Lecture Notes

Quantum Cryptography Week 8:

Quantum cryptography beyond key-distribution

Contents

In the last couple of weeks we discussed what are probably the most well-known quantum cryptographic protocols, namely the BB'84 and Ekert'91 protocols for quantum key distribution (QKD). However, quantum cryptography allows much more than just QKD. Quantum cryptographic protocols can be roughly divided into two categories. The first category concerns the use of quantum communication to implement classical tasks. An example of this is QKD, where the goal is to generate a classical key between two distant, but cooperating, parties. This week we will see other tasks which fall in this category. Contrary to QKD, however, this week the users implementing the task will not trust each other — so we will talk about "malicious Bob" playing against "honest Alice", and vice-versa.

The second category is concerned with genuinely quantum tasks, for which at least one of the inputs or outputs is quantum. An example of such a task is quantum secret sharing, where the goal is to distribute a qubit among several participants in such a way that at least a certain number of them need to come together to reconstruct the secret (we saw an example of a quantum secret sharing scheme in week 2).

This categorization does not cover all possibilities, and some important cryptographic tasks cannot be placed in either category. An example is delegated quantum computation. Here we imagine a user who has access to a powerful, but remote, quantum computer, while herself having only very limited, if any, quantum capabilities. The user's goal is to "delegate" a complex quantum computation she is interested in to the remote computer, without leaking any information about the computation or the input. We will learn more about delegated computation in week 10.

8.1 Two-Party cryptography

Let's imagine two parties, Alice and Bob, who can communicate over a classical, or even quantum, channel. Alice has some classical input x, which contains some private information only she has access to. Similarly Bob has a classical input y. Each of Alice and Bob would like to compute a certain function of both their inputs, $f_A(x,y)$ for Alice and $f_B(x,y)$ for Bob.

An easy solution would be for Alice and Bob to exchange their respective inputs and perform the computation locally. Unfortunately they don't trust each other: neither party wants to reveal more information about his or her input than is absolutely necessary. A typical example of such a task is "Yao's millionaire's problem". Here x and y represent Alice and Bob's respective fortune. The functions $f_A(x,y) = 1_{x>y}$ and $f_B(x,y) = 1_{y>x}$ will tell Alice and Bob if their fortune is larger than the others'. Can they decide who is the richest without announcing their actual fortune? (It turns out they can, but it's not so easy: we'll see how to do it later.)

Note that there is no eavesdropper here — this week we take a break from Eve. The communication channel between Alice and Bob is perfectly secure; the "adversary" is the other player rather than some external entity.

8.1.1 Secure Function Evaluation

The general task we just introduced is called Secure Function Evaluation (SFE). Let's formalize it.

Definition 8.1.1 Secure function evaluation (SFE) is a task involving two parties, Alice and Bob. Alice holds an input $x \in \mathscr{X}$ and Bob holds an input $y \in \mathscr{Y}$. Alice and Bob interact over a communication channel, and output an $a \in \mathscr{A}$ and $b \in \mathscr{B}$ respectively. We will say that a given protocol is a secure protocol computing a pair of functions $(f_A : \mathscr{X} \times \mathscr{Y} \to \mathscr{A}, f_B : \mathscr{X} \times \mathscr{Y} \to \mathscr{B})$ if it satisfies the following properties:

- *Correctness:* If both Alice and Bob follow the protocol (we say they are *honest*) then $a = f_A(x,y)$ and $b = f_B(x,y)$.
- *Security against cheating Bob:* If Alice is honest, then Bob cannot learn more about her

input x than he can infer from $f_B(x,y)$.

- *Security against cheating Alice:* If Bob is honest, then Alice cannot learn more about his input y than she can infer from $f_A(x,y)$.

Note that the definition does not guarantee anything in case Alice and Bob are both dishonest. In this case there is nothing we can do! The goal is only to protect the honest parties.

The definition we gave is rather informal — what does it mean that “Bob cannot learn more about Alice’s input x than he can infer from $f_B(x,y)$”? It turns out that this requirement is rather tricky to make precise. We will return to it in a moment. First let’s consider some examples that illustrate the kind of problems that fit in this framework.

▪ **Example 8.1.1** Alice and Bob are contemplating going to a movie. Here, $x,y \in \{0,1\}$ where ’0’ denotes “no” and ’1’ denotes “yes”. The function they wish to compute is

$$f(x,y) = f_A(x,y) = f_B(x,y) = x \text{ AND } y.$$

Let us see what security means here. If $f(x,y) = 1$, then it must be that $x = y = 1$ and both parties learn the other’s input. Alice and Bob go to the movies. If $f(x,y) = 0$, then it must be that either $x = 0$ or $y = 0$ (or both). If Alice input $x = 1$, i.e., she would like to go to a movie, and the output is $f(x,y) = 0$, then Alice can infer $y = 0$, but Bob will never learn whether $x = 0$ or $x = 1$. So a party only learns the others’ input if they themselves declared they wanted to go to a movie. ▪

▪ **Example 8.1.2** Alice (a customer) wants to identify herself to Bob (an ATM). Here, x is the password honest Alice should know, and y the password (for Alice) that the honest ATM should have stored in its database. The function f is the equality, that is, $f(x,y) = 1$ if and only if $x = y$, and $f(x,y) = 0$ otherwise. Security means that if Alice is dishonest (she might not know x but is still trying to break through the ATM’s authentication system), then Bob should have the guarantee that Alice will never learn anything more about his input y than she can infer from $f(x,y)$ — that is, whatever x she tries, that $x \neq y$! (Unless she happens to be lucky of course.) Similarly, if Bob is a fraudulent ATM who is out to steal passwords from the users, the best he can do is guess a y and see whether it worked. No more information is revealed. ▪

▪ **Example 8.1.3** Alice wants to sell a book to Bob. Here, x is Alice’s asking price, and y is Bob’s bid. The function they wish to compute is $f(x,y) = (ok,y)$ if $y \geq x$, and $f(x,y) = (no,0)$ if $y < x$. If $f(x,y) = (ok,y)$, Alice can proceed to sell the book to Bob. Bob pays what he offers, and Alice gets at least her asking price. Security means that dishonest Bob can never learn what the asking price actually was, only that it was less or equal than his bid. If $f(x,y) = (no,0)$, then Alice will not sell her book. Security means that Alice will never learn exactly what Bob’s bid actually was, only that it was lower than her asking price. Similarly, Bob will only learn that Alice’s asking price was higher than his bid. ▪

8.1.2 The simulation paradigm for security

We won’t attempt to give a fully formal definition of security of SFE, as this would take us too far. Good references to learn more on the topic include a survey by Goldreich [Gol05] (especially Section 7) and one by Lindell [LP09]. For the case of quantum protocols, it is only recently that a satisfactory definition has been introduced; see [Unr10], or [FS09] for a weaker but perhaps more approachable definition.

The key concepts based on which security is defined are those of *ideal functionality* and *simulators*.

Definition 8.1.2 — Ideal functionality. Let $(f_A : \mathscr{X} \times \mathscr{Y} \to \mathscr{A}, f_B : \mathscr{X} \times \mathscr{Y} \to \mathscr{B})$ be a pair of functions corresponding to an SFE task. The *ideal functionality* is a device which takes as input $x \in \mathscr{X}$ and $y \in \mathscr{Y}$, and returns $f_A(x,y)$ and $f_B(x,y)$.

Thus the ideal functionality does precisely what a protocol solving the SFE task is supposed to achieve: it directly returns the values of each of the two functions. It is "ideal" in the sense that it does not require any interaction between the two parties: you should picture a "black box" which takes the inputs and provides the outputs, no questions asked. Informally, we will then say that a protocol for SFE is secure if, provided one of the parties is honest, whatever the other party does there is nothing more they can produce that they could not have produced by interacting with the ideal functionality.

■ **Example 8.1.4** Consider again the millionaire's problem. Here the ideal functionality takes as input x from Alice and y from Bob, and returns $f_A(x,y) = 1_{x>y}$ to Alice and $f_B(x,y) = 1_{y>x}$ to Bob. Now, suppose we are given a protocol for this problem, and suppose a malicious Bob was able to infer Alice's fortune x through his interaction with her. Then the simulation paradigm dictates that, if the protocol was secure, he should be able to do the same through an interaction with the ideal functionality. But the ideal functionality just takes any y' of Bob's choice and returns to him $f_B(x,y') = 1_{y'>x}$. Since only one interaction is allowed, the best Bob can do is find out if $x < y'$ for a single y' of his choice, something which for this particular SFE task is unavoidable. ■

A little more formally, we make the following definition.

Definition 8.1.3 — Security against cheating Bob. A protocol for an SFE task (f_A, f_B) is secure if for any malicious Bob interacting with an honest Alice in the protocol, there exists a *simulator* which, by controlling Bob in an interaction with the ideal functionality (where Alice acts as honest Alice in the protocol), is able to generate a distribution on outputs that is indistinguishable from the distribution produced by malicious Bob in the real interaction.

(A symmetric definition can be given for security against cheating Alice.) The definition refers to the output distributions being "indistinguishable" from one another. This by itself is a subtle notion. Indistinguishability can be either "statistical", meaning that the output distributions are close in total variation distance (the classical analogue of the trace distance), or "computational", meaning that the distributions cannot be distinguished by any efficient (polynomial-time) algorithm. In the classical setting most interesting tasks in two-party cryptography cannot be proven secure for the stronger notion of statistical indistinguishability, so that one has to rely on computational indistinguishability.

In these notes our main focus is on finding certain tasks for which statistical indistinguishability can be achieved by quantum protocols, but not by classical ones. We won't worry too much about giving formal proofs of security, but you should be aware that doing so can be quite tricky. In fact, many *wrong* proofs of security of quantum protocols have been given in the past by relying on "intuitive" arguments rather than the simulation paradigm. We will point out one such example later.

8.2 Oblivious Transfer

Let's discuss an important example of an SFE task, *Oblivious Transfer* (OT). Here $\mathscr{X} = \{0,1\}^\ell \times \{0,1\}^\ell$ and $\mathscr{Y} = \{0,1\}$. That is, Alice receives as input two ℓ-bit strings $x = (s_0, s_1)$, and Bob receives as input a single bit y. The goal is for Bob to obtain $f_B(x,y) = s_y$, while Alice will obtain nothing, $f_A(x,y) = \perp$. Thus a protocol will implement this task securely if, first of all it is correct, and second, malicious Alice will not obtain any information at all about Bob's input bit (since the ideal functionality never returns anything to Alice), and malicious Bob will at best learn one of Alice's strings, but never more.

As an example scenario where this task could be useful, imagine that Alice is a database which contains two entries s_0 and s_1. Bob would like to retrieve one of them, but does not want Alice to know which one he retrieved, preserving his privacy. Alice can also be sure that he does not retrieve her entire database.

What makes OT truly interesting is the fact that it is *universal* for two-party cryptography. That is, if we can build a secure protocol for 1-2 OT then we can solve *any* SFE problem by just using 1-2 OT multiple times. In this respect 1-2 OT is analogous to a universal gate in computing. This universality property can be shown in different ways; see Section 6 in [PS10] for a construction based on secret sharing called "Yao's garbled circuits".

Unfortunately it is also known that OT cannot be implemented securely without computational assumptions… in the classical world. What about quantum protocols? Let's consider the following natural protocol, introduced in [Ben+91].

Protocol 1 — Quantum OT protocol. Alice has input $x = (s_0, s_1) \in \{0,1\}^\ell \times \{0,1\}^\ell$ and Bob has input $y \in \{0,1\}$. Their goal is to compute $(f_A(x,y) = \perp, f_B(x,y) = s_y)$.

1. Alice selects uniformly random $x \in \{0,1\}^{2n}$ and $\theta \in \{0,1\}^{2n}$. She prepares BB'84 states $|x_j\rangle_{\theta_j}$ for $j = 1,\dots,2n$ and sends them to Bob.
2. Bob measures each of the qubits he received from Alice in a random basis $\tilde{\theta}_j$, obtaining outcomes $\tilde{x}_1,\dots,\tilde{x}_n \in \{0,1\}$. He notifies Alice that he is done with his measurements.
3. Alice reveals her choice of bases $\theta_1,\dots,\theta_{2n}$ to Bob.
4. Bob sets $I = \{i : \theta_i = \tilde{\theta}_i\}$, $I_y = I$ and $I_{1-y} = \{1,\dots,2n\}\backslash I$. (For simplicity, assume that $|I_0| = |I_1| = n$.) Bob sends (I_0, I_1) to Alice.
5. Alice sends $t_0 = s_0 \oplus x_{I_0}$ and $t_1 = s_1 \oplus x_{I_1}$ to Bob.
6. Alice outputs $\perp$, and Bob outputs $t_y \oplus \tilde{x}_{I_y}$.

Let's first check that this protocol is correct. This is clear: whenever $j \in I_y$, by definition $\theta_j = \tilde{\theta}_j$, therefore $x_j = \tilde{x}_j$ and $(s_y \oplus x_{I_y}) \oplus \tilde{x}_{I_y} = s_y$.

Is it secure? Security against cheating Alice is not hard to verify. Indeed, the only information she gets from a honest Bob are two sets (I_0, I_1). If Bob is honest, even if Alice sent him arbitratry states in the first step, and misleading basis information in the third, since Bob's choice of $\tilde{\theta}_j$ is uniformly random the sets I_0, I_1 will be a uniformly random partition of $\{1,\dots,2n\}$ that contains no information at all about his input y. So anything a dishonest Alice could do in this protocol can be simulated by an interaction with the ideal functionality, where the simulator would replace Bob's message (I_0, I_1) (which is not provided by the ideal functionality) with a uniformly random choice.

How about security against cheating Bob? The idea is supposed to be that, given Bob's basis choices are random, he can at best learn roughly half of Alice's inputs $\tilde{x}_j$. Of course he could lie about which half he learned, but in any case he will only be able to recover about half of the bits of Alice's input $x = (s_0, s_1)$. Note he could still, for example, learn half of s_0 and half of s_1 (instead of the whole s_0 or s_1 and nothing about the other). This can be prevented by adding in a layer of privacy amplification (recall from Week 4) to the protocol, so let's assume it is not a serious issue.

You might already have noticed there is a more worrisome hitch. The protocol requires Bob to "measure each qubit he received from Alice", and then "notify Alice that he is done with his measurements". But what if Bob is malicious — what if he stores Alice's qubits in a large quantum memory, without performing any immediate measurement, and lies to her by declaring that he is done? Alice would then naively reveal her basis information, and Bob could measure all the qubits he stored using $\tilde{\theta}_j = \theta_j$. He would thus obtain outcomes $\tilde{x}_j = x_j$ for all j, and he could recover both $s_0 = t_0 \oplus \tilde{x}_{I_0}$ and $s_1 = t_1 \oplus \tilde{x}_{I_1}$!

So the protocol we gave is not at all secure. There are two ways to get around the problem. One possibility is to make certain physical assumptions on the capacities of cheating Bob. A possible assumption is that Bob has a bounded quantum memory, in which case he wouldn't be able to store all of Alice's qubits. We will explore this assumption next week. Another possibility would be to somehow force Bob to *commit* to a choice of basis $\tilde{\theta}_j$, and outcomes $\tilde{x}_j$ that he obtained, *before* Alice would accept to reveal her θ_j. Of course, to avoid reversing the difficulty it should be that Alice cannot learn any information about the $\tilde{\theta}_j$ just from Bob's commitments. The task we're trying to solve is called *bit commitment*, and it is another fundamental primitive of multi-party cryptography. Let's explore it next.

8.3 Bit commitment

The goal of a bit commitment protocol is to provide a means for Alice to *commit* to an unbreakable promise, without revealing any information about the promise to Bob until Alice decides to *open* her promise — without being allowed to change her mind in-between the "commit" and "open" phases.

Definition 8.3.1 — Bit Commitment (BC). Bit commitment is a task involving two parties, Alice (the committer) and Bob (the receiver). The input to Alice is a single bit $b \in \{0,1\}$, and she has no output. Bob has no input, and his output is b'. A protocol for bit commitment has two phases, the *commit phase* and the *open phase*, and it should satisfy the following properties:

1. (Correctness) If both Alice and Bob are honest then at the end of the protocol Bob outputs a bit $b' = b$.
2. (Hiding) For any malicious Bob, the state of Bob at the end of the commit phase (including all his prior information and information received from Alice during the commit phase, classical or quantum) is independent of b.
3. (Binding) For any three possible malicious behavior A, A_0 and A_1, the probabilities p_b that Bob outputs $b' = b$ after interacting with A in the commit phase and A_b in the open phase satisfy $p_0 + p_1 \leq 1$.

The hiding property is clear: it states that, after the open phase, Bob still has no information at all about the bit b that honest Alice committed to. The binding property is more subtle. Intuitively, what it is trying to capture is that once Alice has committed to a specific value b (this is the role of A in the definition), then she shouldn't be able to come up with two possible different behavior (A_0 and A_1) such that she has a strictly higher than $1/2$ chance of being able to convince Bob that $b = 0$ (she would run A_0) *or* that $b = 1$ (she would run A_1).

Exercise 8.3.1 Give a secure protocol for Yao's millionnaire's problem, assuming you have access to a protocol securely implementing bit commitment. ■

Bit commitment is a good example of a cryptographic task for which it is crucial to define security as precisely as possible, especially in the quantum setting. Consider the following "intuitive" definition of the binding property: "It should be impossible for malicious Alice to convince honest Bob that $b = 0$ *and* $b = 1$ with probability strictly larger than 1". Do you see the difference? I wouldn't blame you if you didn't — the pioneers of quantum information and cryptography didn't either! In 1991 Brassard et al. [Bra+93] famously proposed an "unconditionally secure" quantum protocol for bit commitment, that satisfied the above intuitive notion of security. However, their protocol was later completely broken! (Indeed, as we will soon see, perfectly secure bit commitment is impossible both in the classical and the quantum world.) Their "mistake" is that they interpreted the italicized "and" in the intuitive definition above in a strong sense: they show that, in their protocol, it wouldn't be possible for a malicious Alice to simultaneously convince Bob that $b = 0$ and $b = 1$, by assuming that, if this where the case, the two final quantum states of the protocol associated with the outcomes "Bob returns $b' = 0$" and "Bob returns $b' = 1$" would exist simultaneously. However, as we know very well by now, quantum information is subtle, and the fact that Alice can "change her mind" after the commit phase does *not* imply that she can generate both the $b' = 0$ and $b' = 1$ states for Bob from the same state at the end of the open phase; only that she can generate either of them.

8.3.1 Universality of bit commitment

Bit commitment is an important task in quantum multiparty cryptography because, just as OT, it is known to be universal. This is demonstrated by the protocol for OT we gave in the previous section: as we discussed, the protocol by itself is not secure; however if one has access to a secure protocol for bit commitment then it can be turned into a secure protocol as well. Since OT itself is universal for multiparty computation we deduce that bit commitment is universal. However, note that the protocol for OT based on bit commitment we gave is quantum, even if bit commitment is implemented using a classical protocol. It is interesting to note that this is unavoidable: indeed, bit commitment is *not* universal for *classical* multiparty computation!

Let's see how the reverse can be accomplished, using OT as a building block to achieve bit commitment. For this, we will consider an approximate version of bit commitment, in which Alice can change her mind with some small error probability ε. That is, the protocol is ε-binding in that the requirement $p_0 + p_1 \leq 1$ is relaxed to $p_0 + p_1 \leq 1 + \varepsilon$.

The following protocol takes 1-2 OT and turns it into bit commitment. We will invert the use of 1-2 OT: Bob will now be the sender, and Alice the receiver.

Protocol 2 — Bit commitment from 1-2 OT. Alice's input is $b \in \{0,1\}$. Bob has no input.

1. Commit phase: Bob chooses two strings $s_0, s_1 \in \{0,1\}^\ell$ uniformly at random. Bob and Alice execute a protocol for OT, with the role of the players reversed: OT-Alice's input is (s_0, s_1) (provided by Bob), and OT-Bob's input is b (provided by Alice). Thus Alice receives s_b, and Bob receives $\perp$.
2. Open phase: Alice sends $\hat{b}$ and $\hat{s} = s_b$ to Bob. If $\hat{s} = s_{\hat{b}}$, then Bob accepts and concludes Alice committed herself to $b = \hat{b}$. If $\hat{s} \neq s_{\hat{b}}$, then Bob rejects.

Why does this give bit commitment? First of all, if both parties behave honestly the protocol is clearly correct. Let's consider the hiding property. We need to show that, at the end of the commit phase, Bob has no information about b. This follows right away from the definition of OT, which guarantees that the sender never receives any information about the receiver's input.

It remains to show the protocol is ε-binding. This again follows from the security of OT, for $\varepsilon = 2^{-\ell}$. Indeed, the ideal functionality for OT is such that the receiver can learn only *one* of the two strings. Suppose Alice has two possible strategies, one to open $\hat{b} = 0$ and the other to open $\hat{b} = 1$. Let p_0 be the probability that the first strategy succeeds, and p_1 the probability that the second succeeds. As a consequence, Alice can recover both of s_0 and s_1 with probability at least $p_0 + p_1 - 1$. By the security of the OT primitive, this can happen with probability at most the probability that a random guess of the non-received string would succeed, i.e. $2^{-\ell}$. By taking ℓ large enough we can achieve any desired ε-security for the binding property.

If you have been reading carefully you may have noted that in the argument above we made a jump from "Alice can recover s_0 with probability p_0, and s_1 with probability p_1" to "Alice can recover both of s_0 and s_1 with probability at least $p_0 + p_1 - 1$". While this is correct if Alice is classical, if her strategies involved incompatible quantum measurements the implication might no longer be true. Hence in case we allow the protocol implementing OT to be a quantum protocol, we have to be additionally careful in showing that the resulting protocol for bit commitment satisfies the required definition. This is possible (so the protocol described above *is* secure provided the implementation of OT is, whether classical or quantum) but one must take even greater care in making the right security definitions to ensure that they satisfy the stringent criteria of "universal composability" [Unr10].

8.3.2 Impossibility of bit commitment

Since, as we argued, bit commitment implies OT (in the quantum world), but perfect OT is impossible, it must be that bit commitment is impossible as well! Let's see why. We give an informal argument, and refer you to the detailed notes by Watrous on the subject [Wat06] for a more rigorous proof.

Consider a protocol for bit commitment that is perfectly hiding, i.e. at the end of the commit phase Bob has absolutely no information about Alice's bit b. Let's show that in this case a malicious Alice can cheat *arbitrarily*: she is able to open any bit that she likes.

Suppose that the initial state of Alice and Bob is a pure state $|\psi\rangle_{AB}$, which in particular contains Alice's input. We can always assume this is the case by considering a purification and giving the purifying system to Alice.

Now suppose Alice executes the bit commitment protocol with input b. At the end of the commit phase, the joint state of Alice and Bob can be described by some pure state $|\psi(b)\rangle_{AB}$. Since the bit commitment protocol is perfectly hiding, it must be the case that

$$\rho_B(0) = \mathrm{tr}_A(|\Psi(0)\rangle\langle\Psi(0)|_{AB}) \quad \text{and} \quad \rho_B(1) = \mathrm{tr}_A(|\Psi(1)\rangle\langle\Psi(1)|_{AB})$$

are absolutely identical, as otherwise there would be a measurement that Bob can make on his system to distinguish (even partially) between the two states, giving him some information about b.

Now is time to take out our quantum information theorist's toolbox and extract one of its magic tools: Uhlman's theorem! The theorem implies that, if $\rho_B(0) = \rho_B(1)$, then necessarily there exists a unitary U_A on Alice's system such that $U_A \otimes \mathbb{I}_B|\Psi(0)\rangle_{AB} = |\Psi(1)\rangle_{AB}$. But this means Alice can perfectly change her mind, thereby completely breaking the binding property for the protocol.

Rather unfortunately for the fate of quantum multiparty cryptography, it is possible to generalize this argument to show that any protocol for *any* task in multiparty quantum cryptography must be "totally insecure" in the following sense: if the protocol is perfectly secure against a malicious Bob, then it must be that a malicious Alice (interacting with honest Bob) can recover the value $f_A(x,y)$ associated with Bob's input y for *all* possible values of x, simultaneously! (To see why this indeed renders the protocol totally insecure, consider for example the millionnaire's problem: Alice would learn if $x > y$ for any x, and could thus perform a quick binary search to learn Bob's fortune y exactly.)

Given such a strong impossibility result, due to [LC97; May97], we are left with two possibilities. Either we place limiting assumptions on any malicious player's abilities. In classical cryptography these are mostly computational assumptions, and we give an example in the next section. In quantum cryptography a very successful approach considers physical assumptions on the adversary, such as it having limited storage capabilities. We will explore such assumptions in detail next week.

The second option consists in taking act of the impossibility of *perfect* protocols for multiparty cryptography and instead settle for protocols with a relaxed notion of security, where e.g. Bob can learn "some" information about both Alice's input strings s_0, s_1 in bit commitment, but not all. This is indeed possible, and can be quite useful in spite of the relaxed security condition. We'll see an example in Section 8.4.

8.3.3 Computationally secure commitments

We have seen that it is impossible to perfectly implement bit commitment, whether we use quantum information or not. The fact that it is such a useful primitive, however, should encourage us to be creative. In many contexts we would be willing to put up with our usual requirement for perfect, information-theoretic security, and start making assumptions — of course, the fewer the better! One possibility is to make physical assumptions, such as that the malicious party has a bounded amount of quantum memory. We will discuss this assumption in much more detail next week.

We can also assume that the malicious party has bounded computational power. This is a very standard assumption in classical cryptography, as indeed very little can be achieved without it (in contrast to quantum cryptography). Of course, here we would only want to make assumptions that hold even if the malicious party has bounded *quantum* computational power. The weakest such assumption under which any interesting cryptographic task is made possible is the existence of *one-way functions*. Informally, a function is one-way if it is easy to evaluate the function on any input (there is an efficient algorithm to compute it), but it is hard to invert the function (given a point in the range of the function, find a pre-image).

There are many candidate constructions of one-way functions, including some that are believed to be hard to invert even for quantum computers. And it turns out that, assuming one-way functions exist, there is a simple protocol for bit commitment that is statistically binding ($p_0 + p_1$ can be made as close to 1 as desired by increasing the amount of communication required in the scheme), and computationally hiding (the hiding property holds as long as it can be assumed that the malicious party cannot invert the one-way function). For a description of the protocol we refer you to the videos; you may also be interested in Section 4.7 in the lecture notes [PS10], which presents a different protocol based on the (somewhat stronger) assumption of existence of one-way permutations, and discusses applications of bit-commitment to zero-knowledge proof systems.

8.4 Coin flipping

Let's see a last example of an interesting two-party task: coin-flipping. This problem was introduced by Blum [Blu83]: imagine Alice and Bob want to flip a fair coin to determine who has to do some chore — for example, prepare the next lecture. But Alice is in Europe, Bob is in North America, so they have to do this over the phone. Is there a good protocol, that would ensure both Alice and Bob obtain the same outcome for the coin flip, but such that neither can bias it one way or the other?

Definition 8.4.1 — Strong coin flipping. In the task of *coin flipping* there are two players, Alice and Bob. Neither has an input. The goal is for both players to output the same value $c \in \{0,1\}$ such that the following properties hold.

- Correctness: if both Alice and Bob are honest then c is uniformly distributed.
- ε-secure: neither player can force $p(c=0) \geq 1/2+\varepsilon$ or $p(c=1) \geq 1/2+\varepsilon$, where $p(c)$ is the probability that the honest player outputs a value c.

The smallest ε for which a protocol is ε-secure is called the *bias*.

Coin flipping does not fall in the framework of SFE, because it is a randomized primitive: there is no fixed function of the players' inputs that determines their outputs (indeed, here they do not have any input at all). Thus the strong impossibility results that we saw for SFE, both in the classical and quantum setting, no longer apply...is perfectly secure coin flipping possible?

8.4.1 Classical coin flipping

Let's see if the following simple protocol does the trick.

Protocol 3 — Blum coin flipping.

1. Alice flips a random bit $a \in \{0,1\}$ and sends it to Bob.
2. Bob flips a random bit $b \in \{0,1\}$ and sends it to Alice.
3. Both players return $c = a \oplus b$.

Is this protocol secure? It is certainly correct: if both players are honest then c is uniformly distributed. In fact, it is sufficient that one player is honest: as long as a or b is random then $a \oplus b$ will be random. Or will it? Note that the protocol forces us to specify an order in which the players exchange their messages (indeed, it is never wise to attempt to speak simultaneously over the phone). Here we made Alice go first, and Bob second. So Bob receives Alice's message a before he sends her his choice of b. But then he can easily force any outcome b' of his choice by setting $b = b' \oplus a$: the protocol is completely insecure.

Unfortunately this is the fate of *any* classical protocol for coin flipping: no value of $\varepsilon < 1/2$ can be achieved for security! That is, if one player cannot completely bias the outcome of the protocol to a certain value, then the other player can: there is always at least one of Alice or Bob who can perfectly cheat. Informally, the reason is the same that makes the Blum protocol insecure: one can argue that, whatever the outcome c of the protocol, it has to be determined at *some* point in the protocol. By considering the messages exchanged from the last to the first, one can find a message such that, before the message is sent the outcome is not yet determined, but once the message has been sent it is (in the case of the Blum protocol, this would be Bob's message). But then whomever sends that message has the ability to bias the outcome to any possibility.

8.4.2 Quantum coin flipping

The impossibility argument given in the previous section does not immediately apply to quantum protocols, as for a quantum protocol the notions of a transcript, and the outcome being determined, are much less clear: everything can happen in superposition. And for once, there is good reason: strong coin-flipping is possible using quantum information, at least for some values of $\varepsilon < 1/2$. Let's see an example protocol, discovered by Aharonov et al. [Aha+00].

Protocol 4 — ATVY coin-flipping. For $a, x \in \{0,1\}$ define the qutrit

$$|\phi_{a,x}\rangle = \frac{1}{\sqrt{2}}|0\rangle + (-1)^x|a+1\rangle .$$

1. Alice selects $x \in \{0,1\}$ and $a \in \{0,1\}$ uniformly at random and sends $|\phi_{a,x}\rangle$ to Bob.
2. Bob selects $b \in \{0,1\}$ uniformly at random and sends b to Alice.
3. Alice sends a and x to Bob.
4. Bob verifies the state he received from Alice in step 1. is $|\phi_{a,x}\rangle$ (e.g. by measuring in any orthonormal basis containing $|\phi_{a,x}\rangle$). If it is not the case then he declares that Alice has been cheating and aborts the protocol.
5. Both players return the outcome $c = a \oplus b$.

Note the similarity between this protocol and the Blum protocol we saw earlier. Here as well Alice and Bob each choose "half" of the outcome c: Alice chooses a, Bob b, and they return $c = a \oplus b$. However, Alice does not fully reveal a to Bob in her first message: instead, she provides him with some form of "weak commitment" to a in the form of the state $|\phi_{a,x}\rangle$. Because the four states $|\phi_{a,x}\rangle$ are not orthogonal, it is impossible for Bob to completely discover the value of a without being revealed x first, which only happens after he has had to make his choice of b.

Exercise 8.4.1 Compute the reduced density matrices $\rho^B_{|a=0}$ and $\rho^B_{|a=1}$ associated with Bob's view of the protocol after Alice's first message has been sent, for a uniformly random choice of $x \in \{0,1\}$. Show that the the trace distance between these two matrices is $1/2$, and conclude that the probability with which Bob can force an outcome c of his choice is at most $3/4$. ■

The exercise shows that the maximum bias that a cheating Bob can induce in the protocol is $\varepsilon = 1/4$. Security for cheating Alice is a bit harder to argue, because we have to consider the possibility for her to prepare an arbitrary state in the first step, which may be entangled with some information she keeps on the side and uses, together with the value b received from Bob, to determine her message in the third step of the protocol. We will not give the details here, but the result is the following:

Theorem 8.4.1 — (Amb01). The ATVY coin-flipping protocol is correct and ε-secure for $\varepsilon = 1/4$.

Can we do even better? Unfortunately it turns out that perfectly secure strong coin-flipping is also impossible for quantum protocols: Kitaev showed that the smallest bias any protocol could achieve is $\varepsilon = (\sqrt{2}-1)/2 \approx 0.207$. Kitaev's proof is an extension of the classical impossibility argument, based on an ingenuous representation of transcripts for quantum protocols; see [Amb+04] for details. If you are interested in Section 8.5 below we give a "dual" argument to Kitaev's, based on [GW07].

The good news, though, is that Kitaev's bound is achievable: for any $\varepsilon > 0$ there is a quantum strong coin-flipping protocol with bias $(\sqrt{2}-1)/2+\varepsilon$. The protocol achieving this, however, is rather complex. It is based on the notion of *weak coin flipping*, that we take a look at next.

8.4.3 Weak coin flipping

The task of weak coin flipping is defined as strong coin flipping, except the security requirement is weaker: instead of requiring that neither player can force $p(c=0) \geq 1/2+\varepsilon$ or $p(c=1) \geq 1/2+\varepsilon$, we only require that malicious Alice cannot force $p(c=0) \geq 1/2+\varepsilon$, and malicious Bob cannot force $p(c=1) \geq 1/2+\varepsilon$. That is, we assume a priori that the malicious behavior of each player will always try to achieve a certain pre-determined outcome. For instance, in the example we used earlier, the outcome of the coin flip determines who has to accomplish a certain chore: if $c=0$ it will be Bob, and if $c=1$ it will be Alice. In this scenario the only thing we're really worried about is that Alice would manage to increase the probability that $c=0$, while Bob would increase the probability that $c=1$. So all we need is a weak coin flipping protocol.

It turns out that weak coin flipping, with arbitrarily small (but non-zero) bias ε, is indeed possible [Moc07]. The best protocol known for achieving this, however, remains very complex, and requires a large number of rounds of interaction, that scales exponentially with $1/\varepsilon$ [Aha+16]. (It is known that some dependence on $1/\varepsilon$ is necessary, but it is an open problem to do better than exponential.) Chailloux and Kerenidis [CK09] showed that any weak coin flipping protocol with bias ε could be used to build a strong coin flipping protocol with bias $(\sqrt{2}-1)/2+O(\varepsilon)$, thereby matching Kitaev's lower bound.

8.5 Kitaev's lower bound on strong coin flipping

In this section we give a proof of Kitaev's lower bound on the bias of secure protocols for strong coin flipping, based on [GW07]. The argument relies on simple notions of linear and semidefinite programming with which you may not already be familiar. If so we encourage you to read a bit about these techniques, as the proof is very elegant and well worth understanding. But if you don't have the time then you can skip the section: it is not required for the course.

For any coin-flipping protocol, define p_{1*} as the probability that Alice outputs a 1, maximized over all possible (cheating) strategies for Bob. Define p_{*1} symmetrically. Then the condition for the protocol to be a secure strong coin-flipping protocol with bias ε is that both $p_{1*}, p_{*1} \in [1/2-\varepsilon, 1/2+\varepsilon]$. (This is equivalent to the condition $p_{1*}, p_{*0} \in [1/2-\varepsilon, 1/2+\varepsilon]$ we introduced earlier.)

Theorem 8.5.1 — Kitaev. For any strong coin-flipping protocol, we have $p_{1*}p_{*1} \geq \frac{1}{2}$.

Note that the condition in the theorem immediately implies that any strong coin flipping protocol has bias at least $(\sqrt{2}-1)/2$, as claimed.

Kitaev's theorem applies to both classical and quantum protocols. We'll see the proof for classical protocols first, and then move to the quantum setting. Both proofs have the same structure: Bob's maximum cheating probability can be expressed as the optimum of a linear program (LP) (or semidefinite program (SDP) in the quantum case), and similarly for Alice's. Any feasible solution to the duals of each LP provides an upper bound on the probability of success of the cheating strategy.

The crucial insight is that the cheating probabilities need to be considered *together*, through the quantity $p_{1*}p_{*1}$: a good upper bound on this quantity expresses the fact that, either Alice can force Bob to output a 1, or, if she can't, then it must be that Bob can force her to produce a 1. We will obtain a bound on this bias by taking the product of some of the dual LP (or SDP) constraints. Let's proceed with the details.

8.5.1 The bound on classical protocols

Fix a classical protocol. We can think of the protocol as a tree, where each node is indexed by a variable u representing the transcript that led to this node: if we are in node u, and Alice plays by sending a message a, then we arrive at node (u,a). The honest protocol is given by probabilities $p_A(a|u)$, $p_B(b|u)$, which are Alice's (resp. Bob's) transition probabilities. Given that Alice is honest, Bob's maximum cheating probability can be expressed as a linear program LP$_B$, in which the variables $p_B(u)$ represent the probability of reaching node u, when Alice is honest and Bob cheats. Bob's goal is to maximize the probability of reaching a leaf labeled with a 1; denote this set L_1. We introduce constraints to express the fact that Bob can choose any distribution on edges when it is his turn to play, but he has to follow Alice's distribution when it is her turn.

$$\begin{array}{lll}
(\mathrm{LP}_B,\ \text{primal}) & \max \displaystyle\sum_{u\in L_1} p_B(u) & \\
& p_B(u)p(a|u) = p_B(u,a) & \forall a,\ \forall u \text{ node for Alice} \\
& p_B(u) = \displaystyle\sum_b p_B(u,b) & \forall u \text{ node for Bob} \\
& p_B(0) = 1 & \\
& p_B(u) \geq 0 & \forall u
\end{array}$$

To write the dual of this linear program, introduce variables $Z_A(u,a)$ for the first set of constraints and $Z_A(u)$ for the second set. With a little work the dual can be written in the form

$$\begin{array}{lll}
(\mathrm{LP}_B,\ \text{dual}) & \min \quad Z_A(0) & \\
& Z_A(u) \geq \displaystyle\sum_a p(a|u)Z_A(u,a) & \forall u \text{ node for Alice} \\
& Z_A(u) \geq Z_A(u,b) & \forall b,\ \forall u \text{ node for Bob} \\
& Z_A(u) \geq 1 & \forall u \in L_1
\end{array}$$

$Z_A(u)$ can be interpreted as the maximum probability with which Bob can cheat, starting at node u. We can consider another linear program LP$_A$, this time for a cheating Alice, which is completely symmetrical. The interpretation of the variables Z_A, Z_B motivates the introduction of the quantity

$$F_\ell = \mathrm{E}_{u\sim\ell}\big[Z_A(u)Z_B(u)\big] \tag{8.1}$$

where $u \sim \ell$ is shorthand for u being taken according to the probability distribution on nodes at depth ℓ which arises from the honest protocol. In this expression, $Z_A(u)Z_B(u)$ should be interpreted as the bias that cheating players can achieve, if any of them starts cheating at node u.

Let Z_A, Z_B be optimal solutions to the duals of LP$_B$ and LP$_A$ respectively. The last constraint of the dual implies that without loss of generality we can assume that both Z_A and Z_B are both exactly 1 at all leaves labeled with a 1 (as if they were larger, a better solution to the LP could be obtained by scaling). Hence if n is the last level of the game, then $F_n = p_{1,1} = 1/2$. Moreover, strong duality implies that $F_0 = p_{1*}p_{*1}$. Finally, by multiplying out the constraints of the two duals one easily gets that $F_\ell \geq F_{\ell+1}$, which proves Theorem 8.5.1 for the case of classical protocols.

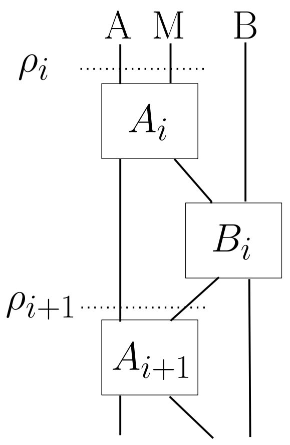

Figure 8.1: Step i in the coin flipping protocol

8.5.2 The bound on quantum protocols

For quantum protocols the bound follows analogously, with a few tweaks. A protocol is modeled by a series of unitary operations A_i for Alice and B_i for Bob. The players are assumed to start in the $|0\dots0\rangle$ state. At the end of the interaction, they measure using $\{\pi_A, \mathbb{I}-\pi_A\}$, $\{\pi_B, \mathbb{I}-\pi_B\}$ respectively and return the outcome. The correctness requirement is that

$$p_{1,1} = \|(\pi_A \otimes \mathbb{I}_M \otimes \pi_B)B_nA_n\cdots B_1A_1|0\dots0\rangle\|^2 = 1/2$$

and $p_{0,0}$, defined symmetrically using $(\mathbb{I}-\pi_A),(\mathbb{I}-\pi_B)$ as the measurements, is also $1/2$. The following SDP captures the maximum cheating probability for Bob:

$$\begin{array}{lll}
(\text{SDP}_B,\ \text{primal}) & \max & \langle \pi_B \otimes \mathbb{I}, \rho_n\rangle \\
& \text{Tr}_M(\rho_{i+1}) = \text{Tr}_M(A_i\rho_iA_{i+1}^\dagger) & \forall i \\
& \rho_0 = |0\rangle\langle 0|_{A\otimes M} & \\
& \rho_i \geq 0 & \forall i
\end{array}$$

Here ρ_i represents the state of Alice's and the message's registers, right before Alice performs her i-th action (see Figure 8.1). The dual of this SDP is

$$\begin{array}{lll}
(\text{SDP}_B,\ \text{dual}) & \min & \langle 0|Z_A(0)|0\rangle \\
& Z_A(i)\otimes \mathbb{I}_M \geq A_{i+1}^\dagger(Z_A(i+1)\otimes\mathbb{I}_M)A_{i+1} & \forall i \\
& Z_A(n) = \pi_A & \\
& Z_A(i) = (Z_A(i))^\dagger & \forall i
\end{array}$$

Let $|\Psi_\ell\rangle$ be the state of the whole system at the ℓ-th round, assuming honest play. Then the analogue of (8.1) is

$$F_\ell = \langle \Psi_\ell|Z_A(\ell)\otimes\mathbb{I}_M\otimes Z_B(\ell)|\Psi_\ell\rangle \tag{8.2}$$

Strong duality then implies the condition $F_0 = p_{1*}p_{*0}$, while $F_n = 1/2$. The relation $F_\ell \geq F_{\ell+1}$ follows from the dual constraints, and we are done.

Acknowledgments

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International Licence. The lecture notes are written by Nelly Ng, Thomas Vidick and Stephanie Wehner. We thank David Elkouss, Kenneth Goodenough, Jonas Helsen, Jérémy Ribeiro, and Andrea Coladangelo for proofreading.

Bibliography

[Aha+00] Dorit Aharonov et al. "Quantum bit escrow". In: *Proceedings of the thirty-second annual ACM symposium on Theory of computing*. ACM. 2000, pages 705–714 (cited on page 11).

[Aha+16] Dorit Aharonov et al. "A Simpler Proof of the Existence of Quantum Weak Coin Flipping with Arbitrarily Small Bias". In: *SIAM Journal on Computing* 45.3 (2016), pages 633–679 (cited on page 12).

[Amb+04] Andris Ambainis et al. "Multiparty quantum coin flipping". In: *Computational Complexity, 2004. Proceedings. 19th IEEE Annual Conference on*. IEEE. 2004, pages 250–259 (cited on page 12).

[Amb01] Andris Ambainis. "A new protocol and lower bounds for quantum coin flipping". In: *Proceedings of the thirty-third annual ACM symposium on Theory of computing*. ACM. 2001, pages 134–142 (cited on page 11).

[Ben+91] Charles H Bennett et al. "Practical quantum oblivious transfer". In: *Annual International Cryptology Conference*. Springer. 1991, pages 351–366 (cited on page 6).

[Blu83] Manuel Blum. "Coin flipping by telephone a protocol for solving impossible problems". In: *ACM SIGACT News* 15.1 (1983), pages 23–27 (cited on page 10).

[Bra+93] Gilles Brassard et al. "A quantum bit commitment scheme provably unbreakable by both parties". In: *Foundations of Computer Science, 1993. Proceedings., 34th Annual Symposium on*. IEEE. 1993, pages 362–371 (cited on page 7).

[CK09] André Chailloux and Iordanis Kerenidis. "Optimal quantum strong coin flipping". In: *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*. IEEE. 2009, pages 527–533 (cited on page 12).

[FS09] Serge Fehr and Christian Schaffner. "Composing quantum protocols in a classical environment". In: *Theory of Cryptography Conference*. Springer. 2009, pages 350–367 (cited on page 4).

[Gol05] Oded Goldreich. *Foundations of cryptography: a primer*. Volume 1. Now Publishers Inc, 2005 (cited on page 4).

[GW07] Gus Gutoski and John Watrous. "Toward a general theory of quantum games". In: *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*. ACM. 2007, pages 565–574 (cited on page 12).

[LC97] Hoi-Kwong Lo and Hoi Fung Chau. "Is quantum bit commitment really possible?" In: *Physical Review Letters* 78.17 (1997), page 3410 (cited on page 9).

[LP09] Yehuda Lindell and Benny Pinkas. "Secure multiparty computation for privacy-preserving data mining". In: *Journal of Privacy and Confidentiality* 1.1 (2009), page 5 (cited on page 4).

[May97] Dominic Mayers. "Unconditionally secure quantum bit commitment is impossible". In: *Physical review letters* 78.17 (1997), page 3414 (cited on page 9).

[Moc07] Carlos Mochon. "Quantum weak coin flipping with arbitrarily small bias". In: *arXiv preprint arXiv:0711.4114* (2007) (cited on page 12).

[PS10] Rafael Pass and Abhi Shelat. *A Course in Cryptography*. Lecture notes available at `http://www.cs.cornell.edu/courses/cs4830/2010fa/lecnotes.pdf`. 2010 (cited on pages 6, 10).

[Unr10] Dominique Unruh. "Universally composable quantum multi-party computation". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2010, pages 486–505 (cited on pages 4, 8).

[Wat06] John Watrous. *Impossibility of quantum bit commitment*. Lecture notes from the Winter 2006 course "Introduction to Quantum Computing". 2006 (cited on page 9).