Superposition Attacks on Cryptographic Protocols

Ivan Damgård, Jakob Funder, Jesper Buus Nielsen, Louis Salvail

Dept. of Computer Science, Aarhus University, Université de Montreal

Abstract. Attacks on classical cryptographic protocols are usually modeled by allowing an adversary to ask queries from an oracle. Security is then defined by requiring that as long as the queries satisfy some constraint, there is some problem the adversary cannot solve, such as compute a certain piece of information. In this paper, we introduce a fundamentally new model of quantum attacks on classical cryptographic protocols, where the adversary is allowed to ask several classical queries in quantum superposition. This is a strictly stronger attack than the standard one, and we consider the security of several primitives in this model. We show that a secret-sharing scheme that is secure with threshold t in the standard model is secure against superposition attacks if and only if the threshold is lowered to $t/2$. We use this result to give zero-knowledge proofs for all of NP in the common reference string model. While our protocol is classical, it is sound against a cheating unbounded quantum prover and computational zero-knowledge even if the verifier is allowed a superposition attack. Finally, we consider multiparty computation and show that for the most general type of attack, simulation based security is not possible. However, putting a natural constraint on the adversary, we show a non-trivial example of a protocol that can indeed be simulated.

1 Introduction

Attacks on classical cryptographic protocols are usually modeled by allowing an adversary to ask queries from an oracle, for instance the adversary specifies a subset of players he wants to corrupt, and gets back their views of the protocol. Security is then defined by requiring that as long as the queries satisfy some constraint (for instance, the corrupted subset is not too large), there is some problem the adversary cannot solve, such as compute a certain piece of information.

Several previous works consider what happens to security if we allow the adversary to be quantum. The model usually considered is that the adversary is now a quantum machine, but otherwise plays exactly the same game as in a classical attack, i.e., he still communicates classically with the protocol he attacks. One example of this is the work of Watrous, showing that a large class of zero-knowledge protocols are also zero-knowledge against a quantum verifier.

It is natural to ask why we constrain a quantum adversary to communicate classically during the attack? The standard answer to this is that since honest players are classical, they would (implicitly) be doing a measurement of anything they receive, thus forcing a collapse of any quantum state they are given.

An important point, however, is that the assumption about honest players being classical is not always justified, *even if the protocol is supposed to be classical*: in the future, honest players may use quantum computing, just to speed up their local computation, even if they sometimes communicate classically. Furthermore, future usage of quantum cryptography will imply that players sometimes communicate quantumly (to do quantum key distribution) and sometimes classically. Finally, one should consider the case where a classical protocol is used as a subrutine for a protocol that handles quantum data. This is exactly what happens in the work of Ben-Or et al. [BCG$^+$05], where classical multiparty computation is used as a tool to obtain quantum multiparty computation.

Now, if a quantum adversary is attacking honest players that use quantum computing or even quantum communication themselves, it does not seem justified to assume that he can only communicate classically with them. Indeed, as an example, consider a zero-knowledge protocol where

the prover is implemented as a small quantum device sitting inside a mobile unit, say a PDA or a smart-phone. If an adversary gets hold of the unit, he may not be able to break in and directly read the prover's secret. But he can try to subject the device to unusual physical conditions, say by cooling it down and in this way perhaps be able to communicate quantumly with the prover, even if the device was not designed for this in the first place.

In this paper, we therefore introduce a new model of quantum attacks on classical cryptographic protocols, where the adversary is allowed to ask several classical queries in quantum superposition. In more concrete terms, we ask, for multiparty protocols: what happens if the adversary can be in superposition of having corrupted several different subsets? or, for zero-knowledge protocols: what happens if a quantum verifier can be in superposition of having issued several different challenges to the prover? As we argued above, we believe such superposition attacks to be a valid physical concern, but they also form a very natural generalization from a theory point of view: in the literature on black-box quantum computing, quantum black-box access to a function is usually defined by extending classical black-box access such that queries are allowed to contain several inputs in superposition. Our superposition attacks extend classical attacks in the same way.

Superposition attacks are strictly stronger than the standard one, and we consider the security of several primitives in this model: We show that a secret-sharing scheme that is perfectly secure with threshold t in the standard model is perfectly secure against superposition attacks if and only if the adversary's superposition is constrained to contain subsets of size at most $t/2$. If this condition is not satisfied, not only does perfect security fail, we show examples where the adversary may even learn the secret with certainty.

We use the secret-sharing result to construct zero-knowledge proofs for all of NP in the common reference string (CRS) model. While our protocol is classical, it is sound against a cheating unbounded quantum prover and computational zero-knowledge even if the verifier is allowed a superposition attack. Since we use the CRS model, the reader may ask why we do not use existing protocols for non-interactive zero-knowledge (NIZK), where the prover just sends a single message to the verifier. In this way, the adversary would not get a chance to do a superposition attack. However, the most general assumption under which NIZK is known to be possible is existence of one-way permutations. They in turn are only known to be realizable under assumptions that are easily broken by a quantum adversary, such as factoring or discrete log. Therefore we do not consider NIZK a satisfactory solution.

Finally, we consider multiparty computation and we define a UC-style model for static and passive superposition attacks on classical MPC protocols. Given our result on secret-sharing schemes, it is natural to speculate that classical MPC protocols that are secure against t corruptions, are secure against superposition attacks corrupting $t/2$ players. The situation turns out to be more complicated, however: We show that for the model that gives the adversary the most power (and hence is the most hostile to the simulator), simulation based security is not possible at all. The adversary can put its query in a state that prevents the simulator from learning any information on the inputs and outputs of corrupted players. However, putting a natural constraint on the adversary, we show a non-trivial example of a protocol that can indeed be simulated. By non-trivial, we mean that although the protocol is secure against a classical attack, we can show that it cannot be proved secure against a superposition attack by simply running the classical simulator in superposition. We therefore come up with techniques that are "more quantum" to do the simulation. We give a (in completely classical terms) a characterization of the protocols that can be simulated using these techniques. The obtained simulators are not necessarily efficient, however.

Whether more general positive results hold in this constrained model remains an open question. Likewise, the very natural question of security of *quantum* protocols against superposition attacks

remains open. Note that in existing work on quantum multiparty computation [BCG+05], the adversary's choice of subset to corrupt is classical. The negative part of our result on secret sharing described above shows that the protocol from [BCG+05] is not secure against superpositions attacks as it stands.

2 Preliminaries

2.1 Notation and terminology

We will model players in protocols in two different ways: when we are not interested in computational limitations on players, a player will be specified by a series of unitary transforms where the i'th transform is done all qubits available to the player, after the i'th message has been received (in the form of a quantum register), and then some designated part of the storage is sent as the next outgoing message.

In cases where we want to bound the computational complexity of a player, we consider a players to be an infinite family of interactive quantum circuits, as in the model from [FS09], and then the complexity is circuit size.

2.2 Running functions in superposition

Consider any function, $f : X \to Y$ and a register of qubits, $|\psi\rangle = \sum_x \alpha_x |x\rangle|0\rangle \in \mathcal{H}_X \otimes \mathcal{H}_Y$, where $dim(\mathcal{H}_X) = |X|$ and $dim(\mathcal{H}_Y) = |Y|$. *Running* f on $|\psi\rangle$ means to apply the unitary transformation, U_f, such that $U_f \sum_x \alpha_x |x\rangle|0\rangle = \sum_x \alpha_x |x\rangle|f(x)\rangle$. In general the register in $\mathcal{H}_Y$, called the *response register*, can contain any superposition of values, not just 0. In this case, we have that, $U_f \sum_{x,a} \alpha_{x,a} |x\rangle|a\rangle = \sum_x \alpha_{x,a} |x\rangle|f(x) \oplus a\rangle$

3 Secret sharing

In (classical) secret sharing n parties are sharing some secret value $s \in \mathbb{S}$ using randomness $r \in \mathcal{R}$, where $\mathbb{S}$ and $\mathcal{R}$ is the set of possible secrets and randomness. We name the parties $P_1, \ldots, P_n$. Let $N = \{1, \ldots, n\}$. Each party, P_i, receives a *share* $v_i(s, r) \in \{0,1\}^k$, also called his *private view* . That is, $v_i : \mathbb{S} \times \mathcal{R} \to \{0,1\}^k$. For $A \subset N$, let $v_A(s, r) = \{v_i(s, r)\}_{i \in A}$ be the string containing the concatenation of views for parties P_i with $i \in A$. For convenience in the following we assume that each such string is padded, so that they have the same length regardless of the size of A. That is, $v_A : \mathbb{S} \times \mathcal{R} \to \{0,1\}^t$. An *adversary structure* G is a family of subsets $G \subset 2^N$. A secret sharing scheme is perfectly secure against classical G-attacks if for any $A \in G$, the distribution of $v_A(s, r)$ does not depend on s. The adversary structure of the secret sharing scheme is the maximal adversary structure F such that the scheme is perfectly secure against F attacks.

We'll model any passive attack on the scheme as an one-time query to an *corruption oracle*. The corruption oracle for a specific run of a secret sharing scheme $O(s, r, A) = v_A(s, r)$ is the function that on input A returns the private view of those parties. That is, $O : \mathbb{S} \otimes \mathcal{R} \otimes F \to \{0,1\}^t$.

3.1 Two-party bit sharing example

Before we give the full model for secret sharing we start with a small example. We consider the case of 2 parties sharing a single bit, $b \in \{0,1\}$ using a random bit $r \in \{0,1\}$. $N = \{1,2\}$, $F = \{(1),(2)\}$, $v_1(b,r) = b \oplus r$, $v_2(b,r) = r$. This scheme is trivially secure in the classical setting. In the follow we'll consider what happens if we allow the adversary to interact with the corruption oracle in superposition. As this is meant to introduce the concept, we'll cut a few corners in terms of technicality and and reserve that for the later sections.
Assuming some specific bit has been shared with some specific randomness, we can write the state of the parties as $|v_1(b,r)\rangle \in \mathcal{H}_2$ and $|v_2(b,r)\rangle \in \mathcal{H}_2$. Consider an adversary supplying the following input to the corruption oracle,

$$|\omega\rangle = \frac{1}{\sqrt{2}}(|1\rangle|0\rangle + |2\rangle|0\rangle)$$

The oracle will run on both of these input in superposition. The state the adversary receives will be a mixed state over different choices of randomness and secrets. We'll assume both of these are uniformly chosen and he'll hence receive,

$$\rho^{adv} = \sum_{b\in\{0,1\},r\in\{0,1\}} \frac{1}{4}\left|\psi_{b,r}^{adv}\right\rangle\left\langle\psi_{b,r}^{adv}\right|$$

where $|\psi_{b,r}^{adv}\rangle = \frac{1}{\sqrt{2}}(|1\rangle|v_1(b,r)\rangle + |2\rangle|v_2(b,r)\rangle) = \frac{1}{\sqrt{2}}(|1\rangle|b\oplus r\rangle + |2\rangle|r\rangle)$. Define the state the adversary sees for a specific secret, b, as $\rho_b^{adv} = \sum_{r\in\{0,1\}} \frac{1}{2}|\psi_{b,r}^{adv}\rangle\langle\psi_{b,r}^{adv}|$. We would consider our bit sharing scheme secure iff for all possible queries, $\rho_0^{adv} = \rho_1^{adv}$. However, note that,

$$\rho_b^{adv} = \sum_{r\in\{0,1\}} \frac{1}{2}\left|\psi_{b,r}^{adv}\right\rangle\left\langle\psi_{b,r}^{adv}\right| = \frac{1}{2}\sum_{A,A'\in\{1,2\}} |A\rangle\langle A'| \otimes \left(\sum_{r\in\{0,1\}} |v_A(b,r)\rangle\langle v_{A'}(b,r)|\right)$$

For $A = 1, A' = 2$, consider the submatrix, $\sum_{r\in\{0,1\}} |v_1(b,r)\rangle\langle v_2(b,r)| = \sum_{r\in\{0,1\}} |b\oplus r\rangle\langle b|$. It should be clear that $\sum_{r\in\{0,1\}} |0\oplus r\rangle\langle 0| \neq \sum_{r\in\{0,1\}} |1\oplus r\rangle\langle 1|$ and hence $\rho_0^{adv} \neq \rho_1^{adv}$, which means the scheme is not secure if we allow the adversary to run the corruption oracle in superposition. This is not surprising since we know that using the Deutch-Josza algorithm you can actually distinguish between two such shares perfectly. Note that to do it perfectly it would require the use of a superposition of values for the response registers.

3.2 Model for secret sharing

We'll now give the full technical description of the model for superposition attacks on general secret sharing. To do this we first consider the state spaces needed to run the protocol and the attack on the protocol. First is the space that contains the shares for all the parties, $\mathcal{H}_{parties}$. The state in the register for this space is unchanged throughout the attack and is,

$$|parties\rangle_p = \sum_{s\in\mathbb{S},r\in\mathcal{R}} \sqrt{p_s}\sqrt{p_r}|s,r,v_N(s,r)\rangle_p = \sum_{s\in\mathbb{S},r\in\mathcal{R}} \sqrt{p_s}\sqrt{p_r}|s,r\rangle \bigotimes_i |v_i(s,r)\rangle \in \mathcal{H}_{parties}$$

where $|s, r\rangle$ is the purification of the secret and randomness choice. This is purely for technical reasons and does not matter for the adversary as he never sees it (and hence they might as well be considered measured). Next is the space for the environment, $\mathcal{H}_{env}$, which the adversary can use to choose his query and use as potential auxiliary register. The initial state for the environment is a general (pure) state,

$$|\psi\rangle_e = \sum_x \alpha_x |x\rangle_e \in \mathcal{H}_{env}$$

Finally is the space holding the adversary's query to the corruption oracle, $\mathcal{H}_{query}$. This is initially a 'blank' state,

$$|\omega\rangle_q = |0, 0\rangle_q \in \mathcal{H}_{query}$$

The space for the entire model is hence, $\mathcal{H}_{total} = \mathcal{H}_{parties} \otimes \mathcal{H}_{env} \otimes \mathcal{H}_{query}$ and the intial state is,

$$|init\rangle_t = \sum_{s\in\mathbb{S}, r\in\mathcal{R}} \sqrt{p_s}\sqrt{p_r}|s, r, v_N(s, r)\rangle_p \otimes \sum_x \alpha_x |x\rangle_e \otimes |0, 0\rangle_q \in \mathcal{H}_{total}$$

The attack will be defined by two operations and an adversary structure F. First the adversary needs to construct his query for the oracle. This includes choosing the superposition of subsets he'll corrupt and associated values for the response registers. This is an arbitrary unitary operation. We'll denote it, $U^{adv,F}_{query}$,

$$U^{adv,F}_{query} : \mathcal{H}_{env} \otimes \mathcal{H}_{query} \to \mathcal{H}_{env} \otimes \mathcal{H}_{query}$$

After this operation the state is,

$$|query\rangle_t = U^{adv,F}_{query}|init\rangle_t = \sum_{s\in\mathbb{S}, r\in\mathcal{R}} \sqrt{p_s}\sqrt{p_r}|s, r, v_N(s, r)\rangle_p \otimes \sum_{x,A\in F, a\in\{0,1\}^t} \alpha_{x,A,a}|x\rangle_e \otimes |A, a\rangle_q \in \mathcal{H}_{total}$$

where we assume it is the identity on $\mathcal{H}_{parties}$ Next the oracle, $O(s, r, A)$, is run. Let $U_{O(s,r,A)}$ denote the unitary applying this function. The state afterwards is,

$$|final\rangle_t = U_{O(s,r,A)}|query\rangle_t =$$
$$\sum_{s\in\mathbb{S}, r\in\mathcal{R}} \sqrt{p_s}\sqrt{p_r}|s, r, v_N(s, r)\rangle_p \otimes \sum_{x,A\in F, a\in\{0,1\}^t} \alpha_{x,A,a}|x\rangle_e \otimes |A, a + v_A(s, r)\rangle_q \in \mathcal{H}_{total}$$

were we, again, assume $U_{O(s,r,A)}$ is padded with appropriate identities. Consider the final state the adversary sees for a specific secret, s,

$$\rho^{adv,F}_s = \sum_{r\in\mathcal{R}} \left|\psi^{adv,F}_r\right\rangle\left\langle\psi^{adv,F}_r\right|$$

where $|\psi^{adv,F}_r\rangle = \sum_{x,A\in F, a\in\{0,1\}^t} \alpha_{x,A,a}|x\rangle_e \otimes |A, a + v_A(s, r)\rangle_q$.

Definition 1. *A secret sharing scheme S is* perfectly secure against superposition F-attacks *if, and only if, for all unitary matrices, $U^{adv,F}_{query} : \mathcal{H}_{env} \otimes \mathcal{H}_{query} \to \mathcal{H}_{env} \otimes \mathcal{H}_{query}$ and all possible pairs of inputs, $s, s' \in \mathbb{S}$,*

$$\rho^{adv,F}_s = \rho^{adv,F}_{s'}$$

For an adversary structure F, we define $F^2 = \{A|\ A = B \cup C \text{ where } B, C \in F\}$.

Theorem 1. *Let G be the classical adversary structure of $\mathcal{S}$. $\mathcal{S}$ is perfectly secure against superposition F-attacks if and only if $F^2 \subseteq G$.*

Proof. For the forward direction, consider the adversary's final state,

$$\begin{aligned}\rho_s^{adv} &= \sum_{r\in\mathcal{R}} p_r \left|\psi_r^{adv}\right\rangle\!\left\langle\psi_r^{adv}\right| \\ &= \sum_{r\in\mathcal{R},x,x',A,A\in F,a,a'\in\{0,1\}^t} p_r \alpha_{x,A,a}\alpha^*_{x',A',a'} |x\rangle_e\langle x|_e \otimes |A, a+v_A(s,r)\rangle_q\langle A', a'+v_{A'}(s,r)\big|_q\end{aligned}$$

Now, for any fixed A, A', a, a' and s, consider the matrix $\sum_{r\in\mathcal{R}} p_r |A, a+v_A(s,r)\rangle_q\langle A', a'+v_{A'}(s,r)|_q$. The crucial observation now is that this matrix is in 1-1 correspondence with the joint distribution of $v_A(s,r)$ and $v_{A'}(s,r)$. Namely, its entries are indexed by pairs of strings (α,β), where α, β are strings of the same length. And furthermore the (α,β)'th entry is the probability that the events $v_A(s,r) = \alpha + a$ and $v_{A'}(s,r) = \beta + a'$ occur simultaneously. Now, if $F^2 \subseteq G$, we have that $\mathcal{S}$ is perfectly secure against classical F^2-attacks. Therefore the joint distribution of $v_A(s,r)$ and $v_{A'}(s,r)$ does not depend on s, consequently each matrix $\sum_{r\in\mathcal{R}} p_r |A, a+v_A(r,s)\rangle_q\langle A', a'+v_{A'}(s,r)|_q$ is independent of s as well. Hence $\forall s,s' \in \mathbb{S} : \rho_s^{adv,F} = \rho_{s'}^{adv,F}$ as required.

For the only-if part, assume for contradiction that $F^2 \not\subseteq G$, i.e., there exist A_0, A_1 such that $A_0 \cup A_1 \notin G$. It follows that a secret shared using $\mathcal{S}$ is uniquely determined from shares in $A_0 \cup A_1$. Then consider the query $|\omega_\alpha\rangle = (|A\rangle|0\rangle + |A'\rangle|0\rangle)/\sqrt{2}$. By the same computation as above, we see that ρ_s^{adv} contains a submatrix of form $\sum_{r\in\mathcal{R}} |a_A + v_A(s,r)\rangle\langle a_{A'} + v_{A'}(s,r)|$, that corresponds to the joint distribution of shares in A and A'. But since the secret is uniquely determined from these shares, it follows that this submatrix is different for different secrets, and hence we get that there exists a measurement with non-zero bias towards the secret, and so $\mathcal{S}$ is not perfectly secure against quantum F-attacks. This is exactly the result we saw in the small example in Section **??**.

3.3 Simplified models for secret sharing

When formalizing superposition attacks on secret sharing we need to consider if we allow the adversary to use different values for the response register. The choice provably make a difference for the strength of the model, and both options can be justified from a physical perspective. We therefore cover both, models. When the adversary runs a classical component in superposition, then the reply will in general be a superposition. This opens the question of how the reply is delivered. In quantum information processing, it is customary that the result is xor'ed onto a response register a supplied along with the input. I.e., for a function f on is given a box which on classical input $|x\rangle|a\rangle$, the output is $|x\rangle|a \oplus f(x)\rangle$. This is convenient, as it is invertible, so the action of the box on a superposition is given by its actions on the classical inputs. This approach is reasonable in quantum information processing, as one is typically designing the boxes one self. If f is a database one can simply design the quantum version to supply the output by xor'ing it onto a response register. Consider, however, the prover in a zero-knowledge proof, which is sent a challenge e and then sends back $z(e)$. Even though the prover might be tricked into running on a superposition without noticing it, it does not seem reasonable that the prover would not notice it if we sent along a response register and asked her to xor her resply onto this register. In such a setting it seems more reasonable that the box/prover creates the response registers and returns them to the attacker/verifier. We model the setting of *created response registers* by restricting the more general setting og *supplied response registers* by allowing only $a = 0$. In that case the response from the box would be $|x\rangle|f(x)\rangle$.

3.4 Attacks on Secret Sharing

Lemma 1. *Consider a two-party Shamir secret sharing scheme $\mathcal{S}$. For any two secrets $s, s' \in \mathbb{S}$: $s \neq s'$, there exists a query with $a = 0$ that will allow an adversary to distinguish between the two with probability at least p_{guess}, where*

$$p_{guess} \geq \frac{3}{4}$$

Proof. The adversary will need no auxiliary register, so let his state simply be the query register. He constructs the following (pure state) query

$$|\omega\rangle = \frac{1}{\sqrt{2}}(|A_0, 0\rangle + |A_1, 0\rangle)$$

where $|\omega\rangle \in \mathcal{H}_{query}$. The final state the adversary sees for different secrets is then,

$$\rho_s^{adv} = \sum_{r \in \mathcal{R}} p_r \left|\psi_r^{adv}\right\rangle\!\left\langle\psi_r^{adv}\right|$$

where $|\psi_r^{adv}\rangle = \sum_{A \in \{A_0, A_1\}} \frac{1}{\sqrt{2}} |A, v_A(s, r)\rangle_q$ and $r \in \mathcal{R}$.

It is well-known that the adversary's probability of distinguishing between two such states, ρ_{adv}^s and $\rho_{adv}^{s'}$, is $p_{guess} = \frac{1}{2} + \frac{1}{4} \times |\rho_{adv}^s - \rho_{adv}^{s'}|_{Tr}$, where $|\ldots|_{Tr}$ denotes the trace norm. Define the difference between the two states as the matrix Δ.

$$\begin{aligned}
\Delta &= \rho_{adv}^s - \rho_{adv}^{s'} \\
&= \frac{1}{2} \sum_{A, A' \in \{A_0, A_1\}, r \in \mathcal{R}} p_r |A, v_A(s, r)\rangle_q \langle A', v_{A'}(s, r)|_q - \frac{1}{2} \sum_{A, A' \in \{A_0, A_1\}, r} p_r \left|A, v_A(s', r)\right\rangle_q \left\langle A', v_{A'}(s', r)\right|_q \\
&= \frac{1}{2} \sum_{A, A' \in \{A_0, A_1\}} |A\rangle\langle A'| \left(\sum_{r \in \mathcal{R}} p_r |v_A(s, r)\rangle_q \langle v_{A'}(s, r)|_q - \frac{1}{2} \sum_{r \in \mathcal{R}} p_r \left|v_A(s', r)\right\rangle_q \left\langle v_{A'}(s', r)\right|_q \right)
\end{aligned}$$

Since the state for any party individually is independent of the secret we have that for all $s, s' \in \mathbb{S}$

$$\sum_{r \in \mathcal{R}} |v_A(s, r)\rangle\langle v_A(s, r)| = \sum_{r \in \mathcal{R}} \left|v_A(s', r)\right\rangle\left\langle v_A(s', r)\right|$$

which means we only need to consider $A, A' \in \{A_0, A_1\} | A \neq A'$.

$$\begin{aligned}
\Delta = &\frac{1}{2} |A_0\rangle\langle A_1| \otimes \left(\sum_{r \in \mathcal{R}} p_r |v_{A_0}(s, r)\rangle_q \langle v_{A_1}(s, r)|_q - \frac{1}{2} \sum_{r \in \mathcal{R}} p_r \left|v_{A_0}(s', r)\right\rangle_q \left\langle v_{A_1}(s', r)\right|_q \right) \\
&+ \frac{1}{2} |A_1\rangle\langle A_0| \otimes \left(\sum_{r \in \mathcal{R}} p_r |v_{A_1}(s, r)\rangle_q \langle v_{A_0}(s, r)|_q - \frac{1}{2} \sum_{r \in \mathcal{R}} p_r \left|v_{A_1}(s', r)\right\rangle_q \left\langle v_{A_0}(s', r)\right|_q \right)
\end{aligned}$$

Define the two submatrices:

$$S = \sum_{r \in \mathcal{R}} p_r |v_{A_0}(s, r)\rangle_q \langle v_{A_1}(s, r)|_q - \frac{1}{2} \sum_{r \in \mathcal{R}} p_r \left|v_{A_0}(s', r)\right\rangle_q \left\langle v_{A_1}(s', r)\right|_q \tag{1}$$

$$S^\dagger = \sum_{r\in\mathcal{R}} p_r |v_{A_1}(s,r)\rangle_q \langle v_{A_0}(s,r)|_q - \frac{1}{2}\sum_{r\in\mathcal{R}} p_r \left|v_{A_1}(s',r)\right\rangle_q \left\langle v_{A_0}(s',r)\right|_q$$

such that Δ is the 2×2^t by 2×2^t matrix

$$\Delta = \frac{1}{2}\begin{pmatrix} 0 & S \\ S^\dagger & 0 \end{pmatrix} \tag{2}$$

It is well-known that if Δ is of the form (2) and $\frac{1}{2}S$ has singular values $s_1 \geq ... \geq s_p$ then Δ has eigenvalues $\pm s_1, ..., \pm s_p$. Since Δ is Hermitian, the trace norm is the sum of the absolute eigenvalues. From this we conclude that $|\Delta|_{Tr} = |S|_{Tr}$ and we can reduce our problem to that of finding the trace norm of S. Let

$$\begin{aligned} S &= M_s - M_{s'} \\ M_s &= \sum_{r\in\mathcal{R}} p_r |v_{A_0}(s,r)\rangle\langle v_{A_1}(s,r)| \end{aligned}$$

Note that for the state to be normalized it must be that

$$\sum_{i,j\in\{0,1\}^t} [M_s]_{i,j} = \sum_{i,j\in\{0,1\}^t, r} p_r \delta_{v_{A_0}(s,r),i}\delta_{v_{A_1}(s,r),j} = 1$$

Now, define the matrix, $\tilde{M}_s$

$$\tilde{M}_s = \sum_{r\in\mathcal{R}} |v_{A_0}(s,r)\rangle\langle v_{A_1}(s,r)| + \sum_{i=|\mathcal{R}|}^{t^2-1} |i\rangle\langle i|$$

It's straight forward to see that $\tilde{M}_s\tilde{M}_s^T = \mathbb{I}$,

$$\begin{aligned} &\left(\sum_{r\in\mathcal{R}} |v_{A_0}(s,r)\rangle\langle v_{A_1}(s,r)| + \sum_{i=|\mathcal{R}|}^{t^2-1} |i\rangle\langle i|\right) \times \left(\sum_{r'\in\mathcal{R}} \left|v_{A_1}(s,r')\right\rangle\left\langle v_{A_0}(s,r')\right| + \sum_{i=|\mathcal{R}|}^{t^2-1} |i\rangle\langle i|\right) \\ &= \sum_{r,r'\in\mathcal{R}} |v_{A_0}(s,r)\rangle\left\langle v_{A_0}(s,r')\right|\langle v_{A_1}(s,r)|\left|v_{A_1}(s,r')\right\rangle + \sum_{i=|\mathcal{R}|}^{t^2-1} |i\rangle\langle i| \\ &= \sum_{r\in\mathcal{R}} |v_{A_0}(s,r)\rangle\langle v_{A_0}(s,r)| + \sum_{i=|\mathcal{R}|}^{t^2-1} |i\rangle\langle i| = \mathbb{I} \end{aligned}$$

Note that $\sum_{i=|\mathcal{R}|}^{t^2-1} |i\rangle\langle i|$ is simply used to pad the subspace to ensure that the matrix is unitary. It will not be of any importance in the following calculations and can simply be ignored. It is well-known that

$$|S|_{Tr} = max_U\{|Tr(SU)|\}$$

where U is any unitary matrix. In other words, any specific matrix U is going to give a lower bound on the trace norm. Both $\tilde{M}_s$ and $\tilde{M}_s^T$ are such unitary matrices,

$$\begin{aligned}
|S|_{Tr} &= max_U\{|Tr(SU)|\} \geq |Tr(S\tilde{M}_s^T)| \\
&= |Tr((M_s - M_{s'})\tilde{M}_s^T)| \\
&= |\sum_{i,j\in\{0,1\}^t} [M_s]_{i,j}[\tilde{M}_s]_{i,j} - \sum_{i,j\in\{0,1\}^t} [M_{s'}]_{i,j}[\tilde{M}_s]_{i,j}| \\
&= |1 - \sum_{i,j\in\{0,1\}^t} [M_{s'}]_{i,j}[\tilde{M}_s]_{i,j}|
\end{aligned}$$

Now note that

$$\sum_{i,j\in\{0,1\}^t} [M_{s'}]_{i,j}[\tilde{M}_s]_{i,j} = \sum_{i,j\in\{0,1\}^t,r,r'} p_r \delta_{v_{A_0}(s,r),i}\delta_{v_{A_1}(s,r),j}\delta_{v_{A_0}(s',r'),i}\delta_{v_{A_1}(s',r'),j}$$

However the pair $(v_{A_0}(s,r), v_{A_1}(s,r))$, uniquely defines s and hence the sum is 0 unless $s = s'$. Therefore for $s \neq s' : |S|_{Tr} \geq 1$.

$$p_{guess} = \frac{1}{2} + \frac{1}{4}|\Delta|_{Tr} = \frac{1}{2} + \frac{1}{4}|S|_{Tr} \geq \frac{1}{2} + \frac{1}{4} \times 1 = \frac{3}{4}$$

which completes the proof

4 Zero-Knowledge

In this section, we present a zero-knowledge proof for any NP problem in the common reference string model. The proof is sound for an unbounded prover (quantum or not) and is computationally zero-knowledge for a polynomially bounded quantum verifier, even if superposition attacks are allowed.

For the protocol, we need a commitment scheme with special properties: we require a *keyed* commitment scheme $\texttt{Commit}_{\texttt{pk}}$, where the corresponding public key $\texttt{pk}$ is generated by one of two possible key-generation algorithms: $\mathcal{G}_{\texttt{H}}$ or $\mathcal{G}_{\texttt{B}}$. For a key $\texttt{pkH}$ generated by $\mathcal{G}_{\texttt{H}}$, the commitment scheme $\texttt{Commit}_{\texttt{pkH}}$ is unconditionally hiding, whereas the other generator, $\mathcal{G}_{\texttt{B}}$, actually produces a key *pair* $(\texttt{pkB}, \texttt{sk})$, so that the secret key $\texttt{sk}$ allows to efficiently extract m from $\texttt{Commit}_{\texttt{pkB}}(m, r)$, and as such $\texttt{Commit}_{\texttt{pkB}}$ is unconditionally binding. Furthermore, we require that keys $\texttt{pkH}$ and $\texttt{pkB}$ produced by the two generators are computationally indistinguishable, for any family of polynomial size quantum circuits. We call such a commitment scheme a *dual-mode* commitment scheme. [1] As a candidate for implementing such a system, we propose the public-key encryption scheme of Regev [Reg05], which is based on a worst-case lattice assumption and is not known to be breakable even by (efficient) quantum algorithms. Regev does not explicitly state that the scheme has the property we need, but this is implicit in his proof that the underlying computational assumption implies semantic security.[2]

[1] The notions of dual-mode *cryptosystems* and of meaningful/meaningless encryptions, as introduced in [PVW08] and [KN08], are similar in spirit but differ slightly technically.

[2] The proof compares the case where the public key is generated normally to a case where it is chosen with no relation to any secret key. It is then argued that the assumption implies that the two cases are computationally indistinguishable, and that in the second case, a ciphertext carries essentially no information about the message. This argument implies what we need.

4.1 The Model

We now describe the framework for our protocol: the proof system is specified w.r.t. a language L, and we have a prover P and a verifier V, both are assumed classical (when playing honestly). They get as input a common reference string CRS chosen with a prescribed distribution σ and a string x. P and V interact and at the end V outputs *accept* or *reject*. The first two properties we require are standard: *Completeness:* if $x \in L$ and P, V follow the protocol, V outputs *accept* with probability 1. *Soundness:* if $x \notin L$ (but CRS is chosen according to σ) then for any prover P^*, V outputs *accept* with probability negligible (in the length of x) when interacting with P^* on input x and CRS.

For zero-knowledge, we extend the capabilities of a cheating verifier V^* so it may do a superposition attack For simplicity, we give our definition of superposition zero-knowledge only for 3-move public coin protocols, i.e., conversations are assumed to have the form (a, e, z), where e is a random challenge issued by the verifier. It is not hard to extend the definition but the notation becomes more cumbersome. First, V^* is assumed to be a quantum machine, and the protocol is executed as follows: V^* receives x, CRS and P's first message a. Now, in stead of sending a classical challenge e, V^* is allowed to send a query

$$\sum_{e,y} \alpha_{e,y} |e\rangle |y\rangle.$$

We assume the the prover will process the query following his normal algorithm in superposition, so the verifier will get the same two registers back, in state

$$\sum_{e,y} \alpha_{e,y} |e\rangle |y + z(x, e, \rho)\rangle,$$

where $z(x, e, \rho)$ is P's response to challenge e on input x and internal randomness ρ. Finally, V^* outputs 0 or 1. Let $p_{real}(x)$ be the probability that 1 is output. We say that the proof system is *superposition zero-knowledge* if there exists an polynomial time quantum machine, the simulator S, such that the following holds for any cheating verifier V^* and $x \in L$: S interacts with V^* on input x, and we let $p_{sim}(x)$ be the probability that V^* outputs 1. Then $|p_{real}(x) - p_{sim}(x)|$ is negligible (in the length of x).

Note that, as usual in the CRS model, S only gets x as input and may therefore generate the reference string itself.

4.2 The Protocol

We now describe the basic ideas behind our protocol: we will let the CRS contain the following: $\mathtt{pkB}, c = \mathtt{Commit}_{\mathtt{pkB}}(0), \mathtt{pkB}'$, where the public keys are both generated by $\mathcal{G}_{\mathtt{B}}$. Then, using a standard trick, we will let P show that either $x \in L$ or c contains a 1. Since of course the latter statement is false, P still needs to convince us that $x \in L$. The simulator, on the other hand, can construct a reference string where c does contain 1 and simulate by following the protocol. The CRS will look the same to the verifier so we just need that the change of witness used is not visible in the proof, i.e., the proof should be witness indistinguishable. In this way, we can simulate without rewinding, and this allows V^* to be quantum.

However, standard techniques for witness indistinguishability are not sufficient to handle a superposition attack. For this, we need to be more specific about the protocol: a first attempt (which does not give us soundness) is that P will secret-share his witness w (where for the honest prover, w will be a witness for $x \in L$), to create shares $s_1, ..., s_n$ where we assume the scheme has t-privacy.

Then P's first message is a set of commitments $a = (\texttt{Commit}_{\texttt{pkB}'}(s_1, r_1), ..., \texttt{Commit}_{\texttt{pkB}'}(s_n, r_n))$. The verifier's challenge e will point out a random subset of the commitments, of size $t/2$, and the prover opens the commitments requested. Intuitively, this is zero-knowledge by Theorem 1: since we limit the number of shares the verifier can ask for to half the threshold of the secret sharing scheme, the state V^* gets back contains no information on the secret w.

On the other hand, this protocol is of course not sound, the verifier cannot check that the prover commits to meaningful shares of anything. To solve this, we make use of the "MPC in the head" technique from [IKOS09]: Here, we make use of an n-party protocol in which the witness w is secret-shared among the player, and a multiparty computation is done to check whether w is correct with respect to claim on the the public input, namely in our case $x \in L$ and the c from the CRS contains 1. Finally all players output *accept* or *reject* accordingly. It is assumed that the protocol is secure against active corruption of t players where t is $\Theta(n)$. We will call this protocol $\pi_{L,CRS}$ in the following. Several examples of suitable protocols can be found in [IKOS09]. In their construction, the prover emulates an execution of π in his head, and we let $v_{\pi_{L,CRS}}(i, \rho)$ denote he view of virtual player i, where ρ is the randomness used. The prover then commits to $v_{\pi_{L,CRS}}(i, \rho)$, for $i = 1...n$ and the verifier ask the prover to open t randomly chosen views that are checked for consistency and adherence to $\pi_{L,CRS}$. It is shown in [IKOS09] that if no valid witness exists for the public input, then the verifier will detect an error with overwhelming probability.

Now, observe that the process of emulating π can be thought of as a secret sharing scheme, where the prover's witness w is shared and each $v_\pi(i, \rho)$ is a share: indeed any t shares contain no information on w by t-privacy of the protocol. Therefore combining this with our rudimentary idea from before gives us the solution.

Superposition-secure zero-knowledge proof for any NP-language L.
The public input is x, of length k bits. The distribution σ generates the common reference string as $\texttt{pkB}, c = \texttt{Commit}_{\texttt{pkB}}(0), \texttt{pkB}'$, where the public keys are both generated by $\mathcal{G}_{\texttt{B}}$ on input 1^k.

1. The prover P emulates $\pi_{L,CRS}$ to generate $v_{\pi_{L,CRS}}(i, \rho)$ and sends $\texttt{Commit}_{\texttt{pkB}'}(v_{\pi_{L,CRS}}(i, \rho), r_i)$, for $i = 1...n$, to the verifier V.
2. V sends a challenge e designating a random subset of the commitments of size $t/2$.
3. P opens the commitments designated by e, V checks the opened views according to the algorithm described in [IKOS09], and accepts or rejects according to the result.

Theorem 2. *If $(\mathcal{G}_{\texttt{B}}, \mathcal{G}_{\texttt{H}}, \texttt{Commit})$ form a secure dual-mode commitment scheme, then the above protocol is complete, sound and superposition zero-knowledge.*

Proof. Completeness is trivial by inspection of the protocol. Soundness follows immediately from the soundness proof in [IKOS09], we just have to observe that the fact that the prover opens $t/2$ and not t views makes no difference, in fact the proof holds as long as $\Theta(n)$ views are opened. For zero-knowledge, we describe a simulator S: It will generate a common reference string as $\texttt{pkH}, c = \texttt{Commit}_{\texttt{pkH}}(1), \texttt{pkH}'$ where both public keys are generated by $\mathcal{G}_{\texttt{H}}$ on inout 1^k. It then plays the protocol with V^*, answering its quantum queries by following the protocol. This is possible since c now contains a 1, so S knows a valid witness. To show that V^* cannot distinguish simulation from protocol, we define series of games

Game 0 The protocol as described above, but where P talks to V^* doing a superposition attack.
Game 1 As Game 0, but the CRS is generated as $\texttt{pkH}, c = \texttt{Commit}_{\texttt{pkH}}(0), \texttt{pkB}'$ where pkH is generated by $\mathcal{G}_{\texttt{H}}$ and $\texttt{pkB}'$ is generated by $\mathcal{G}_{\texttt{B}}$.
Game 2 As Game 1, but the CRS is generated as $\texttt{pkH}, c = \texttt{Commit}_{\texttt{pkH}}(1), \texttt{pkH}'$ where both public keys are generated by $\mathcal{G}_{\texttt{H}}$.

Game 3 As Game 3, but the P uses as witness the fact that c contains a 1.

Now, Game 0 and Game 1 are computationally indistinguishable by assumption on the dual-mode commitment scheme, and the same is true for Game 1 and Game 2. Game 2 and Game 3 are statistically indistinguishable by Theorem 1 and the fact that commitments done using $\texttt{pkH}'$ are statistically hiding. Finally, note that Game 3 is exactly the same game as the simulation.

5 Multiparty computation

In this section we consider the models for MPC protocols. A classical passive attack on a multiparty computation protocol looks a lot like the attacks on secret sharing: you query for a subset and get back the party's entire view of the protocol. Of course, you can generalize this to a quantum attack in the same way. And you can ask if there is some adversary structure for which the protocol would be secure against such an attack, assuming classical security.

Security for MPC protocols is usually defined as an adversary's ability to distinguish between an attack in the *real world* where he's allowed access to a corruption oracle of some subset of the parties and an *ideal world* where the attack is *simulated* towards the adversary using the *ideal functionality* of the protocol. We hence need to describe both models for the real and for the ideal world. They'll need different spaces and different operations to execute. As before, we have n parties running the protocol. We name the parties $P_1, \ldots, P_n$. Let $N = \{1, \ldots, n\}$. An adversary structure is $F \subset 2^N$. Each party, P_i, has private input, $s_i \in \mathbb{S}_i$ and chooses randomness, $r_i \in \mathcal{R}_i$. $s \in \mathbb{S}$ and $r \in \mathcal{R}$ denotes the concatenation of each of these. $v_i(s, r)$ and $o_i(s, r)$ is the private view and output for the party P_i, when the protocol has been run on inputs s and using randomness r. Note these are functions and not general quantum operations as the parties are expected to run the protocol honestly. For $A \subset N$, let $v_A(s, r) = \{v_i(s, r)\}_{i \in A}$, $s_A = \{s_i\}_{i \in A}$ $o_A(s, r) = \{o_i(s, r)\}_{i \in A}$ and be strings containing the concatenation of views, input and output for parties P_i with $i \in A$. For convenience in the following we assume that each such string is padded, so that they have the same length (t bits) regardless of the size of A.

5.1 MPC model in the 'Real world'

First we will consider the case of running and attacking the protocol in the real world. As earlier, all actions taken by the parties and the adversary will be considered purified so the overall state remains pure throughout. Consider the following space,

$$\mathcal{H}_{total} = \mathcal{H}_{parties} \otimes \mathcal{H}_{in} \otimes \mathcal{H}_{out} \otimes \mathcal{H}_{env} \otimes \mathcal{H}_{query}$$

$\mathcal{H}_{parties}$ contains the private views and purification of the randomness for the parties.
$\mathcal{H}_{in}$ contains the input the parties will use to run the protocol.
$\mathcal{H}_{out}$ is where the output will be stored after running the protocol.
$\mathcal{H}_{env}$ is the environment and is used to store auxiliary input and any auxiliary register needed by the adversary. The dimension is therefore arbitrary, though finite.
$\mathcal{H}_{query}$ is where the query to, and response from, the oracle will be stored.
Each of these subspaces are, of course, of appropriate (and finite) dimension.

We will break the superposition run, and attack, of a MPC protocol in the real world down into five operations. After each operation we will consider the change to the description of the state. In

the beginning all the registers are blank (ie. have value 0), except the environment, which might contain some (purified) auxiliary input for the adversary. The initial state is hence,

$$|init\rangle_t^{rw} = \sum_x \alpha_x |0\rangle_p |0\rangle_i |0\rangle_o |x\rangle_e |0\rangle_q$$

where $|init\rangle_t^{rw} \in \mathcal{H}_{total}$, $|0\rangle_p \in \mathcal{H}_{parties}$, $|0\rangle_i \in \mathcal{H}_{in}$, $|0\rangle_o \in \mathcal{H}_{out}$, $|0\rangle_e \in \mathcal{H}_{env}$ and $|0\rangle_q \in \mathcal{H}_{query}$, as should be expected from the notation. The superscript, rw, specifies that it's in the real world. The first operation is applied by the adversary and supplies the inputs to the parties. This is an arbitrary unitary operation. We'll denote it, U_{in}^{adv},

$$U_{in}^{adv} : \mathcal{H}_{in} \otimes \mathcal{H}_{env} \to \mathcal{H}_{in} \otimes \mathcal{H}_{env}$$

The result of this is that the input registers are now filled. These are in superposition over all possible inputs. The state after the first operation is therefore,

$$|1\rangle_t^{rw} = \sum_{x,s} \alpha_{x,s} |0\rangle_p |s\rangle_i |0\rangle_o |x\rangle_e |0\rangle_q$$

The protocol is now run honestly without intervention from the adversary. This is a classical function run in superposition of the possible inputs and produces a corresponding superposition of private views and outputs for the parties. We'll denote this operation, U_{run}^{pro},

$$U_{run}^{pro} : \mathcal{H}_{parties} \otimes \mathcal{H}_{in} \otimes \mathcal{H}_{out} \to \mathcal{H}_{parties} \otimes \mathcal{H}_{in} \otimes \mathcal{H}_{out}$$

Recall that we are purifying all actions, hence also the choice of randomness when the protocol is run. The state after the second operation is therefore,

$$|2\rangle_t^{rw} = \sum_{x,s,r} \alpha_{x,s} \sqrt{p_r} |v_N(s,r)\rangle_p |s\rangle_i |o_N(s,r)\rangle_o |x\rangle_e |0\rangle_q \tag{3}$$

Next the adversary needs to construct his query to the oracle. This includes choosing the superposition of subsets he'll corrupt and associated values for the response registers for input, output and view. For simplicity we'll sometimes refer to these three values by a. That is, $a = (a_i, a_o, a_v)$. This is an arbitrary unitary operation. We'll denote it, $U_{query}^{adv,F}$,

$$U_{query}^{adv,F} : \mathcal{H}_{env} \otimes \mathcal{H}_{query} \to \mathcal{H}_{env} \otimes \mathcal{H}_{query}$$

The state is now,

$$|3\rangle_t^{rw} = \sum_{x,s,r,A,a} \alpha_{x,s,A,a} \sqrt{p_r} |v_N(s,r)\rangle_p |s\rangle_i |o_N(s,r)\rangle_o |x\rangle_e |A,a\rangle_q$$

Next operation is applied by the oracle, that, for each corrupted subset in the query, fills the input, output and view into the response register supplied by the adversary. This is a classical function on each corrupted subset and view in the superposition which fills in the input, output and view into the response register. We'll denote this operation, U_{res}^{oracle},

$$U_{res}^{oracle} : \mathcal{H}_{parties} \otimes \mathcal{H}_{in} \otimes \mathcal{H}_{out} \otimes \mathcal{H}_{query} \to \mathcal{H}_{parties} \otimes \mathcal{H}_{in} \otimes \mathcal{H}_{out} \otimes \mathcal{H}_{query}$$

and the state after the fourth operation is therefore

$$|4\rangle_t^{rw} = \sum_{x,s,r,A,a} \alpha_{x,s,A,a}\sqrt{p_r}|v_N(s,r)\rangle_p|s\rangle_i|o_N(s,r)\rangle_o|x\rangle_e|A, a_i + s_A, a_o + o_A(s,r), a_v + v_A(s,r)\rangle_q$$

The adversary receives the response register and must now guess if he's in the real or ideal world. He can do this using the input register, his auxiliary register and the query register. This is an arbitrary quantum operation. We'll denote it, O_{dist}^{adv}

$$O_{dist}^{adv} : \mathcal{H}_{in} \otimes \mathcal{H}_{env} \otimes \mathcal{H}_{query} \to \{Real, Ideal\}$$

Except for the register holding his guess, the final state is of little interest. To see the adversary's final state we need to trace out the register holding the view of all the parties,

$$\rho_{adv}^{rw} = Tr_p(|4\rangle\langle 4|_t^{rw}) = \sum_{r,r',s,s'} \sqrt{p_r}\sqrt{p_{r'}}\left|\psi_{r,s}^{adv}\right\rangle\left\langle\psi_{r',s'}^{adv}\right| tr(|v_N(s,r)\rangle_p\langle v_N(s',r')|_p \otimes |o_N(s,r)\rangle_o\langle o_N(s',r')|_o) \quad (4)$$

$$= \sum_{r,r',s,s'} \sqrt{p_r}\sqrt{p_{r'}}\left|\psi_{r,s}^{adv}\right\rangle\left\langle\psi_{r',s'}^{adv}\right| \langle v_N(s,r)|_p|v_N(s',r')\rangle_p \times \langle o_N(s,r)|_o|o_N(s',r')\rangle_o) \quad (5)$$

$$= \sum_{r,s} p_r \left|\psi_{r,s}^{adv}\right\rangle\left\langle\psi_{r,s}^{adv}\right| \quad (6)$$

where $|\psi_{r,s}^{adv}\rangle = \sum_{x,A,a} \alpha_{x,s,A,a}|s\rangle_i|x\rangle_e|A, a_i + s_A, a_o + o_A(s,r), a_v + v_A(s,r)\rangle_q$. It's important to note that by letting the parties use his input register to run the protocol, this is now perfectly entangled with their views and outputs, which he has no access to, and has therefore increased the Von Neumann entropy he has on his own state.

We'll sum up these steps below

Inital state:

$$|init\rangle_t^{rw} = \sum_x \alpha_x |0\rangle_p |0\rangle_i |0\rangle_o |x\rangle_e |0\rangle_q$$

1.

$$U_{in}^{adv} : \mathcal{H}_{in} \otimes \mathcal{H}_{env} \to \mathcal{H}_{in} \otimes \mathcal{H}_{env}$$
$$|1\rangle_t^{rw} = \sum_{x,s} \alpha_{x,s} |0\rangle_p |s\rangle_i |0\rangle_o |x\rangle_e |0\rangle_q$$

2.

$$U_{run}^{pro} : \mathcal{H}_{parties} \otimes \mathcal{H}_{in} \otimes \mathcal{H}_{out} \to \mathcal{H}_{parties} \otimes \mathcal{H}_{in} \otimes \mathcal{H}_{out}$$
$$|2\rangle_t^{rw} = \sum_{x,s,r} \alpha_{x,s} \sqrt{p_r} |v_N(s,r)\rangle_p |s\rangle_i |o_N(s,r)\rangle_o |x\rangle_e |0\rangle_q$$

3.

$$U_{query}^{adv,F} : \mathcal{H}_{env} \otimes \mathcal{H}_{query} \to \mathcal{H}_{env} \otimes \mathcal{H}_{query}$$
$$|3\rangle_t^{rw} = \sum_{x,s,r,A,a} \alpha_{x,s,A,a} \sqrt{p_r} |v_N(s,r)\rangle_p |s\rangle_i |o_N(s,r)\rangle_o |x\rangle_e |A,a\rangle_q$$

4.

$$U_{res}^{oracle} : \mathcal{H}_{parties} \otimes \mathcal{H}_{in} \otimes \mathcal{H}_{out} \otimes \mathcal{H}_{query} \to \mathcal{H}_{parties} \otimes \mathcal{H}_{in} \otimes \mathcal{H}_{out} \otimes \mathcal{H}_{query}$$
$$|4\rangle_t^{rw} = \sum_{x,s,r,A,a} \alpha_{x,s,A,a} \sqrt{p_r} |v_N(s,r)\rangle_p |s\rangle_i |o_N(s,r)\rangle_o |x\rangle_e |A, a_i + s_A, a_o + o_A(s,r), a_v + v_A(s,r)\rangle_q$$

5.

$$O_{dist}^{adv} : \mathcal{H}_{in} \otimes \mathcal{H}_{out} \otimes \mathcal{H}_{env} \otimes \mathcal{H}_{query} \to \{Real, Ideal\}$$

Fig. 1. Purified run of multiparty computation in the real world with superposition attacks

5.2 MPC model in the 'Ideal world'

Secondly we consider the ideal world, where a simulator is to simulate a real attack for the adversary using only the ideal functionality. The space for this model is,

$$\mathcal{H}_{total} = \mathcal{H}_{idealF} \otimes \mathcal{H}_{in} \otimes \mathcal{H}_{out} \otimes \mathcal{H}_{sim} \otimes \mathcal{H}_{env} \otimes \mathcal{H}_{query}$$

$\mathcal{H}_{in}, \mathcal{H}_{out}$ and $\mathcal{H}_{env}$ serve the same purpose as in the real world.
$\mathcal{H}_{sim}$ denotes the subspace in which the simulator operates.
$\mathcal{H}_{query}$ is still the register where the adversary constructs his query, but the simulator will be allowed to change this before using it to query the ideal functionality. Finally, we have a space for the ideal functionality, $\mathcal{H}_{idealF}$. This is necessary as we want to purify the random choices made and we need the state of the ideal functionality to be entangled with the input register as the parties would be in the real world. We'll describe the operations that differ and refer to the earlier description for those that don't. There will be seven operations in total. The initial state is,

$$|init\rangle_t^{iw} = \sum_x \alpha_x |0\rangle_{if} |0\rangle_i |0\rangle_o |0\rangle_s |x\rangle_e |0\rangle_q$$

where $|init\rangle_t \in \mathcal{H}_{total}$, $|0\rangle_{if} \in \mathcal{H}_{idealF}$, $|0\rangle_i \in \mathcal{H}_{in}$, $|0\rangle_o \in \mathcal{H}_{out}$, $|0\rangle_s \in \mathcal{H}_{sim}$, $|0\rangle_e \in \mathcal{H}_{env}$ and $|0\rangle_q \in \mathcal{H}_{query}$, as should be expected from the notation. The superscript, iw, denotes that it's in the ideal world. The first operation is exactly as in the real world and hence,

$$U_{in}^{adv} : \mathcal{H}_{in} \otimes \mathcal{H}_{env} \to \mathcal{H}_{in} \otimes \mathcal{H}_{env}$$

$$|1\rangle_t^{iw} = \sum_{x,s} \alpha_{x,s} |0\rangle_{if} |s\rangle_i |0\rangle_o |0\rangle_s |x\rangle_e |0\rangle_q$$

The ideal functionality now entangles itself with the input register and produces the correct outputs in the output register. This is a classical function of each view in superposition. We'll denote this operation, U_{out}^{ideal},

$$U_{out}^{ideal} : \mathcal{H}_{idealF} \otimes \mathcal{H}_{in} \otimes \mathcal{H}_{out} \to \mathcal{H}_{idealF} \otimes \mathcal{H}_{in} \otimes \mathcal{H}_{out}$$

Recall that any random choices are purified onto $\mathcal{H}_{idealF}$. The resulting state is therefore,

$$|2\rangle_t^{iw} = \sum_{x,s,r} \alpha_{x,s} \sqrt{p_r} |s, r\rangle_{if} |s\rangle_i |o_N(s, r)\rangle_o |0\rangle_s |x\rangle_e |0\rangle_q$$

The adversary constructs his query exactly as earlier,

$$U_{query}^{adv,F} : \mathcal{H}_{env} \otimes \mathcal{H}_{query} \to \mathcal{H}_{env} \otimes \mathcal{H}_{query}$$

$$|3\rangle_t^{iw} = \sum_{x,s,r,A,a} \alpha_{x,s,A,a} \sqrt{p_r} |s, r\rangle_{if} |s\rangle_i |o_N(s, r)\rangle_o |0\rangle_s |x\rangle_e |A, a\rangle_q$$

Now the simulator gets the query and must construct an appropriate response to the adversary, only using the ideal functionality. First the simulator is allowed to change the adversary's query using an auxiliary register. Only requirement is that it is corruption preserving [3]. This is an arbitrary quantum operation. We'll denote it, U_{query}^{sim},

$$U_{query}^{sim} : \mathcal{H}_{sim} \otimes \mathcal{H}_{query} \to \mathcal{H}_{sim} \otimes \mathcal{H}_{query}$$

[3] That is, the probability of measuring a specific A is unchanged

$$|4\rangle_t^{iw} = \sum_{x,z,s,r,A,a} \alpha_{x,z,s,A,a}\sqrt{p_r}|s,r\rangle_{if}|s\rangle_i|o_N(s,r)\rangle_o|z\rangle_s|x\rangle_e|A,a\rangle_q$$

Now the ideal functionality is run on the query from the simulator. This is a classical function that each corrupted subset and each view in the superposition which fills in the input and output into the response register. We'll denote this operation, U_{res}^{ideal},

$$U_{res}^{ideal} : \mathcal{H}_{in} \otimes \mathcal{H}_{out} \otimes \mathcal{H}_{query} \rightarrow \mathcal{H}_{in} \otimes \mathcal{H}_{out} \otimes \mathcal{H}_{query}$$

and the response register now contains input and output from the corrupted parties, but, of course, not their views.

$$|5\rangle_t^{iw} = \sum_{x,z,s,r,A,a} \alpha_{x,z,s,A,a}\sqrt{p_r}|s,r\rangle_{if}|s\rangle_i|o_N(s,r)\rangle_o|z\rangle_s|x\rangle_e|A,a_i+s_A,a_o+o_A(s,r),a_v\rangle_q$$

Next the simulator must try to simulate the response the adversary got in the real world. We'll denote this operation, U_{res}^{sim},

$$U_{res}^{sim} : \mathcal{H}_{sim} \otimes \mathcal{H}_{query} \rightarrow \mathcal{H}_{sim} \otimes \mathcal{H}_{query}$$

The resulting state is

$$|6\rangle_t^{iw} = \sum_{x,z,s,r,A,a} \alpha'_{x,z,s,A,a}\sqrt{p_r}|s,r\rangle_{if}|s\rangle_i|o_N(s,r)\rangle_o|z\rangle_s|x\rangle_e|A,\psi_{z,s,A,a}\rangle_q$$

As earlier, the adversary must now guess which world he is in,

$$O_{dist}^{adv} : \mathcal{H}_{in} \otimes \mathcal{H}_{out} \otimes \mathcal{H}_{env} \otimes \mathcal{H}_{query} \rightarrow \{Real, Ideal\}$$

Finally, to see what state the adversary sees we must trace out the ideal functionality, the output and the simulator registers,

$$\begin{aligned}\rho_{adv}^{iw} = Tr_{if,o,s}(|6\rangle\langle 6|_t^{iw}) &= \sum_{r,r',s,s',z,z'} p_r \left|\psi_{r,s,z}^{adv}\right\rangle\left\langle\psi_{r',s',z'}^{adv}\right| tr\left(|s,r\rangle_{if}\langle s',r'|_{if} \otimes |o_N(s,r)\rangle_o\langle o_N(s',r')|_o \otimes |z\rangle_s\langle z'|_s\right) \\ &= \sum_{r,r',s,s',z,z'} p_r \left|\psi_{r,s,z}^{adv}\right\rangle\left\langle\psi_{r',s',z'}^{adv}\right| \left(\langle s,r|_{if}|s',r'\rangle_{if} \times \langle o_N(s,r)|_o|o_N(s',r')\rangle_o \times \langle z|_s|z'\rangle_s\right) \\ &= \sum_{r,s,z} p_r \left|\psi_{r,s,z}^{adv}\right\rangle\left\langle\psi_{r,s,z}^{adv}\right|\end{aligned}$$

where $|\psi_{r,s,z}^{adv}\rangle = \sum_{x,A,a} \alpha_{x,z,s,A,a}|s\rangle_i|x\rangle_e|A,\psi_{z,s,A,a}\rangle_q$. Again, we will sum up the steps below,

A MPC protocol is defined by the operator U_{run}^{pro} (which in turn also defines U_{out}^{ideal}). A specific adversay is defined by the initial state $\sum_x \alpha_a|x\rangle$ and the three operators, $U_{in}^{adv}, U_{query}^{adv,F}, O_{dist}^{adv}$. Finally, a simulator is defined by the two operators, $\{U_{query}^{sim}, U_{res}^{sim}\}$. This allows for the following definition.

Definition 2. *A pair of unitary operators, $\{U_{query}^{sim}, U_{res}^{sim}\}$, are a perfect black-box simulator for the MPC protocol defined by U_{run}^{pro} with adversary structure, F, if, and only if, for all auxiliary inputs, $\sum_x \alpha_x|x\rangle$, and all operators $U_{in}^{adv}, U_{query}^{adv,F}$,*

$$\rho_{adv}^{rw} = Tr_{p,o}(|4\rangle\langle 4|_t^{rw}) = \rho_{adv}^{iw} = Tr_{if,o,s}(|6\rangle\langle 6|_t^{iw})$$

Inital state:

$$|init\rangle_t^{iw} = \sum_x \alpha_x |0\rangle_{if} |0\rangle_i |0\rangle_o |0\rangle_s |x\rangle_e |0\rangle_q$$

1.

$$U_{in}^{adv} : \mathcal{H}_{in} \otimes \mathcal{H}_{env} \to \mathcal{H}_{in} \otimes \mathcal{H}_{env}$$
$$|1\rangle_t^{iw} = \sum_{x,s} \alpha_{x,s} |0\rangle_{if} |s\rangle_i |0\rangle_o |0\rangle_s |x\rangle_e |0\rangle_q$$

2.

$$U_{out}^{ideal} : \mathcal{H}_{idealF} \otimes \mathcal{H}_{in} \otimes \mathcal{H}_{out} \to \mathcal{H}_{idealF} \otimes \mathcal{H}_{in} \otimes \mathcal{H}_{out}$$
$$|2\rangle_t^{iw} = \sum_{x,s,r} \alpha_{x,s} \sqrt{p_r} |s, r\rangle_{if} |s\rangle_i |o_N(s, r)\rangle_o |0\rangle_s |x\rangle_e |0\rangle_q$$

3.

$$U_{query}^{adv,F} : \mathcal{H}_{env} \otimes \mathcal{H}_{query} \to \mathcal{H}_{env} \otimes \mathcal{H}_{query}$$
$$|3\rangle_t^{iw} = \sum_{x,s,r,A,a} \alpha_{x,s,A,a} \sqrt{p_r} |s, r\rangle_{if} |s\rangle_i |o_N(s, r)\rangle_o |0\rangle_s |x\rangle_e |A, a\rangle_q$$

4.

$$U_{query}^{sim} : \mathcal{H}_{sim} \otimes \mathcal{H}_{query} \to \mathcal{H}_{sim} \otimes \mathcal{H}_{query}$$
$$|4\rangle_t^{iw} = \sum_{x,z,s,r,A,a} \alpha_{x,z,s,A,a} \sqrt{p_r} |s, r\rangle_{if} |s\rangle_i |o_N(s, r)\rangle_o |z\rangle_s |x\rangle_e |A, a\rangle_q$$

5.

$$U_{res}^{ideal} : \mathcal{H}_{in} \otimes \mathcal{H}_{out} \otimes \mathcal{H}_{query} \to \mathcal{H}_{in} \otimes \mathcal{H}_{out} \otimes \mathcal{H}_{query}$$
$$|5\rangle_t^{iw} = \sum_{x,z,s,r,A,a} \alpha_{x,z,s,A,a} \sqrt{p_r} |s, r\rangle_{if} |s\rangle_i |o_N(s, r)\rangle_o |z\rangle_s |x\rangle_e |A, a_i + s_A, a_o + o_A(s, r), a_v\rangle_q$$

6.

$$U_{res}^{sim} : \mathcal{H}_{sim} \otimes \mathcal{H}_{query} \to \mathcal{H}_{sim} \otimes \mathcal{H}_{query}$$
$$|6\rangle_t^{iw} = \sum_{x,z,s,r,A,a} \alpha'_{x,z,s,A,a} \sqrt{p_r} |s, r\rangle_{if} |s\rangle_i |o_N(s, r)\rangle_o |z\rangle_s |x\rangle_e |A, \psi_{z,s,A,a}\rangle_q$$

7.

$$O_{dist}^{adv} : \mathcal{H}_{in} \otimes \mathcal{H}_{out} \otimes \mathcal{H}_{env} \otimes \mathcal{H}_{query} \to \{Real, Ideal\}$$

Fig. 2. Purified run of multiparty computation in the ideal world with superposition attacks

where

$$|4\rangle_t^{rw} = U_{res}^{oracle} U_{query}^{adv,F} U_{run}^{pro} U_{in}^{adv} |init\rangle_t^{rw}$$
$$|6\rangle_t^{iw} = U_{res}^{sim} U_{res}^{ideal} U_{query}^{sim} U_{query}^{adv,F} U_{out}^{ideal} U_{in}^{adv} |init\rangle_t^{iw}$$

where we for readability assume that the operators are padded with appropriate identities on the subspaces they don't operate.

5.3 No simulator for general MPC if U_{query}^{sim} is unitary

Consider a simple MPC protocol with a single dealer that deals a secret $s \in \{0,1\}$ to a number of parties. Denote this dealer as the party, P_1. The adversary will not need an auxiliary register in the following and will therefore be left out of the equations. Let the adversary make a query, defined by $U_{query}^{adv,F} : \mathcal{H}_{query} \to \mathcal{H}_{query}$, that is only of the dealer and one more party where he puts the response register for the input in perfect superposition. The complete state after applying $U_{query}^{adv,F}$ on the query register is,

$$|3\rangle_t^{rw} = \sum_{r,a_i} \frac{1}{\sqrt{|\mathbb{S}|}} \sqrt{p_r} |v_N(s,r)\rangle_p |s\rangle_i |o_N(s,r)\rangle_o |A, a_i, 0, 0\rangle_q$$

where $A = \{1,2\}$ The state after applying the oracle is

$$|4\rangle_t^{rw} = \sum_{r,a_i} \frac{1}{\sqrt{|\mathbb{S}|}} \sqrt{p_r} |v_N(s,r)\rangle_p |s\rangle_i |o_N(s,r)\rangle_o |A, a_i + s_A, 0, v_A(s,r)\rangle_q$$

If we look at the final part of the query register ($v_A(s,r)$) we can see that it is in a classical state and contains the view of the dealer and one other party, that is, the randomness and one secret share. This uniquely defines the secret and hence the query register must be orthogonal for two different secrets.
In the ideal world assume, for the time being, that U_{query}^{sim} is the identity. The state after the oracle for the ideal functionality is applied is then,

$$|5\rangle_t^{iw} = \sum_{r,a_i} \frac{1}{\sqrt{|\mathbb{S}|}} \sqrt{p_r} |s,r\rangle_{if} |s\rangle_i |o_N(s,r)\rangle_o |0\rangle_s |A, a_i + s_A, 0, 0\rangle_q$$

Since this is in perfect superposition over all values of a_i we can conclude that,

$$\begin{aligned} |5\rangle_t^{iw} &= \sum_{r,a_i} \frac{1}{\sqrt{|\mathbb{S}|}} \sqrt{p_r} |s,r\rangle_{if} |s\rangle_i |o_N(s,r)\rangle_o |0\rangle_s |A, a_i + s_A, 0, 0\rangle_q \\ &= \sum_{r,a_i} \frac{1}{\sqrt{|\mathbb{S}|}} \sqrt{p_r} |s,r\rangle_{if} |s\rangle_i |o_N(s,r)\rangle_o |0\rangle_s |A, a_i, 0, 0\rangle_q \end{aligned}$$

and the state the simulator sees is therefore independent of the secret. There's hence no way it can produce two orthogonal states depending on the secret and hence no way to simulate. For the case of a different simulator where U_{query}^{sim} is not the identity, but some unitary operation on $\mathcal{H}_{query}$, there exists a different adversary that applies the operation, $\tilde{U}_{query}^{adv} = U_{query}^{adv,F} (U_{query}^{sim})^{-1}$ instead. The register the simulator sends to the oracle of the ideal functionality is now exactly the same as above, and the same argumentation can now be repeated. It follows that for any simulator there exists an adversary that cannot be simulated.

5.4 Simplified MPC models

As with secret sharing we need to consider if we allow for superposition of the response register. In addition we need to consider if we allow the adversary to run the honest parties on a superposition of classical inputs, or we might restrict the adversary to pick classical inputs, and then carrying out only the attack itself in superposition. If a protocol is to be used as a sub-protocol in a classical protocol which might be under a super-position attack, then the sub-protocol might very conceivably be given a superposition of classical inputs by the honest parties of the outer protocol. If, on the other hand, the protocol is to be used in a specific physical context where it can be guaranteed that the inputs are classical, then the weaker model might be the right choice. An example of such a setting could be a prover on a smart card, where the instance and witness is hard-coded into the smart card at production time, but where a malicious verifier might later run the proof in superposition.

5.5 No simulator for quantum attacks by running classical simulator in superposition

Unfortunately, it is not completely clear how to design a good quantum simulator, even assuming restrictions on F as in Theorem 1 for secret sharing and the additional restriction above. A natural first attempt would be to use a classical simulator (which we can assume exists) and produce a superposition of what it produces on those corrupted subsets that occur in the query. We can show, however, that this cannot work in general.

Let us first make clear what we mean by running a classical simulator in superposition: consider a classical machine S which gets as input parties subset A, the inputs and outputs of those parties $(s_A, o_A(s))$ and a random string c. It then outputs $S(A, s_A, o_A(s), c)$. Running S in superposition now means that on input $|\psi^{sim}_{x,s}\rangle = \sum_{A\in F} \alpha_{x,s,A}|0\rangle_s|A, s_A, o_A(s), 0\rangle_q$, we output $\sum_{A\in F} \alpha_{x,s,A}\sqrt{p_c}|c\rangle_s|A, s_A, o_A(s), S(A, s_A, o_A(s), c)\rangle_q$. This means that the state returned to the adversary will be

$$\sum_{A,A'\in F} \alpha_A\alpha^*_{A'}|A\rangle\langle A'| \otimes |s_A, o_A(s)\rangle\langle s_{A'}, o_{A'}(s)| \otimes \sum_c p_c|S(A, s_A, o_A(s), c)\rangle\langle S(A', s_{A'}, o_{A'}(s), c)|$$

Now consider the following simple example protocol: we have 4 parties P_0, P_1, P_2, P_3. Player P_0 gets as input a bit s. He will then secret share it additively among the other parties: he chooses bits r_1, r_2 at random and sends r_1 to P_1, r_2 to P_2 and $s \oplus r_1 \oplus r_2$ to P_4. There is no output defined for anyone. Clearly, this protocol is perfectly secure against a classical attack where at most 2 parties are corrupted. One might therefore hope that it would be perfectly secure against quantum attacks using superpositions of sets containing only 1 party.

However, we now argue that such security cannot be shown by running any classical simulator S in superposition. To this end, consider the case where we corrupt all parties in equally weighted superposition. This will mean that the simulator has to start from the input state.

$$\frac{1}{2}(|P_0\rangle|s\rangle|0\rangle + \sum_{i=1}^{3} |P_i\rangle|\perp\rangle|0\rangle)$$

The state has this form since the input and output for P_0 is just s and the other parties have no input or output. The state returned will be

$$\rho_{sim,s} = \sum_{A,A'\in\{\{P_0\},\{P_1\},\{P_2\},\{P_3\}\}} \frac{1}{2}|A\rangle\langle A'| \otimes |s_A\rangle\langle s_{A'}| \otimes \sum_c p_c|S(A, s_A, c)\rangle\langle S(A', s_{A'}, c)|$$

If we define $s_{\{P_i\}} = s_i$ is some secret s if $i = 0$ and $\perp$ otherwise, and index with $P_i, P_{i'}$ instead of A, A', we can write the state a bit more conveniently as:

$$\rho_{sim,s} = \sum_{i,i'} \frac{1}{2} |P_i\rangle\langle P_{i'}| \otimes |s_i\rangle\langle s_{i'}| \otimes \sum_c p_c |S(i, s_i, c)\rangle\big\langle S(i', s_{i'}, c)\big|$$

On the other hand, we can compute the state $\rho_{real,s}$ that would be returned from a real attack. Define $v_i(s, r_1, r_2)$ to be the view of the protocol for P_i, defined as a 2-bit register. Thus

$$v_0(s, r_1, r_2) = (r_1, r_2), v_1(s, r_1, r_2) = (r_1, 0), v_2(s, r_1, r_2) = (r_2, 0), v_3(s, r_1, r_2) = (r_1 \oplus r_2 \oplus s, 0).$$

This means that the state returned for a particular choice of s, r_1, r_2 is

$$|\Psi_{s,r_1,r_2}\rangle = \frac{1}{2} \sum_{i=0}^{3} |P_i\rangle|s_i\rangle|v_i(s, r_1, r_2)\rangle$$

For fixed s, each choice of r_1, r_2 occurs with probability $1/4$, so

$$\rho_{real,s} = \sum_{r_1,r_2} \frac{1}{4} |\Psi_{s,r_1,r_2}\rangle\langle\Psi_{s,r_1,r_2}| = \frac{1}{16} \sum_{i,i'} |P_i\rangle\langle P_{i'}||s_i\rangle\langle s_{i'}| \sum_{r_1,r_2} |v_i(s, r_1, r_2)\rangle\langle v_{i'}(s, r_1, r_2)|$$

Consider the part of $\rho_{sim,s}$ that corresponds to $P_i = P_0$ and $P_{i'} = P_1$. Then, since we assume perfect simulation, i.e., $\rho_{sim,s} = \rho_{real,s}$, for any c, we must have $S(P_0, s_A, o_A(s), c) = S(0, s, c) = (r_1, r_2)$ and $S(P_{i'}, s_{A'}, o_{A'}(s), c) = S(1, \perp, c) = (r_1, 0)$, where the *same* r_1 occurs in both strings, since in a real execution, P_1 would of course receive the same bit that P_0 sent. We get an exactly similar conclusion for $P_{i'} = P_2$, and for $P_{i'} = P_3$, we can conclude that $S(3, \perp, c) = (s \oplus r_1 \oplus r_2, 0)$.

But now note that, we can simply compute $S(i, \perp, c)$ for $i = 1, 2, 3$ and some fixed c. Then the above shows that if running S in superposition was a perfect quantum simulator, we could compute $(s, 0) = S(1, \perp, c) \oplus S(2, \perp, c) \oplus S(3, \perp, c)$, without any information on s, which is of course a contradiction.

5.6 Limited simulators

The result in 5.4 strongly suggests that it's impossible to construct a simulator for general MPC protocols if we allow for the response registers to be in superposition. In this section we will discuss the problem of constructing a simulator for the model without response registers and protocols where the output is predetermined by the input. While simulators in general consists of two operators, $\{U^{sim}_{query}, U^{sim}_{res}\}$, we will restrict U^{query}_{sim} to be the identity. That is, the query from the adversary is sent directly to the oracle. We can do this wlog because there's no values in the response register and U^{query}_{sim} must be corruption preserving anyway.

Assume all randomness is uniformly chosen. This can be done wlog and helps to unclutter the notation. For a specific protocol, define the matrix (or vector of states) $M(A, s)$ as

$$M(A, s) = (|A, v_A(s, 0)\rangle, \ldots, |A, v_A(s, r)\rangle, \ldots, |A, v_A(s, |\mathcal{R}| - 1)\rangle)$$

We can now express the security of the protocol in the following way

Lemma 2. *A multiparty computation protocol for a deterministic function is perfectly secure against quantum F-attacks without response registers if, and only if, there exists a set of $|\mathcal{R}| \times |\mathcal{R}|$ unitary matrices $\{U_s\}_{s\in\mathbb{S}}$ such that for all $s, s' \in \mathbb{S}, r \in \mathcal{R}, A \in F_{s,s'}$ where $F_{s,s'} = \{A \in F | s_A = s'_A \wedge o_A(s) = o_A(s')\}$*

$$M(A, s)U_s = M(A, s')U_{s'}$$

Proof. Consider the final state the adversary sees (using the output is independent of randomness and no response registers),

$$\rho_{adv}^{rw} = Tr_p(|4\rangle\langle 4|_t^{rw}) = \sum_{r\in\mathcal{R}} \frac{1}{|\mathcal{R}|} \left|\psi_r^{adv}\right\rangle\left\langle\psi_r^{adv}\right|$$

where $|\psi_r^{adv}\rangle = \sum_{x,s,A} \alpha_{x,s,A}|s\rangle_i|x\rangle_e|A, s_A, o_A(s), v_A(s,r)\rangle_q$ and the state the simulator sees after the oracle has been applied in the ideal world (using no randomness in output),

$$\rho_{sim}^{iw} = Tr_{if,e,i}(|5\rangle\langle 5|_t^{rw}) = \sum_{x,s} \left|\psi_{x,s}^{sim}\right\rangle\left\langle\psi_{x,s}^{sim}\right|$$

where $|\psi_{x,s}^{sim}\rangle = \sum_{\mathbf{A}} \alpha_{x,s,A}|0\rangle_s|A, s_A, o_A(s), 0\rangle_q$. The state the simulator must sent back is,

$$\sum_{x,r,s} \frac{1}{|\mathcal{R}|}|\psi_{x,r,s}\rangle\langle\psi_{x,r,s}|$$

where

$$|\psi_{r,s}\rangle = \sum_A \alpha_{x,s,A}|A, s_A, o_A(s), v_A(s,r)\rangle_q$$

To continue we need a claim that is almost equivalent to Theorem 4 in [CJW04], but changed slightly for our specific purposes. For this reason we'll also provide a separate proof. The reader is encouraged to read [CJW04] for additional information on the existence of transformations between sets of quantum states.

Claim. There exists a perfect simulator, i.e. $\rho_{adv}^{rw} = \rho_{adv}^{iw}$, iff there exist unitary operator, U_{res}^{sim} such that for all x, $s \in \mathbb{S}$

$$U_{res}^{sim}\left|\psi_{x,s}^{sim}\right\rangle = U_{res}^{sim}\sum_A \alpha_{x,s,A}|0\rangle_s|A, s_A, o_A(s), 0\rangle_q = \frac{1}{\sqrt{|\mathcal{R}|}}\sum_A \alpha_{x,s,A}\sum_{i,k}[U_s]_{i,k}|k\rangle_s|A, s_A, o_A(s), v_A(s,i)\rangle_q \tag{7}$$

where $[U_s]_{i,k}$ is the i,k index of a unitary matrix in some set of unitary matrices, $\{U_s\}_{s\in\mathbb{S}}$.

Proof. For the forward direction, consider the final state the adversary sees (using no randomness in output)

$$\rho_{adv}^{iw} = Tr_{if,s}(|6\rangle\langle 6|_t^{iw}) = \sum_{k,s} \left|\psi_{k,s}^{adv}\right\rangle\left\langle\psi_{k,s}^{adv}\right|$$

where $|\psi_{k,s}^{adv}\rangle = \sum_{x,A} \alpha_{x,s,A}|s\rangle_i|x\rangle_e \left(\sum_i [U_s]_{i,k}\frac{1}{\sqrt{|\mathcal{R}|}}|A, s_A, o_A(s), v_A(s,i)\rangle_q\right)$. Note that $\forall s, A, A', x, x'$,

$$\frac{1}{|\mathcal{R}|}\sum_{i,j,k'}[U_s^*]_{i,k}[U_s]_{j,k}|A, s_A, o_A(s), v_A(s,i)\rangle_q\langle A', s_{A'}, o_{A'}(s), v_{A'}(s,j)\big|_q \tag{8}$$

$$= \frac{1}{|\mathcal{R}|}\sum_{i,k}[U_s^*]_{i,k}[U_s]_{i,k}|A, s_A, o_A(s), v_A(s,i)\rangle_q\langle A', s_{A'}, o_{A'}(s), v_{A'}(s,i)\big|_q$$

$$= \frac{1}{|\mathcal{R}|}\sum_i p_i|A, s_A, o_A(s), v_A(s,i)\rangle_q\langle A', s_{A'}, o_{A'}(s), v_{A'}(s,i)\big|_q$$

where we used that $[U_s]_{i,k}$ is unitary and only depends on s. This means the adversary sees the state he expects (except for labels) and we've completed the proof for the forward direction.
Conversely, assume for contradiction that there is a simulator that can constructs the correct mixed state for the adversary, but none that can construct one of the form of equation (7). If $\sum_{x,k,s} p_k|\phi_{x,k,s}\rangle\langle\phi_{x,k,s}|$ is the state the simulator sents back, then it must be that $\forall x, s$:

$$\sum_k p_k|\phi_{x,k,s}\rangle\langle\phi_{x,k,s}| = \sum_{r\in\mathcal{R}} \frac{1}{|\mathcal{R}|}|\psi_{x,r,s}\rangle\langle\psi_{x,r,s}|$$

where

$$|\psi_{r,s}\rangle = \sum_A \alpha_{x,s,A}|A, s_A, o_A(s), v_A(s,r)\rangle_q$$

This is true if, and only if, there exists unitary matrices, $\{U_{x,s}\}_{x,s}$, such that,

$$\forall k : \sqrt{p_k}|\phi_{x,k,s}\rangle = \frac{1}{\sqrt{|\mathcal{R}|}}\sum_i [U_{x,s}]_{i,k}|\psi_{x,i,s}\rangle$$

By purifying the state we see we get,

$$\sum_k \sqrt{p_k}|k\rangle|\phi_{x,k,s}\rangle = \frac{1}{\sqrt{|\mathcal{R}|}}\sum_{i,k}[U_{x,s}]_{i,k}|k\rangle|\psi_{x,i,s}\rangle = \frac{1}{\sqrt{|\mathcal{R}|}}\sum_A \alpha_{x,s,A}\sum_{i,k}[U_{x,s}]_{i,k}|k\rangle_s|A, s_A, o_A(s), v_A(s,i)\rangle_q$$

Since the value of x only effects the amplitude we can, by linearity, assume that $U_{x,s}$ does not depend on x. This shows that the state produced by any perfect simulator must be of that form. This is a contradiction and completes the proof.

U_{res}^{sim} exists (and is unitary) if it preserves inner product between all possible states, hence we can conclude that there exists a perfect simulator if there exists a set of unitary matrices, $\{U_s\}_{s\in\mathbb{S}}$ such that for all $x, x', s, s' \in \mathbb{S}$ and all queries $\{\alpha_{x,s,A}\}, \{\alpha'_{x',s',A}\}$

$$\begin{aligned}(\sum_A \alpha^*_{x,s,A}\langle 0|_s\langle A, s_A, o_A(s), 0|_q)(\sum_{A'}\alpha'_{x',s',A'}|0\rangle_s|A', s'_{A'}, o_{A'}(s'), 0\rangle_q)\\ = \frac{1}{|\mathcal{R}|}\left(\sum_A \alpha^*_{x,s,A}\sum_{i,k}[U^*_s]_{i,k}\langle k|_s\langle A, s_A, o_A(s), v_A(s,i)|_q\right)\\ \left(\sum_{A'}\alpha'_{x',s',A'}\sum_{j,k'}u_{j,k'}(s')|k'\rangle_s|A', s'_{A'}, o_{A'}(s'), v_{A'}(s',j)\rangle_q\right)\end{aligned}$$

For the LHS we have that,

$$\begin{aligned}(\sum_A \alpha^*_{x,s,A}\langle 0|_s\langle A, s_A, o_A(s), 0|_q)(\sum_{A'}\alpha'_{x',s',A'}|0\rangle_s|A', s'_{A'}, o_{A'}(s'), 0\rangle_q)\\ = \sum_A \alpha^*_{x,s,A}\alpha'_{x',s',A}\langle s_A, o_A(s)||s'_A, o_A(s')\rangle\end{aligned}$$

For the RHS we get that,

$$\frac{1}{|\mathcal{R}|}\left(\sum_A \alpha^*_{x,s,A}\sum_{i,k}[U^*_s]_{i,k}\langle k|_s\langle A, s_A, o_A(s), v_A(s,i)|_q\right)$$
$$\left(\sum_{A'}\alpha'_{x',s',A'}\sum_{j,k'}u_{j,k'}(s')\big|k'\big\rangle_s\big|A', s'_{A'}, o_{A'}(s'), v_{A'}(s',j)\big\rangle_q\right)$$
$$= \frac{1}{|\mathcal{R}|}\sum_{A,k,i,j}\alpha^*_{x,s,A}\alpha'_{x',s',A}[U^*_s]_{i,k}[U_{s'}]_{j,k}\langle s_A, o_A(s), v_A(s,i)\big|\big|s'_A, o_A(s'), v_A(s',j)\big\rangle$$

$$= \frac{1}{|\mathcal{R}|}\sum_{s,s',A,k}\alpha^*_{x,s,A}\alpha'_{x',s',A}\langle s_A, o_A(s)\big|\big|s'_A, o_A(s')\big\rangle \times \left(\sum_i[U^*_s]_{i,k}\langle A, v_A(s,i)|\right)\left(\sum_j[U_{s'}]_{j,k}\big|A, v_A(s',j)\big\rangle\right)$$

Noting that $|A, v_A(s',j)\rangle$ are normalized and U_s is unitary we see that

$$\left(\sum_i[U^*_s]_{i,k}\langle A, v_A(s,i)|\right)\left(\sum_j[U_{s'}]_{j,k}\big|A, v_A(s',j)\big\rangle\right) = 1$$

if and only if

$$\sum_i[U_s]_{i,k}|A, v_A(s,i)\rangle = \sum_j[U_{s'}]_{j,k}\big|A, v_A(s',j)\big\rangle$$

we see that, there exists a simulator iff there exist $|\mathcal{R}| \times |\mathcal{R}|$ unitary matrices U_s such that for all $s, s' \in \mathbb{S}, r \in \mathcal{R}, A \in F_{s,s'}$

$$\sum_i[U_s]_{i,r}|A, v_A(s,i)\rangle = \sum_j[U_{s'}]_{j,r}\big|A, v_A(s',j)\big\rangle \tag{9}$$

Using the definition of $M(A,s)$ we can now write the requirement in the more convenient way: There exists a simulator for the MPC protocol if, and only if, there exist a set of unitary matrices, $\{U_s\}_{s\in\mathbb{S}}$, such that for all $s, s' \in \mathbb{S}, A \in F_{s,s'}$:

$$M(A,s)U_s = M(A,s')U_{s'}$$

which completes the proof of Lemma 2.

We'd like to warn the reader that it is important not to confuse the unitary matrices, $\{U_s\}_{s\in\mathbb{S}}$, with the simulator itself. They are merely a tool to show the existence of one. Also, at this point the reader would be excused for lacking any intuition on why the lemma is reasonable. We therefore find it instructive to consider two examples. First we'll reconsider secret sharing in this framework and secondly a simple MPC protocol.

5.7 Secret sharing example

For this example we'll show that all secret sharing schemes secure against superposition F-attacks in fact satisfies the requirement in Lemma 2 (as they should). This does not add to what we already knew and is only meant to illustrate the principles. We'll allow each choice of randomness to be non-uniform as it adds only very little clutter. We note that since no parties have any input or output we have that for all $s, s' \in \mathbb{S} : F_{s,s'} = F$. The result follows as a Corollary of Theorem 1.

Corollary 1. *If a secret sharing scheme $\mathcal{S}$ is perfectly secure against quantum F-attacks then there exists a set of $|\mathcal{R}| \times |\mathcal{R}|$ unitary matrices $\{U_s\}_{s\in\mathbb{S}}$ such that for all $s, s' \in \mathbb{S}, r \in \mathcal{R}, A \in F$*

$$\sum_{i\in\mathcal{R}} \sqrt{p_i}[U_s]_{i,r}|v_A(i,s)\rangle = \sum_{j\in\mathcal{R}} \sqrt{p_j}[U_{s'}]_{j,r}\big|v_A(j,s')\big\rangle$$

Proof. Since the secret sharing scheme is secure we know from Theorem 1 that the joint distribution of the view for any $A, A' \in F$ is independent of s and hence, for all $s, s' \in \mathbb{S}$;

$$\sigma_s = \sum_{r\in\mathcal{R}} p_r|\psi_{s,r}\rangle\langle\psi_{s,r}| = \sigma_{s'} = \sum_{r\in\mathcal{R}} p_r\big|\psi_{s',r}\big\rangle\big\langle\psi_{s',r}\big|$$

where

$$|\psi_{s,r}\rangle = \sum_{A\in F} \alpha_A|A\rangle|v_A(r,s)\rangle$$

This is equivalent to saying that the fidelity of two such states is 1. According to Uhlmann's Theorem this implies that there exists purifications of σ_s and $\sigma_{s'}$ such that their inner product is 1. Unitary equivalence of purifications implies you can write any such purification in $\mathcal{H}\otimes\mathcal{R}$, where $dim(\mathcal{R}) = |\mathcal{R}|$, as

$$\sum_{k\in\mathcal{R}} \sqrt{p'_k}\big|\psi'_{s,k}\big\rangle|k\rangle_{\mathcal{R}} = \sum_{i,k\in\mathcal{R}} \sqrt{p_i}[U_s]_{i,k}|\psi_{s,i}\rangle|k\rangle_{\mathcal{R}}$$

where $\{U_s\}_{s\in\mathbb{S}}$ is a set of $|\mathcal{R}| \times |\mathcal{R}|$ unitary matrices. As the fidelity must be 1, it must be that for all $s, s' \in \mathbb{S}$

$$\left(\sum_{i,k\in\mathcal{R}} \sqrt{p_i}[U_s^*]_{i,k}\langle\psi_{s,i}|\langle k|_{\mathcal{R}}\right)\left(\sum_{j,k'\in\mathcal{R}} \sqrt{p_j}[U_{s'}]_{j,k'}\big|\psi_{s',j}\big\rangle\big|k'\big\rangle_{\mathcal{R}}\right) = \sum_{i,j,k\in\mathcal{R}} \sqrt{p_i}\sqrt{p_j}[U_s^*]_{i,k}[U_{s'}]_{j,k}\langle\psi_{s,i}\big|\big|\psi_{s',j}\big\rangle$$

$$= \sum_{A\in F,k\in\mathcal{R}} |\alpha_A|^2\left(\sum_{i\in\mathcal{R}} \sqrt{p_i}[U_s^*]_{i,k}\langle v_A(i,s)|\right)\left(\sum_{j\in\mathcal{R}} \sqrt{p_j}[U_{s'}]_{j,k}\big|v_A(j,s')\big\rangle\right) = 1$$

Because $\sum_{A\in F,r} p_r|\alpha_A|^2 = 1$, we can conclude that

$$\forall s, s' \in \mathbb{S}, r \in \mathcal{R}, A \in F : \left(\sum_{i\in\mathcal{R}} \sqrt{p_i}[U_s^*]_{i,k}\langle v_A(i,s)|\right)\left(\sum_{j\in\mathcal{R}} \sqrt{p_j}[U_{s'}]_{j,k}\big|v_A(j,s')\big\rangle\right) = 1$$

Noting the states are normalized, the result follows.

5.8 Simple MPC simulator example

For this example we return to the simple four-party (P_0, P_1, P_2, P_3) bit-sharing scheme considered in section 5.3. There we showed that we could not simulate an attack by simply running a classical simulator for the protocol in superposition. We are now in position to show that a simulator nonetheless exists. Recall that $|v_0(s, r_0, r_1)\rangle = |r_0, r_1\rangle, |v_1(s, r_0, r_1)\rangle = |r_0, 0\rangle, |v_2(s, r_1, r_2)\rangle =$

$|r_1, 0\rangle, |v_3(s, r_0, r_1)\rangle = |r_0 \oplus r_1 \oplus s, 0\rangle$. We have two possible inputs $s \in \{0, 1\}$ so we need to find two unitary matrices, U_0, U_1 in order to apply Lemma 2. These have been found manually.

$$U_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$U_1 = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

It is a tedious, but straight forward, calculation to show that[4],

$$\begin{aligned} M(P_1, 0) &= M(P_1, 1)U_1 \\ M(P_2, 0) &= M(P_2, 1)U_1 \\ M(P_3, 0) &= M(P_3, 1)U_1 \end{aligned}$$

and since $P_0 \notin F_{0,1}$ we do *not* require that $M(P_0, 0) = M(P_0, 1)U_1$. To get a better feeling for what is going on we'll consider equation (9) for two specific choices for A and r. First let $A = P_2$ and $r = (1, 0)$. The following must be true for the unitary matrices to be correct choices,

$$|P_2, v_{P_2}(0, (1, 0))\rangle = \sum_{(i_0,i_1)\in\mathcal{R}} u_{(i_0,i_1),(1,0)}(1)|P_2, v_{P_2}(1, (i_0, i_1))\rangle$$

Since U_0 is just the identity, the LHS is easy to calculate, $|P_2, v_{P_2}(0, (1, 0))\rangle = |0, 0\rangle$ For the RHS we see that

$$\begin{aligned} &\sum_{(i_0,i_1)\in\mathcal{R}} u_{(i_0,i_1),(1,0)}(1)|P_2, v_{P_2}(1, (i_0, i_1))\rangle \\ &= \frac{1}{2}|P_2, v_{P_2}(1, (0, 0))\rangle - \frac{1}{2}|P_2, v_{P_2}(1, (0, 1))\rangle + \frac{1}{2}|P_2, v_{P_2}(1, (1, 0))\rangle + \frac{1}{2}|P_2, v_{P_2}(1, (1, 1))\rangle \\ &= \frac{1}{2}|0, 0\rangle - \frac{1}{2}|1, 0\rangle + \frac{1}{2}|0, 0\rangle + \frac{1}{2}|1, 0\rangle = |0, 0\rangle \end{aligned}$$

Of particular interest is $A = P_3$ as the view depends on the secret (for fixed randomness). Choose $r = (0, 0)$:

$$|P_3, v_{P_3}(0, (0, 0))\rangle = \sum_{(i_0,i_1)\in\mathcal{R}} u_{(i_0,i_1),(0,0)}(1)|P_3, v_{P_3}(1, (i_0, i_1))\rangle$$

For LHS, $|P_3, v_{P_3}(0, (0, 0))\rangle = |0, 0\rangle$. For RHS,

$$\begin{aligned} &\sum_{(i_0,i_1)\in\mathcal{R}} u_{(i_0,i_1),(0,0)}(1)|P_3, v_{P_3}(1, (i_0, i_1))\rangle \\ &= \frac{1}{2}|P_3, v_{P_3}(1, (0, 0))\rangle + \frac{1}{2}|P_3, v_{P_3}(1, (0, 1))\rangle + \frac{1}{2}|P_3, v_{P_3}(1, (1, 0))\rangle - \frac{1}{2}|P_3, v_{P_3}(1, (1, 1))\rangle \\ &= \frac{1}{2}|1, 0\rangle + \frac{1}{2}|0, 0\rangle + \frac{1}{2}|0, 0\rangle - \frac{1}{2}|1, 0\rangle = |0, 0\rangle \end{aligned}$$

[4] Note that we encode the randomness as $(0, 0) = 0, (0, 1) = 1, (1, 0) = 2, (1, 1) = 3$

5.9 General MPC

In this section we'll give a restatement of Lemma 2, expressing the requirement for the existence of a simulator as an explicit property of the multiparty computation protocol. This will allow for a straight-forward, albeit extremely inefficient, method for checking the security of any deterministic MPC protocol in this model. For all inputs, $s \in \mathbb{S}$, and all sets $A_{s,s'} \in F_{s,s'}$ we'll associate a permutation, $\pi_{s,A_{s,s'}} : \mathcal{R} \to \mathcal{R}$.

Theorem 3. *A multiparty computation protocol for a deterministic function is perfectly secure against quantum F-attacks without response registers if, and only if, there exist permutations, $\pi_{s,A_{s,s'}}$ with the following two properties,*

1.

$$\forall s, s' \in \mathbb{S} | s < s', \forall A_{s,s'} \in F_{s,s'}, \forall r \in \mathcal{R} :$$
$$\left| A_{s,s'}, v_{A_{s,s'}}(s, \pi_{s,A_{s,s'}}(r)) \right\rangle = \left| A_{s,s'}, v_{A_{s,s'}}(s', \pi_{s',A_{s,s'}}(r)) \right\rangle$$

2.

$$\forall s, s', s'' \in \mathbb{S} | s < s', s'', \forall A_{s,s'} \in F_{s,s'}, A'_{s,s''} \in F_{s,s''} :$$
$$\sum_{r \in \mathcal{R}} \left| A_{s,s'}, v_{A_{s,s'}}(s, r) \right\rangle \left\langle A'_{s,s''}, v_{A'_{s,s''}}(s, r) \right|$$
$$= \sum_{r \in \mathcal{R}} \left| A_{s,s'}, v_{A_{s,s'}}(s, \pi_{s,A_{s,s'}}(r)) \right\rangle \left\langle A'_{s,s''}, v_{A'_{s,s''}}(s, \pi_{s,A'_{s,s''}}(r)) \right|$$

Note that property (1) is exactly the statement that a (not necessarily efficient) simulator exists in the classical model.

Proof. First recall that according to Lemme 2 we have that there is a simulator iff

$$\exists U_0, \cdots, U_{|\mathbb{S}|-1} \text{ st. :}$$
$$\forall s, s' \in \mathbb{S} | s < s' :$$
$$\forall A_{s,s'} \in F_{s,s'} : M(A_{s,s'}, s) U_s = M(A_{s,s'}, s') U_{s'}$$

For any solution we can always multiply with U_0^{-1} on the right side of all equations and we can hence wlog assume that $U_0 = I$. Now consider $\forall s \in \mathbb{S}$ the equation,

$$\forall A_{0,s} \in F_{0,s} : M(A_{0,s}, 0) = M(A_{0,s}, s) U_s$$

That is,

$$\forall r \in \mathcal{R} : \sum_{k \in \mathcal{R}} [U_s]_{r,k} \left| A_{0,s} v_{A_{0,s}}(s, k) \right\rangle = \left| A_{0,s} v_{A_{0,s}}(0, r) \right\rangle$$

Hence all rows (and columns) of U_s must sum to 1. We can, in other words, view $M(A_{0,s}, s)U_s$ simply as $M(A_{0,s}, 0)$ under some permutation of the columns. In fact, since U_s must always preserve the length of the columns, any $M(A_{s,s'}, s)U_s$ is always just a permutation of the columns in $M(A_{s,s'}, s)$. Although the permutation is defined by the same unitary it is not necessarily the same permutation. But they're clearly related. We will make this relationship explicit by separating the requirement for a simulator into two parts. Let $M^{\pi_{s,A_{s,s'}}}(A_{s,s'}, s)$ denote the permutation of the columns in

$M(A_{s,s'}, s)$ that corresponds to applying the permutation function to the randomness for the view in each column, that is,

$$M^{\pi_{s,A_{s,s'}}}(A_{s,s'}, s) = \left(\left|A, v_A(s, \pi_{s,A_{s,s'}}(0))\right\rangle, \ldots, \left|A, v_A(s, \pi_{s,A_{s,s'}}(r))\right\rangle, \ldots\right)$$

The first requirement is simply the statement that for $A_{s,s'}$ some kind of permutation of the randomness exist to allow their views to be equal for s and s' (which is true iff a simulator in the classical model exists). The second is the statement that these permutations must be performed by the same unitary matrix, when the input to the parties are the same. That is, there exists a simulator if, and only if, there exist permutations, $\pi_{s,A_{s,s'}}$ with the following two properties,

1.

$$\begin{aligned} \forall s, s' \in \mathbb{S} | s < s' : \\ \forall A_{s,s'} \in F_{s,s'} : \\ M^{\pi_{s,A_{s,s'}}}(A_{s,s'}, s) = M^{\pi_{s',A_{s,s'}}}(A_{s,s'}, s') \end{aligned}$$

2. There exist unitary matrices, $\{U_s\}_{s\in\mathbb{S}}$, such that $\forall s, s', A_{s,s'} : M(A_{s,s'}, s)U_s = M^{\pi_{s,A_{s,s'}}}(A_{s,s'}, s)$.

For a specific choice of permutations such unitary matrices exist iff they preserve the inner product. That is, iff

$$\begin{aligned} \forall s, s', s'' \in \mathbb{S} | s < s', s'' : \forall A_{s,s'} \in F_{s,s'}, A_{s,s''} \in F_{s,s''} : \\ M(A_{s,s'}, s')M(A'_{s,s''}, s)^\dagger = M^{\pi_{s,A_{s,s'}}}(A_{s,s'}, s)M^{\pi_{s,A_{s,s''}}}(A'_{s,s''}, s)^\dagger \end{aligned}$$

Writing out the equations using the definition of $M(A, s)$ we conclude the proof.

Had the choice of corrupted parties been classical we could have let the unitary matrices depend on both the input and which party was corrupted. Specifically, in equation (8), the reason the unitaries cannot depend on A is that $\sum_{i,j,k\in\mathcal{R}} [U^*_{s,A}]_{i,k}[U_{s,A'}]_{j,k}$ does not cancel out correctly if they differ and the adversary would not see the correct state. If the choice of A was classical we would see no such cross-terms[5] and no such relationship would be required.

References

[BCG+05] Michael Ben-Or, Claude Crépeau, Daniel Gottesman, Avinatan Hassidim, and Adam Smith. Secure multiparty quantum computation with (only) a strict honest majority. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 249–260, 2005.

[CJW04] Anthony Chefles, Richard Jozsa, and Andreas Winter. On the existence of physical transformations between sets of quantum states. *International Journal of Quantum Information*, pages 11–21, 2004. http://arxiv.org/abs/quant-ph/0307227.

[FS09] Serge Fehr and Christian Schaffner. Composing quantum protocols in a classical environment. In *Theory of Cryptography Conference (TCC)*, volume 5444 of *Lecture Notes in Computer Science*, pages 350–367. Springer, 2009.

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

[KN08] Gillat Kol and Moni Naor. Games for exchanging information. In *Theory of Cryptography Conference (TCC)*, volume 4948 of *Lecture Notes in Computer Science*, pages 423–432. Springer, 2008.

[5] recall: there are no cross-terms for s because the register is perfectly entangled with the parties/ideal functionality

[PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In *Advances in Cryptology—CRYPTO '08*, volume 5157 of *Lecture Notes in Computer Science*, pages 554–571. Springer, 2008.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 84–93, 2005.