Tight Setup Bounds for Identifiable Abort

Nicholas Brandt

Department of Computer Science, ETH Zurich, nicholas.brandt@inf.ethz.ch

Abstract. We present fundamental (in-)feasibility results for the strongest security notion for Secure Multi-Party Computation (MPC) that is achievable when a majority of parties is malicious, i.e. security with Identifiable Abort. As general Universally Composable (UC) MPC requires a setup, typically in the form of a Common Reference String or Common-Randomness, we investigate whether the setup must provide randomness to *all* parties.
Given broadcast, we give tight bounds for the necessary and sufficient setup cardinality, i.e. number of participating parties, for UC-MPC protocols with Identifiable Abort. Concretely, we improve previous upper bounds by constructing Secure Function Evaluation for n parties (h of which are honest) from setups of cardinality $\beta := \min(n, \lfloor n/h \rfloor + \lfloor (n-1)/h \rfloor - 1)$ and broadcast.
Conversely, we present the first general lower bound on the minimal setup cardinality for Identifiable Abort by proving that setups of cardinality $\beta - 1$ plus broadcast are insufficient even for a commitment among n parties. Somewhat surprisingly, we show that Oblivious Transfer plus broadcast is sufficient for $n = 2h \geq 2$ which is consistent with the fact that in two-party MPC Identifiable Abort comes for free. We present the results in the Universal Composability (UC) framework and assume the setup functionalities to be secure with Identifiable Abort.
Our constructions yield an efficient (poly-time) protocol for any $n \in \text{poly}(\lambda)$ where λ is the security parameter if at least a constant fraction $h \in \Theta(n)$ of parties is honest. However for $h \in o(n)$ our results suggest that for efficient protocols the overall number of parties n is limited quite severely by $\binom{n}{\beta} \in \text{poly}(\lambda)$.

Keywords: Multi-Party Computation · Identifiable Abort · Dishonest Majority · Conflict Graph

1 Introduction

Because a Universally Composable (UC) commitment is impossible in the plain model [CF01], a setup is necessary—typically in the form of a Common Reference String that involves all parties. An important question is whether all parties must be involved in the setup, or if setups with fewer parties suffice for general MPC. Since "smaller" setups may require less communication and may be more efficiently realizable via hardware assumptions or through smaller security parameters [GIS+10; CK88; Cré97], protocols based on smaller setups may be much more efficient in practice.

This question of the *Minimal Complete Cardinality* has been formally posed by Fitzi, Garay, Maurer, and Ostrovsky [FGM+01] for MPC with guaranteed output delivery without a broadcast. In the honest-majority case Ben-Or, Goldwasser, and Wigderson [BGW88] showed that pairwise setups suffice iff $t < n/3$, and Beaver [Bea90] and Rabin and Ben-Or [RB89] showed that pairwise channels plus broadcast suffice for $t < n/2$. On the other hand, Cleve [Cle86] showed that in the dishonest-majority case the notion of fairness is impossible, and hence also guaranteed output delivery. To sidestep this impossibility protocols that are secure against a dishonest majority settle for *security with abort* [IPS08] where the adversary can abort the protocol at any point. Here Oblivious Transfer (OT), a two-party setup, allows for secure protocols even against a dishonest majority. While this ensures that the adversary cannot tamper with the parties' outcome, i.e. manipulate the outcome of a coin-toss, it opens the door for Denial-of-Service attacks. To mitigate this caveat the main idea has been to identify some dishonest party upon abort. This has been considered—among others—by Hofheinz and Müller-Quade [HM04] as *cheater identification*, Aumann and Lindell [AL07] in the form of *covert security*, and by Ishai, Ostrovsky, and Seyalioglu [IOS12] and Ishai, Ostrovsky, and Zikas [IOZ14] as *Identifiable Abort* (IA). In the light of the impossibility of fairness, the notion of IA is the strongest one that is possibly achievable in the dishonest-majority setting.

Motivation: a case for small setups. We want to motivate the relevance of MPC from small setups in both theory in practice. On a theoretical level, one might view the result of a Secure Function Evaluation as a large truth table and the protocol itself as a circuit that computes that truth table. Suppose the protocol has runtime at most $r \in \text{poly}(\lambda)$ and each party i has input and randomness $x_i \in \{0,1\}^\lambda$ then the overall input to the circuit is a matrix $\{0,1\}^{n \times r(\lambda)}$ composed of n rows per λ bits. Local operations on the input and randomness of a party may then be represented by any gate operating on a single row. For the circuit computing non-local functions, i.e. values that depend on all parties' inputs, the circuit also need gates that operator on multiple rows simultaniously. For example the Correlated Randomness setup of [IOZ14] may be viewed as a gate operating on all rows at once. However, one might ask the question: what is the minimal gate size that allows for poly-sized SFE-circuits? A lower bound for the minimal complete cardinality for SFE with IA directly translates into a lower bound for the minimal gate size for circuits realizing SFE with IA. Since circuit lower bounds are notoriously hard to prove, we think this connection legitimizes the quest for minimal complete setups in MPC.

From a more practical point of view, consider a scenario where n parties want to compute some function but they only have a unstable connection to the network which sporadically allows them to communicate with other parties that are online at the same time. Say at any given time $i \in \mathbb{N}$ each party $j \in [n]$ is online with probability $p < 1$, and suppose that any setup can be computed within one timestep. To realize an n-party Correlated Randomness setup all n parties need to be online at the exact same time, i.e. the probability of success at each timestep is p^n and hence the expected number of timesteps $E_n = p^{-n}$ is exponential in n. Conversely, for excepted polynomial runtime $E_n \in \text{poly}(\lambda)$ in the security parameter λ only logarithmically many $n \in \mathcal{O}(\ln \lambda)$ parties are supported.
On the other hand, suppose that an arbitrarily small constant fraction of parties is honest, i.e. $h \in \Theta(n)$. Then the probability of realizing one setup of cardinality $\beta \in \Theta(1)$ is constant $p^\beta \in \Theta(1)$ at each timestep. Hence the expected number of timesteps until all $\binom{n}{\beta}$ succeed is polynomially bounded by $E_\beta \leq \binom{n}{\beta} p^{-\beta} \in \text{poly}(n)$. As such, for expected polynomial runtime $E_\beta \in \text{poly}(\lambda)$ polynomially many $n \in \text{poly}(\lambda)$ parties are supported.
This exponential gap in the expected runtime resp. number of supported parties should exemplify the importance of small setups in a practical scenario. We adapt the question of the Minimal Complete Cardinality originally posed by [FGM+01] to the setting of IA with broadcast.

Related work on Identifiable Abort.

- Ishai, Ostrovsky, and Seyalioglu [IOS12] give a universal n-party setup functionality based on *identifiable secret sharing* and rule out any two-party setup plus broadcast for $t \geq 2n/3$.
- Ishai, Ostrovsky, and Zikas [IOZ14] give a universal n-party setup functionality—the Correlated Randomness Model. This allows to split the protocol in a offline and an online phase where the computationally secure offline phase is oblivious to the actual function and the inputs and the online phase enjoys statistical security.
- Brandt, Maier, Müller, and Müller-Quade [BMM+20] give a first upper bound by constructing a protocol from setups of cardinality $n-1$ and broadcast for $t \leq n-3$. Their construction is based on the observation that replacing two-party OT in the IPS-compiler [IPS08] with an n-party *committed OT* [Cré90; CvT95] directly lifts the resulting protocol to Identifiable Abort.
- Simkin, Siniscalchi, and Yakoubov [SSY21] improve this result slightly by constucting a Correlated Randomness setup for n parties from setups of cardinality $n-1$ for $t \leq n-2$. To this end they introduce a new kind of identifiable secret sharing based on [IOZ14].

Moreover, the settings of Identifiable Abort has recently been adopted to quantum MPC [ACC+20].

Our Contribution. We explore the dishonest-majority setting for Identifiable Abort in the UC framework thoroughly.
In Appendix A we prove that no setup of cardinality $\min(n-1, \lfloor n/h \rfloor + \lfloor (n-1)/h \rfloor - 2)$ is sufficient for a commitment among n parties (at least h of which are honest) even when a broadcast is available. To this end we prove an *abort-lemma* that provides a strategy for the environment to abort setups in such any protocol must either violate the hiding or binding property of the commitment.
On the positive side, in Appendix B we show that our bound is tight by giving a protocol to realize a commitment among n parties from setups of cardinality β and broadcast. In a second step,

we construct a committed version of OT[1] for n parties from setups for β parties and an n-party commitment.
We note that our results are strong in the sense that our impossibility holds even for computationally bounded (efficient) environment, and our constructions are secure against computationally unbounded environments.
Finally, in Section 5 we discuss the limitation $\binom{n}{\beta} \in \text{poly}(\lambda)$ of the number of parties that seems inherent in constructions from smaller setups, and given an outlook.

As such our results subsume or improve upon all previously listed constructions and impossibilities in a unified way.

- The lower bound of 3 from [IOS12] is raised to 4 for $t \geq 2n/3 \geq 4$.
- The upper bound of $n-1$ from [SSY21] reduced to the optimal $n-2$ for $t \geq n-2$.
- The upper bound of n from [IOZ14] is shown to be tight for $t \geq n-1$.

Finally, we want to remark that while one might regard broadcast as an unwarranted assumption for constructing MPC it is still an intuitively weaker assumption than a Common Reference String [CF01] or even arbitrary Common Randomness [IOZ14]. Furthermore, the broadcast makes our impossibility result stronger than an impossibility for the settings where parties can only communicate via pairwise channels.

2 Setting

We focus on **static corruptions** of an arbitrary number of parties at the onset of the protocol. The maximal number of malicious parties is denoted by t and the minimal number of honest parties by $h \coloneqq n - t$.
Our constructions enjoy **statistical security** and our impossibility holds against any adversary, in particular for computationally bounded ones. No computational assumptions are made since we only assume the existence of hybrid functionalities (setups).
We assume that all messages sent between parties and ideal functionalities are **authenticated**.

2.1 The UC Framework & Identifiable Abort

We perform our analysis in the Universal Composability (UC) framework of Canetti [Can01]. In this strong version of *simulation-based* security, an environment (distinguisher) controls the adversary interactively, in particular the simulator cannot rewind the enviroment. This notion ensures that protocol instances can be composed arbitrarly.

Technically, the desired behaviour of a computation is specified by an *ideal functionality*, usually denote by $\mathcal{F}$, and security of a protocol π is proved by giving a simulator $\mathcal{S}$ that produces a simulated protocol transcript only using the ideal functionality. In other words, everything that a real adversary $\mathcal{A}$ can do in a protocol the simulator can pretend to do with the ideal functionality. Due to a fundamental impossibility of UC commitments by Canetti and Fischlin [CF01] protocols are often formulated as *hybrid* protocols that allow the parties access to setups which are again modeled as ideal (hybrid) functionalities. By the Composition Theorem, these hybrid setup functionalities may in turn be realized by any secure protocol.

Because an asynchronous network is essentially susceptible to Denial-of-Service attacks [CM89; BCG93] we assume a **synchronous** communication network.[2] For the sake of simplicity we omit explicit use of the functionality $\mathcal{F}_{\mathsf{SYN}}$ in our analysis. Instead we assume that the parties are activated in rounds, although the order of activations within one round is controlled by the environment. In particular, we assume that parties are implicitly aware of the current round number. The round based communication model is particularly useful in the setting of Identifiable Abort with broadcast as it allows parties to check whether a party honestly responded to detected misbehaviour. We describe this in more detail in Section 4.

As described in Section 1 Identifiable Abort (IA) is the strongest security notion that is not ruled out by Cleve [Cle86]. Formally introduced in [IOZ14] an ideal functionality with IA allows the adversary to send $(\texttt{abort}, \mathsf{P})$ where P is a malicious party—which we call *disruptor*—so that

[1] The idea of a committed OT was first introduced by Crépeau [Cré90] and Crépeau, van de Graaf, and Tapp [CvT95].
[2] Compare Section 6.2.3 of the 2020 version 20200212:021048 of [Can00]

all parties receive $(\texttt{abort}, \mathsf{P})$. The main idea is that although the protocol is aborted, the parties all agree on a common malicious party. So that they can restart the protocol and replace the identified party with so default input.
In particular, we assume that all setups have Identifiable Abort themselves and also allow multiple parties to be identified at once. That is for any protocol with parties P, the adversary can input $(\texttt{abort}, C')$ into any setup with participating parties $P' \subseteq P$ where $\emptyset \neq C' \subseteq P'$ is the set of identified disruptors, and all parties P' obtain $(\texttt{abort}, C')$.
The work [BMM+20] formalized the following natural and intuitive way of handling aborts of setups. Specifically, whenever a setup functionality with parties P' is aborted with $(\texttt{abort}, C')$, we can require all honest parties in P' to declare a conflict with all parties in C'. At the onset of the next round each honest party can check if indeed all parties $P' \setminus C'$ declared the required conflicts. Any honest party should then—by design of our protocol—also declare conflict with any party in P' that did not declare with all disruptors C'—we call these parties *loyalists*. If honest parties behave this way, then after the abort of a setup with parties P' the broadcasted conflicts are always two partitions $P_1 \cup P_2 = P'$ such that all parties in P_1 are in conflict with all parties from P_2. This condition is called *biseparation* in [BMM+20]. We recall the graph-theoretical properties of [BMM+20] in more detail in Section 4.

Remark 1. We note that for two-party functionalities, such as OT, security with abort is equivalent to IA.

Similar in effect to the n-party *Universal Black Box* setup of [FGM+01] for guaranteed output delivery [IOZ14] give an n-party *Correlated Randomness* setup for IA—both are oblivious to the evaluated functionality and inputs. There has also been much effort in producing practical MPC protocols with IA in [DPS+12; SF16; BOS16; BOS+20] among others. However, we are primarily interested in the fundamental (in-)feasibility of achieving MPC with IA from minimal setups. In particular, we do not consider the round complexity of our protocols and leave the optimization to future work.

3 Preliminaries

3.1 Notation

We use λ for the (statistical) security parameter, n for the overall number of parties, h for the number of (guaranteed) honest parties and t for the number of (potentially) malicious parties. We also use $\mathrm{negl}(x)$ and $\mathrm{owhl}(x)$ to denote the set of negligible resp. overwhelming functions w.r.t. x.

Notation 1 (Subsets). *For any set V and $k \in \mathbb{N}$ s.t. $k \leq |V|$ we denote by $\binom{V}{k}$ the set of subsets of cardinality k.*

Notation 2 (Union of disjoint sets). *For any two disjoint sets V and V' s.t. $V \cap V' = \emptyset$ we emphasize their disjointness in the union operation as $V \cup V'$.*

In particular, we use the fact that $|V \cup V'| = |V| + |V'|$.

We use the following notation to clarify our Identifiable Abort property:

Notation 3 (Functionalities with IA). *We denote by $\mathcal{F}^n$ an n-party functionality with Identifiable Abort.*

Notation 4 (Protocol construction). *For any set of ideal functionalities F and any ideal functionality $\mathcal{F}$ we write $F \rightsquigarrow \mathcal{F}$, iff there is a protocol π^F that securely UC-realizes $\mathcal{F}$ in the F-hybrid model. More formally:*

$$F \rightsquigarrow \mathcal{F} \iff \exists \pi^F : \pi^F \geq \mathcal{F} \ . \tag{1}$$

Conversely, we write

$$F \not\rightsquigarrow \mathcal{F} \iff \forall \pi^F : \pi^F \not\geq \mathcal{F} \ . \tag{2}$$

We furthermore use the additional notation $F \overset{\text{stat}}{\rightsquigarrow} \mathcal{F}$ resp. $F \overset{\text{comp}}{\rightsquigarrow} \mathcal{F}$ to denote the contruction is secure against an computationally unbounded resp. efficient environment.

Notation 5 (Minimal complete cardinality generalized from [FGM+01]). *For any number of parties $n \in \mathbb{N}_{\geq 2}$ let $k \in \mathbb{N}$ be the smallest number such that $\forall \mathcal{F}^n \exists \mathcal{F}^k : \{\mathcal{F}^k\} \cup F \rightsquigarrow \mathcal{F}^n$, then k is the* minimal complete cardinality *for n-party MPC relative to some set of ideal setup functionalities F.*

The original work [FGM+01] defined the minimal complete cardinality relative to $F = \emptyset$ while we consider $F = \{\mathcal{F}_{\mathsf{BC}}^n\}$.

Notation 6 (Lists). *For any list $\gamma := (\gamma_i)_{i \in l}$ of length $l \in \mathbb{N}$ we denote the list at index set $H \subseteq [l]$ by $\gamma_H := (\gamma_i)_{i \in H}$.*

Definition 1 (Threshold secret sharing). *For any $a, b \in \mathbb{N} : a \leq b$ an (a, b)-threshold secret sharing scheme for message space M is defined by a probabilistic algorithm* $\mathrm{Share}_{a,b}$ *and a deterministic algorithm* $\mathrm{Recover}_a$ *with syntax*

- $\mathrm{Share}_{a,b} : M \to (\{0,1\}^\lambda)^b : m \mapsto \mu = (\mu_\kappa)_{\kappa \in [b]}$
- $\mathrm{Recover}_b : (\{0,1\}^\lambda)^b \to M : \mu \mapsto m$

for any $H \in \binom{[b]}{a}$. That is, $\mathrm{Share}_{a,b}$ *takes a message $m \in M$ and outputs b shares such that any a of them reconstruct the message but $a - 1$ shares perfectly hide the secret. Formally, we require correctness: i.e. for all messages $m \in M$ it should hold that*

$$\Pr[\mathrm{Recover}_b(\mu) = m \mid \mu \leftarrow \mathrm{Share}_{a,b}(m)] \in \mathrm{owhl}(\lambda) \ .$$

Additionally, we require privacy: i.e. for all possible shares μ output by $\mathrm{Share}_{a,b}$*, for all messages m and m' of the same bitlength and for all sets of indices $H' \in \binom{[b]}{a-1}$ it should hold that*

$$|\Pr[\mu_{H'} = \mu'_{H'} \mid \mu' \leftarrow \mathrm{Share}_{a,b}(m)] - \Pr[\mu_{H'} = \mu'_{H'} \mid \mu' \leftarrow \mathrm{Share}_{a,b}(m')]| \in \mathrm{negl}(\lambda) \ .$$

For example, we could use Shamir's secret sharing [Sha79] with $b = 4a$. An important feature that we require for our construction is that shares of sharings for message x and y can be homomorphically combined to produce a sharing of the message $x + y$. The purpose of these threshold sharing is to ensure that parties input the same shares across multiple setups.[3] To this end, we use the following lemma which bounds the probability that a sufficiently manipulated sharing stays undetected.

Lemma 1 (Error-detection). *Let $u, w, s \in \mathbb{N}_{\geq 1}$ s.t. $s, w \leq u$ and let $W \in \binom{[u]}{w}$.*

$$\Pr_S\left[W \cap S = \emptyset \;\middle|\; S \in \binom{[u]}{s}\right] = \prod_{i=0}^{s-1} \frac{u - w - i}{u - i} \leq \prod_{i=0}^{s-1} \frac{u - w}{u} \leq (1 - w/u)^s \leq 2^{-sw/u} \tag{3}$$

Notation 7 (Complement graph). *For any undirected irreflexive graph $G = (V, E)$ we use the notation $\overline{G} := (V, \overline{E})$ for the undirected, reflexive* complement *graph with $\overline{E} := \binom{V}{2} \setminus E$.*

Notation 8 (Neighbors). *For an undirected irreflexive graph $G = (V, E)$ we use the following notation for neighbors:*

$$\mathrm{N}_G(v) := \{v' \mid \{v, v'\} \in E\} \ . \tag{4}$$

For an conflict graph $G = (P, E)$ we call $\mathrm{N}_{\overline{G}}(\mathsf{P})$ the *associates* of $\mathsf{P} \in P$.

3.2 Functionalities / Setup

We use the following ideal functionalities. First we define a (one-to-many) bit commitment, again in the spirit of [CLO+02], adapted to the IA setting.

[3] The idea is conceptually similar to *public verifiability* [AO12].

Functionality $\mathcal{F}^n_{\mathsf{COM}}$

$\mathcal{F}^n_{\mathsf{COM}}$ proceeds as follows, running with security parameter λ, parties $P = \{\mathsf{S}, \mathsf{R}_1, \ldots, \mathsf{R}_{n-1}\}$, malicious parties $C \subseteq P$ and adversary $\mathcal{S}$. Messages not covered here are ignored.

- When receiving $(\texttt{commit}, m \in \{0,1\}^\lambda)$ from party S, send $(\texttt{receipt commit})$ to all parties. Ignore further messages $(\texttt{commit}, \cdot)$ from S.
- When receiving $(\texttt{open})$ from party S, send $(\texttt{open}, m)$ to all parties and terminate.
- When receiving $(\texttt{abort}, C')$ from $\mathcal{S}$ with $C' \subseteq C$, then output $(\texttt{abort}, C')$ to all parties and terminate.

In the proof of Theorem 1 we use a one-to-one variant, which is essentially the same, except that only one fixed receiver obtains the opened value.

Functionality $\mathcal{F}^n_{\mathsf{COM},1:1}$

$\mathcal{F}^n_{\mathsf{COM},1:1}$ proceeds as follows, running with security parameter λ, parties $P = \{\mathsf{S}, \mathsf{R}, \ldots\}$, malicious parties $C \subseteq P$ and adversary $\mathcal{S}$. Messages not covered here are ignored.

- When receiving $(\texttt{commit}, m \in \{0,1\}^\lambda)$ from party S, send $(\texttt{receipt commit})$ to all parties. Ignore further messages of the type $(\texttt{commit}, \cdot)$ from S.
- When receiving $(\texttt{open})$ from party S, send $(\texttt{open}, m)$ to R and $(\texttt{open}, \perp)$ to all other parties and $\mathcal{S}$, and terminate.
- When receiving $(\texttt{abort}, C')$ from $\mathcal{S}$ with $C' \subseteq C$, then output $(\texttt{abort}, C')$ to all parties and terminate.

Furthermore, we use a verifiable OT originating from [Cré90; CvT95] formulated as an ideal functionality in [BMM+20]: Fully Committed Oblivious Transfer (FCOT). The FCOT has a specific structure with a dedicated sender and receiver with makes it conceptually easier to construct than to provide general MPC protocols directly. Lemma 14 from [BMM+20] claims that general SFE can be based on FCOT via the IPS-compiler [IPS08]. Note that this functionality assigns dedicated role to the participating parties. Because the functionality are modeled in the UC framework, multiple instances can be arbitrarily composed to allow OTs between any two parties.

Functionality $\mathcal{F}^{n}_{\mathsf{FCOT}}$

$\mathcal{F}^{n}_{\mathsf{FCOT}}$ proceeds as follows, running with security parameter λ, parties $P = \{\mathsf{S}, \mathsf{R}, \mathsf{W}_1, \ldots, \mathsf{W}_{n-2}\}$, malicious parties $C \subseteq P$ and adversary $\mathcal{S}$. Messages not covered here are ignored.

- When receiving $(\texttt{messages}, m^0, m^1 \in \{0,1\}^\lambda)$ from S, store m^0, m^1 and send $(\texttt{receipt messages})$ to all parties and $\mathcal{S}$ if $(\texttt{receipt choice})$ has not been sent. Ignore further messages of the type $(\texttt{messages}, \cdot, \cdot)$ from S.
- When receiving $(\texttt{choice}, c \in \{0,1\})$ from R with $c \in \{0,1\}$, store c and send $(\texttt{receipt choice})$ to all parties and $\mathcal{S}$ if $(\texttt{receipt messages})$ has not been sent. Ignore further messages of the type $(\texttt{choice}, \cdot)$ from R.
- When both m^0, m^1 and c are stored, send $(\texttt{output}, m_c)$ to R, and $(\texttt{receipt transfer})$ to all other parties and $\mathcal{S}$.
- When receiving $(\texttt{open message}, b \in \{0,1\})$ from S and m_0, m_1 are stored, send $(\texttt{open message}, b, m_b)$ and to all parties and $\mathcal{S}$. Ignore further messages $(\texttt{open message}, b)$ from S.
- When receiving $(\texttt{open choice})$ from R and c is stored, send $(\texttt{open choice}, c)$ and to all parties and $\mathcal{S}$. Ignore further messages from R.
- When receiving $(\texttt{abort}, C')$ from $\mathcal{S}$ with $C' \subseteq C$, then output $(\texttt{abort}, C')$ to all parties and terminate.

Next, we define variants of the commitment and OT functionality that operate on sharings of the inputs instead of the inputs themselves. First, we define an Externally Verifiable Commitment (EVCOM) functionality that enables the verification of consistency of inputs across instance classes. The idea is that the sender of the commitment commits to a threshold sharing instead of the message directly. Then shares at some indices (chosen uniformly by honest parties) can be opened to ensure that the same sharing was input into multiple setups. This way a commitment to few parties can be extended to many parties. For notational convenience we define the functionality such that any party can perform an OT with any other party.

Functionality $\mathcal{F}^{\beta}_{\mathsf{EVCOM}}$

$\mathcal{F}^{n}_{\mathsf{EVCOM}}$ proceeds as follows, running with security parameter λ, sharing parameter ℓ, parties $P = \{\mathsf{P}_1, ..., \mathsf{P}_\beta\}$. malicious parties $C \subseteq P$, adversary $\mathcal{S}$. Messages not covered here are ignored.

- When receiving $(\texttt{commit}, \mu_i \in \{0,1\}^{4\ell \times \lambda})$ from P_i, send $(\texttt{receipt commit}, \mathsf{P}_i)$ to each party $\mathsf{P} \in P$ and $\mathcal{S}$. Ignore further messages $(\texttt{input}, \cdot)$ from P_k.
- When receiving $(\texttt{open}, \kappa \in [4\ell])$ from P_i, send $(\texttt{output}, \mathsf{P}_i, \kappa, \mu_{i,\kappa})$ to each party $\mathsf{P} \in P$ and $\mathcal{S}$. Ignore further messages $(\texttt{open}, \kappa)$ from P_i.
- When receiving $(\texttt{abort}, C')$ from $\mathcal{S}$ with $C' \subseteq C$, then output $(\texttt{abort}, C')$ to all parties and terminate.

Next, we define an Externally Verifiable Oblivious Transfer (EVOT) functionality—similar to FCOT—that enables all parties to verify that the inputs encoded in the sharings are consistent with the inputs of some other functionality session, in particular a one-to-many commitment. The notation $\mathsf{P}_i \to \mathsf{P}_j$ signifies an OT with sender P_i and receiver P_j.

Functionality $\mathcal{F}_{\mathsf{EVOT}}^{\beta}$

$\mathcal{F}_{\mathsf{EVOT}}^{n}$ proceeds as follows, running with security parameter λ, sharing parameter ℓ and probes ρ, parties $P = \{\mathsf{P}_1, ..., \mathsf{P}_\beta\}$. malicious parties $C \subseteq P$, adversary $\mathcal{S}$. Messages not covered here are ignored.

- When receiving $(\texttt{messages}, \mathsf{P}_i \to \mathsf{P}_j, \mu_i^0, \mu_i^1 \in \{0,1\}^{4\ell\times\lambda})$ from P_i, store $(\mathsf{P}_i \to \mathsf{P}_j, \mu^0, \mu^1)$ and send $(\texttt{receipt messages}, \mathsf{P}_i \to \mathsf{P}_j)$ to each party $\mathsf{P} \in P$ and $\mathcal{S}$. Ignore further messages of this type from P_i.
- When receiving $(\texttt{choice}, \mathsf{P}_i \to \mathsf{P}_j, \gamma \in \{0,1\}^{4\ell\times\lambda})$ s.t. $\text{Recover}_{4\ell}(\gamma) \in \{0,1\}$ from P_j, store $(\mathsf{P}_i \to \mathsf{P}_j, \gamma)$ and send $(\texttt{receipt choice}, \mathsf{P}_i \to \mathsf{P}_j)$ to each party $\mathsf{P} \in P$ and $\mathcal{S}$. Ignore further messages of this type from P_i.
- When both $(\mathsf{P}_i \to \mathsf{P}_j, \mu^0, \mu^1)$ and $(\mathsf{P}_i \to \mathsf{P}_j, \gamma)$ are stored, send $(\texttt{output}, \mathsf{P}_i \to \mathsf{P}_j, \mu^c)$ to P_j where $c \leftarrow \text{Recover}_{4\ell}(\gamma)$. Draw $H' \leftarrow \binom{[4\ell]}{\rho}$ and send $(\texttt{receipt transfer}, \mathsf{P}_i \to \mathsf{P}_j, H', \mu_{i,H'}^0, \mu_{i,H'}^1, \gamma_{i,H'})$ all parties and $\mathcal{S}$.
- When receiving $(\texttt{abort}, C')$ from $\mathcal{S}$ with $C' \subseteq C$, then output $(\texttt{abort}, C')$ to all parties and terminate.

So, far all functionalities have were defined for a single session only. To reduce notational overhead we omit the *session ids* whenever the session is clear from the context.
Next, we describe the functionalities usually related to the actual model of communication among parties.

Functionality $\mathcal{F}_{\mathsf{BC}}^{n}$

$\mathcal{F}_{\mathsf{BC}}^{n}$ proceeds as follows, running with security parameter λ, parties $P = \{\mathsf{P}_1, ..., \mathsf{P}_n\}$, malicious parties $C \subseteq P$ and adversary $\mathcal{S}$. Messages not covered here are ignored.

- When receiving $(\texttt{input}, m \in \lambda)$ from party P_j, send $(\texttt{output}, \mathsf{P}_j, m)$ to all parties.
- When receiving $(\texttt{abort}, C')$ from $\mathcal{S}$ with $C' \subseteq C$, then output $(\texttt{abort}, C')$ to all parties and terminate.

The broadcast is essentially the one from [CLO+02], though we only let parties broadcast messages to all parties not any subset of parties.
Lastly, we define use the two-party functionality Secure Message Transfer (SMT) which often describes the underlying communication model. However, because we consider larger setups of size β the two party setup is not an additional assumption.

Functionality $\mathcal{F}_{\mathsf{SMT}}^{2}$

$\mathcal{F}_{\mathsf{SMT}}^{n}$ proceeds as follows, running with security parameter λ, parties $P = \{\mathsf{P}_1, \mathsf{P}_2\}$, malicious parties $C \subseteq P$ and adversary $\mathcal{S}$. Messages not covered here are ignored.

- When receiving $(\texttt{input}, m \in \{0,1\}^\lambda)$ from party P_j, store (m) to P_2.
- When receiving $(\texttt{abort}, C')$ from $\mathcal{S}$ with $C' \subseteq C$, then output $(\texttt{abort}, C')$ to all parties and terminate.

4 Technical Overview

As our main technical tool we use the notion of the *Conflict Graphs* (CG) from [BMM+20]. Thus, we give a brief recapitulation of the way in which [BMM+20] uses conflicts to identify cheating parties in general. To this end we recall two graph-theoretical properties that directly correspond to Identifiable Abort. Then, we describe two abort lemmas that state which and how many setups

the adversary can abort in terms of these graph-theoretical properties. For the impossibility we show how the the adversary can abort the setups of any protocol attempting to securely UC-realize a commitment among n-party. For our construction we give two protocols inspired by the abort lemmas that use the fact that some setups must succeed or otherwise identification becomes possible for all parties.

Identification via Conflicts.

Definition 2 (Identifiable Abort from [BMM+20] adapted from [IOZ14]). *Let $\mathcal{F}^n$ be an ideal n-party functionality with parties P and malicious parties $C \subseteq P$. $\mathcal{F}^n$ has **(Multi-)Identifiable Abort**, iff on input $(\texttt{abort}, C')$ from the adversary $\mathcal{F}^n$ sends $(\texttt{abort}, C')$ to all parties. If $C' \not\subseteq C \vee C' = \emptyset$, the message is ignored.*

We make extensive use of the Conflict Graph (CG) formalism of [BMM+20], therefore we repeat the corresponding ideal functionality $\mathcal{F}_{\mathsf{CG}}^n$ and the relevant graph-theoretical properties. The intuitive idea of the CG is to capture which parties *definitively* distrust each other and model these mutual distrusts (conflicts) as as edges in a undirected graph on the set of parties. The advantage of this technique is that it allows to formulate graph-theoretical properties of the CG that translate to necessary resp. sufficient conditions for Identifiable Aborts.

Functionality $\mathcal{F}_{\mathsf{CG}}^n$ adapted from [BMM+20]

$\mathcal{F}_{\mathsf{CG}}^n$ proceeds as follows, running with parties $P = \{\mathsf{P}_1, \ldots, \mathsf{P}_n\}$, malicious parties $C \subseteq P$ and adversary $\mathcal{S}$. Messages not covered here are ignored.

- Upon first activation, initiate the set of conflict edges $E := \emptyset$.
- When receiving a message $(\texttt{conflict}, \mathsf{P}_i \in P)$ from P_j, for $i \neq j$ append the new conflict edge $\{\mathsf{P}_i, \mathsf{P}_j\}$ to the set of conflict edges E and send $(\texttt{conflict}, \mathsf{P}_j, \mathsf{P}_i)$ to the adversary.
- When receiving a message $(\texttt{conflict}, P' \subseteq P)$ from P_j, for each $\mathsf{P}_i \in P'$ s.t. $i \neq j$ append the new conflict edge $\{\mathsf{P}_i, \mathsf{P}_j\}$ to the set of conflict edges E and send $(\texttt{conflict}, \mathsf{P}_j, P')$ to the adversary.
- When receiving a message $(\texttt{query})$ from P_j, output the *Conflict Graph* $G := (P, E)$ to P_j.

When receiving $(\texttt{abort}, C')$ from $\mathcal{S}$ with $C' \subseteq C$, then $\mathcal{F}_{\mathsf{CG}}^n$ outputs $(\texttt{abort}, C')$ to all parties, and then terminates.

The object of the Conflict Graph itself is an undirected irreflexive graph on all parties returned by the $\mathcal{F}_{\mathsf{CG}}^n$ functionality. Because $\mathcal{F}_{\mathsf{CG}}^n$ is a functionality on all n parties, it is well-defined and unique at each given time during a protocol execution. By this definition the CG is only formally defined for protocols in some $(F \cup \{\mathcal{F}_{\mathsf{CG}}^n\})$-hybrid model. However Lemmas 8 and 16 of [BMM+20] show that any secure protocol using broadcast can be augmented to use the $\mathcal{F}_{\mathsf{CG}}^n$ in an abort-respecting way that preserves security and functionality. So, we can make the following remark.

Remark 2. When arguing about the impossibility of *any* secure protocol in some F-hybrid model for a certain ideal functionality it suffices to show that no secure *abort-respecting* protocol exists in the $(F \cup \{\mathcal{F}_{\mathsf{CG}}^n\})$-hybrid model.

Notation 9. *We denote the complement of the Conflict Graph $G = (P, E)$ by $\overline{G} := (P, \overline{E})$ where $\overline{E} := \{\{\mathsf{P}, \mathsf{P}'\} \mid \{\mathsf{P}, \mathsf{P}'\} \notin E\}$.*

Definition 3 (t-settledness from [BMM+20]). *Let n be the number of parties of which at most $0 \leq t < n$ are malicious. Let $G = (P, E)$ be the Conflict Graph of a protocol π. Let $M(G, t)$ be the set of all Minimal Vertex Covers (MVCs) of G with size t or less, and let $X(G, t)$ be the intersection of all of these MVCs, that is, the set of parties which are present in all MVCs of size $\leq t$. We call $X(G, t)$ the* settled set *of G. We call G t-*settled*, iff $X(G, t) \neq \emptyset$.*

The notion of t-settledness describe the situation where—given a CG of a protocol—even an outsider can tell some subset of parties to be malicious. In particular, we use the fact that if any party has strictly more that t conflicts, it must be malicious since no honest-honest conflicts are allowed. Another way to put it is that honest parties always form a clique of size $h = |H|$ in the complement graph $\overline{G}$. Consequently, if two parties are not in such a clique, then it is obvious to all parties that at least one of the two is malicious.

Definition 4 (Biseparation from [BMM+20]). *A Conflict Graph $G = (P, E)$ is called* biseparated*, iff there exists a subset of edges $E' \subseteq E$ that forms a complete bipartite graph (biclique) on P. Formally, $\exists P_1, P_2 \subset P : P_1 \cup P_2 = P \wedge E' \coloneqq \{\{\mathsf{P}, \mathsf{P}'\} \mid \mathsf{P} \in P_1 \wedge \mathsf{P}' \in P_2\} \subseteq E$.*

The notion of biseparation describes the situation where participating parties can tell some subset of parties to be malicious (but not necessarily an outsider).

Remark 3. The complement graph $\overline{G} = \left(P, \overline{E}\right)$ of a CG $G = (P, E)$ is connected, iff G is not biseparated.

Theorem 6 from [BMM+20] formally establishes the link between the biseparation of the CG and the Identifiable Abort of a protocol. On a high level it makes two statements. For a protocol with parties P and honest parties $H \subseteq P$:

1. When the CG of a protocol is biseparated let the partitions $P_1 \cup P_2 = P$ be such that w.l.o.g. $P_1 \supseteq H$ is the largest subset of parties such that all parties in P_1 are in conflict with all parties in P_2. Then all (honest) parties can abort the protocol by outputting $(\texttt{abort}, P_2)$ and terminating.
2. When a protocol aborts with $(\texttt{abort}, C')$, then one can construct a biseparated graph as $G \coloneqq (P, E)$ with $E \coloneqq \{\{\mathsf{P}, \mathsf{P}'\} \mid \mathsf{P} \in C' \wedge \mathsf{P}' \in P \setminus C'\}$.

Additionally, when designing protocols we can require honest parties to declare conflicts with the identified parties (disruptors) upon abort, then the actual CG of the protocol will be biseparated after the abort. This behaviour is called *abort-respecting* in [BMM+20].

As mentioned before, we call an instance class (subset of parties) aborted, iff the Conflict Graph on this subset of parties is biseparated.

Now that we have recalled the relevant graph property related to Identifiable Abort we can go ahead and make statements about the CG on a graph-theoretical level and then translate the results back into the (im-)possibly abort of setups.

Graph-Theoretical Lemmas. Next we present two lemmas in purely graph-theoretical terms which we prove the lemmas in Appendices A and B.

Lemma 2. *Let $n, h \in \mathbb{N}$ with $1 \leq h \leq n$. Let $G = (V, E)$ be an undirected irreflexive graph with $n = |V|$ and let $v, v' \in V$ be some nodes s.t. $|\mathrm{N}_{\overline{G}}(v)|, |\mathrm{N}_{\overline{G}}(v')| \geq h$. Furthermore, let $A_u \coloneqq \{u\} \cup \bigcup_{S \subset \mathrm{N}_{\overline{G}}(u) : |S| = h-1} S$ and let $M \coloneqq \{V' \subseteq V \mid V' \cap A_v \neq \emptyset \wedge V' \cap A_{v'} \neq \emptyset \wedge |V'| < \beta\}$.*

For all $n, h \in \mathbb{N}$ s.t. $h \leq n$ such a graph G and some $v, v' \in V$ exists s.t. for all $V' \in M$ the subgraph $(V', E \cap \binom{V'}{2})$ is biseparated, and G is neither biseparated nor $(n-h)$-settled.

Lemma 3. *Let $n, h \in \mathbb{N}$ with $2 \leq h \leq n$. Let $G = (V, E)$ be an undirected irreflexive graph with $n = |V|$ and let $v, v' \in V$ be some nodes s.t. $|\mathrm{N}_{\overline{G}}(v)|, |\mathrm{N}_{\overline{G}}(v')| \geq h$. Furthermore, let $A_u \coloneqq \{u\} \cup \bigcup_{S \subset \mathrm{N}_{\overline{G}}(u) : |S| = h-1} S$ and let $M \coloneqq \{V' \subseteq V \mid V' \cap A_v \neq \emptyset \wedge V' \cap A_{v'} \neq \emptyset \wedge |V'| = \beta\}$. Additionally, let $E^* \coloneqq E \cup \{\{u, u'\} \notin E \mid |\mathrm{N}_{\overline{G}}(u) \cap \mathrm{N}_{\overline{G}}(u')| < h\}$.*

If the subgraph $(V', E \cap \binom{V'}{2})$ is biseparated for all $V' \in M$, then $G^* = (V, E^*)$ is biseparated. Furthermore, the map $\phi : G \mapsto G^*$ is efficiently computable.

These lemmas translate into the context of protocols with setups in the following way.

Impossibility of commitments. For our impossibility we want to show that no (abort-respecting) protocol with setups of cardinality $\beta - 1$ plus broadcast can securely UC-realize the commitment functionality $\mathcal{F}^n_{\mathsf{COM},1:1}$ which has a dedicated sender S and receiver R. To this end we set $V \coloneqq P$, $v \coloneqq \mathsf{S}$ and $\mathsf{R} \in H$ where H are the honest parties. If a setup with parties P' is aborted in an abort-respecting protocol, then the subgraph of the CG on P' becomes biseparated as described in the previous paragraph. Furthermore, as proven in Theorem 6 of [BMM+20] an abort-respecting protocol can be aborted, iff its CG is biseparated. Now, we can plug in Lemma 2 to see that—even after all setups of cardinality $\beta - 1$ that contain the sender and some honest party—the overall CG remains *not* biseparated. Hence an abort is not possible yet the sender has to commit towards the receiver. In a last step we show that any abort-respecting protocol restricted to the non-aborted setups must be non-hiding or non-binding.

The high-level idea is that, in order to be committed towards the receiver, the sender has to send the message to intermediate parties—even when all parties act honestly in the commitment phase. However, this set of intermediates is small enough that an alternative environment can corrupt it and thus extract the message of an honest sender during the commitment phase.

Formally, our impossibility result states the following.

Theorem 1 (No transmitted commitment). *Let $n \in \mathbb{N}$ and $\beta \coloneqq \min(n, \lfloor n/h \rfloor + \lfloor (n-1)/h \rfloor - 1)$. No $\{\mathcal{F}^2, ..., \mathcal{F}^{\beta-1}, \mathcal{F}_{\mathsf{BC}}^n\}$-protocol π can securely UC-realize $\mathcal{F}_{\mathsf{COM},1:1}^n$ against $t = n - h \geq n/2$ malicious parties:*

$$\left\{\mathcal{F}^2, ..., \mathcal{F}^{\beta-1}, \mathcal{F}_{\mathsf{BC}}^n\right\} \overset{\text{comp}}{\not\rightarrow} \mathcal{F}_{\mathsf{COM},1:1}^n \tag{5}$$

where $\mathcal{F}^2, ..., \mathcal{F}^{\beta-1}$ stand for an arbitrary functionalities of the respective cardinality.

Corollary 1. *In particular, for $h = 2$ we get $\left\{\mathcal{F}^2, ..., \mathcal{F}^{n-3}, \mathcal{F}_{\mathsf{BC}}^n\right\} \overset{\text{comp}}{\not\rightarrow} \mathcal{F}_{\mathsf{COM},1:1}^n$ showing that the result $\left\{\mathcal{F}^{n-1}, \mathcal{F}_{\mathsf{BC}}^n\right\} \rightsquigarrow \mathcal{F}_{\mathsf{SFE}}^n$ from [SSY21] is almost tight.*[4]

Contructions of committed OT. For our construction we want to first construct an abort-respecting protocol that securely UC-realizes a *global commitment* functionality which has a sender who commits to all other parties. In particular, we require all honest parties to locally compute $G^* = \phi(G)$ from Lemma 3 when querying the CG G from $\mathcal{F}_{\mathsf{CG}}^n$. Moreover, we require all parties to abort according to G^*, so—in effect—the CG of our protocol is actually G^*. This modification of the abort-condition is justified because the map ϕ preserve the invariant that no two honest parties are in conflict.

Without going to too much detail, we outline the idea of the protocol. The sender inputs its message—in the form of a threshold sharing—into all setups and gives masks to its associates who, in turn, also commit to them in all setups. The associates also obtain the sharing of the message masked with the masks of the other associates—again they commit to the masked sharing in all setups. For $V \coloneqq P$, $v \coloneqq \mathsf{S}$ and the set of honest parties H Lemma 3 guarantees that at least one setup of cardinality β that contains both some associate of the sender and some associate of the receiver must succeed. Otherwise, if all such setups are aborted, then the CG G^* becomes biseparated by Lemma 3 and the (honest) parties can abort the protocol. Two more arguments are necessary to see that the protocol securely realizes a global commitment.

1. All setups indeed contain sharings of the *same* (possibly masked) message. This is ensured by probing some of the shares input into the setups by the sender.
2. Whenever all setups containing some associate of the sender and some associate of the receiver are aborted, then an honest receiver only has honest associates.

To open the message, all parties open all setups and at least one honest receiver is able to recover the message either directly from the sender's sharing of the message, or from the opened masks and some masked sharing. Those receivers then broadcast the recovered message. Any honest receiver that did not receive any opening—because all its setups have been aborted—then it outputs the majority of its associates broadcasted messages.

We state the following theorem.

Theorem 2 (COM expansion). *Let n be the number of parties of which at most $t = n - h \in [n/2, n-2]$ are malicious s.t. $\binom{n}{\beta} \in \text{poly}(\lambda)$ with $\beta = \lfloor n/h \rfloor + \lfloor (n-1)/h \rfloor - 1$. There is a protocol π_{COM}^n that statistically securely UC-realizes $\mathcal{F}_{\mathsf{COM}}^n$ in the $\{\mathcal{F}_{\mathsf{SMT}}^2, \mathcal{F}_{\mathsf{EVCOM}}^\beta, \mathcal{F}_{\mathsf{BC}}^n\}$-hybrid model:*

$$\left\{\mathcal{F}_{\mathsf{SMT}}^2, \mathcal{F}_{\mathsf{EVCOM}}^\beta, \mathcal{F}_{\mathsf{BC}}^n\right\} \overset{\text{stat}}{\rightsquigarrow} \mathcal{F}_{\mathsf{COM}}^n \tag{6}$$

For the second part of our construction we want to construct an abort-respecting protocol that securely UC-realizes a *committed* or *verifiable* OT[5] functionality which has a dedicated sender S and receiver R. This committed OT variant allows the sender and receiver to opend their inputs to all parties after the actual OT. Again, we require all honest parties P to locally compute G^* and abort according to it.

[4] We note that [SSY21] state their results in the stand-alone model.

[5] The idea originates from [Cré90; CvT95].

The main idea of the protocol is to perform the OT directly via some setup containing the sender and the receiver while both parties also commit to their resp. inputs in the global commitment setup. To this end, both parties create threshold sharings of their inputs and globally commit to their shares indiviually. Each party in the setup then request the opening of some shares of all inputs in both the setup and the global commitments. While these shares do not leak any information about the encoded input it allows with overwhelming probability to detect significant inconsistencies in the setup and the global commitments.

If all direct setups are aborted, then the sender and receiver use their associates respectively to carry out the OT for them. To preserve privacy of the sender's messages and the receiver's choice bit our protocol lets each party use an additive secret sharing of its input. Because at least one associate of the honest sender resp. receiver is honest as well, this perfectly hides the input. While this seems straightforward for an honest sender, for an honest receiver it might not be clear how to share its choice bit. We set up the sender's messages in a clever way to that the receiver can still obtain the chosen message while preserving the privacy of the receiver's choice bit.

To open their input the sender and the receiver simply open their global commitment to the sharings of their input.

Theorem 3 (FCOT expansion). *Let n be the number of parties of which at most $t = n - h \in [n/2, n-2]$ are malicious s.t. $\binom{n}{\beta} \in \text{poly}(\lambda)$ with $\beta = \lfloor n/h \rfloor + \lfloor (n-1)/h \rfloor - 1$. There is a protocol π^n_{FCOT} that statistically securely UC-realizes $\mathcal{F}^n_{\mathsf{FCOT}}$ in the $\{\mathcal{F}^2_{\mathsf{SMT}}, \mathcal{F}^\beta_{\mathsf{EVOT}}, \mathcal{F}^n_{\mathsf{COM}}, \mathcal{F}^n_{\mathsf{BC}}\}$-hybrid model:*

$$\left\{\mathcal{F}^2_{\mathsf{SMT}}, \mathcal{F}^\beta_{\mathsf{EVOT}}, \mathcal{F}^n_{\mathsf{COM}}, \mathcal{F}^n_{\mathsf{BC}}\right\} \overset{\text{stat}}{\rightsquigarrow} \mathcal{F}^n_{\mathsf{FCOT}} \tag{7}$$

Lastly we use the observation in [BMM+20] (Lemma 14) that replacing two-party OT with a verifiable OT for n parties in the IPS-compiler [IPS08] lifts the resulting protocol from security with abort to Identifiable Abort.

Corollary 2. *Combining our Theorems 2 and 3 with Lemmas 14 and 16 of [BMM+20] we get*

$$\left\{\mathcal{F}^2_{\mathsf{SMT}}, \mathcal{F}^\beta_{\mathsf{EVCOM}}, \mathcal{F}^\beta_{\mathsf{EVOT}}, \mathcal{F}^n_{\mathsf{BC}}\right\} \overset{\text{stat}}{\rightsquigarrow} \mathcal{F}^n_{\mathsf{SFE}}\ , \tag{8}$$

i.e., the minimal complete cardinality for n-party SFE with Identifiable Abort relative to $F = \{\mathcal{F}^n_{\mathsf{BC}}\}$ is $\beta = \min(n, \lfloor n/h \rfloor + \lfloor (n-1)/h \rfloor - 1)$.

The above result extends to reactive functionalities because the IPS-compiler also supports constructing protocols for reactive functionalities.

5 Discussion

Limitation of the overall number of parties. While our impossibility holds for arbitrary n, our protocols need $\binom{n}{\beta} \in \text{poly}(\lambda)$ to have polynomial runtime. The reason for this is that there are $\binom{n}{\beta}$ instance classes (subsets of parties) of size β which might all be used in case the adversary aborts all others. Unfortunately, this limitation is inherent to all protocols that don't discriminate between different subsets of parties in any meaningful way. In other words, to overcome this limitation a protocol would have to *never* use some a priori fixed instance classes of size β although they are not aborted. This gives the adversary more leverage to abort the other instance classes in a way that breaks the protocol.

The constructions of [BMM+20; SSY21] only support $n \in \text{poly}(\lambda)$ for constant expansions $n - s \in \mathcal{O}(1)$ (where s is the setup size) because of the exponential composition blowup. We have a different situation; see Table 1 for an overview of the supported number of parties vs. honest parties. The first intuitive thing to note is that the smaller the fraction of honest parties the less overall parties are supported. The case $h = 1$ is trivial. Somewhat counterintuitive is the case for $h = 2$ which supports polynomially many parties, the reason for this is that for $h = 2$ the necessary and sufficient setup encompasses all parties but two, i.e. $s = n - 2$. Here all but one setup functionality contains at least one honest party.
Another case worth mentioning is the two-party case $n = 2$ and $h = 1$ where security with abort is trivially equivalent to IA; interestingly $\beta = 2$ carries over to $n = 2h \geq 4$.

Min. honest parties h	Max. supported parties n	Minimal setup size β
1	$\text{poly}(\lambda)$	n
2	$\text{poly}(\lambda)$	$n-2$
$c \geq 3$	$\Theta(\ln \lambda)$	$\approx 2n/c$
$\Theta(\ln n)$	$\mathcal{O}(\ln \lambda \cdot \ln\ln \lambda / \ln\ln\ln \lambda)$	$\Theta(n/\ln n)$
$\Theta\left(\sqrt{n}\right)$	$\mathcal{O}\left(\ln^2 \lambda / \ln^2 \ln \lambda\right)$	$\Theta\left(\sqrt{n}\right)$
$\Theta(n/\ln n)$	$\mathcal{O}\left(\exp \sqrt{\ln \lambda}\right)$	$\Theta(\ln n)$
$\Theta(n)$	$\text{poly}(\lambda)$	$\Theta(1)$
$(n-1)/2$	$\text{poly}(\lambda)$	3
$\geq n/2^{(*)}$	$\text{poly}(\lambda)$	2

Table 1: Overview of the minimal setup size and respective supported number of parties n vs. honest parties h for our construction with broadcast. The setup size is minimal and complete. The limitation of the overall number of parties is only to achieve polynomial-time protocols, for more parties the protocols remain correct and secure but require the parties to have superpolynomial runtime. The case $(*)$ also covers an honest majority of parties treated in early works [Bea90; RB89].

When an arbitrarily small but constant fraction of parties is honest our protocols are efficient for any $n \in \text{poly}(\lambda)$. However when less than a constant fraction of parties are honest, e.g. $\Theta(n/\ln n)$, then the overall number parties drops drastically below $n \in \mathcal{O}(\exp \sqrt{\ln \lambda}) \subset \lambda^{o(1)}$. To relativize, these bounds only apply when trying to design protocols with IA from *minimal* setups. It could be the case that slightly larger setups yield protocols that support many more overall parties. As such, the path to designing secure protocols that support many parties is to find a way not to use all available setups, or resort to larger setups.

Outlook. While we give a complete characterization of the minimal sufficient setups for MPC with Identifiable Abort some questions remain open. First, do our results carry over to the setting of adaptive corruptions? We conjecture this to be true but leave the question open for future work.
In light of the rather unfavorable scaling behaviour of the max. supported parties vs. setup cardinality for $h \in o(n)$, it may be interesting to investigate the utility of slightly larger setup to potentially support many more parties overall.
An obvious and natural question is to ask: what is the minimal complete cardinality without broadcast, i.e. $F = \emptyset$? In particular, what is the minimal complete cardinality for broadcast itself? Based on our Corollary 2 constructing broadcast with Identifiable Abort from setups of cardinality β' implies an upper bound of $\max(\beta, \beta')$ for general MPC with IA.
Another major research direction is the round-complexity of MPC protocols which is not considered in this work. We leave it to future work to strive towards optimally efficient protocols from β-party setups.

References

[ACC+20] B. Alon, H. Chung, K.-M. Chung, M.-Y. Huang, Y. Lee, and Y.-C. Shen. Round efficient secure multiparty quantum computation with identifiable abort. Cryptology ePrint Archive, Report 2020/1464, 2020. https://eprint.iacr.org/2020/1464.

[AL07] Y. Aumann and Y. Lindell. Security against covert adversaries: efficient protocols for realistic adversaries. In pages 137–156, 2007.

[AO12] G. Asharov and C. Orlandi. Calling out cheaters: covert security with public verifiability. In pages 681–698, 2012.

[BCG93] M. Ben-Or, R. Canetti, and O. Goldreich. Asynchronous secure computation. In pages 52–61, 1993.

[Bea90] D. Beaver. Multiparty protocols tolerating half faulty processors. In pages 560–572, 1990.

[BGW88] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In pages 1–10, 1988.

[BMM+20] N.-P. Brandt, S. Maier, T. Müller, and J. Müller-Quade. Constructing secure multi-party computation with identifiable abort. Cryptology ePrint Archive, Report 2020/153, 2020. https://eprint.iacr.org/2020/153.

[BOS+20] C. Baum, E. Orsini, P. Scholl, and E. Soria-Vazquez. Efficient constant-round MPC with identifiable abort and public verifiability. In pages 562–592, 2020.

[BOS16] C. Baum, E. Orsini, and P. Scholl. Efficient secure multiparty computation with identifiable abort. In pages 461–490, 2016.

[Can00] R. Canetti. Universally composable security: a new paradigm for cryptographic protocols. Cryptology ePrint Archive, Report 2000/067, 2000. http://eprint.iacr.org/2000/067.

[Can01] R. Canetti. Universally composable security: a new paradigm for cryptographic protocols. In pages 136–145, 2001.

[CF01] R. Canetti and M. Fischlin. Universally composable commitments. In pages 19–40, 2001.

[CK88] C. Crépeau and J. Kilian. Achieving oblivious transfer using weakened security assumptions (extended abstract). In pages 42–52, 1988.

[Cle86] R. Cleve. Limits on the security of coin flips when half the processors are faulty (extended abstract). In pages 364–369, 1986.

[CLO+02] R. Canetti, Y. Lindell, R. Ostrovsky, and A. Sahai. Universally composable two-party and multi-party secure computation. In pages 494–503, 2002.

[CM89] B. Chor and L. Moscovici. Solvability in asynchronous environments (extended abstract). In pages 422–427, 1989.

[Cré90] C. Crépeau. Verifiable disclosure of secrets and applications (abstract). In pages 150–154, 1990.

[Cré97] C. Crépeau. Efficient cryptographic protocols based on noisy channels. In pages 306–317, 1997.

[CvT95] C. Crépeau, J. van de Graaf, and A. Tapp. Committed oblivious transfer and private multi-party computation. In pages 110–123, 1995.

[DPS+12] I. Damgård, V. Pastro, N. P. Smart, and S. Zakarias. Multiparty computation from somewhat homomorphic encryption. In pages 643–662, 2012.

[FGM+01] M. Fitzi, J. A. Garay, U. M. Maurer, and R. Ostrovsky. Minimal complete primitives for secure multi-party computation. In pages 80–100, 2001.

[GIS+10] V. Goyal, Y. Ishai, A. Sahai, R. Venkatesan, and A. Wadia. Founding cryptography on tamper-proof hardware tokens. In pages 308–326, 2010.

[HM04] D. Hofheinz and J. Müller-Quade. A synchronous model for multi-party computation and the incompleteness of oblivious transfer. In *Foundations of Computer Security, Proceedings of FCS 2004*, volume 31 of *TUCS General Publications*, pages 117–130. Turku Center of Computer Science, 2004.

[IOS12] Y. Ishai, R. Ostrovsky, and H. Seyalioglu. Identifying cheaters without an honest majority. In pages 21–38, 2012.

[IOZ14] Y. Ishai, R. Ostrovsky, and V. Zikas. Secure multi-party computation with identifiable abort. In pages 369–386, 2014.

[IPS08] Y. Ishai, M. Prabhakaran, and A. Sahai. Founding cryptography on oblivious transfer - efficiently. In pages 572–591, 2008.

[RB89] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In pages 73–85, 1989.

[SF16] G. Spini and S. Fehr. Cheater detection in SPDZ multiparty computation. In pages 151–176, 2016.

[Sha79] A. Shamir. How to share a secret. 22(11):612–613, November 1979.

[SSY21] M. Simkin, L. Siniscalchi, and S. Yakoubov. On sufficient oracles for secure computation with identifiable abort. Cryptology ePrint Archive, Report 2021/151, 2021. https://eprint.iacr.org/2021/151.

Supplementary Material

A Impossibility / Lower Bound

First we show that all instance classes of a certain type can be aborted by the adversary without causing biseparation of the Conflict Graph. Then we use this abort strategy to force any protocol into a specific form and show that protocols with such a structure cannot securely realize a one-to-one commitment. First we prove a graph-theoretical lemma.

Lemma 2. *Let $n, h \in \mathbb{N}$ with $1 \leq h \leq n$. Let $G = (V, E)$ be an undirected irreflexive graph with $n = |V|$ and let $v, v' \in V$ be some nodes s.t. $|\mathrm{N}_{\overline{G}}(v)|, |\mathrm{N}_{\overline{G}}(v')| \geq h$. Furthermore, let $A_u \coloneqq \{u\} \cup \bigcup_{S \subset \mathrm{N}_{\overline{G}}(u) : |S| = h-1} S$ and let $M \coloneqq \{V' \subseteq V \mid V' \cap A_v \neq \emptyset \wedge V' \cap A_{v'} \neq \emptyset \wedge |V'| < \beta\}$.*

For all $n, h \in \mathbb{N}$ s.t. $h \leq n$ such a graph G and some $v, v' \in V$ exists s.t. for all $V' \in M$ the subgraph $(V', E \cap \binom{V'}{2})$ is biseparated, and G is neither biseparated nor $(n-h)$-settled.

Proof. As discussed in Section 4 ruling out all abort-respecting protocols in the $F \cup \{\mathcal{F}_{\mathsf{CG}}^n\}$-model also rules out all protocols in the F-model where F is a set ideal functionalities. Henceforth we only consider abort-respecting protocols that use the $\mathcal{F}_{\mathsf{CG}}^n$ setup where the Conflict Graph is well defined.

Let $V = \{v_1, ..., v_n\}$ and w.l.o.g. $v = v_1$ and $v' = v_n$. We define sections as follows $S_j \coloneqq \{v_{jh+2}, ..., v_{(j+1)h}\}$ and $T_j \coloneqq \{v_{jh+1}\}$. In case $h = 1$ note $S_j = \emptyset$. Based on these sections we define the graph $G \coloneqq (V, E)$ by

$$E \coloneqq \left\{ \{u, u'\} \;\middle|\; \begin{array}{ll} & \exists j \in [\lfloor (n-1)/h \rfloor - 1] : u \in T_j \wedge u' \in T_{j+1+\chi} \\ \vee & \exists j \in [\lfloor n/h \rfloor - 2] \quad\;\; : u \in S_j \wedge u' \in S_{j+1} \\ \vee & \exists j \in [\lfloor (n-1)/h \rfloor - 1] : u \in T_j \wedge u' \in S_{j+1} \end{array} \right\}. \tag{9}$$

where $\chi = 1 \iff h = 1$ and $\chi = 0 \iff h \geq 2$. To make the proof more intuitive compare the visual representation Fig. 1. We show three properties of this graph G.

1. G is not biseparated,
2. G is not $(n-h)$-settled,
3. the subgraphs on all $V' \in M$ are biseparated.

To show Property 1 we notice that nodes within a section are not connected, i.e. $\binom{S_j}{2} \cap E = \emptyset$. Furthermore, consecutive sections are not connected, i.e. $h \geq 2 \implies \forall u \in T_j \cup T_{j+1} \forall u' \in S_j : \{u, u'\} \notin E$ and $h = 1 \implies \forall u \in T_j \forall u' \in T_{j+1} : \{u, u'\} \notin E$. So, $\overline{G}$ is connected and G is biseparated.

Next we show Property 2. Define $D_j \coloneqq V \setminus (T_j \cup S_j)$ and $D'_j \coloneqq V \setminus (S_j \cup T_{j+1})$. Note that $\forall j : \binom{T_j \cup S_j}{2} \cap E = \emptyset \wedge \binom{S_j \cup T_{j+1}}{2} \cap E = \emptyset$, as such D_j and D'_j are Vertex Covers (VCs) of G of cardinality $n-h$. So, $\forall v \in V \; \exists \mathrm{VC}\; D : v \notin D$ and thus $\bigcap_{\mathrm{VC}\; D : |D| = n-h} D = \emptyset \iff G$ is $(n-h)$-settled.

Finally, we show Property 3. We show that for each $V' \in M$ the subgraph $(V', E \cap \binom{V'}{2})$ is biseparated. To this end we show that there exists a *splitting* index $j_{V'}$ which defines two partitions $X \coloneqq V' \cap \{v_1, ..., v_{j_{\min, V'} h+1}\}$ and $Y \coloneqq V' \cap \{v_{j_{\min, V'} h+2}, ...v_n\}$ s.t. $X \cup Y = V'$ and $\forall u \in X \, \forall u' \in Y : \{u, u'\} \in E$.
Define for each $V' \in M$

$$J_{V'} \coloneqq \{j \in [\lfloor (n-1)/h \rfloor - 1] \mid v_{jh+1} \notin V'\} \tag{10}$$

and

$$\widetilde{J}_{V'} \coloneqq \left\{ \widetilde{j} \in [\lfloor n/h \rfloor - 2] \;\middle|\; S_{\widetilde{j}} \cap V' = \emptyset \right\}. \tag{11}$$

Note that

$$\left.\begin{array}{l} h = 1 \implies A_v = \{v\} \\ h \geq 2 \implies A_v = \mathrm{N}_{\overline{G}}(v) \end{array}\right\} \implies A_{v_1} = \{v_1, ..., v_h\} \wedge A_{v_n} \supseteq \{v_{n-h}, ..., v_n\} \tag{12}$$

which in turn implies

$$\forall j \in [\lfloor (n-1)/h \rfloor - 1] : S_j \cap (A_{v_1} \cup A_{v_n}) = \emptyset \tag{13}$$

and

$$\forall \widetilde{j} \in [\lfloor n/h \rfloor - 2] : S_{\widetilde{j}} \cap (A_{v_1} \cup A_{v_n}) = \emptyset \ . \tag{14}$$

Next, we show for each $V' \in M$ that $|J_{V'} \cup \widetilde{J}_{V'}| > 0$. Consider the case $h = 1 \iff \beta = n$, then $V' = V \setminus \{v_j\} \implies J_{V'} = \{j-1\}$ and hence $j_{V'} = j - 1$.
Now consider the case $h \geq 2 \iff \beta = \lfloor (n-1)/h \rfloor + \lfloor n/h \rfloor - 1$ and suppose for contraction that

$$(\forall j \in [\lfloor (n-1)/h \rfloor - 1] : v_{jh+1} \in V') \wedge \left(\forall \widetilde{j} \in [\lfloor n/h \rfloor - 2] : S_{\widetilde{j}} \cap V' \neq \emptyset\right) \tag{15}$$

Then

$$V' \supseteq (A_{v_1} \cup A_{v_n}) \cap V' \uplus \bigcup_{j=1}^{\lfloor (n-1)/h \rfloor - 1} \{v_{jh+1}\} \cup \bigcup_{\widetilde{j}=1}^{\lfloor n/h \rfloor - 2} \left(S_{\widetilde{j}} \cap V'\right) \tag{16}$$

implies

$$|V'| \geq 2 + \lfloor (n-1)/h \rfloor - 1 + \lfloor n/h \rfloor - 2 = \beta \tag{17}$$

which contradicts the assumption $|V'| < \beta$. Now, that we have shown that $j_{V'}$ exists, we go on to show that each node in X is connected to each node in Y. Suppose for contradiction that $\{v_a, v_b\} \notin E$ for some $v_a \in X$ and $v_b \in Y$, i.e. $a \leq j_{V'}h + 1 \wedge b > j_{V'}h + 1$. Because v_a and v_b are not connected, they must be in consecutive sections, i.e. $v_a = T_{j_{V'}} \wedge v_b \in S_{j_{V'}} \cup T_{j_{V'}+1}$. However, this contradicts $\{v_a, v_b\} \notin E$ by the definition of E. If $v_a \in S_{j_{V'}-c}$ for some $c \leq 1$, then $b \in T_{j_{V'}-c+1} \implies b = (j_{V'} - c + 1)h + 1 \leq j_{V'}h + 1$ with contradicts $b > j_{V'}h + 1 \iff v_b \in Y$. This concludes the proof.

□

Intuitively, this lemma states that an adversary cannot abort setups in such a way as to separate any party P_1 from the honest ones. In other words, the adversary cannot excuse the failure of a malicious party P_1 to send information by aborting setups.

Theorem 1 (No transmitted commitment). *Let $n \in \mathbb{N}$ and $\beta \coloneqq \min(n, \lfloor n/h \rfloor + \lfloor (n-1)/h \rfloor - 1)$. No $\{\mathcal{F}^2, ..., \mathcal{F}^{\beta-1}, \mathcal{F}_{\mathsf{BC}}^n\}$-protocol π can securely UC-realize $\mathcal{F}_{\mathsf{COM},1:1}^n$ against $t = n - h \geq n/2$ malicious parties:*

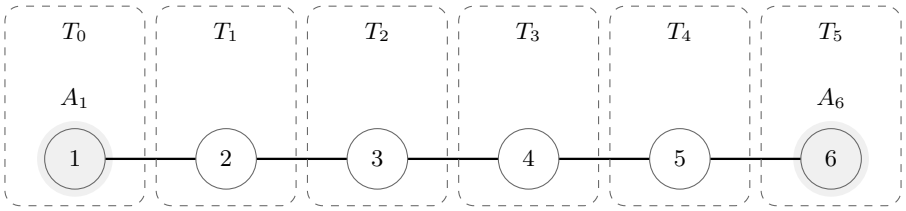
$$\left\{\mathcal{F}^2, ..., \mathcal{F}^{\beta-1}, \mathcal{F}_{\mathsf{BC}}^n\right\} \overset{\text{comp}}{\not\to} \mathcal{F}_{\mathsf{COM},1:1}^n \tag{5}$$

where $\mathcal{F}^2, ..., \mathcal{F}^{\beta-1}$ stand for an arbitrary functionalities of the respective cardinality.

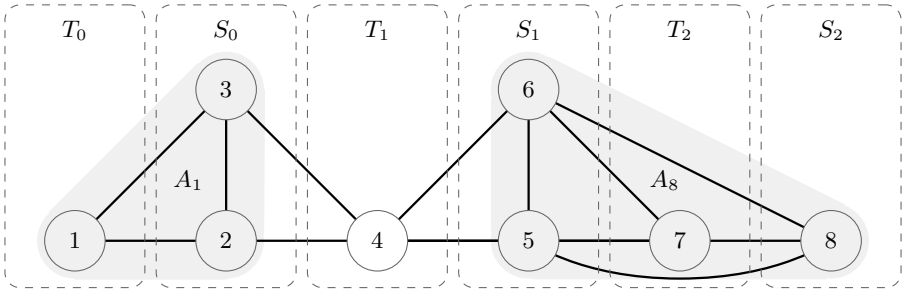
Proof. Overall the proof proceeds similarly to the impossibility of UC-commitments without setup [CF01]. As such this proof can be seen as extending the class of setups that are insufficient for UC-commitments.

Denote the set of parties with $P = \{\mathsf{P}_1, \mathsf{P}_2, ..., \mathsf{P}_n\}$. Suppose for contradiction that there exists an abort-respecting protocol π that securely UC-realizes $\mathcal{F}_{\mathsf{COM},1:1}^n$. W.l.o.g. let $\mathsf{S} = \mathsf{P}_1$ and $\mathsf{R} = \mathsf{P}_n$. Denote the sender's message by m. Furthermore, denote by $\text{ext}_{\mathsf{P}}^{\texttt{commit}} = (\eta, \text{id}, i, s_{\eta,\text{id},i})_{\eta,\text{id},i}$ the list of all messages that P sends to any setup during the commit phase, i.e. before the sender gets input (`open`). Here $s_{\eta,\text{id},i}$ denotes the i-th message that P sends to the setup with session id id in round η. For any set of parties $P' \subseteq P$ let $\text{ext}_{P'}^{\texttt{commit}} \coloneqq (\text{ext}_{\mathsf{P}}^{\texttt{commit}})_{\mathsf{P} \in P'}$. Lastly, denote by $\mathsf{view}_{\mathsf{P}}^{\texttt{commit}} = ((\eta, \text{id}, i, m_{\eta,\text{id},i}, s_{\eta,\text{id},i})_{\eta,\mathsf{P},i}; r_{\mathsf{P}})$ the view of party P, i.e. all messages that party P sends and receives during the commit phase plus its randomness. Here $s_{\eta,\text{id},i}$ denotes the i-th message that P sends to the setup with session id id and $m_{\eta,\text{id},i}$ denotes the i-th message that P receives from the setup with session id id in round η.

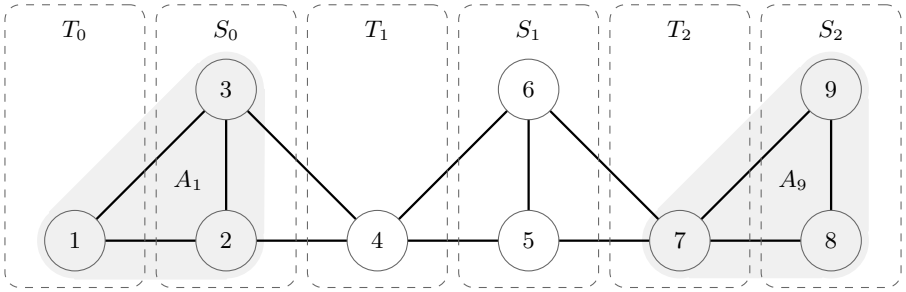
Consider an enviroment $\mathcal{Z}$ that corrupts $C \coloneqq \{\mathsf{P}_1 = \mathsf{S}, ..., \mathsf{P}_{n-h}\}$, i.e. $H \coloneqq P \setminus C = \{\mathsf{P}_{n-h+1}, ..., \mathsf{P}_n\}$ is honest. The environment $\mathcal{Z}$ lets all parties act honestly in the commitment phase, i.e. on input $(\texttt{commit}, m)$ to S. By the assumed secure protocol π there exists a simulator $\mathcal{S}$ that extracts some message m from $\mathsf{view}_C^{\texttt{commit}}$ which the honest receiver will defintively output in the opening phase—if the protocol is not aborted.



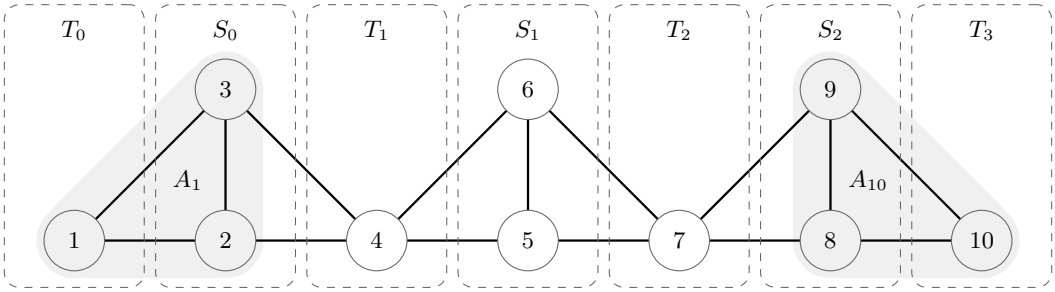
(a) Example of the complement graph $\overline{G}$ from Lemma 2 with $h = 1$, $n = 6$ and $\beta = 6$.



(b) Example of the complement graph $\overline{G}$ from Lemma 2 with $h = 3$, $n = 8$ and $\beta = 3$.



(c) Example of the complement graph $\overline{G}$ from Lemma 2 with $h = 3$, $n = 9$ and $\beta = 4$.



(d) Example of the complement graph $\overline{G}$ from Lemma 2 with $h = 3$, $n = 10$ and $\beta = 5$.

Fig. 1: Examples of the complement graphs from Lemma 2. Sections are represented as dashed boxes, and "associates" of 1 and n in grey. Note that the shortest path from any A_1 to any node in A_n always has at least β nodes.

At the start of the opening phase, i.e. on input (open) to S, the environment $\mathcal{Z}$ immediately aborts setups in the following way according to Lemma 2. For each setup on parties P' s.t. $P' \cap \{\mathsf{P}_1, ..., \mathsf{P}_h\} \wedge P' \cap H \wedge |P'| < \beta$ the environment finds the splitting index $j_{P'}$ as described in Lemma 2 for $V = P$, $v = \mathsf{P}_1$, $v' = \mathsf{P}_n$ and $V' = P'$. Then it aborts the setup with $(\mathtt{abort}, D')$ where $D' \coloneqq P' \cap \{\mathsf{P}_1, ..., \mathsf{P}_{J_{P'}}\}$. Note that

$$j_{P'}h + 1 \leq (\lfloor (n-1)/h \rfloor - 1)h + 1 \leq n - h \implies \{\mathsf{P}_1, ..., \mathsf{P}_{J_{P'}}\} \subseteq C \implies D' \subseteq C \tag{18}$$

therefore the parties with whom $\mathcal{Z}$ aborts are indeed malicious. Because π is abort-respecting the subgraph $(P', E \cap \binom{P'}{2})$ becomes biseparated between D' and $P' \setminus D'$. However, Lemma 2 guarantees that the overall CG G remains neither biseparated nor t-settled, thus Theorem 6 from [BMM+20] guarantees that the protocol cannot abort. Consequently, the honest receiver R must output the message m that the simulator extracted in the commit phase.

Note that—in the opening phase after the aborts—no honest party is in any setup (except broadcast) with any party in $\{\mathsf{P}_1, ..., \mathsf{P}_h\}$. This means that the receiver's output $(\mathtt{output}, m)$ in the opening phase must depend only on $\mathsf{ext}_{P^*}^{\mathtt{commit}}$ and $\mathsf{view}_H^{\mathtt{commit}}$ where $P^* \coloneqq (C \setminus \{\mathsf{P}_1, ..., \mathsf{P}_h\}) \cup H = \{\mathsf{P}_{h+1}, ..., \mathsf{P}_n\}$. As such, the receiver acts as an extractor of m from $(\mathsf{ext}_{P^*}^{\mathtt{commit}}, \mathsf{view}_H^{\mathtt{commit}})$. In turn, because

- $\mathsf{ext}_{P^*}^{\mathtt{commit}}$ can be easily extracted from $\mathsf{view}_{P^*}^{\mathtt{commit}}$ by omitting the received messages and the randomness of all parties,
- the protocol code of R is efficient computable,
- $H \subseteq P^*$,

there exists an efficient extractor $\mathcal{E}$, i.e. for some $\mu \in \mathrm{negl}(\lambda)$

$$\Pr\left[\mathcal{E}\left(\mathsf{view}_{P^*}^{\mathtt{commit}}\right) = m \;\middle|\; \text{input } (\mathtt{commit}\ m) \text{ for honest } \mathsf{S}\right] \geq 1 - \mu\ . \tag{19}$$

Now, consider an alternative environment $\mathcal{Z}'$ that corrupts $C' \coloneqq P^*$, i.e. $H' \coloneqq P \setminus C'$ is honest. Again, $\mathcal{Z}'$ lets all parties act honestly during the commit phase and gives input $(\mathtt{commit}, m)$ to the honest sender S. The new environment $\mathcal{Z}'$ then uses the extractor $\mathcal{E}$ to extract m from $\mathsf{view}_{C'}^{\mathtt{commit}}$. This is possible because $P^* = C'$ and all messages to corrupted parties are reported to $\mathcal{Z}'$ the dummy adversary.

To conclude the proof note that no simulator $\mathcal{S}'$ can extract m from the ideal commitment functionality during the commit phase if the sender S is honest, i.e. $\Pr\left[\mathsf{REAL} \leftarrow \mathcal{Z}' \;\middle|\; \mathsf{IDEAL}\right] \leq 1/2 + \mu'$ with $\mu' \in \mathrm{negl}(\lambda)$. As such $\mathcal{Z}'$ decides REAL iff $\mathcal{E}(\mathsf{view}_{C'}^{\mathtt{commit}}) = m$ and IDEAL otherwise. Then $\mathcal{Z}'$ can decide the real and the ideal execution with advantage

$$\Pr\left[\mathsf{REAL} \leftarrow \mathcal{Z}' \;\middle|\; \mathsf{REAL}\right] - \Pr\left[\mathsf{REAL} \leftarrow \mathcal{Z}' \;\middle|\; \mathsf{IDEAL}\right] \geq 1/2 - \mu - \mu' \notin \mathrm{negl}(\lambda)\ . \tag{20}$$

□

B Construction / Upper Bound

In this section we prove Lemma 3 and present the two construction in detail. Obviously, for $h = 1$ we get $\beta = n$ which—in light of the universal Correlated Randomness setup [IOZ14]—renders any other construction redundant, therefore we consider $h \geq 2 \implies \beta = \lfloor n/h \rfloor + \lfloor (n-1)/h \rfloor - 1$ henceforth.

Lemma 3. *Let $n, h \in \mathbb{N}$ with $2 \leq h \leq n$. Let $G = (V, E)$ be an undirected irreflexive graph with $n = |V|$ and let $v, v' \in V$ be some nodes s.t. $|\mathrm{N}_{\overline{G}}(v)|, |\mathrm{N}_{\overline{G}}(v')| \geq h$. Furthermore, let $A_u \coloneqq \{u\} \cup \bigcup_{S \subset \mathrm{N}_{\overline{G}}(u) : |S| = h-1} S$ and let $M \coloneqq \{V' \subseteq V \mid V' \cap A_v \neq \emptyset \wedge V' \cap A_{v'} \neq \emptyset \wedge |V'| = \beta\}$. Additionally, let $E^* \coloneqq E \cup \{\{u, u'\} \notin E \mid |\mathrm{N}_{\overline{G}}(u) \cap \mathrm{N}_{\overline{G}}(u')| < h\}$.*

If the subgraph $(V', E \cap \binom{V'}{2})$ is biseparated for all $V' \in M$, then $G^* = (V, E^*)$ is biseparated. Furthermore, the map $\phi : G \mapsto G^*$ is efficiently computable.

Proof (of Lemma 3). First we note that the map ϕ is computable in time $\mathcal{O}(n^3)$ by computing $\left|\mathrm{N}_{\overline{G}}(u) \cap \mathrm{N}_{\overline{G}}(u')\right|$ for each pair $u, u' \in V$ and adding the appropriate edges $\{u, u'\}$ to E.

For contradiction suppose that

1. $(V', E \cap \binom{V'}{2})$ is biseparated for all $V' \in M$,
2. G^* is not biseparated, i.e. $\overline{G}^*$ is connected, and
3. $\forall u, u' \in V : |\mathrm{N}_{\overline{G}^*}(u) \cap \mathrm{N}_{\overline{G}^*}(u')| \geq h$.

Because of Assumption 2 there must be a path in $\overline{G}^*$ from any node $u \in A_v$ to any node $u' \in A_{v'}$. Consider any shortest path $p \coloneqq \{p_1 = u, ..., p_{\beta'} = u'\}$ from u to u' with length β'. Note that $p \cap A_v = \{u\} \wedge p \cap A_{v'} = \{u'\}$ and

$$\forall i \in [1, \beta' - 3] : A_{p_i} \cap A_{p_{i+3}} = \emptyset \tag{21}$$

because otherwise a shorter path from some $z \in (A_v \cap p) \setminus \{u\}$ to some $z' \in (A_{v'} \cap p) \setminus \{u'\}$ existed. Any such path p must have length $\beta' > \beta$. Otherwise, because of Assumption 2, there existed some $V' \supseteq p$ s.t. $V' \in M$ yet V' is connected in $\overline{G}^*$, i.e. not biseparated in G^*, which contradicts Assumption 1.
We define common neighbors $J_i \coloneqq (A_{p_i} \cap A_{p_{i+1}}) \setminus p$ and see that

$$J_i \cap J_{i+2} = \left(A_{p_i} \cap A_{p_{i+1}} \cap A_{p_{i+2}} \cap A_{p_{i+3}}\right) \setminus p \subseteq \left(A_{p_i} \cap A_{p_{i+3}}\right) \setminus p = \emptyset \ . \tag{22}$$

Furthermore we see

$$A_{p_i} \cap A_{p_{i+1}} = \underbrace{\left(A_{p_i} \cap A_{p_{i+1}} \setminus p\right)}_{J_i} \cup \underbrace{\left(A_{p_i} \cap A_{p_{i+1}} \cap p\right)}_{\{p_i, p_{i+1}\}} \tag{23}$$

by Eq. (21). From Assumption 3 and $h \geq 2 \implies \forall u \in V : A_u = \mathrm{N}_{\overline{G}}(u)$ it follows that

$$\forall u, u' \in V : \left|\mathrm{N}_{\overline{G}}(u) \cap \mathrm{N}_{\overline{G}}(u')\right| \geq h \implies |J_i| = \underbrace{\left|A_{p_i} \cap A_{p_{i+1}}\right|}_{\geq h} - \underbrace{|\{p_i, p_{i+1}\}|}_{=2} \geq h - 2 \ . \tag{24}$$

Now, we count the number of overall nodes, we find

$$V \supseteq p \cup (A_v \setminus p) \cup (A_{v'} \setminus p) \cup \bigcup_{i=1}^{\lfloor \beta'/2 - 1 \rfloor} J_{2i} \ . \tag{25}$$

Note that $\forall i \in [2, \beta' - 2] : A_{p_i} \cap (A_v \cup A_{v'}) = \emptyset$ and hence $\forall i \in [1, \lfloor \beta'/2 - 1 \rfloor] : J_{2i} \cap (A_v \cup A_{v'}) = \emptyset$ because otherwise $p' \coloneqq \{p_1, ..., p_i, v''\}$ with $v'' \in A_{p_i} \cap H$ would be a path in $\overline{G}^*$ of length $i+1 < \beta'$ and thus contradict the minimal length of the path p. By $\forall x \in \mathbb{N} : x = \lceil x/2 \rceil + \lfloor x/2 \rfloor$ it follows that

$$\begin{aligned} n = |V| &\geq \underbrace{|p|}_{\beta'} + \underbrace{|A_v \setminus p|}_{h-1} + \underbrace{|A_{v'} \setminus p|}_{h-1} + \sum_{i=1}^{\lfloor \beta'/2 - 1 \rfloor} \underbrace{|J_{2i+1}|}_{\geq h-2} \\ &\geq \beta' + 2(h-1) + \lfloor \beta'/2 - 1 \rfloor (h-2) \\ &= \beta' + 2 + \lfloor \beta'/2 + 1 \rfloor (h-2) \\ &\geq \beta + 3 + \lfloor (\beta+3)/2 \rfloor (h-2) \ . \end{aligned} \tag{26}$$

For $n/h \in \mathbb{N} \iff \beta = 2n/h - 2$ we get

$$n \geq 2n/h + 1 + (n/h)(h-2) = n + 1 \ . \tag{27}$$

For $n/h \notin \mathbb{N} \iff \beta = 2\lfloor n/h \rfloor - 1$ we get

$$n \geq 2\lfloor n/h \rfloor + 2 + (\lfloor n/h \rfloor + 1)(h-2) = \lfloor n/h \rfloor h + h = \lceil n/h \rceil h > n \ . \tag{28}$$

Both cases come to a contradiction which concludes the proof. □

The above lemma can be understood visually when looking at Fig. 1. There is always a connected subgraph of $\overline{G}$ of size β which connected an associate (in grey) of the first node and one associate of the last node.

Next, we give a construction of a global (one-to-many) commitment functionality from setup commitments of size β and a global broadcast. The basic idea is for the sender to simply input its

message into all setups, i.e. each subset of parties that contain the sender, and later open all of them. If the sender is honest, all messages will be consistent and all honest parties will output the correct message. A problem arises when a) the sender is malicious and inputs different messages into different setups, or b) many setups are aborted s.t. some (honest) receivers don't obtain any opening information. For a) the remedy is to let the sender commit to a threshold sharing of its message such that the receivers can request the opening of some shares—which the sender then also has to broadcast—because the threshold sharing is robust against a half of manipulated shares any inconsistency between the shares obtained from the setup commitment and the ones broadcasted will be detected with overwhelming probability. For b) we use the structure of the CG to recognize that whenever a receiver was completely cut off, its associates must be honest, thus such receivers simply follow the behaviour of their associates.

Theorem 2 (COM expansion). *Let n be the number of parties of which at most $t = n - h \in [n/2, n-2]$ are malicious s.t. $\binom{n}{\beta} \in \mathrm{poly}(\lambda)$ with $\beta = \lfloor n/h \rfloor + \lfloor (n-1)/h \rfloor - 1$. There is a protocol π^n_{COM} that statistically securely UC-realizes $\mathcal{F}^n_{\mathsf{COM}}$ in the $\{\mathcal{F}^2_{\mathsf{SMT}}, \mathcal{F}^\beta_{\mathsf{EVCOM}}, \mathcal{F}^n_{\mathsf{BC}}\}$-hybrid model:*

$$\left\{\mathcal{F}^2_{\mathsf{SMT}}, \mathcal{F}^\beta_{\mathsf{EVCOM}}, \mathcal{F}^n_{\mathsf{BC}}\right\} \overset{\mathrm{stat}}{\rightsquigarrow} \mathcal{F}^n_{\mathsf{COM}} \tag{6}$$

Proof (of Theorem 2). First, note that $\mathcal{F}^n_{\mathsf{CG}}$ is realizable using only $\mathcal{F}^n_{\mathsf{BC}}$ by Lemma 16 from [BMM+20], so we explicitly use $\mathcal{F}^n_{\mathsf{CG}}$ in the protocol contruction. Denote the set of parties by $P = \{\mathsf{S}, \mathsf{R}_1, ..., \mathsf{R}_{n-1}\}$. We require all honest parties to locally compute the *effective* Conflict Graph $G^* \leftarrow \phi(G)$ when obtaining G from $\mathcal{F}^n_{\mathsf{CG}}$. Moreover, whenever G^* becomes biseparated, the honest parties abort the protocol with the opposing partition as described in Section 4.

Before we give a formal proof we outline the protocol intuitively. The commit phase goes as follows:

1. The sender draws a random bits $b \leftarrow \{0,1\}^n$ and secret-shares its message m as $\mu \leftarrow \mathrm{Share}_{l,4l}(m)$ and each bit of b as $\xi^j \leftarrow \mathrm{Share}_{l,4l}(b_j)$. Then the sender sends the masked sharing $\sigma \coloneqq \mu \oplus \bigoplus_{\mathsf{R}_i \in A} \xi^i$ to the associate receiver R_j and also inputs (σ, A) into all non-aborted EVCOM setups where $A \coloneqq \mathrm{N}_{\overline{G}}(\mathsf{S})$.
2. Each receiver R_j who obtained ξ^j then inputs it into all non-aborted EVOT setups and acknowledges with a broadcasted receipt.
3. Each party draws random probing indices $H \leftarrow \binom{[4l]}{\rho}$ and broadcasts them.
4. Each party opens all its EVCOMs on all probed indices.
5. Each party checks the consistency of the probed shares with the broadcasted one. If the shares opened in a setup are not equal to the broadcasted ones, then that setup will become aborted in the next round, because honest parties declare conflicts with the offender and its loyalists. Next, the parties check consistency of the broadcasted shares, i.e. $\sigma_\kappa \overset{?}{=} \mu_\kappa \oplus \bigoplus_{\mathsf{R}_i \in A} \xi^i_\kappa$ on all probed indices $\kappa \in H$.
   - If all shares are consistent, each receiver outputs a receipt.
   - Otherwise the parties that notice the inconsistency proceed to declare conflicts with the parties whose broadcasted shares are not equal to the ones opened in the EVCOM. If the second consistency check fails, all parties expect the sender to declare a conflict with at leat one of its current associates. If the sender does not by the next round, it must be malicious.

The opening phase goes as follows:

1. Each party opens all EVCOM setups in which it committed to some shares.[6]
2. Each receiver that receives all necessary shares reconstructs the message m and broadcasts it.
3. Each receiver declares a conflict with each receiver that broadcasts another message.
4. Each receiver who did not receiving all sufficient shares either abort the protocol if the CG is biseparated or the output the majority of what their associates broadcasted.

The crux of the protocol can be made intuitive by looking at Fig. 1c). Suppose the malicious sender is represented as 1 and the honest parties are $7, 8, 9$. Now, the adversary can abort setups in such a way that the sender and honest parties are not jointly in any setup—thus cutting of the honest parties from obtaining the opening information. However, the party 7 will always be in some setup with each associates of the sender. Therefore the sender gives its associates an additively masked

[6] Due to the verification step in the commit phase the shares in each setup must reconstruct to the same message except with negligible probability.

version of the message sharing. Because the sender's associates are committed towards party 7, it can recover the message and parties 8 and 9 can output the same message because they know that 7 is honest as well. In turn, if the sender was honest, then malicious receivers don't obtain any information about the message from σ and the ξ^j's because at least one additive share of the mask is held by an honest associate of the sender.

Now, we provide a more formal version of the proof. Let $P = \{\mathsf{S}, \mathsf{R}_1, ..., \mathsf{R}_{n-1}\}$ denote the set of parties, for notational purposes we use the notation $\mathsf{R}_0 = \mathsf{S}$. Let $\ell := n\lambda^2$ be the size of the $(\ell, 4\ell)$-threshold sharing and let $\rho := \lambda$ be the number of probing shares per party.[7] Throughout the protocol let $I(S) := \left\{M \subseteq P \;\middle|\; |M| = \beta \wedge S \subseteq M \wedge (M, E \cap 2^M) \text{ not biseparated}\right\}$ be the set of not (yet) aborted instance classes that contain the subset of parties S. Note that this set might dynamically change during the protocol as setups are aborted. We use the number $\iota := \left|\mathrm{N}_{\overline{G}}(\mathsf{S})\right|$ as a counter, i.e. everytime ι decreases the protocol restarts—the sender behaves as if it received $(\texttt{input}, m)$. We denote an instance of $\mathcal{F}^{\beta}_{\mathsf{EVCOM}}$ on some set of parties $M \in I(\emptyset)$ with session id ι by $\mathcal{V}_{M,\iota}$.

[7] The choice of parameters is not optimal, we chose these because they simplify the analysis and yield a secure protocol even for superpolynomial $\binom{n}{\beta}$ where parties need superpolynomial runtime.

Protocol π^n_{COM}

The protocol π^n_{COM} proceeds as follows, running with parties $P = \{\mathsf{S}, \mathsf{R}_1, \ldots, \mathsf{R}_{n-1}\}$, malicious parties $C \subseteq P$, adversary $\mathcal{A}$ and environment $\mathcal{Z}$. Messages not covered here are ignored.

- **In the first round** each party $G^*_{\text{ref}} \leftarrow (P, \emptyset)$ and stores $\iota_{\text{ref}} \leftarrow \left|\mathrm{N}_{\overline{G}}(\mathsf{S})\right|$.
- **Each round** each party inputs $(\texttt{query})$ into $\mathcal{F}^n_{\mathsf{CG}}$ to obtain the CG in the next round.
- **On output** $G = (G, E)$ from $\mathcal{F}^n_{\mathsf{CG}}$, the party P computes the new $\iota \leftarrow \left|\mathrm{N}_{\overline{G}}(\mathsf{S})\right|$. If $\iota < \iota_{\text{ref}}$, it deletes all previously received messages and updates the stored $G^*_{\text{ref}} \leftarrow \phi(G)$ and $\iota_{\text{ref}} \leftarrow \iota$. If the sender received $(\texttt{commit}, m)$ from $\mathcal{Z}$ before, it behaves as if it received it again.
- **On input** $(\texttt{commit}, m \in \{0,1\})$ from $\mathcal{Z}$, the sender S creates a sharing $\mu \leftarrow \mathrm{Share}_{\ell,4\ell}(m)$, and inputs $(\texttt{commit}, \mu)$ into $\mathcal{V}_{M,\iota}$ for each $M \in I(\{\mathsf{S}\})$. The sender draws bits $b \leftarrow \{0,1\}^{n-1}$ and shares each bit $\xi^j \leftarrow \mathrm{Share}_{l,4l}(b_j)$. Then the sender sends $(\texttt{shares}, \xi^j)$ to R_j where $A \leftarrow \mathrm{N}_{\overline{G}}(\mathsf{S})$ and $\sigma \coloneqq \mu \oplus \bigoplus_{\mathsf{R}_i \in A} \xi^i$. Then S broadcasts $(\texttt{commit}, \sigma, A)$. The sender ignores further input of this type except when the counter ι is decreased.
- **On output** $(\texttt{shares}, \xi^j)$ from $\mathcal{F}^2_{\mathsf{SMT}}$, the party R_j with $j \in [0, n-1]$ inputs $(\texttt{commit}, (\xi^j))$ into all $\mathcal{V}_{M,\iota}$ for each $M \in I(\{\mathsf{R}_j\})$ and stores ξ^j. The R_j broadcasts $(\texttt{receipt})$.
- **On output** $(\texttt{output}, \mathsf{S}, (\texttt{commit}, \sigma, A))$ from $\mathcal{F}^n_{\mathsf{BC}}$, the receiver $\mathsf{R}_i \in A$ checks whether $A \subseteq \mathrm{N}_{\overline{G}}(\mathsf{S})$. If not, then R_i aborts with $(\texttt{abort}, \{\mathsf{S}\})$. Receiver R_i also checks whether it received $(\texttt{shares}, \xi^j)$ before. If not, then it inputs $(\texttt{conflict}, \mathsf{S})$ into $\mathcal{F}^n_{\mathsf{CG}}$. (Note that ι decreases by one.) Next, R_i checks if it received $(\texttt{receipt}, \mathsf{R}_j)$ from $\mathcal{V}_{M,\iota}$ for all $\mathsf{R}_j \in A$ and M s.t. $M \in I(\{\mathsf{R}_i, \mathsf{R}_j\})$. If so, then it inputs $(\texttt{conflict}, \mathsf{S})$ into $\mathcal{F}^n_{\mathsf{CG}}$ and adds all (R_j, M) to its list L_i for which R_i did not receive a receipt. Then R_i samples some probing indices $H_i \leftarrow \binom{[4\ell]}{\rho}$ and broadcasts $(\texttt{probe}, H_i)$. Then R_i stores $A_{\text{fix}} \leftarrow A$.
- **On output** $(\texttt{output}, \mathsf{R}_j, (\texttt{probe}, H_i))$ from $\mathcal{F}^n_{\mathsf{BC}}$ for all $j \in [n-1]$, each party R_i with $i \in [0, n-1]$ opens its shares at the probed indices by inputting $(\texttt{open}, \kappa)$ into $\mathcal{V}_{M,\iota}$ for each $M \in I(\{\mathsf{R}_i\})$ and each $\kappa \in H' \coloneqq \bigcup_{i=1}^{n-1} H_i$. Furthermore, each receiver R_j broadcasts $(\texttt{shares}, \xi^j_{H'})$ while S broadcasts $(\texttt{shares}, \mu^j_{H'})$.
- **On output** $(\texttt{output}, \mathsf{S}, \kappa, \mu_\kappa)$ from $\mathcal{V}_M$, the receiver R_i stores $\mu^M_\kappa \leftarrow \gamma_\kappa$.
- **On output** $(\texttt{output}, \mathsf{R}_j, \kappa, \xi^j_\kappa)$ from $\mathcal{V}_M$, the party R_i stores $\xi^{j,M}_\kappa \leftarrow \xi^j_\kappa$.
- **On output** $(\texttt{output}, \mathsf{S}, (\texttt{shares}, \mu'_{H'}))$ from $\mathcal{F}^n_{\mathsf{BC}}$, the receiver R_i checks consistency with the opened shares. If $\exists M \in I(\{\mathsf{S}, \mathsf{R}_i\}) : \mu^M_{H'} \neq \mu'_{H'}$, then receiver R_i declares a conflict with S, i.e. it inputs $(\texttt{conflict}, \mathsf{S})$ into $\mathcal{F}^n_{\mathsf{CG}}$ and stores $L_k \leftarrow \{(\mathsf{S}, M) \mid \mu^M_{H'} \neq \mu'_{H'}\}$.
- **On output** $(\texttt{output}, \mathsf{R}_j, (\texttt{shares}, \xi'^j_{H'}))$ from $\mathcal{F}^n_{\mathsf{BC}}$, the receiver R_i checks consistency with the opened shares. If $\exists M \in I(\{\mathsf{R}_j, \mathsf{R}_i\}) : \xi^{M,j}_{H'} \neq \xi'^j_{H'}$, then receiver R_i declares a conflict with R_j, i.e. it inputs $(\texttt{conflict}, \mathsf{R}_j)$ into $\mathcal{F}^n_{\mathsf{CG}}$ and stores $L_k \leftarrow L_k \cup \{(\mathsf{R}_j, M) \mid \mu^M_{H'} \neq \mu'_{H'}\}$. The sender checks whether $\xi'^j_{H'} = \xi^j_{H'}$, if not then S inputs $(\texttt{conflict}, \mathsf{R}_j)$ into $\mathcal{F}^n_{\mathsf{CG}}$. Furthermore, if $\sigma_{H'} \neq \mu'_{H'} \oplus \bigoplus_{\mathsf{R}_i \in A_{\text{fix}}} \xi'^i_{H'}$, then receiver R_i stores $\delta \leftarrow 1$, otherwise $\delta \leftarrow 0$. Each party broadcasts $\texttt{OK}^1$.
- **On output** $(\texttt{output}, \mathsf{R}_i, \texttt{OK}^1)$ from $\mathcal{F}^n_{\mathsf{BC}}$ for all $i \in [0, n-1]$, the receiver R_k with $k \in [0, n-1]$ declares conflicts with all parties that stayed loyal to the offending party. I.e. for each $(\mathsf{R}_j, M) \in L_k$ the receiver R_k inputs $(\texttt{conflict}, \mathrm{N}_{\overline{G}}(\mathsf{R}_j) \cap M)$ into $\mathcal{F}^n_{\mathsf{CG}}$. If $\delta = 1 \wedge \mathrm{N}_{\overline{G}}(\mathsf{S}) = A_{\text{fix}}$, then R_k aborts with $(\texttt{abort}, \mathsf{S})$. Otherwise R_k broadcasts $\texttt{OK}^2$.
- **On output** $(\texttt{output}, \mathsf{R}_i, \texttt{OK}^2)$ from $\mathcal{F}^n_{\mathsf{BC}}$ for all $i \in [0, n-1]$, each party outputs $(\texttt{receipt}\ \texttt{commit})$ and stores $\iota_{\text{fix}} \leftarrow \iota$.

Protocol π^n_{COM} (cont'd)

- **On input** $(\texttt{open})$ from $\mathcal{Z}$, sender S opens all shares by inputting $(\texttt{open}, \kappa)$ into $\mathcal{V}_M$ for each $\kappa \in [\ell]$ and each $M \in I(\{\mathsf{S}\})$, and broadcasts $\texttt{open}$.
- **On output** $(\texttt{output}, \mathsf{S}, \kappa, \mu_\kappa)$ from $\mathcal{V}_{M,\iota_{\text{fix}}}$, the receiver R_k stores μ_κ.
- **On output** $(\texttt{output}, \mathsf{S}, \texttt{open})$ from $\mathcal{F}^n_{\mathsf{BC}}$, the receiver R_k opens its EVCOMs by inputting $(\texttt{open}, \kappa)$ into $\mathcal{V}_{M,\iota_{\text{fix}}}$ for each $\kappa \in [\ell]$. If party R_k obtained μ_κ for all $\kappa \in [\ell]$, i.e. if $I(\mathsf{S}, \mathsf{R}_k) \neq \emptyset$, then R_k broadcasts and stores $m \leftarrow \text{Recover}_{4\ell}(\mu)$.
- **On output** $(\texttt{open}, \mathsf{R}_j, \xi^j_\kappa)$ from $\mathcal{V}_M$ for all A_{fix} and $\kappa \in [\ell]$, the receiver R_i recovers $m \leftarrow \text{Recover}_l(\sigma \oplus \bigoplus_{\mathsf{R}_i \in A_{\text{fix}}} \xi^i)$. Then R_i broadcasts and stores m.
- **On output** $(\texttt{output}, \mathsf{S}, \texttt{open})$ and $(\texttt{output}, \mathsf{R}_i \in \mathrm{N}_{\overline{G}}(\mathsf{R}_k), m'^i)$ from $\mathcal{F}^n_{\mathsf{BC}}$, the receiver R_k inputs $(\texttt{conflict}, \mathsf{R}_i)$ into $\mathcal{F}^n_{\mathsf{CG}}$ for all $m'^i \neq m$. In the next round, if the CG is biseparated R_k aborts, otherwise outputs $(\texttt{open}, m)$ and terminates, if it recovered m itself, or it outputs the majority bit of whatever its associates broadcasted.

Before analyzing the protocol in detail we give a simulator for the canonical dummy adversary. The simulator runs a simulated protocol where it executes the protocol code for all non-corrupted parties. Whenever the environment activates an honest dummy party, the simulator is notified and simulates the party's protocol code. Any message from and to malicious parties are forwarded to the simulated setup functionalities.

Simulator for π^n_{COM}

- **On output** $(\texttt{receipt commit})$ from $\mathcal{F}^n_{\mathsf{COM}}$, the simulator $\mathcal{S}$ gives input $(\texttt{commit}, 0)$ to the simulated sender S'.
- **On output** $(\texttt{receipt commit})$ from all simulated (honest) R'_j, the simulator $\mathcal{S}$ sends $(\texttt{commit}, \widetilde{m})$ to the ideal $\mathcal{F}^n_{\mathsf{COM}}$ in the name of S where $\widetilde{m} \leftarrow \text{Recover}_{4\ell}(\widetilde{\mu}_{M',\iota_{\text{fix}}})$ and M' is the canonically smallest set in $I(\{\mathsf{S}\})$.
- **On output** $(\texttt{open}, m')$ from $\mathcal{F}^n_{\mathsf{COM}}$, the simulator $\mathcal{S}$ gives local input $(\texttt{open})$ to the simulated sender S' which will open the remaining setups $\mathcal{V}_{M,\iota_{\text{fix}}}$ for all $M \in I(\{\mathsf{S}\})$. Here, the simulator equivocates each remaining simulated $\mathcal{V}_{M,\iota_{\text{fix}}}$ to a random sharing $\widetilde{\mu} \leftarrow \text{Share}_{\ell,4\ell}(m')$ s.t. $\widetilde{\mu}_{H'} = \mu_{H'}$. The simulated $\mathcal{V}_{M,\iota_{\text{fix}}}$ on input $(\texttt{open}, \kappa \in [\ell])$ from S' sends $(\texttt{open}, \mathsf{S}, \kappa, \widetilde{\mu}_\kappa)$ to the simulated dummy adversary. This is possible because at most $n\rho$ shares have been probed (at least $4\ell - n\rho$ remain veiled) but the simulator only needs to equivocate $3\ell + 1 = 3n\lambda^2 + 1 \leq n(4\lambda^2 - \lambda) = 4\ell - n\rho$. The same strategy applies for the opening of the ξ^j's of the honest receivers.
- **On output** $(\texttt{open}, m')$ from all simulated R'_j, the simulator $\mathcal{S}$ sends $(\texttt{open})$ to $\mathcal{F}^n_{\mathsf{COM}}$ in the name of S.
- **On input** $\big(\texttt{abort}, C'\big)$ for any simulated setup session with parties $M \subseteq P$ and $\emptyset \neq C' \subset M \cap C$, the simulator aborts that session by forwarding $(\texttt{abort}, C')$ to it.
- **On output** $(\texttt{abort}, C')$ from all simulated parties, the simulator $\mathcal{S}$ inputs $(\texttt{abort}, C')$ into $\mathcal{F}^n_{\mathsf{FCOT}}$.

From the description of the simulator it is apparent that the honest dummy parties output the receipt resp. the opened value exactly when the simulated (honest) parties would output the receipt resp. opened value. It is also clear from the simulator's description that for an honest party the simulator can equivocate its input into simulated setup functionalities to let the simulated parties output $\widetilde{m}$ which the simulator receives from $\mathcal{F}^n_{\mathsf{COM}}$ at the start of the opening phase.

Henceforth, we focus on proving that honest (simulated) receivers actually output $\widetilde{m}$. To see why the protocol works, we make three observations conditioned on the fact that the protocol does not abort.

Observation 1 *When all (honest) parties output* $(\texttt{receipt commit})$ *in the real/simulated protocol, for all* $M \in I(\{\mathsf{S}\})$ *s.t.* $M \cap H \neq \emptyset$ *the sharings* $\widetilde{\mu}^M$ *in* $\mathcal{V}_{M,\iota_{\text{fix}}}$ *encode the same message* $\widetilde{m}$ *with*

overwhelming probability. The same holds for the sharings $\widetilde{\xi}^{M,j}$ *for all* $\mathsf{R}_j \in A_{\text{fix}}$ *and* $M \in I(\{\mathsf{R}_j\})$ *s.t.* $M \cap H \neq \emptyset$.

We prove Observation 1 by contradiction. Note that all sets M contain at least one honest party. Therefore, when all (honest) parties output (`receipt commit`), then there exists a set of (at least ρ uniformly chosen) indices H' on which the shares $\widetilde{\mu}^M$ of all setups must equal the broadcasted shares $\widetilde{\mu}^{\text{BC}}$. Formally,

$$\exists H' \in \binom{[4\ell]}{\rho} \forall M \in I(\{\mathsf{S}\}) : M \cap H \neq \emptyset \implies \widetilde{\mu}^M_{H'} = \widetilde{\mu}^{\text{BC}}_{H'} \ .$$

Now, suppose for contradiction that there exist two setups M and M' whose sharings encode different messages. Then $\exists \widetilde{H} \in \binom{[4\ell]}{\ell+1} \forall \kappa \in \widetilde{H} : \widetilde{\mu}^M_\kappa \neq \widetilde{\mu}^{M'}_\kappa$. In turn this means that $H' \cap \widetilde{H} = \emptyset$. By Lemma 1 the probability of that event is bounded by

$$\Pr_{H'}\left[\widetilde{H} \cap H' = \emptyset \;\middle|\; H' \in \binom{[4\ell]}{\rho}\right] \leq 2^{-\rho(\ell+1)/4\ell} \leq 2^{-\rho/4} = 2^{-\lambda/4} \ . \tag{29}$$

Conversely, with probability at least $1 - 2^{-\lambda/4}$ all relevant sharings encode the same message. The same argument holds for the masks $\xi^{M,j}$.

Recall that the simulator extracts the message $\widetilde{m}$ from the malicious sender's inputs into the setups during the commitment phase. Observation 1 combined with the fact that (`receipt commit`) is only output if $\sigma = \mu \oplus \bigoplus_{\mathsf{R}_i \in A_{\text{fix}}} \xi^i$ guarantees that $\widetilde{m}$ is equal to whatever message (if any) *all* honest receivers output during the opening phase. For an honest sender it obviously holds that $\widetilde{m} = m$.

Next, we show that honest receivers can indeed output the message $\widetilde{m}$. To this end we first show that *some* honest receiver can output $\widetilde{m}$ by recovering it from at least one setup.

Observation 2 *At least one honest receiver is able to reconstruct the message by receiving all shares:* $\exists \mathsf{R} \in H \setminus \{\mathsf{S}\}\, \exists \mathsf{A} \in \mathrm{N}_{\overline{G}}(\mathsf{S}) : I(\{\mathsf{A}, \mathsf{R}\}) \neq \emptyset$.

Again, we prove the observation by contradiction. First, note that ϕ preserves the "no honest-honest" property of the CG. That is, if $E \cap 2^H = \emptyset \implies E^* \cap 2^H = \emptyset$. This follows readily from the fact—mentioned in Section 4—that all honest parties form an h-clique in $\overline{G}$ whereas ϕ adds edges between exactly those parties that are not in an h-clique in $\overline{G}$. Therefore at least one party of any new edge must be malicious.

If an honest receiver R is in some setup with S or in some setup with each associate of the sender, the R can recover the correct message. In the first case R can simply recover directly $m \leftarrow \text{Recover}_l(\mu)$, in the second case R can recover $m \leftarrow \text{Recover}_l(\sigma \oplus \bigoplus_{\mathsf{R}_i \in A_{\text{fix}}} \xi^i)$ because R obtains all ξ^j's of all associates of the sender.

Now, suppose for contradiction $\forall \mathsf{R} \in H \setminus \{\mathsf{S}\}\, \forall \mathsf{A} \in \mathrm{N}_{\overline{G}}(\mathsf{S}) : I(\{\mathsf{A}, \mathsf{R}\}) = \emptyset$. The statement $I(\{\mathsf{A}, \mathsf{R}\}) = \emptyset$ means that all setups on parties M s.t. $\mathsf{A}, \mathsf{R} \in M \wedge |M| = \beta$ must be aborted (biseparated). However, in this case Lemma 3 for $V \coloneqq P$, $v = \mathsf{S}$ and honest parties $H \subseteq P$ states that the overall CG G^* must be biseparated. Since the honest parties can compute G^* efficiently from G queried from $\mathcal{F}^n_{\mathsf{CG}}$, they can abort the protocol. Hence if no abort occurs, at least one honest receiver must obtain $\widetilde{m}$.

Lastly, we show that any receiver that does not obtain $\widetilde{m}$ from some setup can rely on its associates to output $\widetilde{m}$.

Observation 3 *If an honest "unlucky" receiver does not receive any shares and the CG is not biseparated, it can output whichever message the majority of its associates output. Formally,* $\forall \mathsf{R} \in H : \big(\forall \mathsf{A} \in \mathrm{N}_{\overline{G}}(\mathsf{S}) : I(\{\mathsf{A}, \mathsf{R}\}) = \emptyset\big) \implies \mathrm{N}_{\overline{G}}(\mathsf{R}) \subseteq H$.

Note that for $h = 2$ a receiver that does not recover the message directly has only one associate who must be honest. For $h \geq 3$ at most $h - 1$ associates of an unlucky receiver can recover the message directly. Furthermore, it is obvious from the CG which ones those are. We want to recall Fig. 1b), here suppose 1 is the sender and 8 is the unlucky receiver, then only 5 and 6 are the receivers that recover the message from the masks and the masked sharings obtained from 2 and 3. If 5 and 6 are honest, they both broadcast m. If one deviates and broadcasts something else,

then 5 and 6 get into a conflict. By the definition of the map ϕ we now also find conflicts between 4 and 5, and 4 and 6, thus the overall CG is biseparated and all (honest) parties can abort.

For a malicious sender we have shown that the simulator extracts the message $\widetilde{m}$ that the honest receivers actually output in the opening phase.

We make a last observation for the case of an honest sender.

Observation 4 *If the sender is honest and the protocol is not aborted, then all honest receiver will output the correct message.*

If the sender is honest it will truthfully input its sharing μ into $\mathcal{V}_{M,\iota_{\text{fix}}}$ for all $M \in I(\{\mathsf{S}\})$. As such, if a receiver R recovers the message from some direct setup $M \in I(\{\mathsf{S},\mathsf{R}\})$, then it obviously recovers the correct message. Here the adversary has no way of interferring with the sharing of the message. Lastly, note that if S is honest and R is honest as well, then $I(\{\mathsf{S},\mathsf{R}\}) \neq \emptyset$ because $\{\mathsf{S},\mathsf{R}\}$ plus their extended associates always form a connected subgraph of size β of the complement CG, otherwise the complement CG would be disconnected and the protocol aborted.

The protocol requires invocation of at most $\mathcal{O}(\binom{n}{\beta}) \subseteq \text{poly}(\lambda)$ setups per restart (decrease of ι) and can be restarted at most $\mathcal{O}(n) \subseteq \text{poly}(\lambda)$ times because for each restart at least one associate of the sender is lost. □

Next, we present a protocol for Fully Committed Oblivious Transfer (FCOT). As in the classical OT the sender holds two messages m^0, m^1 and the receiver holds a choice bit c. To make the contruction easier to analyze we assume parties to have access to the global commitment setup which is provided by Theorem 2. The role of the global commitment is to commit both the sender and the receiver to their input s.t. they cannot change it after the OT phase.

Intuitively, the protocol tries to peform a *direct* OT via a setup that contains both the sender and the receiver. If this does not work, then both the sender and the receiver share their input with their associates who perform multiple OTs on their behalf. While sharing the sender's messages seems straightforward, is may not be obvious how to share the receiver's choice bit. We show a little trick how this can be accomplished. The structure of the CG guarantees that the associates can indeed perform OTs with each other, i.e. those OTs cannot be aborted without causing the overall protocol to abort.

In the proof we use the following multi-sender variant of global commitment

Functionality $\mathcal{F}^n_{\mathsf{MCOM}}$

$\mathcal{F}^n_{\mathsf{MCOM}}$ proceeds as follows, running with security parameter λ, parties $P = \{\mathsf{P}_1, \ldots, \mathsf{P}_n\}$, malicious parties $C \subseteq P$ and adversary $\mathcal{S}$. Messages not covered here are ignored.

- When receiving $(\texttt{commit}, m \in \{0,1\}^\lambda)$ from party P_i, store (P_i, m) and send $(\texttt{receipt commit}, \mathsf{P}_i)$ to all parties. Ignore further messages $(\texttt{commit}, \cdot)$ from P_i.
- When receiving $(\texttt{open})$ from party P_i and (P_i, m) is stored, send $(\texttt{open}, \mathsf{P}_i, m)$ to all parties and terminate.
- When receiving $(\texttt{abort}, C')$ from $\mathcal{S}$ with $C' \subseteq C$, then output $(\texttt{abort}, C')$ to all parties and terminate.

which can be trivially constructed from $\mathcal{F}^n_{\mathsf{COM}}$ by the Universal Composability Theorem of [Can01].

Theorem 3 (FCOT expansion). *Let n be the number of parties of which at most $t = n - h \in [n/2, n-2]$ are malicious s.t. $\binom{n}{\beta} \in \text{poly}(\lambda)$ with $\beta = \lfloor n/h \rfloor + \lfloor (n-1)/h \rfloor - 1$. There is a protocol π^n_{FCOT} that statistically securely UC-realizes $\mathcal{F}^n_{\mathsf{FCOT}}$ in the $\{\mathcal{F}^2_{\mathsf{SMT}}, \mathcal{F}^\beta_{\mathsf{EVOT}}, \mathcal{F}^n_{\mathsf{COM}}, \mathcal{F}^n_{\mathsf{BC}}\}$-hybrid model:*

$$\left\{\mathcal{F}^2_{\mathsf{SMT}}, \mathcal{F}^\beta_{\mathsf{EVOT}}, \mathcal{F}^n_{\mathsf{COM}}, \mathcal{F}^n_{\mathsf{BC}}\right\} \overset{\text{stat}}{\rightsquigarrow} \mathcal{F}^n_{\mathsf{FCOT}} \tag{7}$$

Proof (of Theorem 3). First, note that $\mathcal{F}^n_{\mathsf{CG}}$ is realizable using only $\mathcal{F}^n_{\mathsf{BC}}$ by Lemma 16 from [BMM+20], so we explicitly use $\mathcal{F}^n_{\mathsf{CG}}$ in the protocol contruction. Moreover, note that $\{\mathcal{F}^n_{\mathsf{COM}}\} \overset{\text{UC}}{\rightsquigarrow} \mathcal{F}^n_{\mathsf{MCOM}}$ by the UC theorem. So we use $\mathcal{F}^n_{\mathsf{MCOM}}$ as well, and to reduce the notational complexity of the protocol we omit the session ids as they are clear from the context, i.e. the committed string.

Denote the set of parties by $P = \{\mathsf{S}, \mathsf{R}_1, ..., \mathsf{R}_{n-1}\}$. We require all honest parties to locally compute the *effective* Conflict Graph $G^* \leftarrow \phi(G)$ when obtaining G from $\mathcal{F}^n_{\mathsf{CG}}$. Moreover, whenever G^* becomes biseparated, the honest parties abort the protocol with the opposing partition as described in Section 4.

Before we give a formal proof we outline the protocol intuitively. The OT phase goes as follows:

1. The sender draws two mask bits at random and commits to them globally.
2. The sender creates two threshold sharings of its two masked messages.
3. The sender commits globally to each share individually.
4. The sender inputs its shares into the canonically smallest EVOT-setup that contains both the sender and the receiver.
5. The receiver creates a threshold sharings of its choice bit.
6. The receiver commits globally to each share individually.
7. The receiver inputs its sharing into the canonically smallest EVOT-setup that contains both the sender and the receiver.
8. The receiver obtains the chosen (masked) message.
9. All parties in the setup obtain some probing shares of the masked messages and the choice bit.
10. The sender and the receiver opens the global commitment corresponding to the probed indices.
11. The parties in the setup verify that the opened shares are consistent with the ones obtained from the EVOT-setup.
    - 11.1. If all shares are consistent, the sender opens the mask. The receiver obtains the chosen unmasked bit. All other parties in the setup output a receipt, followed by all other parties.
    - 11.2. If any shares are inconsistent, then all parties in the setup declare a conflict with the resp. offender and its loyalists. The current setup is aborted (biseparated), and the next setup is used with fresh sharings and reference strings, i.e. the protocol goes back to step 4.
    - 11.3. If no setups containing sender and receiver are left, the protocol goes into fallback mode.

The OT-fallback is essentially the same except that we replace the following steps:

4. If the sender has exactly h associates (including itself), then it gives the (globally committed) sharing of the message to its canonically smallest associate who then performs OTs with the receiver or its associates. If that associate performs OTs with the receiver's associate it sets up a pair of random bits for each associate s.t. their xor is equal to the xor of the messages. Otherwise, the sender creates additive sharings of its messages and distributes them among its associates.[8]
7. If the receiver has exactly h associates (including itself), then the receiver simply gives its choice bit to its canonically smallest associate who then performs OTs with the sender or its associates. Otherwise, the receiver create a random additive sharing of its choice bit and gives each associate one random bit of it. This way no (possibly malicious) associate learns the choice bit but the receiver can still recover the correct message.

Furthermore, if the verification step in the fallback mode fails, then the overall Conflict Graph becomes biseparated. The opening phase goes as follows:

1. The sender opens the global commitments to all shares of the resp. message.
2. The receiver open the global commitments to its shares.

Already the informal description of the protocol is non-trivial. To maintain a balance between readability/complexity of the protocol and rigor we decribe the formal protocol only for the case where the sender and the receiver perform a direct OT and argue the other cases informally. We suggest to keep Fig. 1 in mind when reading the protocol as it makes it easier to follow some arguments with a visual aid.

Now, we provide a more formal version of the proof. Let $\ell := n^2\lambda^2$ be the size of the applied secret sharing and let $\rho := \lambda$ be the number of probing shares.[9] Throughout the protocol let $I(S) := \left\{M \subseteq P \;\middle|\; |M| = \beta \wedge S \subseteq M \wedge (M, E \cap 2^M) \text{ not biseparated}\right\}$ be the set of not (yet) aborted instance classes that contain the subset of parties S. Note that this set might dynamically change during the protocol as setups are aborted. We use the number $\iota := |I(\{\mathsf{S}, \mathsf{R}\})|$ as a counter, i.e. everytime ι decreases the protocol restarts. For each two parties $\mathsf{P}, \mathsf{P}' \in P$ let $M_{\mathsf{P},\mathsf{P}'} := \min I(\{\mathsf{P}, \mathsf{P}'\})$ be the canonically smallest, not (yet) aborted setup class that contains the parties. Also, we denote the setup $\mathcal{F}^{\beta}_{\mathsf{EVOT}}$ on parties $M_{\mathsf{P},\mathsf{P}'}$ with session id ι by $\mathcal{T}_{\mathsf{P},\mathsf{P}',\iota}$.

[8] Actually, only those associates that are in a common setup with the receiver.

[9] We use (non-optimal) parameters for simplicity.

Protocol π^n_{FCOT} (simplified)

The protocol π^n_{FCOT} proceeds as follows, running with parties $P = \{\mathsf{S}, \mathsf{R}, \mathsf{W}_1, ..., \mathsf{W}_{n-2}\}$, malicious parties $C \subseteq P$, adversary $\mathcal{A}$ and environment $\mathcal{Z}$. Messages not covered here are ignored.

- **In the first round** each party stores $G^*_{\text{ref}} \leftarrow (P, \emptyset)$, $\iota_{\text{ref}} \leftarrow |I(\{\mathsf{S}, \mathsf{R}\})|$ and $D_{\mathsf{S}}, D_{\mathsf{R}} \leftarrow \emptyset$.
- **Each round** each party inputs (`query`) into $\mathcal{F}^n_{\mathsf{CG}}$ to obtain the CG.
- **On output** $G = (G, E)$ from $\mathcal{F}^n_{\mathsf{CG}}$, the party P computes the new $\iota \leftarrow |I(\{\mathsf{S}, \mathsf{R}\})|$. If $\iota < \iota_{\text{ref}}$, it deletes all previously received messages and updates the stored $G^*_{\text{ref}} \leftarrow \phi(G)$ and $\iota_{\text{ref}} \leftarrow \iota$.
- **On input** $(\texttt{choice}, c)$ from $\mathcal{Z}$, the receiver R creates a sharing $\gamma \leftarrow \text{Share}_{\ell,4\ell}(c)$ and commits to each share μ_κ by inputting $(\texttt{commit}, (\texttt{choice share}, \kappa, \mu_\kappa))$ into $\mathcal{F}^n_{\mathsf{MCOM}}$. Then R inputs its shares $(\texttt{choice}, \mathsf{S} \to \mathsf{R}, c)$ into $\mathcal{T}_{\mathsf{S},\mathsf{R}}$. The receiver ignores further inputs of this type.
- **On output** $(\texttt{receipt commit}, \mathsf{R})$ from $\mathcal{F}^n_{\mathsf{MCOM}}$ for all shares $\mu_\kappa : \kappa \in [\ell]$, the party P outputs (`receipt choice`) if it did not output (`receipt messages`) before.
- **On input** $\left(\texttt{messages}, m^0, m^1\right)$ from $\mathcal{Z}$, the sender S samples a mask $w^0, w^1 \leftarrow \{0,1\}$ and commits globally to $(\texttt{mask}, w^0, w^1)$. Then S creates a sharing $\mu^b \leftarrow \text{Share}_{\ell,4\ell}(m^b \oplus w^b)$ for both $b \in \{0,1\}$ and commits to each share μ^b_κ by inputting $(\texttt{commit}, (\texttt{message share}, b, \mu^b_\kappa))$ into $\mathcal{F}^n_{\mathsf{MCOM}}$. Then S inputs its shares $(\texttt{messages}, \mathsf{S} \to \mathsf{R}, \mu^0, \mu^1)$ into $\mathcal{T}_{\mathsf{S},\mathsf{R}}$. The sender ignored further inputs of this type form $\mathcal{Z}$.
- **On output** $(\texttt{receipt commit}, \mathsf{S})$ from $\mathcal{F}^n_{\mathsf{MCOM}}$ for all sessions containing the shares $\mu^b_\kappa : \kappa \in [\ell], b \in \{0,1\}$ and the masks w^0, w^1, the party P outputs (`receipt messages`) if it did not output (`receipt choice`) before.
- **On output** $(\texttt{output}, \mathsf{S} \to \mathsf{R}, \mu^c)$ from $\mathcal{T}_{\mathsf{S},\mathsf{R}}$, the receiver R stores μ^c.
- **On output** $(\texttt{output}, \mathsf{S} \to \mathsf{R}, H', \widetilde{\mu}^0_{H'}, \widetilde{\mu}^1_{H'}, \widetilde{\gamma}_{H'})$ from $\mathcal{T}_{\mathsf{S},\mathsf{R}}$, the party P stores $(H', \widetilde{\mu}^0_{H'}, \widetilde{\mu}^1_{H'}, \widetilde{\gamma}_{H'})$. The sender S opens all global commitments corresponding to the probed shares by inputting (`open`) into $\mathcal{T}_{\mathsf{S},\mathsf{R}}$ for each session containing $\kappa \in H'$ and $b \in \{0,1\}$. The receiver opens all global commitments for the probed shares by inputting (`open`) into $\mathcal{T}_{\mathsf{S},\mathsf{R}}$ for each session containing $\kappa \in H'$.
- **On output** $(\texttt{open}, \mu^b_\kappa)$ from $\mathcal{F}^n_{\mathsf{MCOM}}$ for all $\kappa \in H'$ and $b \in \{0,1\}$, the party $\mathsf{P}' \in M_{\mathsf{S},\mathsf{R}}$ checks whether $\widetilde{\mu}^b_{H'} = \mu^b_{H'}$ for both $b \in \{0,1\}$. If so, P' sets $D_{\mathsf{S}} \leftarrow \emptyset$. Otherwise, P' declares a conflict with S, i.e. it inputs $(\texttt{conflict}, \mathsf{S})$ into $\mathcal{F}^n_{\mathsf{CG}}$, broadcasts $(\texttt{conflict}, \mathsf{S})$ and stores $D_{\mathsf{S}} \leftarrow M_{\mathsf{S}}$. Then the party P' checks whether $\widetilde{\gamma}_{H'} = \gamma_{H'}$. If so, P' broadcasts (`receipt transfer`) and sets $D_{\mathsf{R}} \leftarrow \emptyset$. Otherwise, P' declares a conflict with R, i.e. it inputs $(\texttt{conflict}, \mathsf{R})$ into $\mathcal{F}^n_{\mathsf{CG}}$, broadcasts $(\texttt{conflict}, \mathsf{R})$ and stores $D_{\mathsf{R}} \leftarrow M_{\mathsf{S}}$.
- **On output** $(\texttt{output}, \mathsf{P}', (\texttt{conflict}, \mathsf{D}))$ from $\mathcal{F}^n_{\mathsf{BC}}$, any party P'' declares a conflict with each party in $D_{\mathsf{D}} \cap \mathrm{N}_{\overline{G}}(\mathsf{D})$, i.e. it inputs $(\texttt{conflict}, D_{\mathsf{D}} \cap \mathrm{N}_{\overline{G}}(\mathsf{D}))$ into $\mathcal{F}^n_{\mathsf{CG}}$. The previously used $\mathcal{T}_{\mathsf{S},\mathsf{R}}$ is now aborted and the next one is used. The sender and the receiver act as if they received $\left(\texttt{messages}, m^0, m^1\right)$ resp. $(\texttt{choice}, c)$ from $\mathcal{Z}$.
- **On output** $(\texttt{output}, \mathsf{P}, (\texttt{receipt transfer}))$ from $\mathcal{F}^n_{\mathsf{BC}}$ for all $\mathsf{P} \in M_{\mathsf{S}}$, the sender opens the global commitment to the initial mask w.
- **On output** $(\texttt{open}, \mathsf{S}, (\texttt{mask}, w^0, w^1))$ from $\mathcal{F}^n_{\mathsf{MCOM}}$, the party $\mathsf{P}' \neq \mathsf{R}$ outputs (`receipt transfer`) and stores (w^0, w^1). The receiver reconstructs $m^c = w^c \oplus \text{Recover}_{4\ell}(\mu^c)$ and outputs $(\texttt{output}, m^c)$. Any party fixes $\iota_{\text{fix}} \leftarrow \iota$.
- **On input** $(\texttt{open message}, b)$ from $\mathcal{Z}$, the sender S opens the global commitments to the shares $(b, \kappa, \mu^b_\kappa)$ for each $\kappa \in [\ell]$.
- **On output** $(\texttt{open}, \mathsf{S}, (\texttt{message share}, b, \kappa, \mu^b_\kappa))$ from $\mathcal{F}^n_{\mathsf{MCOM}}$ for all $\kappa \in [\ell]$, the party P outputs $(\texttt{open message}, b, \widetilde{m}^b)$ where $\widetilde{m}^b \leftarrow \text{Recover}_{4\ell}(\mu^b) \oplus w^b$.
- **On input** $(\texttt{open choice})$ from $\mathcal{Z}$, the receiver R opens the global commitments to the shares (κ, μ_κ) for each $\kappa \in [\ell]$.
- **On output** $(\texttt{open}, \mathsf{R}, (\texttt{choice share}, \kappa, \mu_\kappa))$ from $\mathcal{F}^n_{\mathsf{MCOM}}$, the party P outputs $(\texttt{open choice}, \widetilde{c})$ with $\widetilde{c} \leftarrow \text{Recover}_{4\ell}(\gamma)$.

Before analyzing the protocol in detail we give a simulator for the canonical dummy adversary. The simulator runs a simulated protocol where it executes the protocol code for all non-corrupted

parties. Whenever the environment activates an honest dummy party, the simulator is notified and simulates the party's protocol code. Any message from and to malicious parties are forwarded to the simulated setup functionalities.
Note that when ι is decreased at least one new conflict must be declares, hence ι decreases at most n^2 times. Consequently, at most $n^2\rho$ many shares of the sender's and receiver's committed sharings are probed.

Simulator for π^n_{FCOT} (simplified)

- **On output** (`receipt messages`) from $\mathcal{F}^n_{\mathsf{FCOT}}$ or **on output** (`receipt transfer`) from $\mathcal{F}^n_{\mathsf{FCOT}}$ after (`receipt choice`), the simulator $\mathcal{S}$ gives input $(\texttt{messages}, 0, 0)$ to the simulated sender S'.
- **On output** (`receipt messages`) from all simulated (honest) parties, the simulator $\mathcal{S}$ $(\texttt{messages}, \widetilde{m}^0 \oplus w^0, \widetilde{m}^1 \oplus w^1)$ to the ideal $\mathcal{F}^n_{\mathsf{FCOT}}$ in the name of S where $\widetilde{m}^b \leftarrow \text{Recover}_{4\ell}(\widetilde{\mu}^b)$. Also, (w^0, w^1) are the masks and $\widetilde{\mu}^b$ is the sharing that the malicious sender previously input into the simulated $\mathcal{F}^n_{\mathsf{MCOM}}$.
- **On output** (`receipt choice`) from $\mathcal{F}^n_{\mathsf{FCOT}}$ or **on output** (`receipt transfer`) from $\mathcal{F}^n_{\mathsf{FCOT}}$ after (`receipt messages`), the simulator $\mathcal{S}$ gives input $(\texttt{choice}, 0)$ to the simulated receiver R'.
- **On output** (`receipt choice`) from all simulated (honest) parties, the simulator $\mathcal{S}$ $(\texttt{choice}, \widetilde{c})$ to the ideal $\mathcal{F}^n_{\mathsf{FCOT}}$ in the name of R where $\widetilde{c} \leftarrow \text{Recover}_{4\ell}(\widetilde{\gamma})$ and $\widetilde{\gamma}$ is the sharing that the malicious receiver previously input into the simulated $\mathcal{F}^n_{\mathsf{MCOM}}$.
- **On output** $(\texttt{open choice}, \widetilde{c})$ from $\mathcal{F}^n_{\mathsf{FCOT}}$, the simulator $\mathcal{S}$ gives local input (`open`) to the simulated receiver R' which will open the remaining setups $\mathcal{F}^n_{\mathsf{MCOM}}$. However, the simulator equivocates the remaining simulated $\mathcal{F}^n_{\mathsf{MCOM}}$ containing share $\widetilde{\gamma}_\kappa$ for each $\kappa \in [\ell] \setminus H'$ s.t. all opened shares form a random sharing $\gamma' \leftarrow \text{Share}_\ell(\widetilde{c})$ where the shares on H' match the already opened shares, i.e. $\forall \kappa \in H' : \gamma'_\kappa = \widetilde{\gamma}_\kappa$. I.e. the simulated $\mathcal{F}^n_{\mathsf{MCOM}}$ on input (`open`) from R sends $(\texttt{open}, (\kappa, \widetilde{\gamma}_\kappa))$ to all parties and the simulated dummy adversary. This is possible because at most $n^2\rho$ shares have been probed but the simulator only needs to equivocate $3\ell + 1 = 3n^2\lambda^2 + 1 \leq n^2(2\lambda^2 - \lambda = 4\ell - n^2\rho)$ shares.
- **On output** $(\texttt{open choice}, \widetilde{c})$ from all simulated parties, the simulator sends (`open choice`) to the ideal $\mathcal{F}^n_{\mathsf{FCOT}}$ in the name of R.
- **On output** $(\texttt{open message}, b, \widetilde{m}^b)$ from $\mathcal{F}^n_{\mathsf{FCOT}}$, the simulator $\mathcal{S}$ gives local input (`open`) to the simulated sender S' which will open the remaining setups $\mathcal{F}^n_{\mathsf{MCOM}}$. Again, the simulator equivocates the remaining simulated $\mathcal{F}^n_{\mathsf{MCOM}}$ containing share $\widetilde{\mu}^b_\kappa$ for each $\kappa \in [\ell] \setminus H'$ s.t. all opened shares form a random sharing $\mu'^b \leftarrow \text{Share}_{\ell,4\ell}(\widetilde{m} \oplus w^b)$ where the shares on H' match the already opened shares, i.e. $\forall \kappa \in H' : \mu'^b_\kappa = \widetilde{\mu}_\kappa$. I.e. the simulated $\mathcal{F}^n_{\mathsf{MCOM}}$ on input (`open`) from S sends $(\texttt{open}, (b, \kappa, \widetilde{\mu}_\kappa))$ to all parties and the simulated adversary. This is possible because at most $n^2\rho$ shares have been probed but the simulator only needs to equivocate $3\ell + 1 = 3n^2\lambda^2 + 1 \leq n^2(2\lambda^2 - \lambda = 4\ell - n^2\rho)$ shares.
- **On output** $(\texttt{open message}, b, \widetilde{m}^b)$ from all simulated parties, the simulator inputs $(\texttt{open message}, b)$ to the ideal $\mathcal{F}^n_{\mathsf{FCOT}}$ in the name of S.
- **On input** $\big(\texttt{abort}, C'\big)$ for any simulated setup with parties $M \subseteq P$ and $\emptyset \neq C' \subset M \cap C$, the simulator aborts that instance by forwarding $\big(\texttt{abort}, C'\big)$ to it.
- **On output** $(\texttt{abort}, C')$ from all simulated parties, the simulator $\mathcal{S}$ inputs $(\texttt{abort}, C')$ into $\mathcal{F}^n_{\mathsf{FCOT}}$.

From the description of the simulator it is apparent that the honest dummy parties output the receipt resp. the opened value exactly when the simulated (honest) parties would output the receipt resp. opened value. It is also clear from the simulator's description that for an honest sender resp. receiver the simulator can equivocate its simulated setup functionalities to let the simulated parties output $\widetilde{m}^0, \widetilde{m}^1$ resp. $\widetilde{\gamma}$ which the simulator receives from $\mathcal{F}^n_{\mathsf{FCOT}}$ at the start of the opening phase. In particular, because the simulator extracts its inputs $\widetilde{m}^0, \widetilde{m}^1$ and $\widetilde{\gamma}$ from whatever inputs are encoded in the sharings in the global commitments and all parties output exactly the opened value from the global commitments, the opened messages resp. choice bit are equal in the real and ideal

run.
Henceforth, we focus on proving that

- When all parties output (`receipt transfer`), then the inputs of the last used setup $\mathcal{T}_{\mathsf{S},\mathsf{R}}$ are indeed equal to the inputs encoded in the sharings contained in the global commitments.
- All (honest) parties eventually output (`receipt transfer`) or the protocol aborts.

To this we make some observations.

Observation 5 *When all (honest) parties output* (`receipt transfer`) *in the real/simulated protocol and the last used setup $\mathcal{T}_{\mathsf{S},\mathsf{R}}$ contains some honest party, the inputs encoded in the sharings of $\mathcal{T}_{\mathsf{S},\mathsf{R}}$, i.e. $\widetilde{\mu}^b$ and $\widetilde{\gamma}$, and the globally committed sharings $\widehat{\mu}^b$ and $\widehat{\gamma}$ are equal except with negligible probability.*

We prove Observation 5 similarly to Observation 1.

Suppose the choice bit encoded in the sharing of $\mathcal{T}_{\mathsf{S},\mathsf{R}}$, i.e. $\widetilde{\gamma}$, and the globally committed sharing $\widehat{\gamma}$ differ. Then there exists a set $W \in \binom{[4\ell]}{\ell+1}$ of differing indices s.t. $\forall \kappa \in W : \widetilde{\gamma}_\kappa \neq \widehat{\gamma}_\kappa$. Furthermore W must be disjoint from the uniformly random probing indices $H' \leftarrow \binom{[4\ell]}{\rho}$, otherwise the parties $M_{\mathsf{S}^*} \cap H \neq \emptyset$ would have declared a conflict with the sender and/or the receiver. Lemma 1 bounds the probability that no differing share is probed by

$$\Pr_{H'}\left[W \cap H' = \emptyset \;\middle|\; H' \subseteq [\ell] \wedge \left|H'\right| \geq \rho\right] \overset{\text{Eq. (29)}}{\leq} 2^{-\rho(\ell+1)/4\ell} \leq 2^{-\lambda/4} \in \mathrm{negl}(\lambda)\ . \tag{30}$$

The same argument applies to the two sharings for the sender's messages. Consequently, by the union bound all three inputs are equal in the setup and in the global commitments with probability at least $1 - 3 \cdot 2^{-\lambda/2}$.

It remains to show that all parties eventually output (`receipt transfer`) if the protocol does not abort. In the above description of the protocol we have assumed that at least one setup containing both the sender and the receiver will not be aborted. If this was the case we would already have concluded the proof. In the remaining part we argue that even if the sender and the receiver cannot perform a direct OT the protocol still works. All setups take sharings as inputs so that the inputs can be verified against the global commitments of the sender and the receiver.

To this end we make some observations.

Observation 6 *If the sender and the receiver are connected via a path of length $\beta + 1$ in the complement CG, then the sender can simply create an additive sharing for each message and let its associates perform an OT with the receiver.*

Suppose the sender is honest, then the adversary still cannot learn both messages because at least one associate of the sender who holds an additive share of both messages is honest as well. Suppose the receiver is honest, then the adversary cannot learn the choice bit because the choice bit is only input into the EVOT setups which by definition does not leak it to the adversary.

A more complicated case arises when the sender and the receiver are not connected by a path of length $\beta+1$. However, the structure of the CG (compare 1) dictates that either the sender or the receiver must have h associates respectively (including themselves). This means that if the sender or receiver are honest then so are their associates. If the receiver has h associates the receiver can simply send the sharing of its choice bit to any associate how then performs OTs with the sender's associates exactly as in Observation 6. The honest sender's messages stay private for the same reason as before and the honest receiver's choice stays private because of its honest associate.

If the sender has h associates then it sets up pairs $z_i^0 \leftarrow \{0,1\}$ and $z_i^1 \leftarrow m^0 \oplus m^1 \oplus z_i^0$ for each associate of the receiver $\mathsf{P}_i \in \mathrm{N}_{\overline{G}}(\mathsf{R})$. The last bit is set up as $z_i^0 \leftarrow \oplus \bigoplus_{j \neq i} z_j^0 \oplus m^0$. Then the sender creates sharings $q_i^b \leftarrow \mathrm{Share}_{l,4l}(z_i^b)$ and give it to its canonically smallest associate. The receiver creates an additive sharing of its choice bit $\gamma \coloneqq (\gamma_i)_{\mathsf{P}_i \in \mathrm{N}_{\overline{G}}(\mathsf{R})}$ s.t. $\bigoplus_{\mathsf{P}_i \in \mathrm{N}_{\overline{G}}(\mathsf{R})} \gamma_i = c$ and distributes the share to its associates. Then the sender's associate and the receiver's associates perform OTs and the receiver's associates report back the results. Crucially, the receiver can still recover the chosen message as

$$\bigoplus_{\mathsf{P}_i \in \mathrm{N}_{\overline{G}}(\mathsf{R})} \mathrm{Recover}_l(q_i^{\gamma_i}) = \bigoplus_i z_i^{\gamma_i} = \bigoplus_i z_i^0 \oplus \left(m^0 \oplus m^1\right)\gamma_i = m^0 \oplus \left(m^0 \oplus m^1\right)c = m^c\ . \tag{31}$$

Moreover, the receiver's choice bit remains private because at least one additive share is held by an honest associate.

Overall, the we proved that the simplified version of the protocol is secure. Based on the simplified version we have argued that if the direct OT does not work, then the sender and the receiver can use their resp. associates to carry out the OT for them. □