakaran, A. Rosen, A. Sahai, Concurrent zero knowledge with logarithmic round-complexity, in *43rd FOCS* (IEEE Computer Society Press, November 2002), pp. 366–375
[32] M. Prabhakaran, A. Sahai, New notions of security: Achieving universal composability without trusted setup, in L. Babai, editor, *36th ACM STOC* (ACM Press, June 2004), pp. 242–251
[33] R. Pass, W.-L.D. Tseng, M. Venkitasubramaniam, Concurrent zero knowledge, revisited. *J. Cryptol.* **27**(1), 45–66 (2014).
[34] R. Pass, H. Wee, Black-box constructions of two-party protocols from one-way functions, in O. Reingold, editor, *TCC 2009*, vol. 5444 of *LNCS* (Springer, Heidelberg, March 2009), pp. 403–418

[35] R. Richardson, J. Kilian, On the concurrent composition of zero-knowledge proofs, in J. Stern, editor, *EUROCRYPT'99*, vol. 1592 of *LNCS* (Springer, Heidelberg, May 1999), pp. 415–431
[36] H. Wee, Black-box, round-efficient secure computation via non-malleability amplification, in *51st FOCS* (IEEE Computer Society Press, October 2010), pp. 531–540