Concurrent composition in the bounded quantum storage model*

Dominique Unruh
University of Tartu, Estonia

May 15, 2011

Abstract

We define the BQS-UC model, a variant of the UC model, that deals with protocols in the bounded quantum storage model. We present a statistically secure commitment protocol in the BQS-UC model that composes concurrently with other protocols and an (a-priori) polynomially-bounded number of instances of itself. Our protocol has an efficient simulator which is important if one wishes to compose our protocol with protocols that are only computationally secure. Combining our result with prior results, we get a statistically BQS-UC secure protocol for general two-party computation without the need for any setup assumption. The round complexity of that protocol is linear in the circuit depth.

Contents

*A short version of this paper has appeared at Eurocrypt 2011 [Unr11].

1 Introduction

Since the inception of quantum key distribution by Bennett and Brassard [BB84], it has been known that quantum communication permits to achieve protocol tasks that are impossible given only a classical channel. For example, a quantum key distribution scheme [BB84] permits to agree on a secret key that is statistically secret, using only an authenticated but not secret channel. (By statistical security we mean security against computationally unbounded adversaries, also known as information-theoretical security.) In contrast, when using only classical communication, it is easy to see that such a secret key can always be extracted by a computationally sufficiently powerful adversary. In light of this result, one might hope that quantum cryptography allows to circumvent other classical impossibility results, possibly even allowing for statistically secure multi-party computation protocols. Yet, Mayers [May97] showed that also in the quantum setting, even statistically secure commitment schemes are impossible, let alone general multi-party computation. This is unfortunate, because from commitments one can build OT (Bennett, Brassard, Crépeau, and Skubiszewska [BBCS91]), and from OT general multi-party computation (Kilian [Kil88]). A way to work around this impossibility was found by Damgård, Fehr, Salvail, and Schaffner [DFSS05]. They showed that if we assume that the quantum memory available to the adversary is bounded (we speak of *bounded quantum storage* (BQS)), we can construct statistically secure commitment and OT schemes. Although such a result is not truly unconditional, it avoids hard-to-justify complexity-theoretic assumptions. Also, it achieves long-term security: even if the adversary can surpass the memory bound after the protocol execution, this will not allow him to retroactively break the protocol.

Yet, we still have not reached the goal of statistically secure multi-party computation. Although we have protocols for commitment and OT, we cannot simply plug them into the protocols by Bennett et al. [BBCS91] and by Kilian [Kil88]. The reason is that it is not clear under which circumstances protocols in the BQS model may be composed. For example, Dziembowski and Maurer [DM04] constructed a protocol that is secure in the classical bounded storage model, but that looses security when composed with a computationally secure protocol. To overcome this remaining difficulty, works by Wehner and Wullschleger [WW08] and by Fehr and Schaffner [FS09] give security definitions in the BQS model that enable secure sequential composition. Both works also present secure OT protocols in their respective settings. Based on these, we can construct secure multi-party computation protocols in the BQS model. There are, however, a few limitations. First, since only sequential composition is supported, all instances of the OT protocol used by the multi-party computation need to be executed one after another, leading to a high round-complexity. Second, interactive functionalities such as a commitment are difficult to use: the restriction to sequential composability requires that we have to commit and immediately open a commitment before being allowed to execute the next commitment. Third, the security proof of their OT protocols uses a computationally unlimited simulator. As discussed in [Unr10], a protocol with an unlimited simulator cannot be composed with a computationally secure protocol. Fourth, since we have no concurrent composability, it is not clear what happens if the protocols are executed in an environment where we do not have total control about which protocols are executed at what time.

To overcome the limitations of sequential composition *in the classical setting*, Canetti [Can01] introduced the Universal Composability (UC) model. In this model, protocols can be arbitrarily composed, even concurrently with other protocols and with copies of themselves. The UC model has been adapted to the quantum setting by Ben-Or and Mayers [BOM04] and by Unruh [Unr04, Unr10]. In light of the success of the UC model, it seems natural to combine the ideas of the UC model with those of the BQS model in order to allow for concurrent composition.

1.1 Our contribution.

We define the notion of BQS-UC-security, which is an extension of quantum-UC-security [Unr10]. We have composability in the following sense: If π is a secure realization of a functionality $\mathcal{F}$, and $\sigma^{\mathcal{F}}$ securely realizes $\mathcal{G}$ by using one instance of $\mathcal{F}$, then σ^{π}, the result of replacing $\mathcal{F}$ by π, still securely realizes $\mathcal{G}$. In contrast to quantum-UC-security, however, BQS-UC-security does not allow for concurrent self-composition: if π is secure, this does not automatically imply that two concurrent instances of π are secure.[1]

In order to get protocols that even self-compose concurrently, we design a commitment scheme π_{COM} such that n concurrent instances of π_{COM} securely realize n instances of the commitment functionality in the presence of a-memory bounded adversaries. Here a and n are arbitrary (polynomially-bounded), but the protocol depends on a and n.

The challenging part in the construction of π_{COM} is that BQS-UC-security requires the following: There must be an *efficient* simulator (which is allowed to have more quantum memory than the adversary) that can extract the committed value (extractability) or change it after the commit phase (equivocality). Prior constructions of commitment schemes in the BQS model required computationally unbounded simulators. Also, the fact that we directly analyze the concurrent composition of several instances of π_{COM} requires care: In the proof, we have hybrid networks in which instances of both π_{COM} and of the simulator occur. Since the simulator uses more quantum memory than π_{COM} tolerates, one needs to ensure that the simulator cannot be (mis)used by the adversary to break the commitment.

Finally, using the composition theorem and π_{COM}, for any two-party functionality $\mathcal{G}$, we get a statistically secure protocol π realizing $\mathcal{G}$ in the BQS model.[2] The protocol is secure even when running n concurrent instances of the protocol. (Again, this holds for any n and any memory bound, but the protocol depends on n and the memory-bound.) The round-complexity of the protocol is linear in the circuit depth. It does not use any quantum memory or quantum computation and thus is in the reach of today's technology.

1.2 Preliminaries

General. A nonnegative function μ is called negligible if for all $c > 0$ and all sufficiently large k, $\mu(k) < k^{-c}$. μ is called exponentially-small if for some $c > 0$ and sufficiently large k, $\mu(k) < c^{-k}$. A nonnegative function f is called overwhelming if $f \geq 1 - \mu$ for some negligible μ. Keywords in typewriter font (e.g., `environment`) are assumed to be fixed but arbitrary distinct non-empty words in $\{0,1\}^*$. $\varepsilon \in \{0,1\}^*$ denotes the empty word. We write $\|$ for the concatenation of bitstrings. Given a bitstring x, we write x_I for x restricted to the indices in the set I. Given a set M, we write $M^{\complement}$ for its complement. A b-block (m, κ, d)-linear code is a code with alphabet $\{0,1\}^b$ (that is, each symbol of the code is a b-bit string), codewords

[1]The reader may wonder how it can be that σ and π may compose in general while π and π do not. Might not σ and π be the same protocol? The reason lies in the exact conditions of the composition theorem: In order to compose, σ needs to be secure against adversaries with a higher quantum memory bound than π tolerates. Thus σ and π cannot be the same protocol.

[2]We are restricted to two-party functionalities because our construction uses a subprotocol by Wolf and Wullschleger [WW06, Wul07] to reverse the direction of an OT; this protocol only makes sense in a two-party setting.

of length m (i.e., mb bits), size $2^{b\kappa}$, and detecting $d-1$ errors (i.e., any non-zero codeword contains at least d non-zero blocks). The Hamming distance between x and x^* we denote $\omega(x,x^*)$. (In the case of a block code, this is the number of blocks, not bits, that differ.) We say a family of (m,κ,d)-linear codes, parametrized by a security parameter k, has efficient error-correction if for any x, the codeword x^* with $\omega(x,x^*) \le (d-1)/2$ can be found in deterministic time polynomial in k (if such an x^* exists).

Quantum systems. We can only give a terse overview over the formalism used in quantum computing. For a thorough introduction, we recommend the textbook by Nielsen and Chuang [NC00, Chap. 1–2]. A (pure) state in a quantum system is described by a vector $|\psi\rangle$ in some Hilbert space $\mathcal{H}$. In this work, we only use Hilbert spaces of the form $\mathcal{H} = \mathbb{C}^N$ for some countable set N, usually $N = \{0,1\}$ for qubits or $N = \{0,1\}^*$ for bitstrings. We always assume a designated orthonormal basis $\{|x\rangle : x \in N\}$ for each Hilbert space, called the computational basis. The basis states $|x\rangle$ represent classical states (i.e., states without superposition). Given several separate subsystems $\mathcal{H}_1 = \mathbb{C}^{N_1}, \dots, \mathcal{H}_n = \mathbb{C}^{N_n}$, we describe the joint system by the tensor product $\mathcal{H}_1 \otimes \cdots \otimes \mathcal{H}_n = \mathbb{C}^{N_1 \times \cdots \times N_n}$. We write $\langle\Psi|$ for the linear transformation mapping $|\Phi\rangle$ to the scalar product $\langle\Psi|\Phi\rangle$. Consequently, $|\Psi\rangle\langle\Psi|$ denotes the orthogonal projector on $|\Psi\rangle$. We set $|0\rangle_+ := |0\rangle$, $|1\rangle_+ := |1\rangle$, $|0\rangle_\times := \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, and $|1\rangle_\times := \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. For $x \in \{0,1\}^n$ and $\theta \in \{+,\times\}^n$, we define $|x\rangle_\theta := |x_1\rangle_{\theta_1} \otimes \cdots \otimes |x_n\rangle_{\theta_n}$.

Mixed states. If a system is not in a single pure state, but instead is in the pure state $|\Psi_i\rangle \in \mathcal{H}$ with probability p_i (i.e., it is in a mixed state), we describe the system by a density operator $\rho = \sum_i p_i |\Psi_i\rangle\langle\Psi_i|$ over $\mathcal{H}$. This representation contains all physically observable information about the distribution of states, but some distributions are not distinguishable by any measurement and thus are represented by the same mixed state. The set of all density operators is the set of all positive[3] operators $\mathcal{H}$ with trace 1, and is denoted $\mathcal{P}(\mathcal{H})$. Composed systems are descibed by operators in $\mathcal{P}(\mathcal{H}_1 \otimes \cdots \otimes \mathcal{H}_n)$. In the following, when speaking about (quantum) states, we always mean mixed states in the density operator representation. A mapping $\mathcal{E} : \mathcal{P}(\mathcal{H}_1) \to \mathcal{P}(\mathcal{H}_2)$ represents a physically possible operation (realizable by a sequence of unitary transformations, measurements, and initializations and removals of qubits) iff it is a completely positive trace preserving map.[4] We call such mappings superoperators. The superoperator $\mathcal{E}^m_{init}$ on $\mathcal{P}(\mathcal{H})$ with $\mathcal{H} := \mathbb{C}^{\{0,1\}^*}$ and $m \in \{0,1\}^*$ is defined by $\mathcal{E}^m_{init}(\rho) := |m\rangle\langle m|$ for all ρ.

Composed systems. Given a superoperator $\mathcal{E}$ on $\mathcal{P}(\mathcal{H}_1)$, the superoperator $\mathcal{E} \otimes id$ operates on $\mathcal{P}(\mathcal{H}_1 \otimes \mathcal{H}_2)$. Instead of saying "we apply $\mathcal{E} \otimes id$", we say "we apply $\mathcal{E}$ to $\mathcal{H}_1$". If we say "we initialize $\mathcal{H}$ with m", we mean "we apply $\mathcal{E}^m_{init}$ to $\mathcal{H}$". Given a state $\rho \in \mathcal{P}(\mathcal{H}_1 \otimes \mathcal{H}_2)$, let $\rho_x := (|x\rangle\langle x| \otimes id)\rho(|x\rangle\langle x| \otimes id)$. Then the outcome of measuring $\mathcal{H}_1$ in the computational basis is x with probability $\operatorname{tr}\rho_x$, and after measuring x, the quantum state is $\frac{\rho_x}{\operatorname{tr}\rho_x}$. Since we will only perform measurements in the computational basis in this work, we will omit the qualification "in the computational basis". The terminology in this paragraph generalizes to systems composed of more than two subsystems.

Classical states. Classical probability distributions $P : N \to [0,1]$ over a countable set N are represented by density operators $\rho \in \mathcal{P}(\mathbb{C}^N)$ with $\rho = \sum_{x \in N} P(x)|x\rangle\langle x|$ where $\{|x\rangle\}$ is the computational basis. We call a state classical if it is of this form. We thus have a canonical isomorphism between the classical states over $\mathbb{C}^N$ and the probability distributions

[3]We call an operator positive if it is Hermitean and has only nonnegative Eigenvalues.

[4]A map $\mathcal{E}$ is completely positive iff for all Hilbert spaces $\mathcal{H}'$, and all positive operators ρ on $\mathcal{H}_1 \otimes \mathcal{H}'$, $(\mathcal{E} \otimes id)(\rho)$ is positive.

over N. We call a superoperator $\mathcal{E} : \mathcal{P}(\mathbb{C}^{N_1}) \to \mathcal{P}(\mathbb{C}^{N_2})$ classical iff if there is a randomized function $F : N_1 \to N_2$ such that $\mathcal{E}(\rho) = \sum_{\substack{x\in N_1\\ y\in N_2}} \Pr[F(x) = y] \cdot \langle x|\rho|x\rangle \cdot |y\rangle\langle y|$. Classical superoperators describe what can be realized with classical computations. An example of a classical superoperator on $\mathcal{P}(\mathbb{C}^N)$ is $\mathcal{E}_{class} : \rho \mapsto \sum_x \langle x|\rho|x\rangle \cdot |x\rangle\langle x|$. Intuitively, $\mathcal{E}_{class}$ measures ρ in the computational basis and then discards the outcome, thus removing all superpositions from ρ.

2 Bounded quantum storage UC

2.1 Review of quantum-UC

Machine model. A machine M is described by an identity id_M in $\{0,1\}^*$ and a sequence of superoperators $\mathcal{E}_M^{(k)}$ ($k \in \mathbb{N}$) on $\mathcal{H}^{state} \otimes \mathcal{H}^{class} \otimes \mathcal{H}^{quant}$ with $\mathcal{H}^{state}, \mathcal{H}^{class}, \mathcal{H}^{quant} := \mathbb{C}^{\{0,1\}^*}$ (the *state transition operators*). The index k in $\mathcal{E}_M^{(k)}$ denotes the security parameter. The Hilbert space $\mathcal{H}^{state}$ represents the state kept by the machine between invocations, and $\mathcal{H}^{class}$ and $\mathcal{H}^{quant}$ are used both for incoming and outgoing messages. Any message consists of a classical part stored in $\mathcal{H}^{class}$ and a quantum part stored in $\mathcal{H}^{quant}$. If a machine id_{sender} wishes to send a message with classical part m and quantum part $|\Psi\rangle$ to a machine id_{rcpt}, the machine id_{sender} initializes $\mathcal{H}^{class}$ with $(id_{sender}, id_{rcpt}, m)$ and $\mathcal{H}^{quant}$ with $|\Psi\rangle$. (See the definition of the network execution below for details.) The separation of messages into a classical and a quantum part is for clarity only, all information could also be encoded directly in a single register. If a machine does not wish to send a message, it initializes $\mathcal{H}^{class}$ and $\mathcal{H}^{quant}$ with the empty word ε.

A network $\mathbf{N}$ is a set of machines with pairwise distinct identities containing a machine $\mathcal{Z}$ with $id_{\mathcal{Z}} = \texttt{environment}$. We write $ids_{\mathbf{N}}$ for the set of the identities of the machines in $\mathbf{N}$.

We call a machine M quantum-polynomial-time if there is a uniform[5] sequence of quantum circuits C_k such that for all k, the circuit C_k implements the superoperator $\mathcal{E}_M^{(k)}$.

Network execution. The state space $\mathcal{H}_{\mathbf{N}}$ of a network N is defined as $\mathcal{H}_{\mathbf{N}} := \mathcal{H}^{class} \otimes \mathcal{H}^{quant} \otimes \bigotimes_{id\in ids_{\mathbf{N}}} \mathcal{H}_{id}^{state}$ with $\mathcal{H}_{id}^{state}, \mathcal{H}^{class}, \mathcal{H}^{quant} := \mathbb{C}^{\{0,1\}^*}$. Here $\mathcal{H}_{id}^{state}$ represents the local state of the machine with identity id and $\mathcal{H}^{class}$ and $\mathcal{H}^{quant}$ represent the state spaces used for communication. ($\mathcal{H}^{class}$ and $\mathcal{H}^{quant}$ are shared between all machines. Since only one machine is active at a time, no conflicts occur.)

A step in the execution of $\mathbf{N}$ is defined by a superoperator $\mathcal{E} := \mathcal{E}_{\mathbf{N}}^{(k)}$ operating on $\mathcal{H}_{\mathbf{N}}$. This superoperator performs the following steps: First, $\mathcal{E}$ measures $\mathcal{H}^{class}$ in the computational basis and parses the outcome as $(id_{sender}, id_{rcpt}, m)$. Let M be the machine in $\mathbf{N}$ with identity id_{rcpt}. Then $\mathcal{E}$ applies $\mathcal{E}_M^{(k)}$ to $\mathcal{H}_{id_{rcpt}}^{state} \otimes \mathcal{H}^{class} \otimes \mathcal{H}^{quant}$. Then $\mathcal{E}$ measures $\mathcal{H}^{class}$ and parses the outcome as $(id'_{sender}, id'_{rcpt}, m')$. If the outcome could not be parsed, or if $id'_{sender} \neq id_{rcpt}$, initialize $\mathcal{H}^{class}$ with $(\varepsilon, \texttt{environment}, \varepsilon)$ and $\mathcal{H}^{quant}$ with ε. (This ensures that the environment is activated if a machine sends no or an ill-formed message.)

The output of the network $\mathbf{N}$ on input z and security parameter k is described by the following algorithm: Let $\rho \in \mathcal{P}(\mathcal{H}_{\mathbf{N}})$ be the state that is initialized to $(\varepsilon, \texttt{environment}, z)$ in $\mathcal{H}^{class}$, and to the empty word ε in all other registers. Then repeat the following indefinitely: Apply $\mathcal{E}_{\mathbf{N}}^{(k)}$ to ρ. Measure $\mathcal{H}^{class}$. If the outcome is of the form $(\texttt{environment}, \varepsilon, out)$, return

[5]A sequence of circuits C_k is uniform if a deterministic Turing machine can output the description of C_k in time polynomial in k.

out and terminate. Otherwise, continue the loop. The probability distribution of the return value *out* is denoted by $\mathrm{Exec}_{\mathbf{N}}(k, z)$.

Corruptions. To model corruptions, we introduce *corruption parties*, special machines that follow the instructions given by the adversary. When invoked, the corruption party P^C_{id} with identity id measures $\mathcal{H}^{class}$ and parses the outcome as $(id_{sender}, id_{rcpt}, m)$. If $id_{sender} = \mathtt{adversary}$, $\mathcal{H}^{class}$ is initialized with m. (In this case, m specifies both the message and the sender/recipient. Thus the adversary can instruct a corruption party to send to arbitrary recipients.) Otherwise, $\mathcal{H}^{class}$ is initialized with $(id, \mathtt{adversary}, (id_{sender}, id_{rcpt}, m))$. (The message is forwarded to the adversary.) Note that, since P^C_{id} does not touch the $\mathcal{H}^{quant}$, the quantum part of the message is forwarded.

Given a network $\mathbf{N}$, and a set of identities C, we write $\mathbf{N}^C$ for the set resulting from replacing each machine $M \in \mathbf{N}$ with identity $id \in C$ by P^C_{id}.

Security model. A protocol π is a set of machines with $\mathtt{environment}, \mathtt{adversary} \notin ids(\pi)$. We assume a set of identities $parties_\pi \subseteq ids(\pi)$ to be associated with π. $parties_\pi$ denotes which of the machines in the protocol are actually protocol parties (as opposed to incorruptible entities such as ideal functionalities).

An environment is a machine with identity `environment`, an adversary or a simulator is a machine with identity `adversary` (there is no formal distinction between adversaries and simulators, the two terms refer to different intended roles of a machine).

In the following we call two networks ε-close if for all $z \in \{0,1\}^*$ and $k \in \mathbb{N}$, $|\Pr[\mathrm{Exec}_N(k, z) = 1] - \Pr[\mathrm{Exec}_M(k, z) = 1]| \leq \varepsilon(k)$. We call two networks negligible-close if they are ε-close for negligible ε. We speak of perfect closeness if $\varepsilon = 0$.

Without assuming a memory bound, this leads to the following definition of quantum-UC-security:[6]

Definition 1 (Quantum-UC-security [Unr10]) *Let protocols π and ρ be given. We say π* quantum-UC-emulates *ρ iff for every set $C \subseteq parties_\pi$ and for every adversary* Adv *there is a simulator* Sim *such that for every environment $\mathcal{Z}$, the networks $\pi^C \cup \{\mathrm{Adv}, \mathcal{Z}\}$ (called the real model) and $\rho^C \cup \{\mathrm{Sim}, \mathcal{Z}\}$ (called the ideal model) are negligible-close. We furthermore require that if* Adv *is quantum-polynomial-time, so is* Sim.

2.2 The BQS-UC model

In order to define BQS-UC, we first need a definition of a memory-bounded machine. We call a machine M a-memory bounded if it keeps at most a qubits of quantum memory *between* activations. (An activation is the computation performed by a machine between receiving a message and sending the immediate response.) We do not impose any limitations on the computation-time or memory-use during a single activation of M: since we allow arbitrary state transition operators, this also includes state transition operators that need exponential time and a large number of auxiliary qubits during a single evaluation of that state transition operator.

We also stress that we do not impose any bound on the classical memory.

Definition 2 (Memory bounded machines) *Let a be a function in the security parameter k. We call a machine M a-*memory bounded *if for every k, we can decompose $\mathcal{H}^{state}$ as $\mathcal{H}^{state,q} \otimes \mathcal{H}^{state,c}$ with $\mathcal{H}^{state,q} := \mathbb{C}^{\{0,1\}^{a(k)}}$ and $\mathcal{H}^{state,c} := \mathbb{C}^{\{0,1\}^*}$ such that*

[6]In [Unr10], this notion is called statistical quantum-UC-security. Since we do not consider computational security in this paper, we omit the qualifier statistical from all definitions.

$\mathcal{E}_M^{(k)} = (\mathcal{E}_{class} \otimes id) \circ \mathcal{E}_M^{(k)}$. We write $\mathrm{QM}(M)$ for the smallest a such that M is a-memory bounded.

We can now formulate BQS-UC-security. Intuitively, a protocol is BQS-UC-secure if it is UC-secure for memory-bounded adversaries. To formulate this, we need to explicitly parametrize the definition over a memory bound a. Then we require that the total quantum memory used by environment and adversary is bounded by a. The reason why we include the environment's memory is that the latter can be involved in the actual attack: If only the adversary's memory was bounded, the adversary could use the environment as an external storage to perform the attack (see also our discussion in Section 2.3).[7]

It remains to decide whether the simulator should be memory bounded. If we allow the simulator to be unbounded, composition becomes difficult: In some cases, the simulator of one protocol plays the role of the adversary of a second protocol. Thus, if simulators where not memory bounded, the second protocol would have to be secure against unbounded adversaries. However, if we require the simulator to be a-memory bounded, we will not be able to construct nontrivial protocols: In order to perform a simulation, the simulator needs to have some advantage over an honest protocol participant (in the computational UC setting, e.g., this is usually the knowledge of some trapdoor). In our setting, the advantage of the simulator will be that he has more quantum memory than the adversary. Thus we introduce a second parameter s which specifies the amount of quantum memory the simulator may use for the simulation. More precisely, we allow the simulator to use $s + \mathrm{QM}(\mathrm{Adv})$ qubits because the simulator will usually internally simulate the adversary Adv as a black-box and therefore have to additionally reserve sufficient quantum memory to store the adversary's state.

Definition 3 (BQS-UC-security) *Fix protocols π and ρ. Let $a, s \in \mathbb{N}_0 \cup \{\infty\}$ (possibly depending on the security parameter). We say π (a,s)-*BQS-UC-emulates[8] *ρ iff for every set $C \subseteq parties_\pi$ and for every adversary* Adv *there is a simulator* Sim *with* $\mathrm{QM}(\mathrm{Sim}) \le s + \mathrm{QM}(\mathrm{Adv})$ *such that for every environment $\mathcal{Z}$ with* $\mathrm{QM}(\mathcal{Z}) + \mathrm{QM}(\mathrm{Adv}) \le a$*, the networks $\pi^C \cup \{\mathrm{Adv}, \mathcal{Z}\}$ (called the real model) and $\rho^C \cup \{\mathrm{Sim}, \mathcal{Z}\}$ (called the ideal model) are negligible-close. We furthermore require that if* Adv *is quantum-polynomial-time, so is* Sim.

We first state a few useful properties of BQS-UC.

Lemma 4 *Let π, ρ, σ be protocols.*

(i) Reflexivity: *π $(\infty, 0)$-BQS-UC-emulates π.*

(ii) Transitivity: *If π (a,s)-BQS-UC emulates ρ and ρ $(a+s, s')$-BQS-UC emulates σ, then π $(a, s+s')$-BQS-UC-emulates σ.*

(iii) Monotonicity: *If $a' \le a$ and $s' \ge s$ and π (a,s)-BQS-UC emulates ρ, then π (a',s')-BQS-UC emulates ρ.*

(iv) Relation to quantum-UC: *π quantum-UC emulates ρ iff there is a polynomial s such that π (∞, s)-BQS-UC emulates ρ.*

Proof. Reflexivity and monotonicity follow directly from Definition 3.

To show transitivity, assume that π (a,s)-BQS-UC emulates ρ and ρ $(a+s, s')$-BQS-UC emulates σ. Fix an adversary Adv and an environment $\mathcal{Z}$ with $\mathrm{QM}(\mathcal{Z}) + \mathrm{QM}(\mathrm{Adv}) \le a$. To

[7]This is captured more formally by the completeness of the so-called dummy-adversary (see Section 2.5), which shows that one can even shift the complete attack into the environment.

[8]Since we only consider statistical security in this work, we omit the qualifier "statistical". Similarly, when we speak about classical-UC-security and quantum-UC-security, we mean the statistical variant of that notion.

show that π $(a, s+s')$-BQS-UC-emulates σ we need to show that there exists a simulator Sim that is independent of $\mathcal{Z}$ and that satisfies $\mathrm{QM}(\mathrm{Sim}) \leq s + s' + \mathrm{QM}(\mathrm{Adv})$.

Since π (a,s)-BQS-UC emulates ρ, there exists a simulator Sim' (independent of $\mathcal{Z}$) with $\mathrm{QM}(\mathrm{Sim}') \leq s + \mathrm{QM}(\mathrm{Adv}) \leq a + s$ such that $\pi \cup \{\mathrm{Adv}, \mathcal{Z}\}$ and $\rho \cup \{\mathrm{Sim}', \mathcal{Z}\}$ are negligible-close. Let $\mathrm{Adv}' := \mathrm{Sim}'$. Since ρ $(a+s, s')$-BQS-UC emulates σ, there exists a simulator Sim (independent of $\mathcal{Z}$) with $\mathrm{QM}(\mathrm{Sim}) \leq s' + \mathrm{QM}(\mathrm{Adv}')$ such that $\rho \cup \{\mathrm{Adv}', \mathcal{Z}\}$ and $\sigma \cup \{\mathrm{Sim}, \mathcal{Z}\}$ are negligible-close. Then $\pi \cup \{\mathrm{Adv}, \mathcal{Z}\}$ and $\sigma \cup \{\mathrm{Sim}, \mathcal{Z}\}$ are negligible-close. And $\mathrm{QM}(\mathrm{Sim}) \leq s' + \mathrm{QM}(\mathrm{Adv}') \leq s + s' + \mathrm{QM}(\mathrm{Adv})$. Thus π $(a, s+s')$-BQS-UC-emulates σ.

The proof of (iv) depends on the concept of the dummy-adversary introduced in the next section; we defer the proof until Section 2.5. (Note that the proofs in the next section do not use (iv).) $\square$

2.3 On the memory bound of the environment

In Definition 3, we impose the memory bound on both the adversary and the environment. In this, we differ from the modeling by Wehner and Wullschleger [WW08]. In their definition, the environment (which is implicit in the definition of the indistinguishability $\equiv_\varepsilon$ of quantum channels) provides the input state to protocol and adversary, then gets the outputs of protocol and adversary, and finally the environment has to guess whether it interacted with the real or the ideal model. During the interaction of the protocol, the environment is not allowed to communicate with any other machine. Between its two activations, the environment is allowed to keep an arbitrarily large quantum state.[9] The interesting point here is that, in contrast to our Definition 3, Wehner and Wullschleger do not impose the memory bound on the environment, only on the adversary. They motivate unlimited environments by pointing out that it is more realistic to assume that a particular memory bound (say, 100 qubits) applies to a particular adversary (e.g., a smart card) than to the whole environment (i.e., all computers world-wide). We believe, however, that this reasoning has to be applied with care: Only when we have the guarantee that the adversary (e.g., the smart card) cannot communicate with any other machines can we assume a smart card is limited to 100 qubits. Otherwise, we have to assume that the smart card effectively has (in the worst case) access to all the quantum memory of the environment. Thus, except in very specific cases, the memory bound we assume needs to be large enough to encompass the environment's memory as a whole. Thus, the bound we assume not to be surpassed by the adversary's memory needs to be large enough that it makes sense to assume that the environment does not surpass this bound either. But in this case, we can safely assume in Definition 3 that the environment is restricted by that memory bound.

We stress that even if our environment is memory bounded, we do take into account the fact that an environment can have a quantum state that is entangled with that of the adversary; we just limit this quantum state to the memory bound.

We get, however, an interesting variant of our model if we follow the approach of Wehner and Wullschleger as follows. We call a machine a-$\diamond$-memory-bounded[10] if its state between activations consists of two registers A and B. The register A contains at most a qubits,

[9] Note that strictly speaking, the formalism of [WW08, full version] does not model an environment with quantum memory: For quantum channels Λ, Λ' they define $\Lambda \equiv_\varepsilon \Lambda'$ iff for all quantum states ρ, the trace distance between $\Lambda(\rho)$ and $\Lambda'(\rho)$ is at most ε. To model environments with quantum memory, we should instead require that for all Hilbert spaces $\mathcal{H}$ and all quantum states ρ, the trace distance between $(\Lambda \otimes id_{\mathcal{H}})(\rho)$ and $(\Lambda' \otimes id_{\mathcal{H}})(\rho)$ is at most ε. We believe that the latter was the intended meaning of $\equiv_\varepsilon$.

[10] We use the symbol $\diamond$ because $\diamond$-memory-bounded environments essentially model indistinguishability with respect to the so-called $\diamond$-norm.

and register B is unlimited but is only accessed in the first and the last activation of B. We denote by $\mathrm{QM}_\diamond(\mathcal{Z})$ the $\diamond$-memory bound of $\mathcal{Z}$. We define (a,s)-$\diamond$-BQS-UC-emulation like (a,s)-BQS-UC-emulation (Definition 3), except that we use $\mathrm{QM}_\diamond(\mathcal{Z})$ instead of $\mathrm{QM}(\mathcal{Z})$. (But we still use $\mathrm{QM}(\mathrm{Adv})$ and $\mathrm{QM}(\mathrm{Sim})$.)

Definition 5 ($\diamond$-BQS-UC-security) *Fix protocols π and ρ. Let $a,s \in \mathbb{N}_0 \cup \{\infty\}$. We say π (a,s)-$\diamond$-BQS-UC-emulates ρ iff for every set $C \subseteq parties_\pi$ and for every adversary Adv there is a simulator Sim with $\mathrm{QM}(\mathrm{Sim}) \leq s + \mathrm{QM}(\mathrm{Adv})$ such that for every environment $\mathcal{Z}$ with $\mathrm{QM}_\diamond(\mathcal{Z}) + \mathrm{QM}(\mathrm{Adv}) \leq a$, the networks $\pi^C \cup \{\mathrm{Adv}, \mathcal{Z}\}$ and $\rho^C \cup \{\mathrm{Sim}, \mathcal{Z}\}$ are negligible-close. We furthermore require that if Adv is quantum-polynomial-time, so is Sim.*

We stress that our techniques also work for this definition. All results of this section still hold with essentially unmodified proofs (except that we always have to refer to the $\diamond$-memory bound of the environment instead memory bound). The results from Section 3 (BQS-UC commitments) are based on the existence of certain commitment schemes that are hiding with respect to memory bounded adversaries. We use the commitment scheme from [KWW09] (see Theorem 20). To extend the results from Section 3 to Definition 5, we need schemes that are hiding with respect to $\diamond$-memory bounded adversaries instead. Besides that, the proofs of the results in Section 3 stay essentially the same (except for using $\diamond$-memory bounds instead of memory bounds).

2.4 Ideal functionalities

In most cases, the behavior of the ideal model is described by a single machine $\mathcal{F}$, the so-called ideal functionality. We can think of this functionality as a trusted third party that perfectly implements the desired protocol behavior. For example, the functionality $\mathcal{F}_{\mathrm{OT}}$ for oblivious transfer would take as input from Alice two bitstrings m_0, m_1, and from Bob a bit c, and send to Bob the bitstring m_c. Obviously, such a functionality constitutes a secure oblivious transfer. We can thus define a protocol π to be a secure OT protocol if π quantum-UC-emulates $\mathcal{F}_{\mathrm{OT}}$ where $\mathcal{F}_{\mathrm{OT}}$ denotes the protocol consisting only of one machine, the functionality $\mathcal{F}_{\mathrm{OT}}$ itself. There is, however, one technical difficulty here. In the real protocol π, the bitstring m_c is sent to the environment $\mathcal{Z}$ by Bob, while in the ideal model, m_c is sent by the functionality. Since every message is tagged with the sender of that message, $\mathcal{Z}$ can distinguish between the real and the ideal model merely by looking at the sender of m_c. To solve this issue, we need to ensure that $\mathcal{F}$ sends the message m_c in the name of Bob (and for analogous reasons, that $\mathcal{F}$ receives messages sent by $\mathcal{Z}$ to Alice or Bob). To achieve this, we use so-called dummy-parties [Can01] in the ideal model. These are parties with the identities of Alice and Bob that just forward messages between the functionality and the environment.

Definition 6 (Dummy-party) *Let a machine P and a functionality $\mathcal{F}$ be given. The dummy-party $\tilde{P}$ for P and $\mathcal{F}$ is a machine that has the same identity as P and has the following state transition operator: Let $id_{\mathcal{F}}$ be the identity of $\mathcal{F}$. When activated, measure $\mathcal{H}^{class}$. If the outcome of the measurement is of the form $(\texttt{environment}, id_P, m)$, initialize $\mathcal{H}^{class}$ with $(id_P, id_{\mathcal{F}}, m)$. If the outcome is of the form $(id_{\mathcal{F}}, id_P, m)$, initialize $\mathcal{H}^{class}$ with $(id_P, \texttt{environment}, m)$. In all cases, the quantum communication register is not modified (i.e., the message in that register is forwarded).*

Note the strong analogy to the corruption parties (page 6).

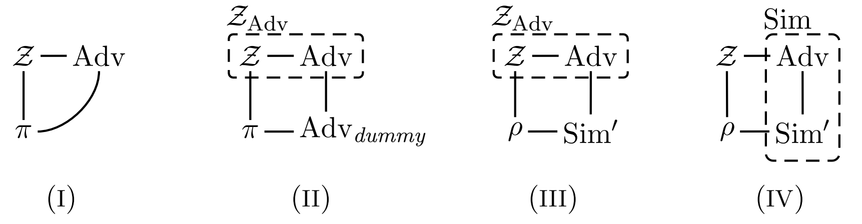

Figure 1: Completeness of the dummy-adversary: proof steps

Thus, if we write π quantum-UC-emulates $\mathcal{F}$, we mean that π quantum-UC-emulates $\rho_{\mathcal{F}}$ where $\rho_{\mathcal{F}}$ consists of the functionality $\mathcal{F}$ and the dummy-parties corresponding to the parties in π. More precisely:

Definition 7 *Let π be a protocol and $\mathcal{F}$ be a functionality. We say that π quantum-UC-emulates $\mathcal{F}$ if π quantum-UC-emulates $\rho_{\mathcal{F}}$ where $\rho_{\mathcal{F}} := \{\tilde{P} : P \in parties_\pi\} \cup \{\mathcal{F}\}$.*

2.5 Dummy-adversary

In the definition of UC-security, we have three entities interacting with the protocol: the adversary, the simulator, and the environment. Both the adversary and the environment are all-quantified, hence we would expect that they do, in some sense, work together. This intuition is backed by the following fact which was first noted by Canetti [Can01]: Without loss of generality, we can assume an adversary that is completely controlled by the environment. This so-called dummy-adversary only forwards messages between the environment and the protocol. The actual attack is then executed by the environment.

Definition 8 (Dummy-adversary Adv$_{dummy}$) *When activated, the dummy-adversary Adv$_{dummy}$ measures $\mathcal{H}^{class}$; call the outcome m. If m is of the form* (`environment`, `adversary`, m')*, initialize $\mathcal{H}^{class}$ with m'. Otherwise initialize $\mathcal{H}^{class}$ with* (`adversary`, `environment`, m)*. In all cases, the quantum communication register is not modified (i.e., the message in that register is forwarded).*

Lemma 9 (Completeness of the dummy-adversary) *Assume that π (a,s)-BQS-UC-emulates ρ with respect to the dummy-adversary (i.e., instead of quantifying over all adversaries* Adv*, we fix* Adv := Adv$_{dummy}$*). Then π (a,s)-UC-emulates ρ.*

Proof of Lemma 9. In [Unr10], an analogous theorem has been shown for quantum-UC-security. Our proof is essentially the same, except that we additionally have to check that the machines we construct satisfy the required memory bounds.

Assume that π (a,s)-BQS-UC-emulates ρ with respect to the dummy-adversary. Fix an adversary Adv. We have to show that there exists a simulator Sim with QM(Sim) $\leq s +$ QM(Adv) such that for all environments $\mathcal{Z}$ with QM($\mathcal{Z}$) + QM(Adv) $\leq a$ we have that $\pi \cup \{\text{Adv}, \mathcal{Z}\}$ and $\rho \cup \{\text{Sim}, \mathcal{Z}\}$ are negligible-close. Furthermore, if Adv is quantum-polynomial-time, Sim has to be quantum-polynomial-time, too.

For a given environment $\mathcal{Z}$ with QM($\mathcal{Z}$) + QM(Adv) $\leq a$, we construct an environment $\mathcal{Z}_{\text{Adv}}$ that is supposed to interact with Adv$_{dummy}$ and internally simulates $\mathcal{Z}$ and Adv, and that routes all messages sent by the simulated Adv to π through Adv$_{dummy}$ and vice versa. Then $\pi \cup \{\text{Adv}, \mathcal{Z}\}$ and $\pi \cup \{\text{Adv}_{dummy}, \mathcal{Z}_{\text{Adv}}\}$ are perfectly close. (Cf. networks (I) and (II) in Figure 1.)

By definition of Adv_{dummy}, we have $\mathrm{QM}(\mathrm{Adv}_{dummy}) = 0$. And by construction of $\mathcal{Z}_{\mathrm{Adv}}$, we have $\mathrm{QM}(\mathcal{Z}_{\mathrm{Adv}}) = \mathrm{QM}(\mathcal{Z}) + \mathrm{QM}(\mathrm{Adv})$. Thus $\mathrm{QM}(\mathcal{Z}) + \mathrm{QM}(\mathrm{Adv}_{dummy}) \leq \mathrm{QM}(\mathcal{Z}) + \mathrm{QM}(\mathrm{Adv}) \leq a$. Since π (a,s)-BQS-UC-emulates ρ with respect to the dummy-adversary, we have that $\pi \cup \{\mathrm{Adv}_{dummy}, \mathcal{Z}_{\mathrm{Adv}}\}$ and $\rho \cup \{\mathrm{Sim}', \mathcal{Z}_{\mathrm{Adv}}\}$ are negligible-close for some Sim' with $\mathrm{QM}(\mathrm{Sim}') \leq s + \mathrm{QM}(\mathrm{Adv}_{dummy})$ and all $\mathcal{Z}$. (Cf. networks (II) and (III).) Since Adv_{dummy} is quantum-polynomial-time, so is Sim'. We construct a machine Sim that internally simulates Sim' and Adv (network (IV)). Then $\rho \cup \{\mathrm{Sim}', \mathcal{Z}_{\mathrm{Adv}}\}$ and $\rho \cup \{\mathrm{Sim}, \mathcal{Z}\}$ are perfectly close. Summarizing, $\pi \cup \{\mathrm{Adv}, \mathcal{Z}\}$ and $\rho \cup \{\mathrm{Sim}, \mathcal{Z}\}$ are negligible-close for all environments $\mathcal{Z}$. Furthermore, since Sim' is quantum-polynomial-time, we have that Sim is quantum-polynomial-time if Adv is. And $\mathrm{QM}(\mathrm{Sim}') = \mathrm{QM}(\mathrm{Sim}) + \mathrm{QM}(\mathrm{Adv}) \leq s + \mathrm{QM}(\mathrm{Adv}_{dummy}) + \mathrm{QM}(\mathrm{Adv}) = s + \mathrm{QM}(\mathrm{Adv})$. Thus π (a,s)-BQS-UC-emulates ρ. □

Using Lemma 9, we can now prove the deferred Lemma 4 (iv).

Proof of Lemma 4 (iv). If the π (∞, s)-BQS-UC-emulates ρ, then π trivially quantum-UC-emulates ρ (the condition $\mathrm{QM}(\mathcal{Z}) + \mathrm{QM}(\mathrm{Adv}) \leq \infty$ is trivially fulfilled).

Assume that π quantum-UC-emulates ρ. Then there is a simulator Sim such that for any environment $\mathcal{Z}$, $\pi \cup \{\mathrm{Adv}_{dummy}, \mathcal{Z}\}$ and $\rho \cup \{\mathrm{Sim}, \mathcal{Z}\}$ are negligible-close. Since Adv_{dummy} is quantum-polynomial-time, so is Sim. Let s be a polynomial upper-bound on the running time of Sim. Then $\mathrm{QM}(\mathrm{Sim}) \leq s$. Thus π (∞, s)-BQS-UC-emulates ρ with respect to the dummy-adversary. By Lemma 9, π (∞, s)-BQS-UC-emulates ρ. □

2.6 Composition

For some protocol σ, and some protocol π, by σ^π we denote the protocol where σ invokes (up to polynomially many) instances of π. That is, in σ^π the machines from σ and from π run together in one network, and the machines from σ access the inputs and outputs of π. (That is, σ plays the role of the environment from the point of view of π. In particular, $\mathcal{Z}$ then talks only to σ and not to the subprotocol π directly.) A typical situation would be that $\sigma^{\mathcal{F}}$ is some protocol that makes use of some ideal functionality $\mathcal{F}$, say a commitment functionality, and then σ^π would be the protocol resulting from implementing that functionality with some protocol π, say a commitment protocol. One would hope that such an implementation results in a secure protocol σ^π. That is, we hope that if π BQS-UC-emulates $\mathcal{F}$ and $\sigma^{\mathcal{F}}$ BQS-UC-emulates $\mathcal{G}$, then σ^π BQS-UC-emulates $\mathcal{G}$. Fortunately, this is the case, as long as we pick the memory bounds in the right way:

Theorem 10 (Composition Theorem) *Let π, ρ, and σ be quantum-polynomial-time protocols. Assume that σ invokes at most n subprotocol instances (n may depend on the security parameter). Assume that π (a,s)-BQS-UC-emulates ρ. Then σ^π $(a - \mathrm{QM}(\sigma) - (n-1)b, ns)$-BQS-UC-emulates σ^ρ where $b := \max\{\mathrm{QM}(\pi), \mathrm{QM}(\rho) + s\}$.*

Proof of Theorem 10. In [Unr10], an analogous theorem has been shown for quantum-UC-security. Our proof is essentially the same, except that we additionally have to check that the machines we construct satisfy the required memory bounds.

Since σ is quantum-polynomial-time, σ invokes at most a polynomial number of instances of its subprotocol π or ρ. We can thus assume that n is polynomially-bounded. Since π (a,s)-BQS-UC-emulates ρ, there is a quantum-polynomial-time simulator Sim' with $\mathrm{QM}(\mathrm{Sim}') \leq s$ such that for all environments $\mathcal{Z}$ with $\mathrm{QM}(\mathcal{Z}) = \mathrm{QM}(\mathcal{Z}) + \mathrm{QM}(\mathrm{Adv}_{dummy}) \leq a$ we have that $\pi \cup \{\mathrm{Adv}_{dummy}, \mathcal{Z}\}$ and $\rho \cup \{\mathrm{Sim}', \mathcal{Z}\}$ are negligible-close. In the following, we call Sim' the dummy-simulator.

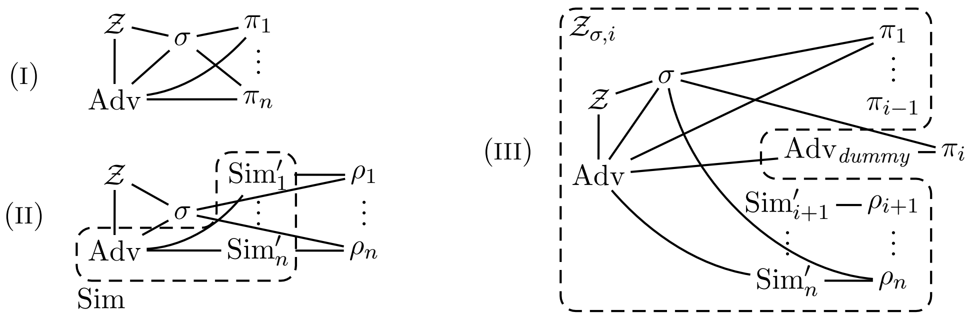

Figure 2: Networks occurring in the proof sketch of Theorem 10. Network (I) represents the real model, (II) the ideal model, and (III) the hybrid case. To avoid cluttering, in (III), the connections to π_{i-1}, Sim'_{i+1}, and ρ_{i+1} have been omitted.

Let a quantum-polynomial-time adversary Adv be given (that is supposed to attack σ^π). We construct a simulator Sim that internally simulates the adversary Adv and n instances $\mathrm{Sim}'_1, \dots, \mathrm{Sim}'_n$ of the dummy-simulator Sim'. The simulated adversary Adv is connected to the environment and to the protocol σ, but all messages between Adv and the i-th instance π_i of π are routed through the dummy-simulator-instance Sim'_i (which is then supposed to transform these messages into a form suitable for instances of ρ). The simulator Sim is depicted by the dashed box in network (II) in Figure 2. We have $\mathrm{QM}(\mathrm{Sim}) \leq ns$.

We have to show that for any environment $\mathcal{Z}$ with $\mathrm{QM}(\mathcal{Z}) + \mathrm{QM}(\mathrm{Adv}) \leq a - \mathrm{QM}(\sigma) - (n-1)b$ we have that $\sigma^\pi \cup \{\mathrm{Adv}, \mathcal{Z}\}$ and $\sigma^\rho \cup \{\mathrm{Sim}, \mathcal{Z}\}$ are negligible-close (networks (I) and (II) in Figure 2).

For this, we construct a hybrid environment $\mathcal{Z}_{\sigma,i}$. ($\mathcal{Z}_{\sigma,i}$ is depicted as the dashed box in network (III) in Figure 2.) This environment internally simulates the machines $\mathcal{Z}$, Adv, the protocol σ, instances $\pi_1, \dots, \pi_{i-1}$ of the real protocol π, and instances $\mathrm{Sim}'_{i+1}, \dots, \mathrm{Sim}'_n$ and $\rho_{i+1}, \dots, \rho_n$ of the dummy-simulator Sim' and the ideal protocol ρ, respectively. The communication between $\mathcal{Z}$, Adv, and σ is directly forwarded by $\mathcal{Z}_{\sigma,i}$. Communication between Adv and the j-th protocol instance is forwarded as follows: If $j < i$, the communication is simply forwarded to π_j. If $j > i$, the communication is routed through the corresponding dummy-simulator Sim'_j (which is then supposed to transform these messages into a form suitable for ρ_j). And finally, if $j = i$, the communication is passed to the adversary/simulator outside of $\mathcal{Z}_{\sigma,i}$. Communication between σ and the instances of π or ρ is directly forwarded.

Since $\mathcal{Z}_{\sigma,i}$ internally simulates one copy of $\mathcal{Z}$, one copy of Adv, one copy of σ, $i-1$ copies of π, and $n-i$ copies of Sim' and ρ. Thus

$$\begin{aligned}\mathrm{QM}(\mathcal{Z}_{\sigma,i}) &= \mathrm{QM}(\mathcal{Z}) + \mathrm{QM}(\mathrm{Adv}) + \mathrm{QM}(\sigma) + (i-1)\mathrm{QM}(\pi) + (n-i)(\mathrm{QM}(\mathrm{Sim}') + \mathrm{QM}(\rho)) \\ &\leq a - \mathrm{QM}(\sigma) - (n-1)b + \mathrm{QM}(\sigma) + (i-1)\mathrm{QM}(\pi) + (n-i)(\mathrm{QM}(\mathrm{Sim}') + \mathrm{QM}(\rho)) \\ &\leq a - \mathrm{QM}(\sigma) - (n-1)b + \mathrm{QM}(\sigma) + (i-1)b + (n-i)b = a.\end{aligned}$$

We will now show that there is a negligible function μ such that $|\Pr[\mathrm{Exec}_{\pi\cup\{\mathrm{Adv}_{dummy},\mathcal{Z}_{\sigma,i}\}}(k,z) = 1] - \Pr[\mathrm{Exec}_{\rho\cup\{\mathrm{Sim}',\mathcal{Z}_{\sigma,i}\}}(k,z) = 1]| \leq \mu(k)$ for any security parameter k and any $i = 1, \dots, n$. For this, we construct an environment $\mathcal{Z}_\sigma$ which expects as its initial input a pair (i, z), and then runs $\mathcal{Z}_{\sigma,i}$ with input z. Since $\mathrm{QM}(\mathcal{Z}_{\sigma,i}) \leq a$, we have $\mathrm{QM}(\mathcal{Z}_\sigma) \leq a$. Since $\pi \cup \{\mathrm{Adv}_{dummy}, \mathcal{Z}\}$ and $\rho \cup \{\mathrm{Sim}', \mathcal{Z}\}$ are negligible-close for all quantum-polynomial-time environments $\mathcal{Z}$ with $\mathrm{QM}(\mathcal{Z}) \leq a$, $\pi \cup \{\mathrm{Adv}_{dummy}, \mathcal{Z}_\sigma\}$ and $\rho \cup \{\mathrm{Sim}', \mathcal{Z}_\sigma\}$ are negligible-close. Hence there exists a negligible function μ such that the difference of $\Pr[\mathrm{Exec}_{\pi\cup\{\mathrm{Adv}_{dummy},\mathcal{Z}_{\sigma,i}\}}(k,z)=1]=\Pr[\mathrm{Exec}_{\pi\cup\{\mathrm{Adv}_{dummy},\mathcal{Z}_{\sigma}\}}(k,(i,z))=1]$ and $\Pr[\mathrm{Exec}_{\rho\cup\{\mathrm{Sim}',\mathcal{Z}_{\sigma,i}\}}(k,z)=1]=\Pr[\mathrm{Exec}_{\rho\cup\{\mathrm{Sim}',\mathcal{Z}_{\sigma}\}}(k,(i,z))=1]$ is bounded by $\mu(k)$ for all i,k,z.

The game $\mathrm{Exec}_{\pi\cup\{\mathrm{Adv}_{dummy},\mathcal{Z}_{\sigma,i}\}}(k,z)$ is depicted as network (III) in Figure 2 (except that we wrote π_i instead of π). Observe that $\mathrm{Exec}_{\rho\cup\{\mathrm{Sim}',\mathcal{Z}_{\sigma,i+1}\}}(k,z)$ (note the changed index $i+1$) contains the same machines as $\mathrm{Exec}_{\pi\cup\{\mathrm{Adv}_{dummy},\mathcal{Z}_{\sigma,i}\}}(k,z)$ (when unfolding the simulation performed by $\mathcal{Z}_{\sigma,i}$ into individual machines) except for the difference that the communication with the i-th instance of π is routed through the dummy-adversary Adv_{dummy}. However, the latter just forwards messages, so $\pi\cup\{\mathrm{Adv}_{dummy},\mathcal{Z}_{\sigma,i}\}$ and $\rho\cup\{\mathrm{Sim}',\mathcal{Z}_{\sigma,i+1}\}$ are perfectly close.

Using the triangle inequality, it follows that $|\Pr[\mathrm{Exec}_{\pi\cup\{\mathrm{Adv}_{dummy},\mathcal{Z}_{\sigma,n}\}}(k,z)=1]-\Pr[\mathrm{Exec}_{\rho\cup\{\mathrm{Sim}',\mathcal{Z}_{\sigma,1}\}}(k,z)=1]|$ is bounded by $n\cdot\mu(k)$ which is negligible. Moreover, $\mathrm{Exec}_{\pi\cup\{\mathrm{Adv}_{dummy},\mathcal{Z}_{\sigma,n}\}}(k,z)$ and $\mathrm{Exec}_{\sigma^\pi\cup\{\mathrm{Adv},\mathcal{Z}\}}(k,z)$ describe the same game (up to unfolding of simulated submachines and up to one instance of the dummy-adversary). Similarly, $\mathrm{Exec}_{\rho\cup\{\mathrm{Sim}',\mathcal{Z}_{\sigma,1}\}}(k,z)$ and $\mathrm{Exec}_{\sigma^\rho\cup\{\mathrm{Sim},\mathcal{Z}\}}(k,z)$ describe the same game (up to unfolding of simulated submachines). Thus $\big|\Pr[\mathrm{Exec}_{\sigma^\pi\cup\{\mathrm{Adv},\mathcal{Z}\}}(k,z)=1]-\Pr[\mathrm{Exec}_{\sigma^\rho\cup\{\mathrm{Sim},\mathcal{Z}\}}(k,z)=1]\big|$ is negligible and thus $\sigma^\pi\cup\{\mathrm{Adv},\mathcal{Z}\}$ and $\sigma^\rho\cup\{\mathrm{Sim},\mathcal{Z}\}$ are negligible-close. Furthermore, since Sim' is quantum-polynomial-time, we have that Sim is quantum-polynomial-time if Adv is. As mentioned above, $\mathrm{QM}(\mathrm{Sim})\le ns$.

Since this holds for all $\mathcal{Z}$ with $\mathrm{QM}(\mathcal{Z})+\mathrm{QM}(\mathrm{Adv})\le a-\mathrm{QM}(\sigma)-(n-1)b$, and the construction of Sim does not depend on $\mathcal{Z}$, we have that σ^π $(a-\mathrm{QM}(\sigma)-(n-1)b,ns)$-BQS-UC-emulates σ^ρ. □

Theorem 11 (Composition Theorem) *Let π and σ be quantum-polynomial-time protocols and $\mathcal{F}$ and $\mathcal{G}$ be quantum-polynomial-time functionalities. Assume that σ invokes at most one subprotocol instance. Assume that π (a,s)-BQS-UC-emulates $\mathcal{F}$ and that $\sigma^{\mathcal{F}}$ $(a-\mathrm{QM}(\sigma)+s,s')$-BQS-UC-emulates $\mathcal{G}$. Then σ^π $(a-\mathrm{QM}(\sigma),s+s')$-BQS-UC-emulates $\mathcal{G}$.*

The proof of this theorem is very similar to that in [Unr10], except that we have to keep track of the quantum memory used by various machines constructed in the proof.
Proof. From Theorem 10, with $n=1$, we get that σ^π $(a-\mathrm{QM}(\sigma),s)$-BQS-UC-emulates $\sigma^{\mathcal{F}}$. By assumption, $\sigma^{\mathcal{F}}$ $(a-\mathrm{QM}(\sigma)+s,s')$-BQS-UC-emulates $\mathcal{G}$. By transitivity (Lemma 4 (ii)), we get that σ^π $(a-\mathrm{QM}(\sigma),s+s')$-BQS-UC-emulates $\mathcal{G}$. □

Notice that in the composition theorem (Theorem 11), the outer protocol σ is only allowed to invoke one instance of the subprotocol π. This stands in contrast to the universal composition theorem for classical-UC [Can01] and for quantum-UC [Unr10] where any polynomially-bounded number of concurrent instances of π is allowed. In fact, this is not just a limitation of our proof technique.[11] For example, assume a protocol $\pi^{A\to B}_{\mathrm{COM}}$ that (a,s)-BQS-UC-emulates the commitment functionality $\mathcal{F}^{A\to B}_{\mathrm{COM}}$ with sender A and recipient B. Assume further that $\pi^{A\to B}_{\mathrm{COM}}$ does not use any functionalities as setup. As we will see later, such a protocol exists. Now let $\pi^{B\to A}_{\mathrm{COM}}$ be the protocol that results from exchanging the roles of A and B. Then $\pi^{B\to A}_{\mathrm{COM}}$ (a,s)-BQS-UC-emulates $\mathcal{F}^{B\to A}_{\mathrm{COM}}$. Consider the concurrent composition of $\pi^{A\to B}_{\mathrm{COM}}$ and $\pi^{B\to A}_{\mathrm{COM}}$. In this protocol, a corrupted Bob may reroute all messages between the Alice in the first protocol and Alice in the second protocol. Thus, if Alice commits to a

[11]In the proof, the difficulty arises from a hybrid argument where the protocol π is executed together in one network with the protocol ρ and the corresponding simulator. Since the simulator may use more quantum memory than π is resistant against, we cannot guarantee security of π in this hybrid setting and the proof cannot proceed.

random value v in the first protocol, Bob commits to the same value v in the second protocol without knowing it. It is easy to see that in a concurrent composition of $\mathcal{F}_{\text{COM}}^{A\to B}$ and $\mathcal{F}_{\text{COM}}^{B\to A}$, this is not possible. Thus the composition of $\pi_{\text{COM}}^{A\to B}$ and $\pi_{\text{COM}}^{B\to A}$ does not (a', s')-BQS-UC-emulate the composition of $\mathcal{F}_{\text{COM}}^{A\to B}$ and $\mathcal{F}_{\text{COM}}^{B\to A}$ (for any parameters a', s'). To convert this into an example of a protocol that does not even compose with itself, just consider the protocol $\pi_{\text{COM}}^{A\leftrightarrow B}$ in which Bob may choose whether $\mathcal{F}_{\text{COM}}^{A\to B}$ or $\mathcal{F}_{\text{COM}}^{B\to A}$ should be executed. It might be possible to make $\pi_{\text{COM}}^{A\leftrightarrow B}$ self-composable by adding suitable tags inside the messages, but the definition of BQS-UC-security does not enforce this.

Although BQS-UC-security does not guarantee for concurrent self-composability, individual protocols may have this property. In order to formulate this, we introduce the concept of the multi-session variant of a protocol. Given a protocol π and a polynomially-bounded n, we define π^n to be the protocol that executes n instances of π concurrently.

Then, from Theorem 11, we immediately get the following corollary:

Corollary 12 *Let π and σ be quantum-polynomial-time protocols and $\mathcal{F}$ and $\mathcal{G}$ be quantum-polynomial-time functionalities. Let $n, m \geq 0$ be integers (depending on the security parameter). Assume that σ invokes at most m subprotocol instances. Assume that π^{nm} (a, s)-BQS-UC-emulates $\mathcal{F}^{nm}$ and that $(\sigma^{\mathcal{F}})^n$ $(a - n\text{QM}(\sigma) + s, s')$-BQS-UC-emulates $\mathcal{G}^n$. Then $(\sigma^\pi)^n$ $(a - n\text{QM}(\sigma), s + s')$-BQS-UC-emulates $\mathcal{G}^n$.*

2.7 Quantum lifting

In [Unr10], it has been shown that for classical protocols, classical-UC-security implies quantum-UC-security (quantum lifting theorem). This theorem is very useful when reusing results from the classical-UC setting: For example, if a classical protocol $\sigma^{\mathcal{F}}$ classical-UC-emulates a functionality $\mathcal{G}$, and a quantum protocol π quantum-UC-emulates $\mathcal{F}$, then the quantum lifting theorem gives us that $\sigma^{\mathcal{F}}$ quantum-UC-emulates $\mathcal{G}$, and then by the composition theorem σ^π quantum-UC-emulates $\mathcal{G}$.

A similar result also holds for BQS-UC:

Theorem 13 (Quantum lifting) *If π and ρ are classical protocols and π classical-UC-emulates ρ, then π $(\infty, 0)$-BQS-UC emulates ρ.*

Proof. In [Unr10], an analogous theorem has been shown for quantum-UC-security. Our proof is essentially the same, except that we additionally have to check that the simulator Sim we construct has QM(Sim) = 0. However, since Sim is a classical machine, this is trivial.

Given a machine M, let $\mathcal{C}(M)$ denote the machine which behaves like M, but measures incoming messages in the computational basis before processing them, and measures outgoing messages in the computational basis. More precisely, the superoperator $\mathcal{E}_{\mathcal{C}(M)}^{(k)}$ first invokes $\mathcal{E}_{class}$ on $\mathcal{H}^{class} \otimes \mathcal{H}^{quant}$, then invokes $\mathcal{E}_M^{(k)}$ on $\mathcal{H}^{state} \otimes \mathcal{H}^{class} \otimes \mathcal{H}^{quant}$, and then again invokes $\mathcal{E}_{class}$ on $\mathcal{H}^{class} \otimes \mathcal{H}^{quant}$. Since it is possible to simulate quantum Turing machines on classical Turing machines (with an exponential overhead), for every machine M, there exists a classical machine M' such that $\mathcal{C}(M)$ and M' are perfectly indistinguishable.[12]

We define the classical dummy-adversary $\text{Adv}_{dummy}^{class}$ to be the classical machine that is defined like Adv_{dummy} (Definition 8), except that in each invocation, it first measures $\mathcal{H}^{class}$, $\mathcal{H}^{quant}$, and $\mathcal{H}^{state}$ in the computational basis (i.e., it applies $\mathcal{E}_{class}$ to $\mathcal{H}^{state} \otimes \mathcal{H}^{class} \otimes \mathcal{H}^{quant}$) and then proceeds as does Adv_{dummy}. Note that $\text{Adv}_{dummy}^{class}$ is probabilistic-polynomial-time.

[12]More precisely, for any set of machines N, the networks $N \cup \{M\}$ and $N \cup \{\mathcal{C}(M)\}$ are perfectly close.

By Lemma 9, we only need to show that for any set C of corrupted parties, there exists a quantum-polynomial-time machine Sim with $\mathrm{QM}(\mathrm{Sim}) \leq 0 + \mathrm{QM}(\mathrm{Adv}_{\mathit{dummy}}) = 0$ such that for every machine $\mathcal{Z}$, the real model $\pi^C \cup \{\mathcal{Z}, \mathrm{Adv}_{\mathit{dummy}}\}$ and the ideal model $\rho^C \cup \{\mathcal{Z}, \mathrm{Sim}\}$ are negligible-close.

The protocol π is classical, thus π^C is classical, too, and thus all messages forwarded by $\mathrm{Adv}_{\mathit{dummy}}$ from π^C to $\mathcal{Z}$ have been measured in the computational basis by π^C, and all messages forwarded by $\mathrm{Adv}_{\mathit{dummy}}$ from $\mathcal{Z}$ to π^C will be measured by π^C before being used. Thus, if Adv would additionally measure all messages it forwards in the computational basis, the view of $\mathcal{Z}$ would not be modified. More formally, $\pi^C \cup \{\mathcal{Z}, \mathrm{Adv}_{\mathit{dummy}}\}$ and $\pi^C \cup \{\mathcal{Z}, \mathrm{Adv}^{\mathit{class}}_{\mathit{dummy}}\}$ are perfectly close. Furthermore, since both π^C and $\mathrm{Adv}^{\mathit{class}}_{\mathit{dummy}}$ measure all messages upon sending and receiving, $\pi^C \cup \{\mathcal{Z}, \mathrm{Adv}^{\mathit{class}}_{\mathit{dummy}}\}$ and $\pi^C \cup \{\mathcal{C}(\mathcal{Z}), \mathrm{Adv}^{\mathit{class}}_{\mathit{dummy}}\}$ are perfectly close. Since it is possible to simulate quantum machines on classical machines (with an exponential overhead), there exists a classical machine $\mathcal{Z}'$ that is perfectly indistinguishable from $\mathcal{C}(\mathcal{Z})$. Then $\pi^C \cup \{\mathcal{C}(\mathcal{Z}), \mathrm{Adv}^{\mathit{class}}_{\mathit{dummy}}\}$ and $\pi^C \cup \{\mathcal{Z}', \mathrm{Adv}^{\mathit{class}}_{\mathit{dummy}}\}$ are perfectly close. Since $\mathrm{Adv}^{\mathit{class}}_{\mathit{dummy}}$ and $\mathcal{Z}'$ are classical and $\mathrm{Adv}^{\mathit{class}}_{\mathit{dummy}}$ is polynomial-time, there exists a classical probabilistic-polynomial-time simulator Sim (whose construction is independent of $\mathcal{Z}'$) such that $\pi^C \cup \{\mathcal{Z}', \mathrm{Adv}^{\mathit{class}}_{\mathit{dummy}}\}$ and $\rho^C \cup \{\mathcal{Z}', \mathrm{Sim}\}$ are negligible-close.

Then $\rho^C \cup \{\mathcal{Z}', \mathrm{Sim}\}$ and $\rho^C \cup \{\mathcal{C}(\mathcal{Z}), \mathrm{Sim}\}$ are perfectly close by construction of $\mathcal{Z}'$. And since both ρ^C and Sim measure all messages they send and receive, $\rho^C \cup \{\mathcal{C}(\mathcal{Z}), \mathrm{Sim}\}$ and $\rho^C \cup \{\mathcal{Z}, \mathrm{Sim}\}$ are perfectly close.

Summarizing, we have that $\pi^C \cup \{\mathcal{Z}, \mathrm{Adv}_{\mathit{dummy}}\}$ and $\rho^C \cup \{\mathcal{Z}, \mathrm{Sim}\}$ are negligible-close for all environments $\mathcal{Z}$. Furthermore, Sim is classical probabilistic-polynomial-time and hence quantum-polynomial-time and its construction does not depend on the choice of $\mathcal{Z}$. And since Sim is classical, $\mathrm{QM}(\mathrm{Sim}) = 0$. Thus π $(\infty, 0)$-BQS-UC-emulates ρ. $\square$

3 Commitments

3.1 Extractable commitments

In this section, we present the notion of online-extractable commitments in the BQS model. These will be used as a building block for constructing BQS-UC commitments in the next section.

Definition 14 ((ε, a)-BQS-hiding) *Given a commitment protocol π with sender Alice and recipient Bob, and an adversary B' corrupting Bob, we denote with $\langle A(m), B'\rangle_{B'}$ the output of B' in an interaction between Alice and B' where Alice commits to m.*

We call π (ε, a)-BQS-hiding iff for all a-memory bounded B' and all $m_1, m_2 \in M$, we have that $\left|\Pr[\langle A(m_1), B'\rangle_{B'} = 1] - \Pr[\langle A(m_2), B'\rangle_{B'} = 1]\right| \leq \varepsilon$. Here M is the message space of the commitment scheme.

Instead of the binding property, we will need a stronger property: online-extractability. This property guarantees that there is a machine (the *extractor*) that, when running as the recipient of the commit protocol, is able to output the committed value V already after the commit phase. This extractor should be indistinguishable from an honest recipient. Note that this does not contradict (ε, a)-BQS-hiding since we allow the extractor's quantum memory to contain more than a qubits. For our purposes, we will only need a definition of online-extractability that does not impose a memory bound on the adversary. We do, however, make the memory bound s of the extractor explicit.

Parameters: Integer n (the length of the transmitted string). The parameters may depend on the security parameter k.
Parties: The sender Alice A and the recipient Bob B.
Inputs: None.
Protocol:

1. Alice picks a random $x \in \{0,1\}^n$ and $\theta \in \{+,\times\}^n$. Then she encodes and sends each bit x_i in the basis θ_i, i.e., she sends $|\Psi\rangle := |x\rangle_\theta$ to Bob.
2. Bob picks a random $\tilde\theta \in \{0,1\}^n$ and measures the i-th qubit of $|\Psi\rangle$ in basis θ_i. Call the outcomes $\tilde x_i$.
3. Both parties wait until the quantum memory bound becomes effective.[13]
4. Alice sends θ to Bob and outputs x.
5. Bob computes $\mathcal{I} := \{i : \theta_i = \tilde\theta_i\}$ and outputs $(\mathcal{I}, \tilde x_{\mathcal{I}})$.

Figure 3: Weak string erasure protocol π_{WSE} from [KWW09].

Definition 15 ((ε, s)-online-extractable) *Given a commitment protocol π with sender Alice and recipient Bob, an* extractor *is a machine B_S that, after the commit phase, gives an output V' and then executes the (honest) code of Bob for the open phase and outputs a value V (the accepted value). (In particular, B_S needs to provide an initial state for the program of the open phase of Bob that matches the interaction so far.) We write $V = \bot$ if the open phase fails.*

For an adversary A', we denote with $\langle A', B\rangle_{A'}$ ($\langle A', B_S\rangle_{A'}$) the output of A' in an interaction between A' and Bob (B_S) where A' is given V after Bob (B_S) terminates.

We call π (ε, s)-online-extractable *iff there exists an s-memory bounded quantum-polynomial-time extractor B_S such that for all adversaries A', we have that* $\big|\Pr[\langle A', B\rangle_{A'} = 1] - \Pr[\langle A', B_S\rangle_{A'} = 1]\big| \le \varepsilon$ *and in an interaction of A' and B_S, we have* $\Pr[V \notin \{V', \bot\}] \le \varepsilon$.

Constructing online-extractable commitments. König, Wehner, and Wullschleger [KWW09] present a commitment scheme in a generalization of the bounded quantum storage model. Although they only show that it is hiding and binding, their scheme turns out to be also online-extractable. Their construction proceeds in two steps. First, they give a protocol for what they call a *weak string erasure* (WSE). A WSE is a protocol between Alice and Bob after which Alice outputs a bitstring X and Bob outputs a set of indices $\mathcal{I}$ and the bitstring $X_{\mathcal{I}}$ consisting of X restricted to the indices $\mathcal{I}$. The properties guaranteed by a weak string erasure are – informally – that Bob does not learn too much about $X_{\mathcal{I}^{\mathtt{C}}}$, and that Alice has no information about $\mathcal{I}$. From a WSE, we can construct a commitment scheme as follows: If Alice wishes to commit to a string m, Alice and Bob perform a WSE. Then Alice sends $m \oplus F(X)$ to Bob where F is a universal hash function with suitable parameters. Furthermore, Alice sends $\mathbf{S}(X)$, the syndrome of X under a suitable linear code. Since Bob only learns $X_{\mathcal{I}}$, $\mathbf{S}(X)$, and little about $\mathsf{X}_{\mathcal{I}^{\mathtt{C}}}$, the string X has high min-entropy from Bob's point of view. Thus $F(X)$ looks random to Bob and the commitment is hiding. To open the commitment, Alice sends X. Since she does not know $\mathcal{I}$, Alice will be detected if she sends an $\hat X$ that is substantially different from X. If Alice sends an $\hat X$ that differs from X only at a few indices, the syndromes $\mathbf{S}(X)$ and $\mathbf{S}(\hat X)$ will not match. Thus Alice is forced to send $\hat X = X$; the commitment is binding. The precise protocols are given in Figures 3 and 4.

[13]Formally, in our model this step does not exist because our model assumes that the quantum memory bound is effective between any two activations.

Parameters: Integers ℓ (the length of the committed value), $n < \ell$, and $\kappa, d \le n$. A binary (n,κ,d)-linear code where $\mathbf{S}(\omega) \in \{0,1\}^{n-\kappa}$ denotes the syndrome of a codeword $\omega \in \{0,1\}^n$. A family $\mathbf{F}$ of universal hash functions $F : \{0,1\}^n \to \{0,1\}^\ell$. All parameters may depend on the security parameter k.
Subprotocols: The n-bit WSE protocol from Figure 3.
Parties: The sender Alice A and the recipient Bob B.
Inputs: In the commit phase, Alice gets v with $v \in \{0,1\}^\ell$. Bob gets no inputs.
Commit phase:
C1. Alice and Bob execute the WSE protocol. Alice gets x, Bob gets $\mathcal{I}$ and $x_\mathcal{I}$.
C2. Alice picks a hash function $F \leftarrow \mathbf{F}$, computes $p := v \oplus F(x)$, computes the syndrome $\sigma := \mathbf{S}(x)$, and sends (F, σ, p) to Bob.
Open phase:
O1. Alice sends $\hat{x} := x$ to Bob.
O2. Bob checks whether $\sigma = \mathbf{S}(\hat{x})$ and $\hat{x}_\mathcal{I} = x_\mathcal{I}$. Otherwise, Bob aborts.
O3. Bob computes $v := p \oplus F(\hat{x})$ and outputs v. (I.e., Bob accepts the opened value v.)

Figure 4: Commitment scheme π_{KWW} from [KWW09].

The hiding property of π_{KWW} has already been proven in [KWW09], we only need to specialize their result to the BQS model.

Lemma 16 *Fix constants $\delta, \nu > 0$ with $\delta + \nu < \frac{1}{2}$. Fix some integer a (the adversary's memory bound; dependent on the security parameter k). Assume that $a \le \nu n$ and $n \ge k$. Assume that $\ell \le \frac{1}{2}n - \delta n - \nu n - (n - \kappa) - k$. ($n, \kappa, \ell$ refer to the parameters in Figure 4.) Then there is an exponentially-small $\varepsilon > 0$ such that π_{KWW} is (ε, a)-BQS-hiding.*

Proof. In [KWW09], the notion of an $(n, \lambda, \varepsilon')$-WSE protocol is introduced. Here n is the length of the transmitted string, and λn lower bounds the knowledge that Bob has about the string (in terms of min-entropy, and up to an error probability ε'). The precise definition is given in [KWW09]; for the present proof we will not need any details. Their definition is parametrized by an operator $\mathcal{F}$ that describes the evolution of the quantum state in Bob's memory. To model the special case of the BQS model in which the adversary's memory has a qubits, we have to choose $\mathcal{F}$ to be the identity on a-qubit states. Let $P^{\mathcal{F}}_{succ}(n') := \max_{(D_x)_x, (\rho_x)_x} \frac{1}{2^{n'}} \sum_{x \in \{0,1\}^{n'}} \operatorname{tr}\bigl(D_x \mathcal{F}(\rho_x)\bigr)$ where $(D_x)_x$ goes over all POVMs (measurements), and $(\rho_x)_x$ over all families of a-qubit mixed states. Intuitively, $P^{\mathcal{F}}_{succ}(n')$ denotes the maximum probability of correctly decoding a random n'-bit string after it has been stored in an a qubit state. In [KW09, page 1] it is shown that $P^{\mathcal{F}}_{succ}(n') \le 2^{-n'(R-1)} = 2^{-a+n'}$ for $R := \frac{a}{n'}$. By [KWW09, Theorem 3.2], for any constant $\delta \in (0, \frac{1}{2})$, we have that π_{WSE} is an $(n, \lambda, \varepsilon')$-WSE protocol where $\lambda = -\frac{1}{n} \log P^{\mathcal{F}}_{succ}\bigl((\frac{1}{2} - \delta)n\bigr)$ and $\varepsilon' = 2e^{-\frac{n\delta^2}{512(4+\log\frac{1}{\delta})^2}}$. We have that $\lambda \ge -\frac{1}{n} \log 2^{-a+(\frac{1}{2}-\delta)n} = (\frac{1}{2} - \delta) - \frac{a}{n} \ge \frac{1}{2} - \delta - \nu$. Let $\varepsilon'' := \max\{2^{-k/2}, \varepsilon'\}$. Since π_{WSE} is an $(n, \lambda, \varepsilon')$-WSE protocol it is also an $(n, \lambda, \varepsilon'')$-WSE protocol. Furthermore, $\ell \le \lambda n - (n - \kappa) - 2 \log 1/\varepsilon''$. Then, by [KWW09, Lemma 4.3],[14] if π_{WSE} is an $(n, \lambda, \varepsilon'')$-WSE protocol, then π_{KWW} is (ε, a)-BQS-hiding with $\varepsilon := 3\varepsilon''$.[15] Since δ is constant and $n \ge k$,

[14]Strictly speaking, [KWW09, Lemma 4.3] analyzes a protocol that commits to a random value (and in particular does not send the value p in the commit phase). It is, however, straightforward to see that their proof also applies to π_{KWW} as presented in Figure 4.

[15]The definition of hiding is formulated slightly differently in [KWW09] but can easily be seen to imply our definition of BQS-hiding.

we have that ε' and hence also $\varepsilon = 3\max\{2^{-k/2}, \varepsilon'\}$ is exponentially-small. $\square$

We proceed to show that π_{KWW} is online-extractable. Since König et. only showed the binding property, we need to extend the proof of [KWW09]. The main ideas are, however, already present in [KWW09]. Intuitively, online-extractability of π_{KWW} is not surprising: If Bob has an unbounded amount of quantum memory, he can delay the measurement of the qubits in π_{WSE} and thus get all the bits x_i. That is, π_{WSE} is online-extractable. Then, in π_{KWW}, Bob will know x already in the commit phase and can thus compute $v = p \oplus F(x)$ (after error-correcting x to match the syndrome σ). Since v is the committed value, this implies that π_{KWW} is online-extractable. To make this argument formal, we need to ensure that even if Alice cheats, she is not able to send some $\hat{x}$ in the open phase which differs from the value x that Bob extracted. The proof of this fact is similar to the proof of the binding property in [KWW09].

We begin by making the fact formal that Bob (given sufficient quantum memory) can extract in π_{WSE}.

Definition 17 (Online-extractable WSE) *A n-bit WSE protocol π consisting of sender A^{WSE} and recipient B^{WSE} is* s-online-extractable *if there exists an s-memory bounded quantum-polynomial-time machine B_S^{WSE} with the following properties:*

- *B_S^{WSE} outputs an n-bit string x.*
- *For any adversary A', let $\rho_{A'B}$ denote the state of A' after interacting with B^{WSE}. Here A' gets the outputs $x_{\mathcal{I}}$ and $\mathcal{I}$ made by B^{WSE}. Let $\sigma_{A'\mathcal{I}\hat{X}_{\mathcal{I}}}$ denote the state of A' after interacting with B_S^{WSE}. Here A' gets $x_{\mathcal{I}}$ and $\mathcal{I}$ where x is the output of B_S^{WSE} and $\mathcal{I}$ is a uniformly random subset of $\{1, \dots, n\}$.[16] Then $\rho_{A'B} = \sigma_{A'\mathcal{I}\hat{X}_{\mathcal{I}}}$.*

Note that in this definition, we do not require the extractor B_S^{WSE} to produce the set $\mathcal{I}$. Instead, in an execution of A' and B_S^{WSE}, we choose $\mathcal{I}$ randomly. Alternatively, we could also let B_S^{WSE} pick $\mathcal{I}$ and require that $\mathcal{I}$ is uniformly random and independent of x and the state of A'; this would lead to an equivalent definition.

The definition could be generalized by allowing an error ε; we would then require that the trace distance ε between $\rho_{A'B}$ and $\sigma_{A'\mathcal{I}\hat{X}_{\mathcal{I}}}$ is bounded by ε. For our purposes, however, the present definition is sufficient.

Lemma 18 (Online-extractability of π_{WSE}) *π_{WSE} is n-online-extractable. (Here n is as in Figure 3.)*

[KWW09, Theorem 3.5] states a weaker property (called "security for Bob"). The proof of that theorem, however, explicitly constructs a simulator and proves the properties required by Definition 17. Thus, their proof also constitutes a proof for our Lemma 18.

Given an online-extractable WSE protocol, we get an online-extractable commitment scheme:

Lemma 19 *Assume that the code with syndrome $\mathbf{S}$ has efficient error-correction. If π_{WSE} is an s-online-extractable WSE protocol for some s, then π_{KWW} is an $(s, 2^{-d/2})$-online-extractable commitment scheme. Here $d, \mathbf{S}$ are the parameters from Figure 4. Notice that we do not require that π_{WSE} is the protocol from Figure 3.*

Proof. To show that π_{KWW} is online-extractable, we first construct the extractor B_S. B_S follows the program of Bob (Figure 4), except that in Step C1, Bob runs the extractor

[16]The names of the states $\rho_{A'B}$ and $\sigma_{A'\mathcal{I}\hat{X}_{\mathcal{I}}}$ are chosen to better match the notation in [KWW09].

B_S^{WSE} from Definition 17 (which yields an n-bit string x), chooses a random subset $\mathcal{I}$ of $\{1, \dots, n\}$, and computes $x_\mathcal{I}$. In all other protocol steps, B_S behaves like Bob in Figure 4. After the commit phase, B_S computes an x^* with Hamming distance $\omega(x, x^*) \leq (d-1)/2$ and $\mathbf{S}(x^*) = \sigma$. (This can be done in polynomial-time since the code with syndrome $\mathbf{S}$ has efficient error-correction.) Then B_S computes $v' := p \oplus F(x^*)$ and outputs v'. (If no x^* with $\omega(x, x^*) \leq (d-1)/2$ exists, then B_S sets $v' := \perp$ instead.)

Since B_S^{WSE} satisfies Definition 17, we have that $\Pr[\langle A', B\rangle_{A'} = 1] = \Pr[\langle A', B_S\rangle_{A'} = 1]$. To show that π_{KWW} is $(s, 2^{-d/2})$-online-extractable, we thus are left to show that $P_{\mathit{fail}} := \Pr[v \notin \{v', \perp\}] \leq 2^{-d/2}$.

From the construction of π_{KWW}, we have that P_{fail} is upper bounded by the probability that $\hat{x} \neq x^*$ and $\mathbf{S}(\hat{x}) = \sigma$ and $\hat{x}_\mathcal{I} = x_\mathcal{I}$. Fix some $\hat{x}, x^*$ with $\hat{x} \neq x^*$ and $\mathbf{S}(\hat{x}) = \sigma$. From $\mathbf{S}(\hat{x}) = \sigma = \mathbf{S}(x^*)$ we have $\mathbf{S}(\hat{x} - x^*) = 0$. Hence $\hat{x} - x^*$ has Hamming-weight at least d. Thus $\omega(\hat{x}, x^*) \geq d$. Since $\omega(x, x^*) \leq (d-1)/2$, it follows that $\omega(x, \hat{x}) \geq d - (d-1)/2 \geq d/2$. (In the case where $v' = \perp$, we have $\omega(x, x^*) > (d-1)/2$ for all x^* and hence $\omega(x, \hat{x}) > (d-1)/2$ and hence $\omega(x, \hat{x}) \geq d/2$.) Since $\mathcal{I}$ is chosen independently of $x, \hat{x}$, and each i is in $\mathcal{I}$ with probability $\frac{1}{2}$, it follows that $x_\mathcal{I} = \hat{x}_\mathcal{I}$ with probability at most $(\frac{1}{2})^{\omega(x,\hat{x})} \leq 2^{-d/2}$. Thus $P_{\mathit{fail}} \leq 2^{-d/2}$. □

Combining our results on π_{WSE} and π_{KWW}, we get the following theorem:

Theorem 20 (Online-extractable commitments) *For any polynomially-bounded integers a and ℓ, there is a constant-round 0-memory bounded (ε, a)-BQS-hiding (ε, s)-online-extractable commitment scheme π for some exponentially-small ε and some polynomially-bounded s. The message space of π is $M = \{0,1\}^\ell$.*

Proof. Reed-Solomon codes [RS60] are $(2^b - 1, 2^b - d, d)$-linear codes over $\mathrm{GF}(2^b)$ (for any b, d). By representing each symbol as a b-bit string, they can be converted into binary $((2^b-1)b, (2^b-d)b, d)$-linear codes. Error-correction is efficiently possible using the Berlekamp-Massey algorithm [Ber67, Ber84].

Let $\delta, \nu > 0$ be some constants with $\delta + \nu < \frac{1}{2}$. (E.g., $\delta = \nu = \frac{1}{6}$.) Let k denote the security parameter. All values introduced below will depend on k. Let $d := k$. Let b be the smallest integer such that $n := (2^b - 1)b$ satisfies $\frac{1}{2}n - \delta n - \nu n - (d-1)b - k \geq \ell$ and $\nu n \geq a$ and $n \geq k$. Such a b exists and then n is polynomially-bounded. Let $\kappa := (2^b - d)b$. Let $\mathbf{S}$ be the syndrome of the (n, κ, d)-Reed-Solomon code. Let $\mathbf{F}$ be a family of universal hash functions $F : \{0,1\}^n \to \{0,1\}^\ell$. Such functions exist for any n, ℓ; for example, the set of all affine transformations from $\{0,1\}^n$ to $\{0,1\}^\ell$ is easily seen to be strongly universal.

Since $\ell \leq \frac{1}{2}n - \delta n - \nu n - (d-1)b - k = \frac{1}{2}n - \delta n - \nu n - (n - \kappa) - d$, by Lemma 16, we have that π_{KWW} is (ε_1, a)-BQS-hiding for some exponentially-small ε_1.

And with $s := n$, by Lemmas 18 and 19, we get that π_{KWW} is (s, ε_2)-online-extractable with $\varepsilon := 2^{d/2}$.

Since both ε_1 and ε_2 are exponentially-small, the theorem holds with $\varepsilon := \max\{\varepsilon_1, \varepsilon_2\}$. □

3.2 BQS-UC commitments

In this section, we present a commitment scheme π_{COM} that is BQS-UC-secure for memory bound a and for n concurrent instances of π_{COM}. The parameters a and n can be arbitrary, but π_{COM} depends on them. To state our result, we first define the ideal functionality for commitments.

Definition 21 (Commitment) *Let A and B be two parties. The functionality $\mathcal{F}_{\mathrm{COM}}^{A\to B,\ell}$ behaves as follows: Upon (the first) input* $(\texttt{commit}, x)$ *with $x \in \{0,1\}^{\ell(k)}$ from A, store x and send* `committed` *to B. Upon (the first) input* `open` *from A send* $(\texttt{open}, x)$ *to B (unless x is still undefined). All communication/input/output is classical.*

We call A the sender and B the recipient.

Note that this definition also defined the behavior of the functionality in the case where A or B is corrupted. In this case, the adversary (or simulator) is allowed to send/receive the inputs/outputs in the name of A or B, respectively. For example, if A is corrupted, the adversary can decide when to commit to what message and when to open.

Intuition. The protocol π_{COM} is depicted in Figure 5. Before we prove its security, we first explain the underlying intuition. In order to prove the BQS-UC-security of π_{COM}, it is necessary to construct a simulator (that may use more quantum memory than the adversary) that achieves the following: When being in the role of the recipient, the simulator is able to extract the commitment after the commit phase. When being in the role of the sender, the simulator should be able to open the commitment to any value of his choosing (equivocality). The first requirement can easily be achieved by using the online-extractable commitment scheme from Theorem 20. That scheme, however, is not equivocal. In order to make our protocol equivocal, we intentionally weaken the binding property of the commitment. Instead of committing to a single value v, the sender commits using a commitment scheme C_2 to random values $\underline{R} := R_1, \dots, R_m$. Then he sends $v \oplus F(\underline{R})$ with F being a universal hash function and sends the syndrome σ of $\underline{R}$ with respect to a suitable linear code. In the open phase, the sender does not open all commitments R_i, but instead just sends $\underline{R}$ to the recipient. The recipient chooses a test set T, and the sender opens R_i for $i \in T$. The modified scheme is still binding: Assume the sender wishes to be able to open the commitment with two different values. Then he has to find values $\underline{R}' \neq \underline{R}$ that both pass the recipients checks in the open phase. If $\underline{R}'$ differs from $\underline{R}$ in many blocks R_i, with high probability the verifier will require that one of these R_i is opened and the sender will be caught. If $\underline{R}'$ and $\underline{R}$ differs in only few blocks, then $\underline{R} - \underline{R}'$ has a low Hamming weight and is not in the code. Hence the syndrome of $\underline{R} - \underline{R}'$ is not zero, and, since the code is linear, the syndromes of $\underline{R}'$ and $\underline{R}$ cannot both equal σ. Thus the sender is caught, too. Furthermore, our scheme is online-extractable if C_2 is online-extractable since the simulator can extract the committed values $\underline{R}$. However, we have not yet achieved the equivocality. In order to open the commitment to a different value, the sender needs to know T before sending $\underline{R}'$. To achieve this, the recipient commits to T before the commit phase (using an online-extractable commitment scheme C_2). A simulator wishing to change the value of the commitment simply extracts T. Then he knows which R_i can be changed without being detected and can thus change $F(R_1, \dots, R_m)$ to any value he wishes.

Difficulties with concurrent composition. The main difficulty in showing the BQC-UC-security of π_{COM} lies in coping with the fact that several (say n) instances of π_{COM} might run concurrently. Consider for example the case that Alice is corrupted. In this case, the adversary may produce the C_2-commitments to $R_1, \dots, R_m$ in n instances of Alice. The simulator needs to run the nm extractors to extract these commitments. Each of these extractors needs some quantum memory s_2. Thus our simulator needs nms_2 bits of quantum memory. On the other hand, we need to make sure that the C_1-commitments to T, produced by the simulator, are hiding. C_1 needs to be hiding against a_1-bounded adversaries with

[17]Note that this does not refer to the memory bound of the adversary. We only state that honest Alice and Bob do not need to use quantum memory in C_1 and C_2.

Parameters: Integers ℓ (the length of the committed value), m, $c < m$, b, d, $\kappa < m$. A b-block (m, κ, d)-linear code where $\mathbf{S}(\omega) \in \{0,1\}^{(m-\kappa)b}$ denotes the syndrome of a codeword $\omega \in \{0,1\}^{mb}$. A family $\mathbf{F}$ of strongly universal hash functions $F : \{0,1\}^{mb} \to \{0,1\}^{\ell}$. All parameters may depend on the security parameter k.
Subprotocols: A commitment scheme C_1 with sender Bob, and a commitment scheme C_2 with sender Alice, both 0-memory bounded (not using quantum memory).[17]
Parties: The sender Alice A and the recipient Bob B.
Inputs: In the commit phase, Alice gets $(\texttt{commit}, v)$ with $v \in \{0,1\}^{\ell}$. In the open phase, Alice gets `open`. Bob gets no inputs.
Commit phase:
C1. Bob picks a random $T \subseteq \{1, \dots, m\}$ with $\#T = c$. Then Bob commits to T using C_1. (We assume some encoding of sets T that does not allow to encode sets with $\#T \neq c$.)
C2. Alice picks $R_1, \dots, R_m \in \{0,1\}^b$. For each i, Alice commits to R_i using C_2. (The commitments may be performed concurrently.)
C3. Alice picks a hash function $F \leftarrow \mathbf{F}$, computes $p := v \oplus F(R_1\|\dots\|R_m)$, computes the syndrome $\sigma := \mathbf{S}(R_1\|\dots\|R_m)$, and sends (F, σ, p) to Bob. (This may be done concurrently with the commitments to R_i.)
C4. Bob outputs `committed`.
Open phase:
O1. Alice sends $R_1\|\dots\|R_m$ to Bob.
O2. Bob opens T using C_1.
O3. For each $i \in T$, Alice opens R_i using C_2. (The open phases may be executed concurrently.)
O4. Bob checks that the values R_i sent by Alice match the values R_i opened by Alice for all $i \in T$, and that $\sigma = \mathbf{S}(R_1\|\dots\|R_m)$. Otherwise, Bob aborts.
O5. Bob computes $v := p \oplus F(R_1\|\dots\|R_m)$ and outputs $(\texttt{open}, v)$. (I.e., Bob accepts the opened value v.)

Figure 5: Our commitment protocol π_{COM}.

$a_1 > nms_2 \geq s_2$ (because we cannot be sure that the memory used by the simulator is not misused by the adversary). But then the extractor for C_1 needs to use $s_1 > a_1$ qubits; otherwise the adversary could run the extractor to break the protocol. Similarly, we can see that when Bob is corrupted, C_2 needs to be hiding against a_2-memory bounded adversaries with $a_2 > ns_1 \geq s_1$, and $s_2 > a_2$. Thus we need $a_1 > s_2 > a_2 > s_1 > a_1$ which is impossible.

Solving the difficulties. The way out is to carefully track the memory used by the simulators; it turns out that in the proof of security against corrupted Alice, we can make sure that the adversary is not able to "misuse" the memory of the simulator. When Alice is corrupted, we need to construct a simulator that extracts the values v of n concurrent commitments produced by Alice, while being indistinguishable from an execution of the honest recipient Bob. More precisely, we show that the simulator is indistinguishable if C_1 is a_1-BQS-hiding and C_2 is s_2-online-extractable and environment and adversary are a_1-memory-bounded. We do not require that $a_1 > s_2$, thus breaking the above-mentioned circularity in the choices of a_1, s_1, a_2, s_2.

Let B^* be defined like the honest Bob, except that instead of honestly running the recipient's code for C_2, B^* runs the extractor B_S for C_2 to extract the committed values $\underline{R}$ in the C_2-commitments. From $\underline{R}$, B^* computes a guess v' for the committed value v. B^*

does not, however, use this guess at any point.

First, note that B^* is indistinguishable from honest Bob: This follows from the fact that the extractor for C_2 is indistinguishable from the recipient for C_2. Furthermore, as discussed in the section "Intuition" above, extracting v will be successful as long as Alice does not learn anything about T, i.e., as long as C_1 is hiding. But C_1 is only a_1-BQS-hiding. And B^* uses ms_2 qubits to run the extractors for C_2, so the total memory used in the network is $a_1 + ms_2$ which is beyond the memory bound tolerated by C_1. Fortunately, however, honest Bob runs the recipient of C_2 after the end of the commit phase and before the beginning of the open phase of C_1. Thus, from the point of view of C_1, the extractors are executed within a single atomic computation. And we defined BQS-hiding to hold even if the adversary uses unlimited memory within a single activation. Thus the ms_2 qubits used by the extractors do not break the hiding property, and we get that B^* guesses the right v' with overwhelming probability.

This argument does, however, only work when a single instance of B^* is executed. If several instances of B^* are executed, one instance may run the commit or open phase of C_1 concurrently with another instance's extractors. Two show that n concurrent instances of B^* extract successfully, we use the following argument: For each j, we have that if only the j-th Bob instance is replaced by B^*, then B^* extracts correctly. Furthermore, Bob and B^* are indistinguishable, thus if we replace all the other instances of Bob by B^*, the j-th instance still extracts correctly. Thus, for any j, if there are n instances of B^*, then the j-th instance extracts correctly. Thus all instances of B^* extract correctly. And the instances of B^* are indistinguishable from the instances of Bob.

Finally, we can construct a simulator that runs B^* instead of Bob and uses the value v' extracted by B^* as input to the commitment functionality. Since $v = v'$ with overwhelming probability, this simulator is successful.

Thus we have shown that a_1 can be chosen independently of s_2. This allows to break the circularity in the choices of a_1, s_1, a_2, s_2: We first start with an arbitrary $a = a_1$. Then we pick an arbitrary a_1-BQS-hiding and s_1-online-extractable C_1, and then an arbitrary $a_2 := a+ns_1$-BQS-hiding and s_2-online-extractable commitment C_2. For the case of corrupted Bob, we then construct a simulator that uses ns_1 qubits and is secure against $a = a_2 - ns_1$-memory bounded environments and adversaries. And for the case of corrupted Alice, using the argument above, we get a simulator that uses nms_2 qubits and is secure against $a = a_1$-bounded environments and adversaries.

The analysis. We proceed with the formal analysis of π_{COM}. We first consider the case where the recipient is corrupted.

Lemma 22 *Assume that ε, δ are negligible, n, c are polynomially-bounded, and $2\kappa b - mb - 2cb - \ell$ is superlogarithmic (in the security parameter k). Assume that C_1 is (ε, s_1)-online-extractable and C_2 is $(\delta, a + ns_1)$-BQS-hiding. Assume that $\mathbf{F}$ is a family of affine strongly universal hash functions.*

Then π^n_{COM} (a, ns_1)-BQS-UC-emulates $(\mathcal{F}^{A\to B,\ell}_{\mathrm{COM}})^n$ for corrupted recipient B.

Proof. First, we describe the structure of the real and ideal model in the case that the party B (Bob) is corrupted:

In the real model, we have the environment $\mathcal{Z}$, the adversary Adv, the honest party A (Alice), the corruption party B^C. The adversary controls the corruption party B^C, so effectively he controls the communication between Alice and Bob. The environment provides Alice's inputs $(\texttt{commit}, v)$ and $\texttt{open}$. See Figure 6 (a).

(a)
$$\begin{tikzcd}
\mathcal{Z} \arrow[r, shift left, "(\texttt{commit},v)"] \arrow[r, shift right, "\texttt{open}"'] \arrow[rrr, leftrightarrow, bend left] & A \arrow[r, leftrightarrow, shift left] \arrow[r, leftrightarrow, shift right] & B^C \arrow[r, leftrightarrow, shift left] \arrow[r, leftrightarrow, shift right] & \text{Adv}
\end{tikzcd}$$

(b)
$$\begin{tikzcd}
\mathcal{Z} \arrow[r, shift left, "(\texttt{commit},v)"] \arrow[r, shift right, "\texttt{open}"'] \arrow[rrrr, leftrightarrow, bend left] & \tilde{A} \arrow[r, shift left, "(\texttt{commit},v)"] \arrow[r, shift right, "\texttt{open}"'] & \mathcal{F}_{\text{COM}} \arrow[r, shift left, "\texttt{committed}"] \arrow[r, shift right, "(\texttt{open},v)"'] & B^C \arrow[r, shift left, "\texttt{committed}"] \arrow[r, shift right, "(\texttt{open},v)"'] & \text{Sim}
\end{tikzcd}$$

Figure 6: Networks occurring in the proof of Lemma 22.

In the ideal model, we have the environment $\mathcal{Z}$, the simulator Sim (to be defined below), the dummy-party $\tilde{A}$, the corruption party B^C, and the commitment functionality $\mathcal{F}_{\text{COM}}$. The inputs $(\texttt{commit}, v)$ and $\texttt{open}$ of $\mathcal{F}_{\text{COM}}$ are provided by the dummy-party $\tilde{B}$ and thus effectively by the environment $\mathcal{Z}$. The simulator Sim controls the corruption party B^C and hence gets the outputs $\texttt{committed}$ and $(\texttt{open}, v)$ of $\mathcal{F}_{\text{COM}}$. See Figure 6 (b).

Fix an adversary Adv. To show Lemma 22, we need to find a simulator Sim with $\text{QM}(\text{Sim}) \leq ns_1$ such that, for any environment $\mathcal{Z}$ with $\text{QM}(\mathcal{Z}) + \text{QM}(\text{Adv}) \leq a$, the real model and the ideal model are negligible-close. This simulator is described in Figure 7. We use the abbreviations $\underline{R} := R_1\|\dots\|R_m$ and $\underline{R}' := R_1'\|\dots\|R_m'$.

To show that the real and the ideal model are negligible-close, we start with the real model, and change the machines in the real model step-by-step until we end up with the ideal model. In each step, we show that the network before and after that step are negligible-close.

Game 1. We change the machine A as follows: Instead of executing the program of the honest recipient of C_1, A executes the extractor A_S. $\diamond$

Let T' denote the extracted value. The modified A does not use T'. Since there are up to n copies of A, and since C_1 is (ε, s_1)-online-extractable, the real model and Game 1 are $n\varepsilon$-close.

Game 2. We change the machine A to abort if the opening of T succeeds and reveals a value $T \neq T'$. $\diamond$

Since C_1 is (ε, s_1)-online-extractable, in each instance of A, this happens with probability at most ε, thus Game 1 and Game 2 are $n\varepsilon$-close.

Notice that the only machines that use quantum memory in Game 2 are $\mathcal{Z}$, Adv, and n copies of A_S. Since A_S is s_1-memory bounded, and $\text{QM}(\mathcal{Z}) + \text{QM}(\text{Adv}) \leq a$ we have that the total amount of quantum memory used in Game 2 is bounded by $a + ns_1$.

Game 3. We change the machine A to commit to 0^b instead of R_i for each $i \notin T'$. $\diamond$

To see that Game 2 and Game 3 are negligible-close, we introduce an intermediate hybrid game, Game 3_j, in which only the first j of the commitments to $R_i, i \notin T$ are replaced by commitments to 0^b. Since at most $a + ns_1$ qubits of quantum memory are used in Game 2 and therefore also in Game 3_j, and since the C_2-commitments to $R_i, i \notin T$ are never opened, from the fact that C_2 is $(\delta, a + ns_1)$-BQS-hiding it follows that Game 3_j and Game 3_{j+1} are δ-close. Note that there are, in the whole game, up to n copies of A and thus up to nc C_2-commitments to some $R_i, i \notin T$. Thus Game 2 = Game 3_0 and Game 3 = Game 3_{nc} are $nc\delta$-close.

Commit phase (on input `committed`):

- When Bob commits to T using C_1, the simulator runs the extractor A_S for C_1 instead of the honest recipient's program. (Since C_1 has recipient Alice, we write A_S, not B_S.) Let T' denote the value extracted by A_S.
- The simulator picks $R_1, \dots, R_m \in \{0,1\}^b$. For each i, Sim commits (honestly) to R_i (if $i \in T$) or to 0^b (if $i \notin T$) using C_2.
- Sim picks a hash function $F \overset{R}{\leftarrow} \mathbf{F}$, picks a random $p \overset{R}{\leftarrow} \{0,1\}^\ell$, computes the syndrome $\sigma := \mathbf{S}(\underline{R})$, and sends (F, σ, p) to Bob.

Open phase (on input $(\texttt{open}, v)$ with $v \in \{0,1\}^\ell$):

- Sim picks $\underline{R}' \in \{\underline{R}' : \forall i \in T'. R_i = R'_i, \sigma = \mathbf{S}(\underline{R}'), p \oplus F(\underline{R}') = v\}$ uniformly.[18]
- Sim sends $\underline{R}'$ to Bob.
- Sim waits for Bob to open T using C_1. If $T \neq T'$, Sim aborts.
- For each $i \in T'$, Sim (honestly) opens R_i using C_2.

Figure 7: Simulator Sim for the case of corrupted Bob. The program described in this figure is executed for each instance of the n instances of π_{COM}. Communication with Bob is sent to an internally simulated instance of the adversary Adv.

Game 4. We modify A to set $\underline{R}' := \underline{R}$ and to send $\underline{R}'$ instead of $\underline{R}$ to Bob in step O1. $\diamond$

This modification is for notational purposes only, Game 3 and Game 4 are perfectly close.

Game 5. We modify the way A chooses $F, \underline{R}, \underline{R}', \sigma, p$: In Game 4, we have $F \overset{R}{\leftarrow} \mathbf{F}$, $\underline{R} \overset{R}{\leftarrow} \{0,1\}^{mb}$, $\sigma := \mathbf{S}(\underline{R})$, $p := v \oplus F(\underline{R})$, $\underline{R}' := \underline{R}$. (We call this distribution $\mathcal{D}_1$.) In Game 5 we use $F \overset{R}{\leftarrow} \mathbf{F}$, $\underline{R} \overset{R}{\leftarrow} \{0,1\}^{mb}$, $p \overset{R}{\leftarrow} \{0,1\}^\ell$, $\sigma := \mathbf{S}(\underline{R})$, $\underline{R}' \overset{R}{\leftarrow} \{\underline{R}' : \forall i \in T'. R_i = R'_i, \sigma = \mathbf{S}(\underline{R}'), p \oplus F(\underline{R}') = v\} =: \mathcal{R}_{F,\underline{R},p}$. (We call this distribution $\mathcal{D}_2$.) $\diamond$

To show that Game 4 and Game 5 are negligible-close, we use the following claim:

Claim 1 *Let $R_T := (R_i)_{i \in T}$. For any $v \in \{0,1\}^\ell$, the statistical distance between $(F, R_T, \underline{R}', \sigma, p)$ chosen according to $\mathcal{D}_1$ and $(F, R_T, \underline{R}', \sigma, p)$ chosen according to $\mathcal{D}_2$ is at most $2^{cb+mb/2+\ell/2-\kappa b-1}$.*

We prove this claim below. Using Claim 1 and the fact that we have n instances of A, we immediately get that Game 4 and Game 5 are $n(2^{cb+mb/2+\ell/2-\kappa b-1})$-close because the values $(R_i)_{i \notin T}$ are never used by A (except indirectly through $\underline{R}'$, σ, and p).

Finally, note that by construction of Sim, Game 5 and the ideal model are perfectly close. Thus the real and the ideal model are γ-close with $\gamma := 2n\varepsilon + nc\delta + n(2^{cb+mb/2+\ell/2-\kappa b-1})$. Since ε, δ are negligible, and n, c are polynomially-bounded, and $2\kappa b - mb - 2cb - \ell$ is super-logarithmic, we have that γ is negligible. Thus π^n_{COM} (a, ns_1)-BQS-UC-emulates $(\mathcal{F}^{A \to B, \ell}_{\mathrm{COM}})^n$ in the case of corrupted Bob.

Proof of Claim 1. We now prove the pending Claim 1. Let $F \overset{R}{\leftarrow} \mathbf{F}$, $\underline{R} \overset{R}{\leftarrow} \{0,1\}^{mb}$, $\sigma := \mathbf{S}(\underline{R})$, $p^{(1)} := v \oplus F(\underline{R})$, $p^{(2)} \overset{R}{\leftarrow} \{0,1\}^\ell$, $U \overset{R}{\leftarrow} \{0,1\}^\ell$, $\underline{R}'^{(1)} := \underline{R}$, and $\underline{R}'^{(2)} \overset{R}{\leftarrow} \mathcal{R}_{F,\underline{R},p^{(2)}}$. We write $X \equiv Y$ if the random variables X and Y have the same distribution, and $X \approx_\eta Y$ if their distributions have statistical distance at most η. Then we have to show that $(F, R_T, \underline{R}'^{(1)}, \sigma, p^{(1)}) \approx_\eta (F, R_T, \underline{R}'^{(2)}, \sigma, p^{(2)})$ with $\eta := 2^{cb+mb/2+\ell/2-\kappa b-1}$.

[18]Note $\underline{R}'$ can be sampled efficiently since the conditions $\forall i \in T'. R_i = R'_i$, $\sigma = \mathbf{S}(\underline{R}')$, and $p \oplus F(\underline{R}') = v$ are a system of linear equations. This uses that $\mathbf{S}$ is the syndrome of a linear code, and that $\mathbf{F}$ is a family of affine functions.

(a)
$$\begin{tikzcd}
\text{Adv} \arrow[r, Leftrightarrow] & A^C \arrow[r, Leftrightarrow] & B \arrow[r, shift left, "\texttt{committed}"] \arrow[r, shift right, "(\texttt{open},v)"'] & \mathcal{Z} \arrow[lll, bend right, end anchor=west]
\end{tikzcd}$$

(b)
$$\begin{tikzcd}
\text{Sim} \arrow[r, shift left, "(\texttt{commit},v)"] \arrow[r, shift right, "\texttt{open}"'] & A^C \arrow[r, shift left, "(\texttt{commit},v)"] \arrow[r, shift right, "\texttt{open}"'] & \mathcal{F}_{\text{COM}} \arrow[r, shift left, "\texttt{committed}"] \arrow[r, shift right, "(\texttt{open},v)"'] & \tilde{B} \arrow[r, shift left, "\texttt{committed}"] \arrow[r, shift right, "(\texttt{open},v)"'] & \mathcal{Z} \arrow[llll, bend right, end anchor=west]
\end{tikzcd}$$

Figure 8: Networks occurring in the proof of Lemma 23.

The leftover hash lemma [HILL99] in the variant from [HU08] states that $(F, F(\underline{R}), R_T\|\sigma)$ and $(F, U, R_T\|\sigma)$ have statistical distance at most $\lambda := 2^{|(R_T\|\sigma)|+|U|/2-H_2(\underline{R})/2-1}$ where $H_2(\underline{R})$ denotes the collision-entropy of $\underline{R}$. (Intuitively, $R_T\|\sigma$ is considered as leaked information about the source $\underline{R}$, and $F(\underline{R})$ is the randomness extracted from $\underline{R}$.) We have $\lambda = 2^{cb+(m-\kappa)b+\ell/2-mb/2-1} = \eta$.

Hence $(F, F(\underline{R}), R_T, \sigma) \approx_\eta (F, U, R_T, \sigma)$. Thus $(F, R_T, p^{(1)}, \sigma) \equiv (F, R_T, v \oplus F(\underline{R}), \sigma) \approx_\eta (F, R_T, v \oplus U, \sigma) \equiv (F, R_T, p^{(2)}, \sigma)$. We define the probabilistic function W, which on input (F, R_T, p, σ) returns a uniform element from $\mathcal{R}_{F,\underline{R},p}$ (note that $\mathcal{R}_{F,\underline{R},p}$ does not depend on $R_i, i \notin T$). Then $(F, R_T, p^{(1)}, \sigma, W(F, R_T, p^{(1)}, \sigma)) \approx_\eta (F, R_T, p^{(2)}, \sigma, W(F, R_T, p^{(2)}, \sigma))$. By definition of $\underline{R}'^{(2)}$ we have $(F, R_T, p^{(2)}, \sigma, W(F, R_T, p^{(2)}, \sigma)) \equiv (F, R_T, p^{(2)}, \sigma, \underline{R}'^{(2)})$. Furthermore, since $W(F, R_T, p^{(1)}, \sigma))$ uniformly samples from those $\underline{R}$ compatible with F, R_T, $p^{(1)}$, and σ, we have that $(F, R_T, p^{(1)}, \sigma, W(F, R_T, p^{(1)}, \sigma)) \equiv (F, R_T, p^{(1)}, \sigma, \underline{R}) \equiv (F, R_T, p^{(1)}, \sigma, \underline{R}')$. Thus $(F, R_T, p^{(1)}, \sigma, \underline{R}') \approx_\eta (F, R_T, p^{(2)}, \sigma, \underline{R}'^{(2)})$ and Claim 1 is shown. □

Lemma 23 *Assume that ε, δ are negligible, n is polynomially-bounded, and $(1-\frac{d}{m})^c$ is negligible (in the security parameter k). Assume that C_1 is (ε, a)-BQS-hiding and that C_2 is (δ, s_2)-online-extractable. Assume that the code with syndrome* $\mathbf{S}$ *has efficient error-correction.*

Then π^n_{COM} (a, nms_2)-BQS-UC-emulates $(\mathcal{F}^{A\to B,\ell}_{\text{COM}})^n$ for corrupted sender A.

Proof. First, we describe the structure of the real and the ideal model in the case that the party A (Alice) is corrupted:

In the real model, we have the environment $\mathcal{Z}$, the adversary Adv, the corruption party A^C, and the honest party B (Bob). The adversary controls the corruption party A^C, so effectively he controls the communication between Alice and Bob. The environment gets Bob's outputs `committed` and $(\texttt{open}, v)$. See Figure 8 (a).

In the ideal model, we have the environment $\mathcal{Z}$, the simulator Sim (to be defined below), the corruption party A^C, the dummy-party $\tilde{B}$, and the commitment functionality $\mathcal{F}_{\text{COM}}$. The inputs $(\texttt{commit}, v)$ and `open` of $\mathcal{F}_{\text{COM}}$ are provided by the corruption party A^C and thus effectively by the simulator Sim. The environment $\mathcal{Z}$ controls the dummy-party $\tilde{B}$ and hence gets the outputs `committed` and $(\texttt{open}, v)$ of $\mathcal{F}_{\text{COM}}$. See Figure 8 (b).

Fix an adversary Adv. To show Lemma 23, we need to find a quantum-polynomial-time simulator Sim with $\text{QM}(\text{Sim}) \le nms_2$ such that, for any environment $\mathcal{Z}$ with $\text{QM}(\mathcal{Z}) + \text{QM}(\text{Adv}) \le a$, the real model and the ideal model are negligible-close. This simulator is described in Figure 9. Note that Sim is quantum-polynomial-time: The extractor B_S is quantum-polynomial-time by definition, and computing $\underline{R}^*$ is possible in polynomial-time

Commit phase:
- Sim picks a random $T \subseteq \{1, \dots, m\}$ with $\#T = c$. Then Sim (honestly) commits to T using C_1.
- When Alice commits to $R_1, \dots, R_m$, the simulator runs the extractor B_S for C_2 instead of the honest recipient's program. Let $R'_1, \dots, R'_m$ denote the extracted values.
- Sim waits for (F, σ, p) from Alice.
- Sim computes an $\underline{R}^* \in \{0,1\}^{mb}$ with $\mathbf{S}(\underline{R}^*) = \sigma$ and $\omega(\underline{R}', \underline{R}^*) \leq (d-1)/2$ (remember that ω is the *block-wise* Hamming distance), computes $v' := p \oplus F(\underline{R}^*)$, and sends $(\texttt{commit}, v')$ to $\mathcal{F}_{\text{COM}}$. (If no such $\underline{R}^*$ exists, we set $v' := \perp$.)

Open phase:
- Sim waits for $\underline{R}$ from Alice.
- Sim (honestly) opens T using C_1.
- For each $i \in T$, Sim waits for Alice to open R_i using C_2.
- Sim checks that the values R_i sent by Alice match the values R_i opened by Alice for all $i \in T$, and that $\sigma = \mathbf{S}(R_1 \| \dots \| R_m)$.
- Sim sends $\texttt{open}$ to $\mathcal{F}_{\text{COM}}$.

Figure 9: Simulator Sim for the case of corrupted Alice. The program described in this figure is executed for each instance of the n instances of π_{COM}. Communication with Alice is sent to an internally simulated instance of the adversary Adv.

because the code with syndrome $\mathbf{S}$ has efficient error-correction. Since C_2 is (δ, s_2)-online-extractable and Sim uses m instances of B_S per copy of B, $\text{QM}(\text{Sim}) \leq nms_2$. We use the abbreviations $\underline{R} := R_1 \| \dots R_m$ and similarly for $\underline{R}'$ and $\underline{R}^*$.

Before we proceed, we introduce two variants of the honest recipient B. The machine B^* behaves like B, but when Alice commits to $R_1, \dots, R_m$ using C_2, B^* runs the extractor B_S for C_2 instead of the honest recipient's program. Call the extracted values $R'_1, \dots, R'_m$. Further, B^* computes an $\underline{R}^*$ with $\mathbf{S}(\underline{R}^*) = \sigma$ and $\omega(\underline{R}', \underline{R}^*) \leq (d-1)/2$ and then computes $v' := p \oplus F(\underline{R}^*)$. (If no such $\underline{R}^*$ exists, $v' := \perp$.) In the open phase, B^* behaves like B. In particular, B^* outputs $(\texttt{open}, v)$, not $(\texttt{open}, v')$. That is, v' is computed but never used.

The machine B^+ behaves like B^*, but outputs $(\texttt{open}, v')$ instead of $(\texttt{open}, v)$.

By definition of online-extractability, and since B^* does not use the value extracted by B_S, we have that B and B^* are δ-indistinguishable. More precisely, for any network S, we have that $S \cup \{B\}$ and $S \cup \{B^*\}$ are δ-close. Since online-extractability was defined with respect to non-memory bounded adversaries, this holds even if S is not memory bounded.

As in Lemma 22, we proceed by investigating a sequence of games.

Game 1. In the game Game 1_j, the j-th instance of B is replaced by an instance of B^*. (Note: only one instance is replaced, not the first j instances.) ◇

We use the following claim:

Claim 2 *Let S be an a-memory bounded network. In an execution of $S \cup \{B^*\}$, let v, v' denote the values v, v' computed by B^*. Then* $\Pr[v \notin \{v', \perp\}] \leq \varepsilon + (1 - \frac{d}{m})^c := \eta$.

We prove this claim below. Since C_1 and C_2 are 0-memory bounded, we have that the machine B is 0-memory bounded. $\text{QM}(\mathcal{Z}) + \text{QM}(\text{Adv}) \leq a$. Thus we can apply Claim 2 to Game 1_j. Hence in Game 1_j, $\Pr[v_j \notin \{v'_j, \perp\}] \leq \eta$ where v_j, v'_j are the values v, v' computed by B^*. We write $v_j := \perp$ if the open phase fails or does not take place (and hence v_j is not computed by B^*).

Game 2. This game is defined like the real model, except that we use n instances of B^* instead of the n instances of B. $\diamond$

Using the fact that B and B^* are δ-indistinguishable, we get that the real model and Game 2 are $n\delta$-close.

Again using that B and B^* are δ-indistinguishable, we get that $\big|\Pr[v_j \notin \{v'_j, \bot\} : \text{Game } 1_j] - \Pr[v_j \notin \{v'_j, \bot\} : \text{Game } 2]\big| \leq (n-1)\delta$. Thus $\Pr[v_j \notin \{v'_j, \bot\} : \text{Game } 2] \leq \eta + (n-1)\delta$. Since this holds for all $j = 1, \dots, n$, we get:

$$\Pr[\exists j.\ v_j \notin \{v'_j, \bot\} : \text{Game } 2] \leq n\eta + n(n-1)\delta. \tag{1}$$

Game 3. This game is defined like the real model, except that we use n instances of B^+ instead of the n instances of B. $\diamond$

Notice that Game 2 and Game 3 only differ in the fact that in Game 2 we use instances of B^* and in Game 3 instances of B^+. By definition, B^* and B^+ only differ in the value they output: B^* outputs $(\mathtt{open}, v)$ and B^+ outputs $(\mathtt{open}, v')$. By (1), the probability that the values v, v' are different in some instance of B^* is bounded by $n\eta + n(n-1)\delta$. Hence Game 2 and Game 3 are $(n\eta + n(n-1)\delta)$-close.

Finally, note that by construction of Sim, Game 3 and the ideal model are perfectly close. Thus the real and the ideal model are γ-close with $\gamma := n\delta + n\eta + n(n-1)\delta = n^2\delta + n\varepsilon + n(1-\frac{d}{m})^c$. Since δ, ε are negligible, n is polynomially-bounded, and $(1-\frac{d}{m})^c$ is negligible, we have that γ is negligible. Thus π^n_{COM} (a, nms_2)-BQS-UC-emulates $(\mathcal{F}^{A \to B, \ell}_{\text{COM}})^n$ in the case of corrupted Alice.

Proof of Claim 2. We now prove the pending Claim 2. Let $\underline{R}$, $\underline{R}'$, $\underline{R}^*$ and T denote the corresponding values as computed by B^*. We abbreviate $R_T := (R_i)_{i \in T}$ and $R'_T := (R'_i)_{i \in T}$. By *Bad* we denote the event that $R_T = R'_T$ and $\mathbf{S}(\underline{R}) = \sigma$ and $\underline{R} \neq \underline{R}^*$. By construction of B^*, $v \neq \bot$ implies $R_T = R'_T$ and $\mathbf{S}(\underline{R}) = \sigma$. And $v \notin \{v', \bot\}$ implies $\underline{R} \neq \underline{R}^*$. Thus $v \notin \{v', \bot\}$ implies *Bad*. Therefore, to show Claim 2, it is sufficient to show $\Pr[\mathit{Bad}] \leq \eta$ in $S \cup \{B^*\}$. To show this, we again proceed using a sequence of games:

Game 4. An execution of $S \cup \{B^*\}$. $\diamond$

Game 5. We change B^* to halt after receiving $\underline{R}$ from Alice. $\diamond$

Then $\Pr[\mathit{Bad} : \text{Game } 4] = \Pr[\mathit{Bad} : \text{Game } 5]$.

Game 6. We change B^* to commit to some (arbitrary) fixed value T_0 instead of committing to T. $\diamond$

We wish to apply the (ε, a)-BQS-hiding property of C_1 in order to show that $\big|\Pr[\mathit{Bad} : \text{Game } 5] - \Pr[\mathit{Bad} : \text{Game } 6]\big| \leq \varepsilon$. Let B_1 denote the sender in the commitment scheme C_1. By definition, to commit to T (or T_0), B^* internally runs B_1. We construct an adversary A'_1 that interacts with B_1. This adversary simulates $S \cup \{B^*\}$ (with B^* as in Game 5) except for the machine B_1 inside B^*. Note that in Game 5, only the commit phase of C_1 is executed. We let A'_1 output 1 iff *Bad* happens. We define $\hat{B}_1$ like B_1, except that $\hat{B}_1$ ignores its input and commits to T_0. Let P be the probability that A'_1 outputs 1 when running with B_1, and let $\hat{P}$ be the probability that A'_1 outputs 1 when running with $\hat{B}_1$. By construction, $P = \Pr[\mathit{Bad} : \text{Game } 5]$ and $\hat{P} = \Pr[\mathit{Bad} : \text{Game } 6]$. Thus we only have to show that $|P - \hat{P}| \leq \varepsilon$. To apply the (ε, a)-BQS-hiding property of C_1 we have to check that A'_1 is a-memory bounded. A'_1 simulates $\mathcal{Z}$, Adv, and B^*. We have $\text{QM}(\mathcal{Z}) + \text{QM}(\text{Adv}) \leq a$ by assumption. But B^* contains the extractor B_S for C_2 which uses additional s_2 qubits of quantum memory. Yet, B_S is executed after the end of the commit phase of C_1. That is, B^* is executed within a single activation of A_1' (since B_1 is not activated any more after the commit phase). Note that, although A_1' might use more than a qubits during the activation in which B^* is simulated, it stores at most a qubits between activations. Thus A_1' is a-memory bounded (remember that our definition of "a-memory bounded", Definition 2, only requires that the memory bound holds between activations). Hence $|P - \hat{P}| \leq \varepsilon$ and thus $\big|\Pr[\mathit{Bad} : \text{Game } 5] - \Pr[\mathit{Bad} : \text{Game } 6]\big| \leq \varepsilon$.

Game 7. We change B^* to choose T only after receiving $\underline{R}'$. $\diamond$

Since T is not used earlier by B^*, $\Pr[\mathit{Bad} : \text{Game } 6] = \Pr[\mathit{Bad} : \text{Game } 7]$. Fix values $\underline{R}$, $\underline{R}'$ and σ with $\mathbf{S}(\underline{R}) = \sigma$. We distinguish two cases, depending on whether there exists an $\underline{R}^*$ with $\mathbf{S}(\underline{R}^*) = \sigma$ and $\omega(\underline{R}^*, \underline{R}') \leq (d-1)/2$. Case "$\underline{R}^*$ exists": Since $\mathbf{S}$ is the syndrome of a b-block (m, κ, d)-linear code, $\mathbf{S}(\underline{R} - \underline{R}^*) = 0$, hence $\underline{R} - \underline{R}^*$ is a codeword. Hence $\underline{R} = \underline{R}^*$ or $\omega(\underline{R}, \underline{R}^*) \geq d$. Using the triangle inequality and $\omega(\underline{R}^*, \underline{R}') \leq (d-1)/2$, we get that $\underline{R} = \underline{R}^*$ or $\omega(\underline{R}, \underline{R}') \geq d - (d-1)/2 \geq d/2$. Case "$\underline{R}^*$ does not exist": Since no $\underline{R}^*$ with $\mathbf{S}(\underline{R}^*) = \sigma$ and $\omega(\underline{R}^*, \underline{R}') \leq (d-1)/2$ exists, and since $\mathbf{S}(\underline{R}) = \sigma$, we have that $\omega(\underline{R}, \underline{R}') > (d-1)/2$. Hence $\omega(\underline{R}, \underline{R}') \geq d/2$.

Thus, for any fixed choice of $\underline{R}, \underline{R}', \sigma$, we have $\mathbf{S}(\underline{R}) \neq \sigma$ or $\underline{R} = \underline{R}^*$ or $\omega(\underline{R}, \underline{R}') \geq d/2$.

If $\underline{R} = \underline{R}^*$ or if $\mathbf{S}(\underline{R}) \neq \sigma$, the event Bad does not occur by definition.

If $\omega(\underline{R}, \underline{R}') \geq d/2$, we bound the probability of Bad occurring as follows: Let $D := \{i : R_i \neq R_i'\}$. Then, for random $T \subseteq [m]$ with $\#T = c$, we have $\Pr[\mathit{Bad}] \leq \Pr[R_T = R_T'] = \Pr[T \cap D = \varnothing] \leq (1 - \frac{\#D}{m})^c \leq (1 - \frac{d}{m})^c$.

Thus for any fixed $\underline{R}, \underline{R}', \sigma$ we have $\Pr[\mathit{Bad}] \leq (1 - \frac{d}{m})^c$. By averaging over the choice of $\underline{R}, \underline{R}', \sigma$, we get $\Pr[\mathit{Bad} : \text{Game } 7] \leq (1 - \frac{d}{m})^c$.

Summarizing, we have $\Pr[\mathit{Bad} : \text{Game } 4] \leq \varepsilon + (1 - \frac{d}{m})^c = \eta$. This shows Claim 2. $\square$

Using Reed-Solomon codes for $\mathbf{S}$, and the extractable commitments from Theorem 20 for C_1 and C_2, we can instantiate the parameters of π_{COM} to satisfy the conditions of Lemmas 22 and 23. Thus we get the following theorem:

Theorem 24 *Let ℓ, n, and a be polynomially-bounded. Then there are choices for the parameters of π_{COM} and a polynomially-bounded integer s such that π_{COM} is polynomial-time, constant-round and π_{COM}^n (a,s)-BQS-UC-emulates $(\mathcal{F}_{\mathrm{COM}}^{A\to B,\ell})^n$.*

Proof. Let k be the security parameter. Let $\ell' := \max\{\ell, k\}$. Let $b := \lceil \log \ell' \rceil$, $m := 2^b - 1$, $c := d := \lceil \frac{m}{6} \rceil$, and $\kappa := 2^b - d$. For any b and any $d < 2^b$, there exists a Reed-Solomon code [RS60] over $\mathrm{GF}(2^b)$ that is a $(2^b - 1, 2^b - d, d)$-linear code. Thus there exists an efficient b-block (m, κ, d)-linear code. Let $\mathbf{S}$ be its syndrome. Let $\mathbf{F}$ be a family of affine strongly universal hash functions $F : \{0,1\}^{mb} \to \{0,1\}^{\ell}$. Such functions exist for any m, b, ℓ; for example, the set of all affine transformations from $\{0,1\}^{mb}$ to $\{0,1\}^{\ell}$ is easily seen to be strongly universal. By Theorem 20 there exists a polynomially-bounded s_1, a negligible ε, and a constant-round 0-memory bounded (ε, a)-BQS-hiding (ε, s_1)-online-extractable commitment scheme C_1. Again by Theorem 20, there exists a polynomially-bounded s_2, a negligible δ, and a constant-round 0-memory bounded $(\delta, a + ns_1)$-BQS-hiding (δ, s_2)-online-extractable commitment scheme C_2. Let $s := \max\{ns_1, nms_2\}$.

We have that

$$\begin{aligned} 2\kappa b - mb - 2cb - \ell &= 2(2^b - d)b - (2^b - 1)b - 2db - \ell \\ &= 2^b b - 4db + b - \ell \geq \ell' \lceil \log \ell' \rceil - 4\lceil \tfrac{\ell' - 1}{6} \rceil \lceil \log \ell' \rceil - \ell' \end{aligned}$$

is superlogarithmic in ℓ' and thus also in $k \leq \ell'$. Thus the conditions for Lemma 22 are fulfilled. Hence π^n_{COM} (a, ns_1)-BQS-UC-emulates $(\mathcal{F}^{A\to B,\ell}_{\mathrm{COM}})^n$ in the case of corrupted Bob. Furthermore, we have that

$$(1 - \tfrac{d}{m})^c \leq (1 - \tfrac{1}{6})^{\lceil m/6 \rceil} \leq (\tfrac{5}{6})^{\ell'/6-1} \leq (\tfrac{5}{6})^{k/6-1}$$

is negligible. Thus the conditions for Lemma 23 are fulfilled. Hence π^n_{COM} (a, nms_2)-BQS-UC-emulates $(\mathcal{F}^{A\to B,\ell}_{\mathrm{COM}})^n$ in the case of corrupted Alice. Since $ns_1, nms_2 \leq s$, it follows that π^n_{COM} (a, s)-BQS-UC-emulates $(\mathcal{F}^{A\to B,\ell}_{\mathrm{COM}})^n$.

Note that π_{COM} executes all instances of C_2 concurrently, so π_{COM} is constant-round. □

4 General two-party computation

We now show that general two-party computation is possible in the BQS-UC-framework. More precisely, for any arbitrary functionality $\mathcal{G}$, any arbitrary memory bound a, and any bound n on the number of concurrent instances of the protocol, we construct a protocol π_{bqs2pc} such that π^n_{bqs2pc} (a, s)-BQS-UC-emulates $\mathcal{G}^n$. The protocol has linear round-complexity in the depth of the circuit that is evaluated. (In the following, we just say "linear round-complexity".)

To construct this protocol, we only need to plug together known results:

Theorem 25 (OT reversal [Wul07]) *There is a classical constant-round protocol π_{revOT} that classical-UC-emulates $\mathcal{F}^{A\to B}_{\mathrm{OT}}$ and invokes one instance of $\mathcal{F}^{B\to A}_{\mathrm{OT}}$. Here $\mathcal{F}^{A\to B}_{\mathrm{OT}}$ denotes the functionality for $\binom{2}{1}$-oblivious transfer with sender A and recipient B. $\mathcal{F}^{B\to A}_{\mathrm{OT}}$ is defined analogously.*

Theorem 26 (Classical two-party computation [IPS08]) *Let $\mathcal{G}$ be a well-formed*[19] *classical probabilistic-polynomial-time functionality. Then there is a classical protocol π_{2pc} with linear round-complexity*[20] *that classical-UC-emulates $\mathcal{G}$ and invokes polynomially-many instances of $\mathcal{F}^{A\to B}_{\mathrm{OT}}$ and $\mathcal{F}^{B\to A}_{\mathrm{OT}}$.*

Theorem 27 (OT from commitment [Unr10]) *There is a constant-round 0-memory bounded protocol π_{QOT} that quantum-UC-emulates $\mathcal{F}^{B\to A}_{\mathrm{OT}}$ and invokes polynomially-many instances of $\mathcal{F}^{A\to B,1}_{\mathrm{COM}}$.*

Theorem 28 (BQS two-party computation) *Let $\mathcal{G}$ be a classical well-formed probabilistic-polynomial-time functionality. Let n and a be polynomially-bounded. Then there is a polynomially-bounded s and a 0-memory bounded protocol π_{bqs2pc} with linear round-complexity*[21] *not invoking any functionality such that π^n_{bqs2pc} (a, s)-BQS-UC-emulates $\mathcal{G}^n$.*

[19]Well-formedness describes certain technical restrictions stemming from the proof by Ishai et al. [IPS08]: Whenever the functionality gets an input, the adversary is informed about the length of that input. Whenever the functionality makes an output, the adversary is informed about the length of that output and may decide when this output is to be scheduled.

[20]The short version [Unr11] misquotes this theorem and claims that the protocol from [IPS08] is constant-round.

[21]The short version [Unr11] erroneously claims that the protocol is constant-round. This was due to the mistake in Theorem 26 described in Footnote 20.

Proof. By composing the protocols from Theorem 26 and Theorem 25, we get a classical protocol σ_1 with linear round-complexity that classical-UC-emulates $\mathcal{G}$ and invokes polynomially-many instances of $\mathcal{F}_{\mathrm{OT}}^{B\to A}$. Using the quantum lifting theorem from [Unr10], this implies that σ_1 *quantum*-UC-emulates $\mathcal{G}$. Using the composition theorem (for quantum-UC, [Unr10]), we can compose σ_1 with π_{QOT} (from Theorem 27) and get a protocol σ_2 with linear round-complexity that quantum-UC-emulates $\mathcal{G}$ and invokes polynomially-many instances of $\mathcal{F}_{\mathrm{COM}}^{A\to B,1}$. Using the composition theorem again, we get that σ_2^n quantum-UC-emulates $\mathcal{G}^n$. (Remember that in the case of quantum-UC-security, the composition theorem allows for concurrent composition.) By Lemma 4 (iii, iv), σ_2^n $(a - n\mathrm{QM}(\sigma_2) + s'', s')$-BQS-UC-emulates $\mathcal{G}^n$ for some polynomially-bounded s' and all s''. Let m be a polynomial upper bound on the number of instances of $\mathcal{F}_{\mathrm{COM}}^{A\to B,1}$ invoked by σ_2. By Theorem 24, there is a polynomially-bounded s'' and a constant-round protocol π_{COM} such that π_{COM}^{nm} (a, s'')-BQS-UC-emulates $(\mathcal{F}_{\mathrm{COM}}^{A\to B,1})^{nm}$. Let π_{bqs2pc} be the protocol resulting from replacing invocations of $\mathcal{F}_{\mathrm{COM}}^{A\to B,1}$ in σ_2 by invocations of π_{COM}. By Corollary 12, π_{bqs2pc} $(a - n\mathrm{QM}(\sigma_2), s'' + s')$-BQS-UC-emulates $\mathcal{G}^n$. Since $\mathrm{QM}(\sigma_2) = 0$ by construction, the theorem follows with $s := s'' + s'$. □

5 Conclusions

We have defined the BQS-UC model for analyzing protocols in the bounded quantum storage model. We have given a composition theorem that permits us to replace arbitrary subprotocols; the composition theorem does not, however, allow for concurrent composition (except in rare cases where the simulation overhead s is smaller than the memory bound a). We gave a statistically secure commitment protocol in the BQS-UC model. Our protocol allows for concurrent composition of a polynomially-bounded number n of instances of itself, but the protocol's parameters depend on n. Our protocol has an efficient simulator. Combining our result with prior results, we constructed a statistically BQS-UC secure protocol with linear round-complexity for general two-party computation without any setup assumption.

Open questions. We believe that the following questions constitute interesting directions for future work:

- Find a protocol that concurrently composes with any polynomial-number n of instances of itself; the protocol's parameters should not depend on n. The protocol should also compose with instances of itself in which Alice and Bob exchanged their roles.
- Find a more efficient construction. Currently, our commitment scheme executes a number of instances of the scheme from Theorem 20. The communication complexity of the latter is in the order of magnitude of the adversary's memory bound a. Is it possible to find a BQS-UC secure commitment scheme whose overall communication complexity is in the order of magnitude of a? Furthermore, can we directly implement an OT in the BQS-UC model without going through the construction from [Unr10]?
- Extend our protocols to tolerate noise on the quantum channel. If the protocols tolerate a sufficient amount of noise, they can be implemented with today's technology. Essentially, this boils down to making the protocols from [KWW09] and [Unr10] tolerate noise.

Acknowledgements. We thank Christian Schaffner and Jürg Wullschleger for valuable discussions. This work was funded by the Cluster of Excellence "Multimodal Computing and Interaction". Most of the work was done while the author was at Saarland University.

References

[BB84] Charles H. Bennett and Gilles Brassard. Quantum cryptography: Public-key distribution and coin tossing. In *IEEE International Conference on Computers, Systems and Signal Processing 1984*, pages 175–179. IEEE Computer Society, 1984.

[BBCS91] Charles H. Bennett, Gilles Brassard, Claude Crépeau, and Marie-Hélène Skubiszewska. Practical quantum oblivious transfer. In *Crypto '91*, volume 576 of *LNCS*, pages 351–366. Springer, 1991.

[Ber67] Elwyn R. Berlekamp. Nonbinary BCH decoding. In *International Symposium on Information Theory, San Remo, Italy*, 1967.

[Ber84] Elwyn R. Berlekamp. *Algebraic coding theory.* Aegean Park Press, 2, revised edition, 1984.

[BOM04] Michael Ben-Or and Dominic Mayers. General security definition and composability for quantum & classical protocols, September 2004. Online available at `http://xxx.lanl.gov/abs/quant-ph/0409062`.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS 2001*, pages 136–145. IEEE Computer Society, 2001. Full and revised version is [Can05].

[Can05] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. IACR ePrint Archive, January 2005. Full and revised version of [Can01], online available at `http://eprint.iacr.org/2000/067.ps`.

[DFSS05] Ivan Damgård, Serge Fehr, Louis Salvail, and Christian Schaffner. Cryptography in the bounded quantum-storage model. In *FOCS 2005*, pages 449–458, 2005. A full version is available at `http://arxiv.org/abs/quant-ph/0508222`.

[DM04] Stefan Dziembowski and Ueli Maurer. On generating the initial key in the bounded-storage model. In Christian Cachin and Jan Camenisch, editors, *Eurocrypt 2004*, volume 3027 of *LNCS*, pages 126–137. Springer-Verlag, 2004. Online available at `ftp://ftp.inf.ethz.ch/pub/crypto/publications/DziMau04b.pdf`.

[FS09] Serge Fehr and Christian Schaffner. Composing quantum protocols in a classical environment. In *TCC 2009*, volume 5444 of *LNCS*, pages 350–367. Springer, 2009.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999. Full version online available at `http://www.icsi.berkeley.edu/~luby/PAPERS/hill.ps`.

[HU08] Dennis Hofheinz and Dominique Unruh. Towards key-dependent message security in the standard model. In *Eurocrypt 2008*, volume 4965 of *LNCS*, pages 108–126. Springer, 2008. Preprint on IACR ePrint 2007/333.

[IPS08] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer – efficiently. In *CRYPTO 2008*, volume 5157 of *LNCS*, pages 572–591. Springer, 2008.

[Kil88] Joe Kilian. Founding cryptography on oblivious transfer. In *STOC 1988*, pages 20–31. ACM, 1988.

[KW09] Robert König and Stephanie Wehner. A strong converse for classical channel coding using entangled inputs. *Phys. Rev. Lett.*, 103(7):070504, Aug 2009. Long version on arXiv:0903.2838v1 [quant-ph].

[KWW09] Robert König, Stephanie Wehner, and Jürg Wullschleger. Unconditional security from noisy quantum storage. arXiv:0906.1030v2 [quant-ph], June 2009.

[May97] Dominic Mayers. Unconditionally Secure Quantum Bit Commitment is Impossible. *Physical Review Letters*, 78(17):3414–3417, 1997. Online available at `http://arxiv.org/abs/quant-ph/9605044`.

[NC00] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information.* Cambridge University Press, 2000.

[RS60] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society of for Industrial and Applied Mathematics*, 8(2):300–304, 1960.

[Unr04] Dominique Unruh. Simulatable security for quantum protocols, September 2004. Online available at `http://arxiv.org/ps/quant-ph/0409125`.

[Unr10] Dominique Unruh. Universally composable quantum multi-party computation. In *Eurocrypt 2010*, LNCS, pages 486–505. Springer, 2010. Preprint on arXiv:0910.2912 [quant-ph].

[Unr11] Dominique Unruh. Concurrent composition in the bounded quantum storage model. In Kenneth G. Paterson, editor, *Eurocrypt 2011*, volume 6632 of *LNCS*, pages 467–486. Springer, 2011.

[Wul07] Jürg Wullschleger. *Oblivious-Transfer Amplification.* PhD thesis, ETH Zurich, March 2007. arXiv:cs/0608076v3 [cs.CR].

[WW06] Stefan Wolf and Jürg Wullschleger. Oblivious transfer is symmetric. In Serge Vaudenay, editor, *Eurocrypt 2006*, volume 4004 of *LNCS*, pages 222–232. Springer, 2006.

[WW08] Stephanie Wehner and Jörg Wullschleger. Composable security in the bounded-quantum-storage model. In *ICALP 2008, track C*, LNCS, pages 604–615. Springer, 2008. Full version available at `http://arxiv.org/abs/0709.0492v1`.

Index