12 Delegated computation

Virtually all the cryptographic protocols that we studied in this book can be implemented using very simple equipment: essentially, a way to create and manipulate single qubits, to send them over a dedicated channel, and to measure (and sometimes store) them at the receiving end. And indeed, as discussed in Chapter **??** such equipment is already available today, making these protocols particularly appealing. Nevertheless, as experimental capabilities start to scale up to quantum computers of larger and larger sizes, cryptographic tasks that may involve more complex quantum computations start to become relevant. In this chapter we study the most fundamental such task, the problem of delegated computation. Delegated computation is a two-party task where there is a large asymmetry between the two parties: on the one hand, Alice would like to execute a quantum computation, but she does not have a powerful enough quantum computer to execute it. On the other hand, Bob has a quantum computer, but he is not trusted by Alice. Can Alice make sure that Bob executes her computation correctly for her? In the chapter we will see three very different approaches to this problem. Studying them gives us a good opportunity to introduce some basics of quantum computation, which you may find useful as you continue to study more and more complex tasks.

12.1 Definition of the task

Week 10, Lecture 10.4,Lecture 1: Delegating quantum circuits

Let's formalize the problem of delegated computation and its security guarantees. Suppose a user Alice has a quantum circuit $\mathcal{C}$ in mind, that she would like to execute on some input x in order to learn the outcome $\mathcal{C}(x)$. All of Alice's data, x and $\mathcal{C}$, is classical: we will assume that x is a classical bit string, and $\mathcal{C}$ is specified by a sequence of two-qubit gates. We can also assume that the outcome is classical: let's say it is the outcome of a standard basis measurement performed on a specially designated output qubit of $\mathcal{C}$. For simplicity we further assume that this circuit always returns deterministic outputs, and write $\mathcal{C}(x)$ for the outcome.[1]

Now, unfortunately Alice herself does not have a universal quantum computer! Maybe she has a tiny desktop machine that lets her play around with BB'84-like operations: prepare or measure single qubits, possibly store a couple qubits at a time in memory, but no more. Luckily Alice has the possibility of buying computation time on a quantum server, appropriately named Bob, with which she could interact over the internet, or maybe even over a simple BB'84-type quantum communication channel that allows the exchange of one qubit at a time. So Alice could send x and the description of $\mathcal{C}$ to Bob, who would perform the computation and return the outcome — right?

Remember that this is a crypto book! Alice might not trust Bob. For one she'd like to have a way to verify that the outcome provided to her is correct. What if Bob is lazy and systematically claims that the outcome of her computation, $\mathcal{C}(x)$, equals '0'? Since Alice has no quantum computer herself she has no means of checking this! A second property Alice could require is that the computation remains private:

[1] For example, $\mathcal{C}$ could be a circuit for factoring.

while she certainly wants to learn $\mathcal{C}(x)$, she'd rather not let Bob know that she is interested in circuit $\mathcal{C}$, or in input x, as these might contain private data.

Let's restate these conditions as the requirements that the computation is *correct*, *verifiable* and *blind*.

Definition 12.1.1 (Delegated computation) *In the task of delegated computation, a user Alice has an input $(x, \mathcal{C})$, where x is a classical string and $\mathcal{C}$ the classical description of a quantum circuit. Alice has a multiple-round interaction with a quantum server Bob. At the end of the interaction, Alice either returns a classical output y, or she aborts. A protocol for delegated computation is called:*

- Correct *if whenever both Alice and Bob follow the protocol, Alice accepts (she does not abort) and $y = \mathcal{C}(x)$.*
- Verifiable *if for any server deviating from the protocol, Alice either aborts or returns $y = \mathcal{C}(x)$.*
- Blind *if for any server deviating from the protocol, at the end of the protocol the reduced density matrix that describes the entire state of the server is independent of Alice's input $(x, \mathcal{C})$.*

Each of these properties is stated informally. In particular, to make them formal we would have to introduce more quantitative versions, writing that "with high probability," and "almost independent," etc. We learned how to do this on many examples throughout the book. Delegated computation is the most complex task that we see so far, and to focus on the essentials we give ourselves a break and accept the informal definition for now.

Remark 12.1.1 *In spite of being rather similar neither of the properties of verifiability or blindness directly implies the other. In practice verifiability ofton follows from blindness by arguing, using "traps", that if a protocol is already blind, the server's trustworthiness can be tested by making it run "dummy" computations for which Alice already knows the output, without the server being able to distinguish whether it is asked to do a real or dummy computation. We will see an example of this technique later on.*

Are there good protocols for delegating quantum computations? It turns out that we don't have a fully satisfactory answer yet: this is an active area of research! In this chapter we'll outline three of the most prominent approaches. The first construction shows how arbitrary quantum circuits can be delegated, as long as the verifier has the ability to prepare certain specific single-qubit states and communicate them to the server. The second construction achieves a similar result, using a very different idea: *measurement-based* quantum computation. The third construction itself has a wholly different flavor. It achieves delegated computation by a purely classical Alice, with no quantum capabilities whatsoever. However, the downside is that Alice now has to interact with *two* isolated Bob's, which moreover need to share entanglement. This third method relies on similar techniques as we have seen in the analysis of device-independent QKD in Chapter**??**.

12.1.1 Preliminaries on efficient quantum computation

Before we find out how to delegate quantum computations, let's first review briefly what a quantum computation *is*. As hinted earlier, there are many distinct models for quantum computation, each capable of universal computation (and thus of simulating each other).

The most straightforward model is called the quantum circuit model. Here a computation is represented by the action of a circuit $\mathcal{C}$ on n input qubits. The input qubits are initialized in an arbitrary quantum state that contains the input of the computation. Here we will only work with classical inputs of the form $|x\rangle$, where $x \in \{0,1\}^n$, but quantum inputs can be considered as well. The circuit $\mathcal{C}$ itself is specified by a list of m gates, which are one- or two-qubit unitaries that act on a subset of the qubits. For instance, a simple 4-qubit circuit could be specified by the ordered list $((H,1),(\mathrm{CNOT},(2,3)),(H,3),(Z,4))$. For simplicity we'll look only at circuits that return a single it of output, which we'll take to be the result of a standard basis measurement of the first qubit. We will write $\mathcal{C}(x)$ for the outcome of a measurement of the output qubit of $\mathcal{C}$ in the standard basis, when $\mathcal{C}$ is executed on the state $|x\rangle$. In general this would be a random value but here we'll focus on circuits such that $\mathcal{C}(x)$ takes a determinitic value for its output qubit.

In general we allow the circuit $\mathcal{C}$ to be based on any family of single- or two-qubit gates. An important theorem in quantum computing, the Solovay-Kitaev theorem, states that it is in fact possible to restrict the set of gates used to some simple sets of gates, called "universal gate sets": a gate set is universal if any circuit that can be implemented using *some* set of gates can also be implemented using that particular set of gates. Moreover, the number of gates required should be at most polynomially larger. An example of a universal gate set that we will use later on is the set

$$\mathcal{G} = \left\{ G = \begin{pmatrix} \cos(\pi/8) & -\sin(\pi/8) \\ \sin(\pi/8) & \cos(\pi/8) \end{pmatrix}, \, \mathrm{CNOT} \right\}$$

where G implements a $\pi/4$ rotation around the y axis of the Bloch sphere and CNOT is a controlled-X operation (on any two qubits of the circuit). Another example of a popular universal gate set is the set

$$\mathcal{G}' = \left\{ H, \, T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}, \, \mathrm{CNOT} \right\},$$

and there are many others.

We will use one more useful feature of quantum (as well as classical) circuits, which is the notion of a "universal" circuit. Informally, a universal circuit is a circuit that can simulate the execution of any other circuit, provided that the other circuit's source code is provided to the universal circuit as input.

Theorem 12.1.2 (Universal circuit) *For any integer n and size parameter s there exists a fixed circuit $\mathcal{C}_U$ acting on $n+m$ qubits, where m is at most a polynomial in n and s, such that the following holds. For any circuit $\mathcal{C}$ of size at most s expressed using the gate set $\mathcal{G}$, and any input $x \in \{0,1\}^n$ to $\mathcal{C}$, there is a $z \in \{0,1\}^m$ which can be efficiently computed from $\mathcal{C}$ and is such that $\mathcal{C}_U(x,z)$ has the same distribution as $\mathcal{C}(x)$.*

12.1.2 The Pauli group and Clifford gates

We've already met the 4 single-qubit Pauli matrices, $\mathcal{P} = \{I,X,Y,Z\}$. These form a group, in the sense that for any two Pauli matrices P and Q, the product PQ is again a Pauli matrix. A single-qubit *Clifford gate U* is any operation that preserves the Pauli group, in the sense that $UPU^\dagger$ is a Pauli matrix for any Pauli P. If U is a two-qubit gate, we similarly require that $UPU^\dagger \in \pm\mathcal{P}^{\otimes 2}$ for any P that is a tensor product of two Paulis.

Exercise 12.1.1 Verify that the Pauli matrices are Clifford gates. Show that the Hadamard, phase $P = T^2$ and CNOT gates are Clifford gates. Do you see other examples? Is the G gate also a Clifford gate? How about the T gate?

The defining property of Clifford gates is very useful, and plays an important role in delegated computation — we will see why. Unfortunately it turns out that there is no universal gate set made only of Clifford gates: any universal set of gates for quantum computation must include at least one non-Clifford gate. This will be a source of many headaches when trying to implement delegated computation.

12.2 Verifiable delegation of quantum circuits

Our first approach to delegated computation is based on the idea of *computing on encrypted data*. Recall the quantum one-time pad from Chapter **??**. Suppose that we have an n-qubit density matrix ρ. To encrypt it using the one-time pad we select two n-bit strings $a, b \in \{0,1\}^n$ uniformly at random, and return $\tilde{\rho} = X^a Z^b \rho (X^a Z^b)^\dagger$, where $X^a = X^{a_1} \otimes \cdots \otimes X^{a_n}$ denotes applying a Pauli X operator on all qubits i such that $a_i = 1$, and the identity on all other qubits. The notation for Z^b is similar, with Pauli Z operators instead. If ρ represents the input x to the circuit, $\rho = |x\rangle\langle x|$, then the result of applying the quantum one-time pad is still classical, $\tilde{\rho} = |x \oplus a\rangle\langle x \oplus a|$.

So here is an idea: Alice can encrypt her input x into a quantum one-time padded state $\tilde{\rho}$, and send $\tilde{\rho}$ to Bob. If she keeps a copy of the strings a, b and does not communicate them to Bob then her input x remains perfectly private. This is a good start: we already understand how we can use the server, Bob, as a quantum memory and maintain privacy of our (classical) input.

Of course there is much more that we would like to do: Alice wants to make Bob execute a circuit $\mathcal{C}$. Can she somehow guide him through this by working directly on $\tilde{\rho}$? For this we need to find a quantum operation $\tilde{\mathcal{C}}$ that the server could apply, such that $\tilde{\mathcal{C}}(\tilde{\rho}) = \widetilde{\mathcal{C}(\rho)}$, an encrypted version of $\mathcal{C}(\rho)$ from which Alice can recover the real output $\mathcal{C}(x)$. (And of course, we want even more—we want $\tilde{\mathcal{C}}$ to hide $\mathcal{C}$, to achieve blindness, and we haven't even discussed verifiability yet—one step at a time!)

The circuit $\mathcal{C}$ can always be expressed using gates from a universal gate set, for example the set $\mathcal{G}' = \{H, \mathrm{CNOT}, T\}$ introduced earlier. Even though it is not needed, to warm up let's assume that we also allow X gates, and that the first gate in $\mathcal{C}$ is an X applied on the first qubit. Now notice that

$$X\big(X^a Z^b \rho (X^a Z^b)^\dagger\big)X^\dagger = X^{a\oplus e_1} Z^b \rho (X^{a\oplus e_1} Z^b)^\dagger = X\tilde{\rho}X^\dagger,$$

where e_1 is the bit string with a single 1 in the first position. This equations shows that Bob can in fact directly apply the X gate on $\tilde{\rho}$, and the effect is as if it had been applied directly on the real ρ! So an X gate is easy, and you can check that any Pauli gate, single- or multi-qubit, will be similarly easy. The main property that is used here is that different Paulis either commute or anti-commute with each other. In other words, Pauli gates can be "commuted" past each other with at most a sign change, which disappears as a global phase.

Let's move one step further and consider a Clifford gate. Let's take the example of a Hadamard gate on the second qubit, H^{e_2}, with e_2 the n-bit string $e_2 = (0, 1, 0, \ldots, 0)$, i.e. $H^{e_2} = \mathbb{I} \otimes H \otimes \mathbb{I} \otimes \cdots \mathbb{I}$. Using the equation $HXH = Z$, we get

$$(X^a Z^b)(H^{e_2} \rho (H^{e_2})^\dagger)(X^a Z^b)^\dagger \;=\; (-1)^{a_2 b_2} H^{e_2} X^{a'} Z^{b'} \rho (X^{a'} Z^{b'})^\dagger (H^{e_2})^\dagger \;,$$

where (a', b') is obtained from (a, b) by exchanging the bits a_2 and b_2. We see that if Alice instructs Bob to apply an H gate on the second *encrypted* qubit, the effect is the same as if the server had applied the H gate directly on the second *unencrypted* qubit — as long as she updates her one-time pad key (a, b) to (a', b') as described above. As long as Alice does this simple classical operation on her side, when Bob returns the encrypted qubits she will be able to undo the one-time pad to recover the correct outcome of the computation.

The following exercise asks you to show that a similar trick can be employed for any Clifford gate.

Exercise 12.2.1 Let U be any one- or two-qubit Clifford gate. Show that the effect of applying U to the encrypted state $\tilde{\rho}$ is equivalent to the application of U on ρ, up to an update rule on the one-time pad key (a, b). Work out the update rule in the case of the phase and CNOT gates.

So Alice can orchestrate the whole computation with Bob, while only having to keep track of simple updates on her one-time pad keys? Unfortunately, remember from Section 12.1.2 that no set of Clifford gates is universal — we need to show how to implement one more gate, for example the T gate considered in the universal set $\mathcal{G}'$. Because the T gate is non-Clifford, applying it to the encrypted state $\tilde{\rho}$ will have a more complicated effect, that we can't keep track of by a simple modification of the one-time pad keys. Instead, we'll show how Alice can make the server implement a T gate on the encrypted state by using the idea of *magic states*.

Quiz 12.2.1 *Suppose that a server applies a T gate directly to an encrypted qubit $X^aZ^b|\psi\rangle$. Which of the states below represents the resulting state? Recall that the T gate is $T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{pmatrix}$, and $P = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$.*

a) $X^aZ^{a\oplus b}PT|\psi\rangle$
b) $X^aZ^{a\oplus b}P^{a\oplus b}T|\psi\rangle$
c) $\longrightarrow X^aZ^{a\oplus b}P^aT|\psi\rangle$
d) $X^aP^aTZ^b|\psi\rangle$

Quiz 12.2.2 *$P^cX^aZ^b|\psi\rangle$ is equal to:*

a) $X^aZ^aP^c|\psi\rangle$
b) $X^aZ^{a\oplus b}P^c|\psi\rangle$
c) $X^aZ^{a\oplus b\oplus c}P^c|\psi\rangle$
d) $\longrightarrow X^aZ^{a\cdot c\oplus b}P^c|\psi\rangle$

12.2.1 Computation with magic states

Week 10, Lecture 10.4,Lecture 2: Computing using magic states

The idea for magic states is that the computation of certain complicated gates on an arbitrary state can be replaced by a simple computation using certain auxiliary states called "magic states" as some form of catalyst. Let's see this for the T gate, the only gate that we still need to figure out how to implement. The magic state we need is

$$|\pi/4\rangle = T|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{e^{i\pi/4}}{\sqrt{2}}|1\rangle\,. \tag{12.1}$$

Preparing this state itself requires applying a T gate. But the point is that we only need to apply the gate to a fixed, known input state, that is independent of the state $|\psi\rangle$ on which we really want to apply the T gate. So the preparation of single-qubit magic states is a relatively simple and computation-independent task that Alice should be able to perform herself, as long as she has access to a small single-qubit quantum computer.

Suppose we are given a single-qubit state $|\psi\rangle$ on a register A_1, and initialize a second qubit in register A_2 in the $|\pi/4\rangle$ state. Consider the following circuit: first, apply a CNOT, controlled on A_2 and acting on A_1. Second, measure register A_1 in the computational basis, obtaining an outcome c. Third, if $c = 1$ we apply a P gate (recall that $P = T^2$) to register A_2, and if $c = 0$ we do nothing (see Figure 12.1). What is the post-measurement state in register A_2? It is a good exercise to verify that this is $X^cZ^cT|\psi\rangle$. That is, up to the possible apparition of a product (XZ) in front (depending on the value obtained for c) the effect of our small circuit is that the T gate got applied to $|\psi\rangle$. Moreover, the only resources we used to achieve this are a magic state, a CNOT gate, a measurement in the computational basis, and a controlled-P gate. Since the P gate is a Clifford gate, applying this transformation to all T gates in a circuit allows to transform any quantum circuit into a circuit that only uses magic states and Clifford gates (with adaptive measurements).

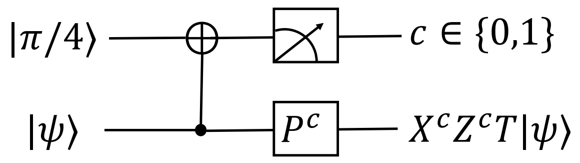

Fig. 12.1 Teleporting into a T gate. The state $|\pi/4\rangle$ is defined in (12.1).

Exercise 12.2.2 Suppose that instead of being applied directly to the state $|\psi\rangle$, the circuit described above is applied to an encrypted version of $|\psi\rangle$, $X^aZ^b\,|\psi\rangle$. Show that the outcome of the circuit is then $P^aX^{a'}Z^{a'}T\,|\psi\rangle$, for some bits a' and b' depending on a, b and c. (Convince yourself that the same calculation works out in case $|\psi\rangle$ is not pure, but a reduced density ρ on a single qubit.)

Note the P gate that we picked up in the exercise. This also needs to be corrected. But the gate is applied on the encrypted state. So Alice could instruct Bob to apply $(P^a)^\dagger$ directly, to remove the gate. Unfortunately this would require revealing the bit a, which is part of Alice's secret one-time pad key. There is a way around this that involves adding a little bit of randomization in the choice of magic state we use (essentially, considering a one-time padded magic state). This will guarantee that the phase correction is always independent of the one-time pad key.

We now have the outline of a delegated computation protocol. Alice first prepares a one-time padded version of her classical input x, and sends it to Bob. She also prepares many (one-time padded) magic states, and sends them to Bob as well. Finally, Alice and Bob both go through the circuit $\mathcal{C}$ one gate at a time. For Clifford gates, Bob applies the gate directly on the encrypted qubit, and Alice updates her keys as in Exercise 12.2. For a T gate, Bob executes the circuit from Figure 12.1 and sends the outcome c back to Alice, who tells him to apply a gate $(P)^\dagger$ or not, depending on her one-time pad keys.[2]

12.2.2 Blindness

Is our delegation protocol blind? We were pretty careful to ensure that Alice's input x remains perfectly private. However, to implement the protocol she needs to completely reveal her circuit $\mathcal{C}$! Luckily there is a simple way out: Alice can instruct Bob to execute a fixed "universal" circuit (as in Theorem 12.1.2), and instead encode the actual circuit $\mathcal{C}$ she is interested in as part of the input x.

12.2.3 Verifiability

So our protocol satisfies the blindness property (provided we use a universal circuit), but so far it is not verifiable: Alice has no guarantee that Bob performs the required computation! Indeed, no check is performed at all. For all we know, Bob never does anything and simply returns the original $\tilde{\rho}$ back to Alice at the end of the protocol. So what kind of test could we make to check that Bob is not being lazy (or even malicious)?

[2] This last step was not described in detail; believe us that it can be done without leaking information about Alice's keys—or check the chapter notes for details!

The idea is to combine the protocol with some "test runs". The original protocol is now called a "computation run". In contrast, in a test run the computation is set up in such a way that Alice knows what the outcome should be, and she will check that the Bob returns the correct value (after decoding). But Bob will not be able to distinguish test runs from computation runs, and as a consequence we'll have the guarantee that Bob is also being honest in a computation run.

There are two types of test runs, X-test and Z-test. In an X-test run, the computation is executed on an encryption of the all-0 input $|0\rangle^{\otimes n}$. In a Z-test run, the same computation is executed on an encryption of $|+\rangle^{\otimes n}$. The main trick to ensure that Alice can keep track of the computation is that all gates in a test run are replaced by *identity* gates, without the prover noticing! Note that we already know how to do this for Pauli gates, as these do not involve the server anyways (Alice only has to update her one-time pad keys). The H gate requires a bit more work, but the idea is simple: since an H exchanges the standard basis and the Hadamard basis we can think of it as exchanging between an X-test run and an Z-test run. So in that case as well Alice can perfectly keep track of the state that the encrypted quantum state maintained by Bob should be in. The T gate, of course, is the interesting one. The idea is to modify the implementation described in Section 12.2.1 by changing the magic state, as well as the update rule, in a way that is un-noticeable by Bob but will result in an application of the identity gate instead of the T. The following exercise asks you to work out how this can be done.

Exercise 12.2.3 Consider the following procedure for implementing a T gate on the single-qubit state $|\psi\rangle_{A_1}$ using a magic state in register A_2. Alice first selects two bits $d, y \in \{0,1\}$ uniformly at random, and prepares the magic state $Z^d P^y T\,|+\rangle_{A_2}$, where $P = T^2$ is the phase gate. Alice sends register A_2 to Bob, who performs a CNOT controlled on A_1 and with target A_2. Bob measures A_1 in the computational basis, obtaining an outcome $c \in \{0,1\}$ that he sends to Alice. Alice then sends back $x = y \oplus c$ to Bob, who applies a gate P^x to the remaining system A_2.

1. Show that the state of A_2 at the end of this procedure is $X^c Z^{c(y\oplus 1)\oplus d\oplus y} T\,|\psi\rangle$, i.e. it is an encryption (using a key known to Alice) of $T\,|\psi\rangle$.

Next let's suppose we're doing a computation run, so that $|\psi\rangle = X^a\,|0\rangle_{A_1}$ for some $a \in \{0,1\}$. Alice would like to perform the identity instead of a T gate, without Bob noticing. This can be done by executing precisely the same circuit, except that the magic state is replaced by $X^d\,|0\rangle_{A_2}$ (it does not depend on y).

2. Show that with the magic state replaced by $X^d\,|0\rangle_{A_2}$ the interaction results in a register A_2 in state $X^d\,|0\rangle_{A_2}$. Show that in this case the outcome c of Bob's measurement is deterministically related to a and d in a simple way.
3. Can you find a similar modification, with a different magic state, that will implement the identity for the case of a Z-test run, where $|\psi\rangle_{A_1} = Z^b\,|+\rangle_{A_1}$ for some $b \in \{0,1\}$?

The exercise shows that simply by changing the magic state used in the implementation of the T gate, Alice can force that gate to act as identity in an X- or Z-test run. Moreover, due to the random bits d, y used in the preparation of the magic state you can verify that, from the point of view of Bob, these magic states look uniformly distributed, and thus he has no way of telling which "gadget" — for a T gate or the identity — he is really implementing.

In a test run Alice knows exactly what the outcome of the circuit should be, so she can verify the answer provided by Bob. Is this enough to ensure that Bob cannot cheat in a computation run? After all, we can imagine that Bob may be able to perform certain "attacks" that do not affect simple computations, where the state is always a tensor product of single qubits encoded in the computational or Hadamard bases, but such that the attack would perturb the kind of highly entangled states that will show up at intermediate stages in Alice's more complex circuit $\mathcal{C}$.

To show that this is not the case — that any significant attack will necessarily have a noticeable effect on either the X- or Z-test runs, the idea is to use an observation called the "Pauli twirl", that you are asked to work out in the next exercise.

Exercise 12.2.4 **Pauli twirl.** Let ρ be a single-qubit density matrix, and $P, P' \in \mathcal{P}$, where $\mathcal{P} = \{I, X, Y, Z\}$ is the set of single-qubit Pauli operators. Show that $\frac{1}{4}\sum_{Q\in\mathcal{P}}(Q^\dagger PQ)\rho(Q^\dagger(P')^\dagger Q)$ equals $P\rho P^\dagger$ if $P = P'$, and is 0 otherwise. Show that the same result holds for n-qubit Pauli operators.

The Pauli twirl allows us to argue that, thanks to the use of the quantum one-time pad, any "attack" of Bob boils down to the application of a Pauli operator at the last step of the circuit. Indeed, suppose first that the interaction performed between Alice and Bob results in the correct circuit $\mathcal{C}$ being implemented, except at the last step Bob applies an arbitrary "deviating unitary" U. Thus the outcome is $U\tilde{C}\tilde{\rho}\tilde{C}^\dagger U^\dagger$, where $\tilde{C}$ is the unitary Alice instructed Bob to implement, and $\tilde{\rho}$ the initial one-time-padded state sent by Alice. Due to the one-time pad, $\tilde{\rho}$ has the form $\tilde{\rho} = \sum_{Q\in\mathcal{P}} Q\,|x\rangle\,\langle x|\,Q^\dagger$, where $|x\rangle$ denotes the real input state that Alice would like the computation to be performed on. Moreover, for any Q there is a correction $c(Q) \in \mathcal{P}$ applied by Alice, which is such that $c(Q)\tilde{C}Q|x\rangle\langle x|Q^\dagger\tilde{C}^\dagger(c(Q))^\dagger = C|x\rangle\langle x|C^\dagger$. Thus, after applying $c(Q)$ to the corrupted circuit,

$$\begin{aligned}
&\sum_{Q\in\mathcal{P}} c(Q)U\tilde{C}Q|x\rangle\langle x|Q^\dagger\tilde{C}^\dagger U^\dagger(c(Q))^\dagger \\
&\quad= \sum_{Q\in\mathcal{P}} c(Q)U(c(Q))^\dagger c(Q)\tilde{C}Q|x\rangle\langle x|Q^\dagger\tilde{C}^\dagger(c(Q))^\dagger c(Q)U^\dagger(c(Q))^\dagger \\
&\quad= \sum_{Q\in\mathcal{P}} c(Q)U(c(Q))^\dagger C|x\rangle\langle x|C^\dagger c(Q)U^\dagger(c(Q))^\dagger \\
&\quad= \sum_{P\in\mathcal{P}} |\alpha_P|^2 PC|x\rangle\langle x|C^\dagger P^\dagger\ ,
\end{aligned}$$

where for the last step we decomposed $U = \sum_{P\in\mathcal{P}}\alpha_P P$ in the Pauli basis, and used the property of the Pauli twirl proved in Exercise 12.2.3.

This computation shows that any unitary applied by a malicious Bob at the end of the honest circuit is equivalent to a convex combination of Pauli operators. But any such non-trivial operator will be detected in either the X- or Z-test runs, as it will result in one of the outcomes being flipped in either the standard or Hadamard bases.

To conclude, we need to deal with the case where Bob applies a deviating unitary, not at the end of the circuit, but at some intermediate step. This case can be reduced to the previous one! Indeed, we can always think of a "purified" version of the whole protocol, where all measurements are deferred until the end. Suppose the unitary $\tilde{C}$ that Bob is supposed to implement decomposes as $\tilde{C} = \tilde{C}_2\tilde{C}_1$, and that Bob applies a deviating unitary U in-between the two circuits. The result can be written as

$$\tilde{C}_2U\tilde{C}_1 = (\tilde{C}_2U\tilde{C}_2^\dagger)\tilde{C}_2\tilde{C}_1 = (\tilde{C}_2U\tilde{C}_2^\dagger)\tilde{C},$$

where we used that $\tilde{C}_2$ is unitary, and hence $\tilde{C}_2^\dagger\tilde{C}_2 = \mathbb{I}$. Thus the deviation U is equivalent to applying another deviating unitary $U' = \tilde{C}_2U\tilde{C}_2^\dagger$ at the end of the circuit, and we are back to the analysis performed in the previous case: if the deviation has a non-trivial effect it will be detected by Alice in one of the test runs.

12.3 Delegation in the measurement-based model

Week 10, Lecture 10.5,Lecture 1: Delegation in the measurement-based model

Our second scheme for delegated computation has a similar flavor to the previous one, but at its heart it is based on a completely different approach to universal quantum computation. So far we have encountered the circuit model for performing quantum computations. From the point of view of computer science this is the most straightforward model, as it is a direct analogue of the classical circuit model on which the architecture of our (classical) computers is based. However, quantum information allows for other, much more exotic, models of computation. Many of these models were originally proposed with the idea that they might be more powerful than the circuit model, although ultimately they were proved equivalent. This includes the adiabatic model for computation and the measurement-based model that we will discuss in this section.

The highlight of measurement-based quantum computation (MBQC) is that it realizes any arbitrary quantum computation (specified by a circuit using some universal gate set) through an (adaptive) sequence of single-qubit measurements on a fixed, universal starting state. Seems impossible? Let's first give an overview of how this model works, and then we'll explain how MBQC can be used to achieve blind, verifiable delegated computation.

12.3.1 Measurement-based computation

Measurement-based computation is based on an idea very similar to *teleportation-based computation*, a model to which we return in the next section. This is the idea that a complete quantum computation, including the preparation of the initial state and the application of gates from a universal set, can be performed by making a sequence of adaptive measurements on a fixed universal state, successively "teleporting" the input state from one qubit to the next while at the same time applying unitary transformations on the state.

Let's do a simple example first. Suppose we have a qubit initialized in the state $|\psi\rangle_A = \alpha|0\rangle_A + \beta|1\rangle_A$. Suppose a second qubit is created in the state $|+\rangle_B$, and a CTL-Z operation is performed, controlling on the first qubit to perform a phase flip on the second. Then the joint state of the system becomes $|\psi\rangle_{AB} = \alpha|0\rangle_A|+\rangle_B + \beta|1\rangle_A|-\rangle_B$. Suppose that we now measure the first qubit in the Hadamard basis. What happens to the second qubit? Let's re-write

$$\begin{aligned}|\psi\rangle_{AB} &= \alpha|0\rangle_A|+\rangle_B + \beta|1\rangle_A|-\rangle_B\\ &= \frac{1}{\sqrt{2}}|+\rangle_A(\alpha|+\rangle_B + \beta|-\rangle_B) + \frac{1}{\sqrt{2}}|-\rangle_A(\alpha|+\rangle_B - \beta|-\rangle_B).\end{aligned}$$

The measurement rule states that if we get the outcome "+" the second qubit is projected to $|\psi'\rangle_B = \alpha|+\rangle_B + \beta|-\rangle_B$, and if we get a "$-$" it is projected to $\alpha|+\rangle_B - \beta|-\rangle_B$. In the first case, $|\psi'\rangle_B = H|\psi\rangle$, and in the second $|\psi'\rangle_B = XH|\psi\rangle$. More succinctly put, $|\psi'\rangle_B = X^m H|\psi\rangle$ where $m \in \{0,1\}$ denotes the outcome of the measurement: $m = 0$ in case of "+" and $m = 1$ in case of "$-$". Thus, up to a "Pauli correction" X^m, we managed to apply a Hadamard gate simply by making a single-qubit measurement on the appropriate state.

Exercise 12.3.1 Write the circuit described above in the same form as the T-gate gadget from Figure 12.1, where the state $|\pi/4\rangle$ is replaced by a different "magic" state, and the P correction by a different correction. (Be careful not to confuse CNOT, which is CTL-X, with CTL-Z! How are these related?

For an arbitrary $\phi \in [0, \pi/2)$ let

$$|+_\phi\rangle = \frac{1}{\sqrt{2}}|0\rangle + e^{i\phi}\frac{1}{\sqrt{2}}|1\rangle \qquad \text{and} \qquad |-_\phi\rangle = \frac{1}{\sqrt{2}}|0\rangle - e^{i\phi}\frac{1}{\sqrt{2}}|1\rangle, \tag{12.2}$$

and recall the rotations

$$R_z(\phi) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}, \qquad R_x(\phi) = HR_z(\phi)H.$$

The rotations $R_z(\phi)$ and $R_x(\phi)$ together generate a universal set of single-qubit gates. This is because any rotation on the Bloch sphere can be implemented as $R_z(\varphi_3)R_x(\varphi_2)R_z(\varphi_1)$ for an appropriate choice of φ_1, φ_2 and φ_3.

The following exercise asks you to generalize the example of the Hadamard gate to any single-qubit rotation that can be decomposed in this way.

Exercise 12.3.2 Modify the method we described to apply a Hadamard gate by instead performing a measurement of the first qubit in the basis $\{|+_\varphi\rangle, |-_\varphi\rangle\}$. Show that the second qubit is then projected on the state $X^m HR_z(\varphi)|\psi\rangle$, where $m \in \{0, 1\}$ indicates the measurement outcome.

Now consider a sequence of three measurements with angles φ_1, φ_2 and φ_3. That is, suppose a first qubit is in state $|\psi\rangle_A$, and three additional qubits are created in the $|+\rangle$ state and organized on a line. Three CTL-Z operations are performed from left to right. Then the first qubit is measured in basis $\{|+_{\varphi_1}\rangle, |-_{\varphi_1}\rangle\}$, obtaining an outcome $m_1 \in \{0, 1\}$, the second qubit is measured with angle φ_2, obtaining outcome m_2, and finally the third qubit is measured with angle φ_3, obtaining outcome m_3. Show that the state of the fourth qubit can then be written as

$$|\psi'\rangle_D = X^{m_3}Z^{m_2}X^{m_1}HR_z((-1)^{m_2}\varphi_3)R_x((-1)^{m_1}\varphi_2)R_z(\varphi_1)|\psi\rangle. \tag{12.3}$$

[Hint: you may use the identities $XR_z(\phi) = R_z(-\phi)X$ and $HR_z(\phi)H = R_x(\phi)$, valid for any real ϕ.]

The exercise *almost* lets us apply an arbitrary rotation $R_z(\varphi_3)R_x(\varphi_2)R_z(\varphi_1)$, except that there are these annoying "corrections," the X and Z operations and the Hadamard to the left, as well as extra $(-1)^{m_i}$ phases in the angles. But these are easy to handle! For the phases, note that we perform the measurements sequentially, and the phase flip that got applied to a certain angle only depends on the outcome of the measurement performed right before. For the case of the calculation performed in the exercise, if we *really* had wanted to end up with $U_x(\varphi_2)$, after having obtained outcome m_1 we could have updated our choice of angle in which to measure to $\varphi_2' = (-1)^{m_1}\varphi_2$. As for the X, Z and H corrections at the end of the computation, we can handle those at the time of final measurement: they correspond to corrections that will need to be applied once we measure the final qubit (this is similar to how we handled the one-time pad in the previous section).

So we now know how to apply any sequence of single-qubit rotations to a qubit by using only measurements. To do this we start with a line of m qubits, each initialized in the $|+\rangle$ state. Then we apply CTL-Z operations on all pairs of neighboring qubits, from left to right. This corresponds to preparing a $1 \times m$-dimensional "brickwork state", a universal resource for single-qubit computation. Suppose for simplicity the initial qubit is meant to be initialized in the $|+\rangle$ state (if it is not you can modify the circuit so that the first gate applied prepares the correct qubit). Any rotation can be applied by decomposing it in the form $R_z(\varphi_3)R_x(\varphi_2)R_z(\varphi_1)$ and making the correct sequence of measurements on three qubits, keeping track of successive measurement outcomes to update the angles and the X, Z and H "corrections" that tag along to the left of the description of the state of the qubit, as in (12.3) (note that you do not need to remember all measurement outcomes, but only their combined effect in terms of a power of X and a power of Z).

What if we have a multi-qubit computation? We won't give the details, but the general idea is the same. Since we already know how to implement arbitrary single-qubit gates, to get a universal gate set it suffices to implement a 2-qubit CNOT gate. This can be done by using multiple lines of qubits, one for each qubit of the original computation. The lines are connected by vertical CTL-Z operations once every three qubits (in a slightly shifted manner). A two-qubit CNOT gate can then be applied using similar ideas as we described, but performing measurements on the two lines associated with the two qubits on which the gate acts. We'll leave the details as an exercise, and refer you to the chapter notes for detailed explanations.

12.3.2 Blind delegation in the MBQC model

Week 10, Lecture 10.5,Lecture 2: Hiding a Hadamard

Now that we have seen how to perform an arbitrary computation in the MBQC model, let's see how the computation can be delegated to an untrusted Bob. Let's imagine that Alice has a sequence of single-qubit measurements, specified by angles $\{\varphi_{ij}\}_{1\leq i\leq n, 1\leq j\leq m}$ and update rules (depending on prior measurement outcomes), that she wishes to apply on an $n \times m$ brickwork state in order to implement a n-qubit quantum circuit that she is interested in. Let's also assume for simplicity that the outcome of the last measurement would (possibly after a Pauli correction if needed) give her the answer she is looking for.

Of course Alice could tell the server to prepare the $n \times m$ brickwork state and then instruct it, through a classical interaction, to perform the measurements specified by the φ_{ij}. The server would report the outcomes, Alice would perform the updates, and tell the server the next angle to measure in. But clearly this would be neither blind nor verifiable.

The key idea is for Alice to (partially) prepare some kind of "one-time padded" version of the brickwork state, on which the server will implement the computation without ever having any information about the "real" angles φ_{ij}.

Consider the following protocol outline.

Protocol 1 *Fix a set of "hiding angles" $D = \{0, \pi/4, 2\pi/4, \ldots, 7\pi/4\}$.*

1 *For each of the nm qubits of the brickwork state, Alice chooses a random $\theta_{i,j} \in D$, prepares the state $\left|+_{\theta_{i,j}}\right\rangle$, and sends it to Bob.*
2 *Bob arranges all the qubits he receives in the shape of an $n \times m$ brickwork state, and performs CTL-Z operations on neighboring qubits as required.*
3 *Alice and Bob have a classical interaction over nm rounds. In each round,*
    1 *Alice computes an angle δ_{ij} as a function of θ_{ij}, φ_{ij}, private randomness $r_{ij} \in \{0,1\}$, and previous outcomes b_{ij} reported by Bob. She sends δ_{ij} to the Bob.*
    2 *Bob measures the (i, j)-th qubit of the brickwork state in the $\{|+\rangle_{\varphi_{ij}}, |-\rangle_{\varphi_{ij}}\}$ basis and reports the outcome $b_{ij} \in \{0,1\}$ to Alice.*
4 *Alice infers the outcome of her circuit from her private data and Bob's last reported outcome.*

There are many details missing to fully specify the protocol. The idea is to design rules for Alice to update the measurement angles δ_{ij} that she sends to Bob in such a way that, from the point of view of the server, δ_{ij} is always uniformly random in D (so it reveals no information about the computation being performed), yet Alice is able to keep track of the actual computation being performed under her one-time pad. To see how this can be done, first attempt the following exercise.

Exercise 12.3.3 Based on Exercise 12.3.1 we know that applying a Hadamard gate to a qubit A can be performed by measuring the qubit in the basis $\{|+\rangle, |-\rangle\}$, and adding an X^m correction, where m is the measurement outcome.

This is correct when the second qubit, B has been initialized in a $|+\rangle$ state, as it would be for the un-hidden brickwork state. Now suppose that the qubit has in fact been initialized in the state $|+_\theta\rangle$, for some real angle θ (and a CTL-Z operation has been performed on the two qubits). Show that the result of measuring the first qubit in the basis $\{|+_\delta\rangle, |-_\delta\rangle\}$ is to project the second qubit on $X^m H R_z(\theta+\delta)|\psi\rangle$, where m is the measurement outcome.

Suppose then that Alice would like to apply a rotation $R_z(\varphi)$, for some angle $\varphi \in D$. The exercise shows that by communicating the angle $\delta = \varphi - \theta$ to Bob instead, where θ is the initial angle using which she prepared the corresponding qubit of the brickwork state, will have the desired effect of implementing $X^m H R_z(\varphi)$. However, this still poses a problem: if Bob is given both the quantum state $|+_\theta\rangle$, *and* the real angle $\varphi - \theta$, we can't argue that the computation is blind, as the joint distribution of these two pieces of information depends on φ.

Exercise 12.3.4 Fix φ, and suppose an adversary is given a classical value $\eta = \varphi - \theta$ and a single-qubit state $|\psi\rangle = |+_\theta\rangle$, where θ is chosen uniformly at random. Design a strategy for the adversary to recover φ, given $(\eta, |\psi\rangle)$. What is its success probability (averaged over the random choice of θ)?

The role of the additional values r_{ij} specified in the protocol is to hide φ_{ij} completely from Bob. Here r_{ij} is chosen uniformly at random in $\{0,1\}$, and Alice communicates the angle $\varphi - \theta + r\pi$ to Bob. Based on exercise 12.3.2 the effect of $r\pi$ on the computation is to add an extra Z^r correction, which Alice can easily keep track of. To see that it is sufficient to ensure blindness, imagine that instead $r\pi$ had been added to the initial angle θ. For any fixed θ, a random choice of $r \in \{0,1\}$ suffices to make sure that Bob gains no information from receiving $|+_{\theta+r\pi}\rangle$, as $\frac{1}{2}|+_\theta\rangle\langle +_\theta| + \frac{1}{2}|+_{\theta+\pi}\rangle\langle +_{\theta+\pi}| = \frac{1}{2}\mathbb{I}$. But as θ varies in D the angle $\theta - \varphi$ itself is uniformly distributed in D. Therefore from the point of view of Bob the joint distribution of the pair $(|+_\theta + r\pi\rangle, \theta - \varphi)$ is indistinguishable from that of a uniformly random qubit and a uniformly random value from D. Bob receives completely random data, so the computation is perfectly blind.

12.3.3 Verifiability

In the previous section we showed that blind delegation could be implemented in the MBQC model. Can we make the protocol verifiable? Note that so far Alice does not perform any checks, so Bob could just as easily report random outcomes to her at each step. Already though, due to blindness there is no way that Bob can *force* a particular outcome on Alice; the best it can do is mislead her into thinking that the outcome of the computation is some random bit.

There are different techniques available to make the protocol verifiable. The main idea is to introduce *trap qubits*. Those are particular rows of the brickwork state that Alice randomly inserts into her circuit but on which the only operation performed is a sequence of identity gates: they are meant to remain in the $|0\rangle$ state (hidden, as usual, under the quantum one-time pad). By asking Bob to measure a qubit on such a line, Alice can verify the measurement outcome. Due to the blindness property, even the application of identity gates cannot be detected by Bob, so he does not know that he is being tested.

Implementing this idea requires a little care, as it is important to ensure that even the tiniest attack by Bob, such as reporting a single false measurement outcome, is detected with good probability: a single such deviation could suffice to ruin the whole computation. This can be achieved by introducing ideas from fault-tolerant computation that we will not go into here.

12.4 Classically delegating to two quantum servers

Week 10, Lecture 10.6,Lecture 1: Delegating to multiple provers

Both schemes for delegated computation we've seen so far, in the circuit model or using measurement-based computation, require Alice to prepare single-qubit states taken from a small fixed set and send them to Bob. What if Alice has no quantum capability whatsoever? Intuitively, the aim of the qubits sent by Alice in the two previous schemes is to establish some kind of "trusted space" within Bob's quantum memory, in which he is constrained to perform the computation. The quantum one-time pad is used to guarantee that if Bob tries to cheat by not using these qubits then Alice interprets the results, at best, as garbage. In fact the verifiability property, enforced through the use of test runs or trap qubits, ensures that Bob's cheating will be detected with high probability.

How can we establish a "trusted computation space" without sending the qubits in the first place? You know the answer! In Chapter **??** we saw that simple tests based on the CHSH game could be used to guarantee that *two* arbitrary but *non-communicating* players share a specific state, the EPR pair $|\mathrm{EPR}\rangle$. Even if it is limited to a single qubit per player, this gives us a solid starting point: a test which ensures that a certain little corner of the servers' workspace behaves in a way that we can control.

Let's see how this idea can be leveraged to devise a scheme for delegated computation in which Alice is completely classical, but has access to *two* non-communicating servers, both untrusted. To avoid confusion we'll call the servers Charlie and Dave — these are the Alice and Bob from Chapter **??**, but we already have an Alice and a Bob here! This method is the most technical of the three we are presenting, and we'll remain at an intuitive level of presentation.

12.4.1 Establishing a trusted computation space

In Chapter **??** we saw the CHSH rigidity theorem, which states that if Charlie and Dave successfully play the CHSH game then up to local isometries the operations they perform are equivalent to those specified in the ideal strategy for the CHSH game. Thus the CHSH game provides a simple test, not only to certify the presence of an EPR pair between the servers, but also the specific measurements that the servers perform on their respective half of the EPR pair when asked certain questions. The central idea for using this in delegated computation will be to alternate between playing the CHSH game with the servers, and playing other games, some of which involve the actual computation Alice wants the servers to implement; this will be done in a way that the servers individually can never tell whether they are being "CHSH-tested" or actually "used" to implement a useful part of the computation. Therefore the servers have to apply the honest CHSH strategy all the time, test or computation, and this gives us a way to control which operations they apply.

The first thing to deal with is that we're going to need many EPR pairs. One idea to certify n EPR pairs would be to play n CHSH games "in parallel": Alice could select n pairs of questions $(x_j, y_j)_{j=1,\ldots,n}$ to send to Charlie and Dave, collect n pairs of answers (a_j, b_j), and check how many satisfy the CHSH condition $a_j \oplus b_j = x_j \cdot y_j$. If this estimate is close enough to the optimal $\cos^2(\pi/8) \cdot n$ she would accept the interaction. Although this is a sensible idea it is currently not known how well it works; in particular the effect of small errors in the Charlie and Dave's answers is not clear.

Instead of executing the games in parallel Alice will perform them sequentially. That is, she sends the questions (x_j, y_j) to Charlie and Dave one pair at a time, waiting for their answer before sending the next pair of questions. After having repeated this procedure for n rounds, she counts the number of rounds in which the CHSH condition was satisfied, and accepts if and only if it is at least $(\cos^2(\pi/8) - \delta)n$, for some error threshold δ. The following sequential rigidity theorem states the consequences of this test in the idealized setting where $\delta = 0$.

Theorem 12.4.1 (Idealized) *Suppose the two servers, Charlie and Dave, successfully play n sequential CHSH games. Then up to local isometries their initial state is equivalent to $|\mathrm{EPR}\rangle_{CD}^{\otimes n} \otimes |\mathrm{junk}\rangle_{CD}$. Moreover, at each step $j \in \{1, \dots, n\}$ the measurements performed by each server are equivalent to those of the ideal strategy for CHSH (Z and X for Charlie and H and $\tilde{H}$ for Dave) applied on the j-th EPR pair.*

You may notice that the protocol for the n sequential CHSH tests is similar to how the CHSH tests are performed in the protocol for device-independent quantum key distribution from Chapter **??**. The analysis uses similar tools: a first step uses a (martingale) concentration inequality to argue that, if a fraction about $\cos^2(\pi/8) - \delta$ of the games are won by the servers, then for most $j \in \{1, \dots, n\}$ the *a priori* probability that the servers would have won in round j must be of the same order, say at least $\cos^2(\pi/8) - 2\delta$. For any such j the basic CHSH rigidity theorem can be applied to conclude that the measurements applied, and the state on which they were applied, are (up to local isometries) equivalent to the ideal CHSH strategy.

This reasoning by itself is not sufficient to imply that the servers' initial state is a tensor product of EPR pairs. Indeed, the different EPR pairs used in each round could partially "overlap", or even be the same pair! Intuitively we know this is not possible, as any measurement destroys the EPR pair, so it cannot be re-used. But this is delicate to establish rigorously, because the EPR pair need not be completely destroyed; could many "leftover EPR pairs" be combined together to make a fresh one? Nevertheless, the analysis can be done, and for the remainder of the section we will assume that a "robust" version of the "idealized" theorem above can be proven dealing with the more realistic setting where the servers are not required to play the CHSH games strictly optimally, a far too stringent requirement for any practical application.

12.4.2 State tomography

Now that we have a way to establish a "secure computation space", as a second step let's see how the client Alice can use that space, and additional CHSH tests, to certify that one of the servers has prepared certain single- or two-qubit states in that space.

Consider the following protocol. With Charlie, Alice behaves exactly as if she was executing the n sequential CHSH games described in the previous section. With Dave, however, she does something different: she instructs him to measure each half of the EPR pairs he is supposed to share with Charlie in a certain basis, say $\{|+_\theta\rangle, |-_\theta\rangle\}$ for some real θ (defined as in (12.2)), and to report the outcome.

Dave of course knows that something special is going on. So we have no guarantee as to what action he performs. In contrast, Charlie is told the exact same thing as in the n-sequential CHSH test. He must thus behave exactly as if this is the test Alice was performing, and Theorem 12.4.1 applies: in each round, Charlie applies the ideal CHSH measurements, in the standard or Hadamard bases, on his half of the j-th EPR pair, in a way that, if Dave had been measuring using his own CHSH measurements, they would have succeeded with near-optimal probability.

But now Dave is doing something different — we don't know what. But *if* Dave performs the measurement asked by Alice, and reports the right outcome, we know what should happen: Dave's half-EPR pair gets projected onto one of the basis states, $|+_\theta\rangle$ or $|-_\theta\rangle$, and by the special properties of EPR pairs so does Charlie's half. In particular, whenever BobCharlieperforms a measurement in the Hadamard basis the average value of his outcome (considered as a value in $\{\pm 1\}$) should be $\langle +_\theta| X |+_\theta\rangle = \cos(\theta)$ or $\langle -_\theta| X |-_\theta\rangle = -\cos(\theta)$. Thus by collecting all of Charlie's answers associated to measurements in the X basis Alice can check whether the average outcome over the rounds in which Dave reported a $+$ is approximately $\cos(\theta)$, and $-\cos(\theta)$ over those rounds when Dave reported a $-$. Alice is using Charlie's answers to perform tomography on the state that Dave claims to have prepared, without Charlie being able to detect what is going on! (Even though Charlie knows that he *might* be currently tested, since he is aware of the structure of the protocol, there is nothing he can do about it — if he deviates he risks failing too many CHSH games, in case this is what Alice is doing.)

In the CHSH game the only measurements made by Charlie are in the computational or Hadamard bases. To perform tomography of arbitrary multi-qubit states we would also need him to sometimes apply a Pauli Y. It is possible to do this via a simple modification of the CHSH game. For our purposes the modification will not be necessary, as the set of states that are characterized by their expectation value with respect to Pauli X and Z observables (we call such states *XZ-determined*) is sufficient to implement the delegated computation protocol.

Exercise 12.4.1 Show that the family of all single-qubit states in the xz-plane of the Bloch sphere, i.e. all states of the form

$$\rho = \frac{1}{2}\big(\mathbb{I} + \cos(\theta)X + \sin(\theta)Z\big)\ , \qquad \theta \in [0, 2\pi),$$

are XZ-determined.

Exercise 12.4.2 Show that the family of two-qubit states of the form

$$|\psi\rangle = U \otimes P\,|\mathrm{EPR}\rangle\ ,$$

for any single-qubit real unitary U and $P \in \{I, X, Y, Z\}$, is XZ-determined.

Exercise 12.4.3 Give an example of two distinct single-qubit states that have the same expectation values with respect to both X and Z observables, and are thus not XZ-determined.

12.4.3 Process tomography

Beyond state tomography, our protocol for delegated computation will require us to implement some limited form of *process tomography*: we need to find a way to guarantee that at least one of the servers, Charlie or Dave, is performing the right computation! At first this task may appear overwhelming: while as described in the previous section it is possible to use one server to perform tomography against the other server's state, how can we test that a certain *gate* has been applied? For the case of state preparation we know what the correct states are, and as long as they are restricted to simple single- or two-qubit states we can do full state tomography. But our ultimate goal is to implement an arbitrary quantum circuit, which may generate highly entangled states of its n qubits; there is no hope to perform full tomography on such states, as it would require an exponential number of measurements.

We will sidestep the difficulty and use a model of computation which only requires the application of a very special type of gate — a measurement in the Bell basis, i.e. the simultaneous eigenbasis of $X \otimes X$ and $Z \otimes Z$, given by

$$|\psi_{00}\rangle_{AB} = \frac{1}{\sqrt{2}}(|00\rangle_{AB} + |11\rangle_{AB}), \qquad |\psi_{01}\rangle_{AB} = \frac{1}{\sqrt{2}}(|00\rangle_{AB} - |11\rangle_{AB}),$$
$$|\psi_{10}\rangle_{AB} = \frac{1}{\sqrt{2}}(|01\rangle_{AB} + |10\rangle_{AB}), \qquad |\psi_{11}\rangle_{AB} = \frac{1}{\sqrt{2}}(|01\rangle_{AB} - |10\rangle_{AB}),$$

where $|\psi_{00}\rangle = |\mathrm{EPR}\rangle$ is the familiar EPR pair. This model of computation is called *teleportation-based computation* (recall that a measurement in the Bell basis is precisely the operation required of the sender in the teleportation protocol), and we'll review it in the next section. But let's already see how it can be used for delegated computation.

Similar to the previous section, suppose Alice instructs Dave to measure his n qubits in the Bell basis, where the qubits are paired in an arbitrary way chosen by Alice (so she tells Dave the whole set of measurements to be performed at the outset). Of course as usual Dave does what he wants — he may not even have n qubits in the first place. But Alice also instructs Charlie to play sequential CHSH games, so that from his point of view the protocol is perfectly indistinguishable from the tests. Once Alice has collected all of Charlie and Dave's outcomes, she groups Charlie's outcomes when they are associated to the same state, and uses them to check that Dave did not lie. For example, if Dave reports $|\psi_{00}\rangle$ then whenever Charlie measured the two corresponding qubits using the same basis, computational or Hadamard, his two outcomes should be the same. (Note that not all Charlie's measurements are useful, as it will sometimes be the case that the qubits were measured in different bases, in which case there is no useful test Alice can perform — she simply discards those rounds.)

The following exercise asks you to make this argument more formal.

Exercise 12.4.4 Suppose that Charlie and Dave share two EPR pairs, $|\mathrm{EPR}\rangle_{C_1D_1} \otimes |\mathrm{EPR}\rangle_{C_2D_2}$. Dave measures his two halves, D_1D_2, using an arbitrary four-outcome POVM, obtaining a result $(d_1, d_2) \in \{0,1\}^2$. Charlie measures each of C_1 and C_2 using observables $O_1, O_2 \in \{X, Z\}$ chosen uniformly at random.

Suppose that if $(O_1, O_2) = (X, X)$ then Charlie's outcomes (as values in $\{\pm 1\}$) satisfy $c_1c_2 = a$, and if $(O_1, O_2) = (Z, Z)$ they satisfy $c_1c_2 = d$, for some fixed values $a, d \in \{\pm 1\}$ (i.e. imagine the same experiment is repeated many times, and Charlie's outcomes consistently satisfy these equations, for the same values of a and d). Show that Dave must have been implementing a measurement in the Bell basis. Which Bell state is associated to each of the four possible values for (a, d)?

The exercise shows that, provided we can trust that Charlie and Dave indeed share EPR pairs, and Charlie's measurements are made in the computational or Hadamard bases, then Alice has a way to verify that Dave has been implementing a Bell basis measurement on certain pre-specified pairs of qubits. Just as for the case of state tomography, these assumptions are guaranteed by the fact that Charlie cannot tell the difference between when Alice is executing the process tomography protocol described here, or when she is executing sequential CHSH games.

12.4.4 Teleportation-based computation

The final ingredient needed for our delegation protocol is a method of computation adapted to the kinds of operations we are able to certify: preparation of EPR pairs and single- or two-qubit XZ-determined states (Exercise 12.4.2), and measurements of pairs of qubits in the Bell basis (Exercise 12.4.3).

Computation by teleportation is a model of computation which does just that. The main idea is that a gate can be applied to a qubit by "teleporting the qubit into the gate". The following exercise fleshes out the main gadget used in computation by teleportation.

Exercise 12.4.5 Let $|\psi\rangle_A$ be an arbitrary single-qubit state and let $|\phi\rangle_{BC} = (I \otimes UP\,|\mathrm{EPR}\rangle)$, where U is an arbitrary single-qubit unitary and $P \in \{I, X, Y, Z\}$. Suppose a measurement of qubits A and B is performed in the Bell basis, yielding a pair of outcomes $(b_1, b_2) \in \{0,1\}^2$. Show that there exists a Pauli operator Q (depending only on (b_1, b_2)) such that the post-measurement state of the qubit in C is $(UQPU^\dagger)U\,|\psi\rangle$.

The idea is then the following. Suppose that Alice wishes to implement an arbitrary computation on n qubits, specified by a circuit $\mathcal{C}$ using the universal gate set $\mathcal{G} = \{\mathrm{CNOT}, G\}$ introduced in Section 12.1.1. Assume for simplicity the input to the circuit is $|0\rangle^{\otimes n}$; this is without loss of generality since the input can always be hardcoded into the circuit by using X gates where appropriate. Alice initializes her work space with a large number of "magic states" from the set

$$\{\,|0\rangle\,,\,(I\otimes H)\,|\mathrm{EPR}\rangle\,,\,(I\otimes G)\,|\mathrm{EPR}\rangle\,,\,\mathrm{CNOT}_{B_1B_2}(|\mathrm{EPR}\rangle_{A_1B_1}\,|\mathrm{EPR}\rangle_{A_2B_2})\}. \tag{12.4}$$

At each stage of the computation Alice keeps track of a special set of n qubits which represent the current state of the circuit. We can label these as $A_1\cdots A_n$, even though they will change over time. Initially $A_1\cdots A_n$ point to any n of the "magic" $|0\rangle$ qubits she has prepared in her workspace.

Now suppose Alice would like to apply a gate to one of her qubits A_j, for example a G gate. Then she can perform the circuit described in Exercise 12.4.4, where the role of A is played by A_j, and the roles of B and C by one of her "magic" $(I\otimes G)\,|\mathrm{EPR}\rangle$. As a result the state of C is projected to $(GQG^\dagger)G\,|\psi\rangle_C$, where initially A_j is in state $|\psi\rangle$ (the same computation would work for mixed states as well). This is the operation Alice wanted to perform, except for the correction $GQG^\dagger$. How do we deal with this?

Depending on Q, $GQG^\dagger$ will amount to a Pauli correction, possibly multiplied by a Hadamard: $GXG^\dagger = iHY$, $GYG^\dagger = Y$ and $GZG^\dagger = H$. By now we are used to Pauli corrections: Alice can keep track of these as a form of one-time pad that is tagged along the whole computation. The Hadamard gate is a little more annoying, but in fact it can be easily corrected using one more step of "teleportation", this time using a "magic" $(I\otimes H)\,|\mathrm{EPR}\rangle$. This will induce yet another correction $HQ'H^\dagger$, but this time whatever Q' is the result is a Pauli correction that Alice can again tag along as part of the one-time pad.

Thus, aside from the preparation of the magic states, the whole computation boils down to a simple sequence of Bell basis measurements. Note however that, due to the necessity of performing Hadamard corrections in an unpredictable way (as it depends on measurement outcomes obtained when teleporting into a G gate), this sequence is adaptive. This is similar to the scenario of MBQC, but it will require us to proceed with a little extra care in the final delegation protocol.

Quiz 12.4.1 *True or False: Consider the following circuit on two qubits. Qubit* 1 *is in state* $|\psi\rangle$. *Qubits* 2 *and* 3 *are prepared in the state* $(I\otimes T)\,|\mathrm{EPR}\rangle$. *Perform a Bell measurement on Qubits* 1 *and* 2. *If the two-bit outcome is* a, b *corresponding to eigenvector* $|\psi_{a,b}\rangle = I\otimes X^aZ^b\,|\mathrm{EPR}\rangle$, *apply the unitary* X^bZ^a *to the third qubit. Then the state of the third qubit after the measurement is* $T\,|\psi\rangle$. ***Solution: False***

Quiz 12.4.2 *True or False: Same setup as in the previous question, with the only difference that qubits* 2 *and* 3 *are initially prepared in the state* $(I\otimes H)\,|\mathrm{EPR}\rangle$. *Then the state of the third qubit after the measurement is* $H\,|\psi\rangle$. ***Solution: True***

Quiz 12.4.3 *Suppose Alice and Bob share two EPR pairs. Bob then applies a H gate on the second of his two qubits. He then performs a Bell measurement on his two qubits. What is the resulting post-measurement state on Alice's side, up to Pauli corrections?*

a) $\longrightarrow(\mathbb{I}\otimes H)\,|\mathrm{EPR}\rangle$
b) $|+\rangle\otimes H\,|+\rangle$
c) $(CNOT_{1\to 2})\,|\mathrm{EPR}\rangle$
d) $|\mathrm{EPR}\rangle$

Quiz 12.4.4 *True or False: Suppose Alice and Bob share two EPR pairs. Bob then applies a G gate on the second of his two qubits. Recall that* $G = \begin{pmatrix}\cos(\frac{\pi}{8}) & -\sin(\frac{\pi}{8})\\ \sin(\frac{\pi}{8}) & \cos(\frac{\pi}{8})\end{pmatrix}$. *He then performs a Bell measurement on his two qubits. The resulting post-measurement state on Alice's side, up to Pauli corrections, is* $(I\otimes G)\,|\mathrm{EPR}\rangle$. ***Solution: False***

12.4.5 Blind and verifiable delegated computation

With all the ingredients in place we are ready to describe a delegation protocol for the classical client Alice to delegate the computation of an n-qubit quantum circuit $\mathcal{C}$, specified using the universal gate set $\{\mathrm{CNOT}, G\}$, to two isolated quantum servers, Charlie and Dave, sharing entanglement.

Protocol 2 *Alice compiles a description of $\mathcal{C}$ in the teleportation-based model. Let $t = \mathrm{poly}(|\mathcal{C}|)$ be chosen large enough, and $\delta = \mathrm{poly}^{-1}(|\mathcal{C}|)$ small enough. Alice performs one of the following four tests (the first three with equal probability, and the last with smaller probability depending on the size of $\mathcal{C}$):*

1 *She plays t sequential CHSH games with the servers. If the fraction of successes is below $\cos^2(\pi/8) - \delta$ she rejects.*
2 *Alice executes a state tomography protocol in which Dave is asked to perform measurements that correspond to $t/9$ copies of each of the four magic states in* (12.4)*. Charlie is instructed to play t CHSH games. Charlie's measurement outcomes are used to check the results reported by Dave, as described in Section 12.4.2.*
3 *Alice executes a process tomography protocol in which Charlie is instructed to perform Bell basis measurements according to the pattern that corresponds to the teleportation-based circuit associated with $\mathcal{C}$. Charlie is instructed to play t CHSH games. Dave's measurement outcomes are used to verify that Bob is reporting the correct outcomes for his Bell measurements, as described in Section 12.4.3.*
4 *Alice orchestrates the computation of $\mathcal{C}$: she instructs Dave to prepare the magic states as in step 2., and Charlie to perform Bell basis measurements as in step 3. She uses the measurement outcome reported by Charlie for the output qubit as the output of the computation (after having applied any required Pauli corrections).*

The main reason this protocol works is that Charlie cannot distinguish a computation run from a process tomography run, or a CHSH run from a state tomography run; similarly Dave cannot distinguish between a computation run and a state tomography run, or between a CHSH run and a process tomography run. The protocol can be made blind: blind to Charlie, whom without knowledge of which magic states his Bell basis measurements are performed on gains no useful information from the pattern of Bell measurements Alice instructs him to perform (the pattern can be made independent from the circuit $\mathcal{C}$, aside from its size); blind to Dave, who prepares magic states in a way that is completely independent from the computation. Verifiability follows directly from the tests performed in cases 1., 2. and 3. of the protocol.

There is one difficulty we hinted at earlier and we have glossed over so far. This is the fact that, after application of a Bell basis measurement corresponding to teleportation into a G gate, Alice needs to make an adaptive choice: either apply an H correction, or not. However, Charlie should be ignorant of this choice, as otherwise the protocol would no longer be blind. The solution is to switch the focus over to Dave. Charlie will always be asked to perform the same pattern of Bell basis measurements, but Dave will be (adaptively) asked by Alice to create certain magic states as $|\mathrm{EPR}\rangle$, and others as $(I \otimes H)\,|\mathrm{EPR}\rangle$, as a function of the outcomes reported by Charlie. Since these outcomes are uniformly distributed, the pattern of state preparation requests Dave sees is still random, so that he does not gain any information about the computation either. (Note however that a third observer able to eavesdrop on both the messages exchanged with Bob and with Dave would learn valuable information about the computation; however such an attack falls outside of the scope of the security definition of delegated computation.)

Only one task remains: performing a soundness analysis of the protocol! Given that it is not possible to require that the servers *exactly* pass all the tests, some error should be tolerated. How does this error effect the quality and trustworthiness of the computation? This is quite delicate. The best analysis known to-date makes this protocol, compared to the ones we saw in the previous two sections, highly inefficient, as it requires T to be a very large power of n before even relatively weak security guarantees can be obtained. Nevertheless, it is the only protocol known for purely classical delegated computation, and improving it is an important research problem.

12.5 Chapter notes

In our definition we stated the properties of verifiability and blindness rather informally. A precise definition satisfying all the desired properties (universal composability in particular) would take many pages. Such a definition was given using the framework of *abstract cryptography* in [**?**].

There are many works on delegated quantum computation. Already in 2001, Childs [**?**] provided a protocol that allowed the blind delegation of quantum computations to a server, provided the client has a quantum memory and the ability to implement Pauli gates. Futher protocols in the circuit model, that substantially weakened the requirements on the client and introduced verifiability, were given by Aharonov et al. [**?**, **?**]. The protocol which we present here is due to Broadbent [**?**]. Delegated quantum computation in the measurement-based model was put forward by Broadbent et al. [**?**, **?**]. For some lecture notes on measurement-based quantum computation itself, we refer to [**?**]. The delegated computation protocol in the two-server model which is presented in this chapter is adapted from the protocol by Reichardt et al. [**?**]. A different class of delegation protocols, which we did not describe here, operate in the so-called "receive-and-measure" model. In this type of protocol it is the server, Bob, who sends single qubits to Alice, and Alice measures them. See for example [**?**].

A survey on protocols may be found in [**?**]. For very recent work, focusing on optimizing delegated quantum computation protocols for real-world implementations we refer to [**?**].