ALMA MATER STUDIORUM · UNIVERSITÀ DI BOLOGNA

SCUOLA DI SCIENZE

Laurea Magistrale in Informatica

Verification of Quantum Bit Commitment Protocols using Bisimulation Techniques

Relatore:
Chiar.mo Prof.
Ugo Dal Lago

Correlatore:
Dott.
Andrea Colledan

Presentata da:
Marco Vitale

Sessione III
Anno Accademico 2020-2021

A nonno Angelo

Introduction

Given the current progress in the development of more and more performing quantum computers, the field of quantum cryptography is making a return, after its golden age in the 1990s. The main feature that made quantum cryptography so interesting is that, in certain cases, it can provide the so-called *unconditional security*, which is a security notion that declares that the system is secure against adversaries with *unlimited* computing resources and time. This allows the creation of cryptographic protocols based on the principles of quantum mechanics and mathematically verifiable. However, the security proof related to these protocols can often be tedious and very complicated, as demonstrated by the first proof of the BB84 protocol by Dominic Meyers [34].

A different approach for the verification of quantum cryptographic protocols is given by the so-called *process algebras.* This method has been used successfully in the field of classical cryptographic protocols, for example to prove the security of El Gamal encryption from the Decision Diffie-Hellman (DDH) [41]. The main step of the process algebra approach is given by the use of the notion of *bisimulation.* In the last years, indeed, several quantum process algebras have been proposed, like qCCS [18, 49] and CQP [20], and they have been used to verify different types of quantum protocols, starting from quantum teleportation and arriving to quantum key distribution protocols like the BB84 [17] and the EDP-based protocol [29].

During this thesis, focusing on qCCS, we use the process algebra approach combined with different types of notions deriving from *bisimulation*, to analyze the security proprieties of two different types of *quantum bit commitment protocols*: the BB84 quantum bit commitment protocol [6] and the Kent relativistic bit commitment protocol [28].

In details:

- In Chapter 1 we briefly introduce quantum computing, focusing mainly on the peculiarities that the quantum world brings to computer science.
- In Chapter 2 we first present classical cryptography and the concept of perfect security and then introduce its quantum derivations. We also show the two reference protocols of this thesis, together with some execution examples and their security details.
- In Chapter 3 we introduce process algebras, using CCS as a baseline and presenting different notions of *behavioral equivalences*.
- In Chapter 4 we broaden the discussion on behavioral equivalences, introducing the fundamental notion of our work, bisimulation, and also presenting its probabilistic version.
- In Chapter 5 we introduce qCCS, the extension of CCS dedicated to quantum processes, and we expand our discussion on bisimulation, presenting three variants designed to compare quantum systems.
- Lastly, Chapter 6 is dedicated to our research work, in which, using qCCS and different notions of bisimulation, we analyze the security proprieties of the BB84 quantum bit commitment and the Kent relativistic bit commitment protocols.

Contents

List of Figures

Chapter 1

Introduction to Quantum Computing

The first step in the work must undoubtedly be to present the fundamental principle of the quantum world. In fact, the first chapter will serve us as an introduction. Starting with qubits and continuing with circuits and algorithms, the intersection of the quantum world with computer science will be discussed.

Note that the purpose of this chapter is not to present a fully fledged introduction to the foundations of quantum mechanics, but rather to help the reader understand the following chapters by providing what is strictly necessary.

1.1 Qubits

When we talk about classical computing, the **bit** is the fundamental unit of information. A bit can be seen as a bidimensional classical deterministic system, i.e., two different values can be "encapsulated" in a single bit, 0 or 1. A computer uses different logic gates, such as NOT, AND, OR, etc., to control and manipulate these bits.

Example 1.1.1. A bit can be viewed as a coin, with 0 corresponding to heads and 1 to tails.

Going into the quantum world, the analogue of the bit is the **quantum bit**, or **qubit**. But what is a qubit? A qubit is a physical object that represents a bidimensional quantum system. But, for the purpose of our introduction, the optimal way to describe a qubit is as a **mathematical object**, as Nielsen and Chuang do in [38].

Two of the possible states for a qubit are $|0\rangle$ and $|1\rangle$, that correspond to states 0 and 1 of a classical bit. The '$|\rangle$' notation is called *Bra-Ket* or *Dirac's notation*, and it is the standard notation for quantum states.

Then, what are the main differences between a bit and a qubit? The peculiarities that make qubits stand out are quantum-mechanical in nature.

The first characteristic of a qubit is that it can be in a state different from $|0\rangle$ or $|1\rangle$. Such a state is called *superposition* and is given through a linear combination of $|0\rangle$ and $|1\rangle$:

$$|q\rangle = c_0 |0\rangle + c_1 |1\rangle , \tag{1.1}$$

with $c_0, c_1 \in \mathbb{C}$

Example 1.1.2. A qubit can be viewed like a special kind of coin which, when in a superposition, is both heads and tails *at the same time*:

$$|coin\rangle = c_0 |heads\rangle + c_1 |tails\rangle .$$

With a bit of basic linear algebra, we can represent a qubit as a column vector of dimension two composed of complex numbers c_0, c_1, where $|c_0|^2 + |c_1|^2 = 1$. The states $|0\rangle$ and $|1\rangle$ mentioned above are also called *computational basis states* and form an orthonormal basis for the two-dimensional complex vector space in which the qubit stays.

We take advantage of this use of linear algebra to expose the first postulate of quantum mechanics. This postulate affirms that associated to each quantum system, there is a *Hilbert space* called *state space* of the system. The column vector presented ahead to describe a quantum system is also called *state vector*, and it is in the *state space* of the system.

But what it is a *Hilbert space*? To answer this question, we need to take a step back, starting from the definition of *inner product.*

Definition 1.1.3. An **inner product** on a complex vector space $\mathbb{V}$ is a function $\langle \cdot , \cdot \rangle$ from $\mathbb{V} \times \mathbb{V}$ to $\mathbb{C}$, that satisfies, for all $V_i \in \mathbb{V}$, $c \in \mathbb{C}$:

1. *Nondegenerate*: $\langle V_0, V_0 \rangle \geq 0$, $\langle V_0, V_0 \rangle = 0$ iff $V_0 = 0$.
2. *Respect addition*:$\langle V_0 + V_1, V_2 \rangle = \langle V_0, V_2 \rangle + \langle V_1, V_2 \rangle$
3. *Respect scalar multiplication*: $\langle c \cdot V_0, V_1 \rangle = c \times \langle V_0, V_1 \rangle$, $\langle V_0, c \cdot V_1 \rangle = \overline{c} \times \langle V_0, V_1 \rangle$, with $\overline{c}$ as the complex conjugate of c.
4. *Skew symmetric*: $\langle V_0, V_1 \rangle = \overline{\langle V_1, V_0 \rangle}$, with $\overline{\langle V_1, V_0 \rangle}$ as the complex conjugate of $\langle V_1, V_0 \rangle$.

Having this, we can proceed, giving the definition of Hilbert space.

Definition 1.1.4. A **Hilbert space** is a complex vector space along with an inner product, that is also *complete.*

During this introduction, we will not argue about the notion of completeness, which can be deepened by the reader in [48].

Returning to our journey, the second main difference from a bit is that a qubit cannot be directly *observed* to determine the value of c_0 and c_1. This idea is as counterintuitive as it seems and lies at the heart of quantum computation. When we perform a *measurement* on a qubit we get either the result 0 with probability $|c_0|^2$, bringing the system to the state $|0\rangle$, or the result 1 with probability $|c_1|^2$, bringing the system to the state $|1\rangle$.

In other words, the act of measuring a qubit will *collapse* the whole system into a classical one, a bit. In quantum mechanics theory, this relies on the *Heisenberg's uncertainty principle* [22], that states that it is impossible to do an *observation*, like a measurement, on a quantum system without *disturbing* it. Leaving aside the incredibly interesting theoretical implications of the principle, in practice measurements make us **lose** almost all the additional information introduced by a qubit.

Example 1.1.5. The measurement collapses our special coin in superposition to either heads or tail:

$$Measure(|coin\rangle) = \begin{cases} |heads\rangle & \text{with probability } |c_0|^2 \\ |tails\rangle & \text{with probability } |c_1|^2. \end{cases}$$

Because $|c_0|^2+|c_1|^2 = 1$, another way to describe a qubit is using a geometrical representation. Equation 1.1 can be rewritten as

$$|q\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle\,. \tag{1.2}$$

Coordinates θ and φ are used to define a point in a three-dimensional sphere called *Bloch Sphere*, represented in Figure 1.1.

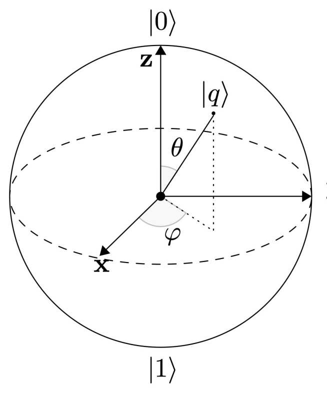


Figure 1.1: Bloch Sphere

Using the Block sphere, the measurement operation can be viewed as a collapse to the north or the south pole, and the probability of which pole it will collapse to depends entirely on the direction in which the qubit is pointing. More precisely, the angle θ controls the chance of collapsing north, that is $|0\rangle$, or south, that is $|1\rangle$.

Example 1.1.6. When a qubit represented in the Bloch sphere is pointing to the "equator" ($\theta = 90°$), there is a perfect $\frac{1}{2}$ chance that the measurement operation will collapse to either $|0\rangle$ or $|1\rangle$.

Obviously, a qubit, like a bit, is not enough for computing. When we want to combine multiple qubits, for instance $|q_0\rangle = |0\rangle$ and $|q_1\rangle = |1\rangle$, the composite system, denoted using the usual Dirac notation, becomes $|\tilde{q}\rangle = |01\rangle$ with respect to the *computational basis states.* Also in this case, all the peculiarities of a single qubit are here, e.g., the two qubits can also exist in superpositions of these four states:

$$|\tilde{q}\rangle = c_{00}\,|00\rangle + c_{01}\,|01\rangle + c_{10}\,|10\rangle + c_{11}\,|11\rangle\,. \tag{1.3}$$

Like in the previous case, the measurement projects the state in one of the possible states, let us call it x, with probability given by $|c_x|^2$.

It can be noticed that the number of complex coefficients rises exponentially when we are combining qubits. This happens because the basic operation used to combine quantum systems is a tensor product. In fact, a string of qubits of length n is an element of $(\mathbb{C}^2)^{\otimes n}$ and we can describe it with a 2^n dimensional column vector.

Example 1.1.7. A qubyte is an element of

$$\mathbb{C}^2 \otimes \mathbb{C}^2 \otimes \mathbb{C}^2 \otimes \mathbb{C}^2 \otimes \mathbb{C}^2 \otimes \mathbb{C}^2 \otimes \mathbb{C}^2 \otimes \mathbb{C}^2 = (\mathbb{C}^2)^{\otimes 8}$$

and can be described as a **256-dimensional** column vector.

The act of considering multiple qubits leads us to another concept of quantum mechanics, namely *entanglement.* A good way to show *entanglement* is through a special two-qubit state called *Bell state* or *EPR-pair*[16]:

$$\frac{|00\rangle + |11\rangle}{\sqrt{2}}. \tag{1.4}$$

In this particular state, the act of measuring one of the two qubits has an interesting effect. The measuring of the first qubit, for example, results in 0 or 1 with the same probability $\frac{1}{2}$, propelling the whole system into either state $|00\rangle$ or $|11\rangle$, respectively. This means that any following measurement operations on the second qubit will lead to the same result. In other words, the two qubits are *entangled* or *correlated.*

The EPR-pair will be useful when we discuss quantum cryptography in Chapter 2.

We have talked about the measurement of a single qubit, describing it in a simple manner in Equation 1.1, but quantum mechanics allows us to do more. Given any *basis states* $|x\rangle, |y\rangle$, it is possible to express an arbitrary qubit state as a linear combination of those states:

$$|q'\rangle = c_0 |x\rangle + c_1 |y\rangle. \tag{1.5}$$

What is even more interesting, is that if the states are *orthonormal*, it becomes possible to perform a *measurement in the* $|x\rangle, |y\rangle$ *basis.* In this case, the measurement gives as result $|x\rangle$ with probability $|c_0|^2$ and $|y\rangle$ with probability $|c_1|^2$.

Example 1.1.8. Given two orthonormal basis states $|+\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$, $|-\rangle = \frac{|0\rangle-|1\rangle}{\sqrt{2}}$, the qubit $|q\rangle$ in Equation 1.1 can be re-expressed in terms of those states in this way:

$$|q\rangle = c_0 |0\rangle + c_1 |1\rangle = \frac{c_0 + c_1}{\sqrt{2}} |+\rangle + \frac{c_0 - c_1}{\sqrt{2}} |-\rangle.$$

During this section, we have presented qubits (and more in general quantum systems) using the language of state vectors. A good way to conclude this part is to introduce another possible formulation, also known as *density matrix.* These matrices are a representation of a linear operator, called *density* *operator*, and allow us to have a different (but mathematically equivalent) way of describing systems whose state is not fully known.

Let us take the general system described in Equation 1.4 as an example and call $|q_0\rangle$ and $|q_1\rangle$ the two possible outcome given by a measurement on the first qubit. The density matrix (or operator) ρ that describe the system is

$$\rho \equiv \sum_i \frac{1}{2} |q_i\rangle \langle q_i| \, . \tag{1.6}$$

This different solution for describing a quantum system gives the same results as the so-called vector language, but in some cases it can make our work easier, as we will see in the following chapters.

To conclude this section, note that the details of the implementation of the qubits are not discussed because the interest of this thesis is towards the theoretical aspect of quantum computing.

1.2 Gates and Circuits

As we have already said, a classical computer uses classical logic gates to control and manipulate bits, but what about quantum gates?

Definition 1.2.1. A complex square matrix U is **unitary** if

$$U^\dagger U = UU^\dagger = I,$$

where I is the identity matrix and $U^\dagger$ is the adjoint of U, which is obtained by transposing and then complex conjugating U.

Definition 1.2.2. A quantum gate is an operator, represented by a **unitary matrix**, that acts on qubits.

The main intuition behind quantum gates is that a unitary matrix is a **reversible** matrix. Having reversible matrices that act on the basic unit of information means that **any** operation, omitting the act of measurement, can be reversed.

Example 1.2.3. The OR gate cannot be applied directly to qubits because, given an output value of $|1\rangle$, we cannot determine if the input was $|01\rangle$, $|10\rangle$ or $|11\rangle$. In contrast, the NOT operation can be carried out without problems.

Being most of the classical gates not reversible, what are the main operators in the quantum world?

Pauli gates

The simplest single-qubit gates are the Pauli **X**, **Y**, **Z** gates:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \qquad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \qquad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \tag{1.7}$$

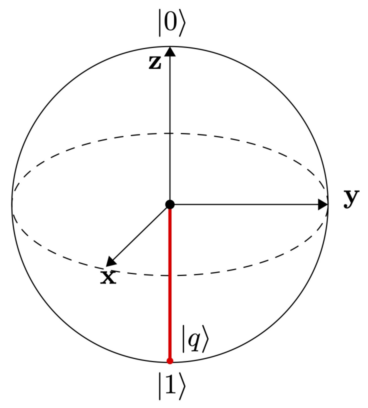
The names of these three gates derive from their behavior, since they respectively perform a rotation around the **x**, **y** or **z** axes of the Bloch sphere by π radians. The **X** gate is also called **NOT** gate with respect to the standard basis because, as is clearly visible from the Bloch sphere, a π rotation around the x-axis turns $|0\rangle$ into $|1\rangle$ and vice-versa. The **Z** gate is sometimes called **phase-flip** gate because, with respect to the standard basis, it leaves $|0\rangle$ unchanged and maps $|1\rangle$ to $-|1\rangle$.

Example 1.2.4. The application of the X gate to the state $|1\rangle$ is represented very clearly using the Bloch sphere:

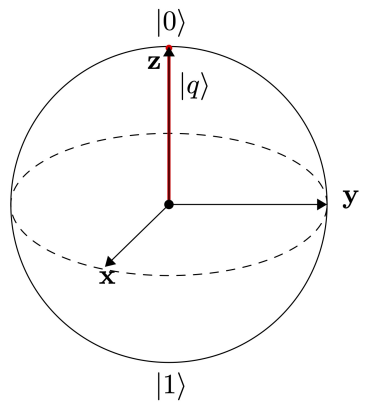
The Hadamard gate

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \tag{1.8}$$

The Hadamard gate is one of the most widely used gates that act on single qubits. The reason is that it maps the basis states into a perfect superposi-



(a) Before the application of X



(b) After the application of X

Figure 1.2

tion:

$$H|0\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}, \qquad H|1\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}. \tag{1.9}$$

Both ending states have equal probabilities of being observed in either of the basis states, the only notable difference being the sign of the second component, which, in the jargon, indicates a different *phase*. This does not affect the probabilities, and therefore does not worry us.

The CNot Gate

$$\text{CNot} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{1.10}$$

When acting on multiple qubits, the *controlled not gate* is an interesting gate. Let us see how it works in details:

- When the first qubit is $|0\rangle$ then the CNot gate does nothing:

$$\text{CNot}\,|00\rangle = |00\rangle \qquad \text{CNot}\,|01\rangle = |01\rangle \tag{1.11}$$

- When the first qubit is $|1\rangle$, then the second qubit is flipped:

$$\text{CNot}\,|10\rangle = |1\rangle\,|1\rangle \qquad \text{CNot}\,|11\rangle = |1\rangle\,|0\rangle \tag{1.12}$$

In other words, the CNot gate use the first qubit as the "control qubits" for reversing the second one:

$$\text{CNot}\,|xy\rangle = |x\rangle\,|x \oplus y\rangle\,, \tag{1.13}$$

where $\oplus$ is the xor operation. In general, the pattern obtained from CNot can be used on any single qubit reversible gate. Given a reversible single-bit gate U, the "controlled-U" ("CU") gate can be described as

$$\text{CU} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & U_{00} & U_{01} \\ 0 & 0 & U_{10} & U_{11} \end{bmatrix}. \tag{1.14}$$

Also note that this pattern can be easily extended to multiple-qubit gates, although the details are beyond the scope of this thesis.

Toffoli gate

$$\text{Toffoli} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \tag{1.15}$$

To conclude the series of presented gates, we introduce the Toffoli gate. It is a 3-qubit gate, which is known to be a *universal gate* for classical computation.

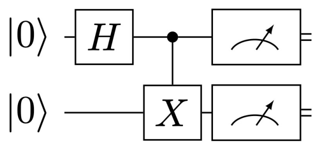


Figure 1.3: EPR-pair and measuring circuit

This means that we can express any other classical logical gate in terms of one or more Toffoli gates. It is also called **CCNOT** gate because the last qubit is flipped only if the first two are $|1\rangle$.

Circuits

Like in the classical case, a combination of sequential and parallel quantum gates is called a **quantum circuit**. However, contrary to their classical counterpart, quantum circuits have some limitations:

- The *fan-in* of the wires is not permitted. This happens simply because the OR operation is not reversible. Note that this is true for all non-injective operations.
- The *fan-out* of the wires, the inverse of *fan-in*, in which we duplicate a bit, is not permitted, as we will see in Theorem 1.2.5.
- Loops are not permitted. Quantum circuits need to be *acyclic*.

Gates in a circuit are represented as boxes, together with a letter which describes the gate we are referring to, connected by wires.

Figure 1.3 represents the circuit that creates an EPR-Pair (Equation 1.4) with a measurement operation downstream. The H box is a Hadamard gate, the following one is a CNot gate, with the topmost qubit acting as the control qubit, and the second qubit acting as the target. In the end, the last two boxes with the meter symbol represent the measurement operation.

No-Cloning Theorem

The most important concept of this section, derived directly from the definition of quantum gates, is called the *No-Cloning Theorem* [47].

Theorem 1.2.5. *It is **impossible** to create an independent and identical copy of an arbitrary quantum state.*

Proof. To be able to copy a qubit into another, the "copy" operation needs to be a represented by a unitary matrix, and therefore needs to be a **linear map**, so let us assume it is and let us call it C. Then we can define $C : \mathbb{C}^2 \otimes \mathbb{C}^2 \to \mathbb{C}^2 \otimes \mathbb{C}^2$ on a pair of qubits **independent** from each other as

$$C(|q\rangle \otimes |0\rangle) = (|q\rangle \otimes |q\rangle). \tag{1.16}$$

Now suppose $q = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$. Applying C needs to result in

$$C\left(\frac{|0\rangle + |1\rangle}{\sqrt{2}} \otimes |0\rangle\right) = \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle + |1\rangle}{\sqrt{2}}\right). \tag{1.17}$$

However, being C linear we would have

$$\begin{aligned} C\left(\frac{|0\rangle + |1\rangle}{\sqrt{2}} \otimes |0\rangle\right) &= \frac{1}{\sqrt{2}}C((|0\rangle + |1\rangle) \otimes |0\rangle) && (1.18)\\ &= \frac{1}{\sqrt{2}}C(|0\rangle\,|0\rangle + |1\rangle\,|0\rangle) && (1.19)\\ &= \frac{1}{\sqrt{2}}(C(|0\rangle\,|0\rangle) + C(|1\rangle\,|0\rangle)) && (1.20)\\ &= \frac{|00\rangle + |11\rangle}{\sqrt{2}} \neq \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle + |1\rangle}{\sqrt{2}}\right) && (1.21) \end{aligned}$$

Thus C cannot be a linear map.

□

The no-cloning theorem is a milestone in the field of quantum information, with enormous consequences on quantum cryptography, as we will see in Chapter 2.

1.3 Algorithms

To conclude the chapter, we will give a brief introduction to quantum algorithms.

What makes quantum computing so interesting from the scientific perspective is the idea of *quantum advantage* or *quantum supremacy* [40, 19]. *Quantum advantage* is the goal of demonstrating that a controlled quantum system can perform tasks faster than the best solution in the classical world. Note that this would allow solving problems that a classical computer cannot solve due to the unfeasible amount of time required. One of the simplest ways of achieving quantum supremacy, Preskill suggests, is to concretely run a quantum algorithm which has a super-polynomial speedup over its classical counterpart.

To achieve this speedup, many algorithms use the concept of *quantum parallelism*, but what is it and how does it work? *Quantum parallelism* relies on the exponential number of states that a number of qubits can represent, and allows quantum computers to evaluate a function f on many values *simultaneously*. But not all that glitters is gold. The fact that we are able to evaluate f simultaneously on many values does not mean that the information contained in the result of f can then be extracted easily. As we stated in Section 1.1, the measurement operation collapses the system to a single state, making us lose almost all the information related to the multitude of evaluations. Therefore, most quantum algorithms are designed in such a way as to extract useful information *before* the act of measuring.

During the years, several algorithms were discovered and proposed, but they all follow a basic framework: The algorithm starts with the qubits in a particular state. From there, the system is put into a superposition of states, which is transformed using unitary operations and then measured. In the continuation of this section, a number of quantum algorithms are presented, in a simplified way.

Deutsch's Algorithm

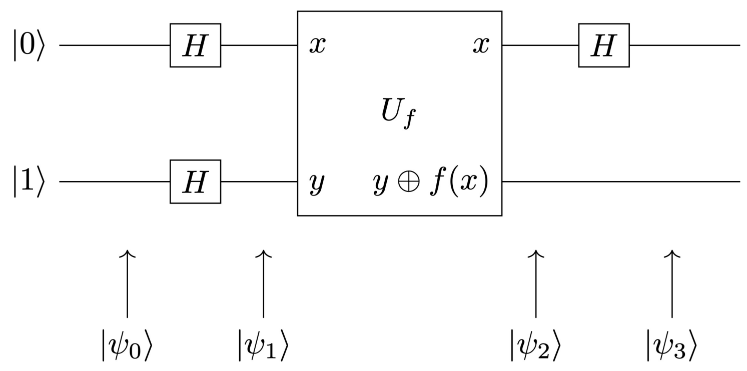


Figure 1.4: Circuit for Deutsch's algorithm

A function $f : \{0,1\} \to \{0,1\}$ is called **constant** if $f(0) = f(1)$, and is called **balanced** otherwise. Deutsch's algorithm is used to determine, given a function f as a "black box", or rather without any knowledge of its internal workings, if it is constant or balanced.

While a classical computer needs to evaluate all the inputs to give the correct answer, Deutsch's algorithm can evaluate both inputs *simultaneously.*

We have already seen the Hadamard gate, but we do not know anything about U_f. We can define it as follows

$$U_f = \begin{bmatrix} f(x) & 0 & 0 \\ f(x) \oplus 1 & 0 & 0 \\ 0 & 0 & f(x) \\ 0 & 0 & f(x) \oplus 1 \end{bmatrix}. \tag{1.22}$$

Example 1.3.1. If we have a balanced f like

$$f(0) = 1 \qquad f(1) = 0,$$

we can define U_f as

$$U_f = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Let us begin to describe Deutsch's algorithm. First, two Hadamard gates are used to put the input qubits $|0\rangle$ and $|1\rangle$ into a superposition:

$$|\psi_1\rangle = H(|0\rangle) \otimes H(|1\rangle) = \left[\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]. \tag{1.23}$$

The application of U_f takes the resulting system to the following state:

$$|\psi_2\rangle = \begin{cases} \pm\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] & \text{if } f \text{ is balanced,} \\ \pm\left[\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] & \text{otherwise.} \end{cases} \tag{1.24}$$

One last Hadamard gate reverts the superposition to a state where the first qubits is either $|1\rangle$ if f is balanced, or $|0\rangle$ otherwise:

$$|\psi_3\rangle = \begin{cases} \pm|1\rangle\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] & \text{if } f \text{ is balanced,} \\ \pm|0\rangle\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] & \text{otherwise.} \end{cases} \tag{1.25}$$

As we already saw, the sign does not impact on the measurement operation, which returns 1 on the first qubit if f is balanced, 0 otherwise.

Deutsch-Jozsa Algorithm

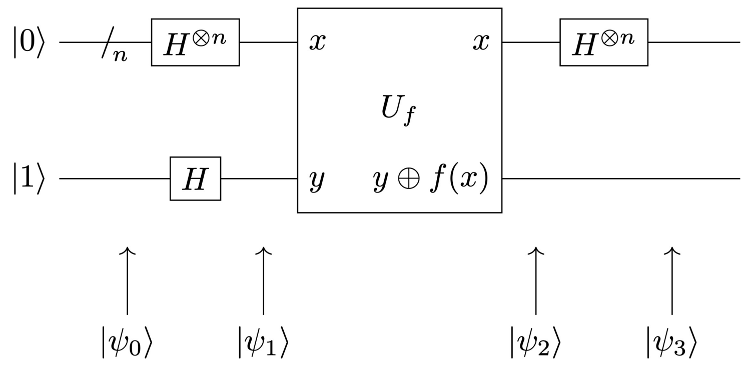


Figure 1.5: Circuit for Deutsch-Jozsa algorithm

The Deutsch-Jozsa algorithm is the generalization of Deutsch's algorithm. In this case, $f : \{0,1\}^n \to \{0,1\}$. Let us start as always by analyzing the classical case. A classical computer performing a deterministic computation needs to query f on at least $\frac{2^n}{2}+1$ values, because the first $\frac{2^n}{2}$ could always output 0 before getting a 1, in which case f is balanced. With the Deutsch-Jozsa Algorithm, using the usual unitary transform U_f, we need only **one** query, obtaining an exponential speedup.

Also in this case, Hadamard gates are used to create a superposition:

$$|\psi_1\rangle = \sum_{q\in\{0,1\}^n} \frac{|q\rangle}{\sqrt{2^n}} \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]. \tag{1.26}$$

Then U_f is applied, resulting in

$$|\psi_2\rangle = \sum_{q\in\{0,1\}^n} \frac{(-1)^{f(q)}\,|q\rangle}{\sqrt{2^n}} \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]. \tag{1.27}$$

Note that the first n amplitudes represent all the function evaluations. The last Hadamard gates are used to interfere, and then to extract the results:

$$|\psi_3\rangle = \sum_{q\in\{0,1\}^n} \sum_{r\in\{0,1\}^n} \frac{(-1)^{q\cdot r+f(q)}\,|r\rangle}{2^n} \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]. \tag{1.28}$$

If the measurement operation of the n qubits results in at least one 1 then the function is balanced; otherwise the function is constant.

Concluding, there are a few other algorithms which are certainly more practical than the aforementioned ones, like Grover's search algorithm, which gives a quadratic speed-up over the best classical solution, or Shor's algorithm, that solves the problem of factoring integers and computing the discrete logarithm in polynomial time, giving a super-polynomial speedup over classical solution. Although they are that are very fascinating, their presentation is far beyond the goal of this brief introduction.

Chapter 2

Quantum Cryptography

Having presented the principles of quantum mechanics that interest us and the way in which they bond to quantum computing, we can delve into one of the areas in which our research work is based, quantum cryptography. The area of quantum cryptography is obtained by merging quantum computation and classical cryptography. Having already robustly introduced the former, the first section of this chapter is dedicated to the latter.

2.1 Classical Cryptography

Being cryptography an immense and fascinating field, exactly as in the previous chapter, this section wants to give a minimal idea of the concept used in the continuation of the work.

Cryptography is the *science* of securing digital information, systems, and distributed computations against adversarial attacks using mathematical techniques. This definition is something different from what we can find in dictionaries, in which cryptography is often defined as an *art*. Historically, cryptography was undoubtedly an art but beginning with Shannon [44] and continuing in the 70s, the world of cryptography radically changed. Through a succession of scientific papers, a rich theory began to appear, enabling its study as a mathematical discipline and therefore a *science*.

To achieve the goal just stated, focusing on digital information security, an *encryption scheme* (also called *cipher*) is used to combine messages with some extra information, to produce a *ciphertext*. Beginning with a little terminology, the action of transforming a message, or *plaintext*, into a ciphertext is called *encryption*, while the opposite operation is called *decryption*.

To allow us to present cryptographic protocols, we need to outline a common scenario. We obviously need two entities, the sender and the receiver, who historically are called **Alice** and **Bob** in the literature, while the villain, or the malevolent entity who is attempting to intercept messages, is often called **Eve**.

In the area called *private key cryptography*, the security of the cryptosystem relies on a *key*, shared in advance between Alice and Bob and unknown to Eve. It may be useful for us to define a more specific scenario in which Alice wants to send Bob a message, let us call it m, through an *insecure* channel. To do this Alice uses an algorithm, let us call it *Enc*, which turns m into a ciphertext c, using a key k. Bob then uses k and another algorithm *Dec* for turning c back into m. Having these elements, we can give the first definition of encryption scheme:

Definition 2.1.1. An **encryption scheme** Π is composed by three sets $(\mathcal{M}, \mathcal{C}, \mathcal{K})$ and a triple of algorithms (Gen, Enc, Dec) that satisfy the following conditions:

- $\mathcal{M}$ is a finite set containing all possible plaintexts.
- C is a finite set containing all possible ciphertexts.
- $\mathcal{K}$ is a finite set containing all possible keys.
- $Gen : 1 \to \mathcal{K}$, $Enc : \mathcal{M} \times \mathcal{K} \to C$, $Dec : C \times \mathcal{K} \to \mathcal{M}$.
- $Dec(Enc(m, k), k) = m$.

2.1.1 Perfect Secrecy

The concept of *perfect secrecy* is probably one of the most interesting concepts in cryptography. Introduced by Claude Shannon in [44], it responds to the question: When can an encryption scheme be defined as *secure*? For the definition of perfect secrecy, it makes sense to present an often quoted cipher in the literature, called **Vernam cipher** or more commonly **one-time pad** (OTP).

Definition 2.1.2. The **Vernam cipher** is a triple of algorithms (Gen, Enc, Dec) such that:

- $\mathcal{K} = \mathcal{M} = C = (0, 1)^n$,
- $Pr(K = k) = \frac{1}{2^n}$ with n as the length of k,
- $Enc(m, k) = m \oplus k$ e $Dec(c, k) = c \oplus k$.

Remark. it is simple to notice that the Vernam cipher is correct:

$$Dec(Enc(m, k), k) = (m \oplus k) \oplus k = m \oplus (k \oplus k) = m \oplus 0^n = m.$$

Note that in the context of perfect secrecy, the processes of choosing the message and the key, and the processes of encryption and the decryption are seen as *probabilistic* processes. This last point therefore allows us to compute probabilities such as $Pr(K = k)$. We have therefore arrived to the definition of perfect secrecy.

Definition 2.1.3. An encryption scheme (Gen, Enc, Dec) is **perfectly secret** if and only if, for every $p \in \mathcal{P}$ and $c \in C$ such that $Pr(C = c) > 0$:

$$Pr(M = m \mid C = c) = Pr(M = m).$$

But what does this mean? It means that the event that a specific plaintext has been encrypted is *independent* of the observed ciphertext. In other words, knowing c brings no advantage when trying to recover the original plaintext m.

In his article, Shannon shows how, for each perfectly secure encryption scheme, the uncertainty of the key used for encryption measured in the form of *entropy* of the key space is at least as large as the uncertainty of the encrypted message. In addition to this, the following theorem derives directly from Definition 2.1.3:

Theorem 2.1.4. *If* (Gen, Enc, Dec) *is a perfectly secret encryption scheme, we necessarily have that* $|\mathcal{K}| \geq |\mathcal{M}|$.

Proof. Assume by contradiction that if $|\mathcal{K}| < |\mathcal{M}|$ then the encryption scheme can be perfectly secret. Considering the uniform distribution over $\mathcal{M}$, let $c \in C$ to be a ciphertext that occurs with probability $(C = c) > 0$ and $\mathcal{M}(c)$ to be the set of messages $m_1, m_2, ...$ that are possible decryptions of c for some key k. Then, obviously $|\mathcal{M}(c)| \leq |\mathcal{K}|$. Indeed, if $|\mathcal{K}| < |\mathcal{M}|$ there is some $m' \in \mathcal{M}$ such that $m' \notin \mathcal{M}(c)$, but:

$$Pr(M = m' \mid C = c) = 0 \neq Pr(M = m') \tag{2.1}$$

Then the encryption scheme cannot be perfectly secret. □

Hence, connecting the dots, we can join Definition 2.1.3 and one-time pad with the following theorem:

Theorem 2.1.5. *Vernam's cipher or one-time pad is perfectly secret.*

Proof. For OTP to be perfectly secret we need:

$$Pr(M = m \mid C = c) = Pr(M = m). \tag{2.2}$$

Then, recalling Bayes' Theorem:

$$Pr(M = m \mid C = c) = \frac{Pr(C = c \mid M = m) \cdot Pr(M = m)}{Pr(C = c)}. \tag{2.3}$$

Having this, we proceed by parts. For an arbitrary $c \in C$, $m \in \mathcal{M}$, $k \in \mathcal{K}$ and $|k| = n$:

$$\begin{aligned} Pr(C = c \mid M = m) &= Pr(m \oplus k = c) \\ &= Pr(k = m \oplus c) \\ &= 2^{-n}. \end{aligned} \tag{2.4}$$

Fixing any distribution over $\mathcal{M}$, for any $c \in C$:

$$\begin{aligned} Pr(C = c) &= \sum_{m \in \mathcal{M}} Pr(C = c \mid M = m) \cdot Pr(M = m) \\ &= \sum_{m \in \mathcal{M}} 2^{-n} \cdot Pr(M = m) \\ &= 2^{-n}. \end{aligned} \tag{2.5}$$

Then, back to Equation 2.3:

$$\begin{aligned} Pr(M = m \mid C = c) &= \frac{Pr(C = c \mid M = m) \cdot Pr(M = m)}{Pr(C = c)} \\ &= \frac{2^{-n} \cdot Pr(M = m)}{2^{-n}} \\ &= Pr(M = m). \end{aligned} \tag{2.6}$$

Then OTP is perfectly secret. □

Aside from the public key cryptography revolution, which lies well beyond the scope of this short introduction, a pair of observations can be made to end this section. As neat as it is and with excellent security properties, Vernam's cipher is hardly ever used, due to known limitations. Its worst flaw lies in the fact that *the key needs to be as long as the message*, and this causes several problems regarding the choice, *distribution* and archiving of the keys in case of very long messages. Furthermore, the cipher is secure if *the key is used once*. These downsides did not stop it from being used historically, but nowadays other symmetrical or asymmetrical cryptosystems are preferred even if they do not achieve perfect secrecy.

2.2 BB84 Quantum Key Distribution

While we are presenting perfect secrecy and one-time pad, one of the greatest limitations to its use lies in the problem of keys distribution. In 1984, Charles H. Bennett and Gilles Brassard published a paper [6] in which the principles of quantum mechanics are exploited to design the first quantum key distribution (QKD) protocol.

But what peculiarities of quantum mechanics are the most useful in the environment of cryptography?

1. The no cloning Theorem (1.2.5) asserts that an intruder cannot make perfect copies of qubits.
2. The act of measuring modifies the qubits irreparably.

Having this in mind, we can proceed to describe the BB84 protocol.

First, let us start by describing the environment. In a QKD setting, the goal is to share a private key between two parties, Alice and Bob, using an *insecure* public channel. In this protocol, Alice uses two different orthogonal bases for the preparation of some qubits, which she will then send to Bob. For a better understanding, each qubit sent by Alice is in the form $|q_{x,y}\rangle$ with x representing the actual bit to be sent and y representing which one of the two bases is used. Thus, they can be in one of the four following possible states:

$$|q_{0,0}\rangle = |0\rangle\,, \qquad |q_{1,0}\rangle = |1\rangle\,, \qquad |q_{0,1}\rangle = |+\rangle\,, \qquad |q_{1,1}\rangle = |-\rangle\,. \tag{2.7}$$

Remark. $|+\rangle$ and $|-\rangle$ have been presented in Example 1.1.8.

For clarity, let the computational basis states $|0\rangle, |1\rangle$ be called **Plus** or "$+$ basis" and $|+\rangle, |-\rangle$ be called **Times** or "$\times$ basis".

Example 2.2.1. The action of preparing a qubit can be viewed as an *encoding*. Suppose Alice wants to encode 1 using the $\times$ basis, then we can represent the qubit as $|q_{1,1}\rangle = |-\rangle$.

Now we can give the specification of the protocol:

Protocol 1 BB84 Quantum Key Distribution

```
QKD84($n$):
1: Alice randomly generates two bistring of length $n$.
2: Alice then prepares a string of qubits of length $n$ using as value the
   values of the first bistring, and as bases the values of the second one.
   Subsequently, she sends the string to Bob.
3: Bob receives the qubits and randomly generates another bitstring of
   length $n$.
4: Bob measures each qubit using the randomly generated bistring as bases.
5: Alice and Bob publicly compare which bases they use, every time they
   disagree they discard the respective bit.
6: Bob then chooses half of the remaining bistring and announces it to Alice.
   If they disagree on more than a small percentage of bits, they abort the
   protocol. Otherwise, the remaining bitstring can be use as the private
   key.
```

Example 2.2.2. An example of the protocol running can lead to a better understanding.

1. Let us call $\tilde{K}_a$ and $\tilde{B}_a$ the two bitstrings generated by Alice. Let us assume $n = 8$, $\tilde{K}_a = 11010010$ and $\tilde{B}_a = 10110001$
2. Let $\tilde{q}$ be the resulting string of qubits. Alice prepares $\tilde{q}$ using $\tilde{K}_a$ and $\tilde{B}_a$. Then $q_0 = |-\rangle, q_1 = |1\rangle, q_2 = |+\rangle, q_3 = |-\rangle\, q_4 = |0\rangle, q_5 = |0\rangle, q_6 = |1\rangle, q_7 = |+\rangle$
3. Let us call $\tilde{B}_b$ the bitstring generated by Bob and assume $\tilde{B}_b = 01100101$.
4. The measurement on $\tilde{q}$ with $\tilde{B}_b$ as bases lead to a bistring $\tilde{K}_b = 01010110$ with $K_{b_{0,1,3,5}}$ to be random due to basis discrepancy.
5. After the comparison and the discarding, $\tilde{K}_a = 0010$ and $\tilde{K}_b = 0010$

$\tilde{K}_a$	1	1	0	1	0	0	1	0
$\tilde{B}_a$	$1_\times$	0_+	$1_\times$	$1_\times$	0_+	0_+	0_+	$1_\times$
Alice prepares								
$\tilde{q}$	$\|-\rangle$	$\|1\rangle$	$\|+\rangle$	$\|-\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|+\rangle$
$\tilde{B}_b$	0_+	$1_\times$	$1_\times$	0_+	0_+	$1_\times$	0_+	$1_\times$
Bob measures								
$\tilde{K}_b$	0	1	0	1	0	1	1	0
Basis compare	–	–	✓	–	✓	–	✓	✓
Privacy amplification			✓		✓		key_1	key_2

Figure 2.1: Example of QKD execution

6. Bob now chooses half of the remaining bits, for example the third and the fifth, to announce. Since the comparison of the same values with Alice gives a positive result, they conclude with the same string 10.

The previous procedure can be reduced to Figure 2.1.

The security of this protocol is *provable* and relies on quantum mechanics. Let us start a little analysis by imposing a new requirement: the message that Alice and Bob use to compare and announce the bitstring needs to be **authenticated**, otherwise the protocol is vulnerable to a *man-in-the-middle attack*. Having said that, we can go ideally through the protocol steps. After the preparation of the string of qubits, they pass through a **public** channel before arriving to Bob. During this passage, an intruder, let us call her Eve as always, can obviously intercept the string since the channel is public. But what exactly can she do with it? Theorem 1.2.5 asserts that qubits cannot be perfectly copied. This means that if she wants to intercept some qubits, she also needs to resend them to Bob, without being able to keep a copy. Furthermore, she cannot even measure the string without changing it and consequently be detected. The formal proof of the unconditional security of the protocol can be found in [34] and will not be discussed, in order to give more space to the following protocols.

2.3 BB84 Quantum Bit Commitment

The *bit commitment* (or coin tossing) version of BB84 is designed to achieve a different goal using the same bases and the same two mistrusting parties of QKD. In this setting, Alice wants to *commit* a single bit to Bob, but she does not want Bob to learn the value of her commitment until she chooses to *unveil* it at a later time. We can view this scenario as a real experiment: for the commitment phase, Alice writes on a sheet of paper the value of a certain commitment, locks the sheet into a safe and sends the safe to Bob. For the unveiling phase, Alice sends the key used to lock the safe to Bob, who will be able to see the value contained in the sheet of paper and be sure of the safety of the process. Having already presented the principles of BB84 quantum key distribution, we can go directly into the protocol definition.

Protocol 2 BB84 Quantum Bit Commitment

```
QBC84(n):
1: (Commit): Alice randomly generates a bitstring of length $n$ and chooses
   the value of the commitment.
2: Alice then prepares a string of qubits of length $n$ using as values the values
   of the bitstring and as basis the value of the commitment. Subsequently,
   she sends the string to Bob.
3: Bob receives the qubits and randomly generates another bitstring of
   length $n$.
4: Bob measures each qubit using the randomly generated bistring as bases.
5: (Unveil): Alice sends to Bob the value of the commitment and the ran-
   domly generated bistring
6: Bob discards all the measurement results done with an incorrect basis. If
   the length of remaining bitstring is below a certain threshold, he aborts
   the protocol.
7: Bob checks if all the remaining bits are consistent with Alice bitstring.
   Otherwise, he aborts the protocol
```

Example 2.3.1. As in Example 2.2.2, it can be useful to give an example of a protocol execution done correctly:

1. *(Commit)* Let suppose Alice chooses $n = 8$ and threshold $t = n/2$, and randomly generates $\tilde{K}_a = 11010010$ and $b_a = 1$, with b_a value of the commitment
2. As $b_a = 1$, Alice prepares $\tilde{q}$ using the $\times$ basis. Then $q_1 = \left|-\right\rangle, q_2 = \left|-\right\rangle, q_3 = \left|+\right\rangle, q_4 = \left|-\right\rangle q_5 = \left|+\right\rangle, q_6 = \left|+\right\rangle, q_7 = \left|-\right\rangle, q_8 = \left|+\right\rangle$
3. Bob randomly generates $\tilde{B}_b = 01100101$
4. Then he measures $\tilde{q}$ with $\tilde{B}_b$, obtaining $\tilde{K}_b = 01011010$. Due to basis discrepancy, $K_{b_1}, K_{b_4}, K_{b_5}, K_{b_7}$ are completely random value.
5. *(Unveil)* Bob receives $\tilde{K}_a$ and b_a then discards all the random bits obtained with a wrong basis, leading to a new bitstring $\tilde{K}'_a$. Given that $|\tilde{K}'_b| \geq t$, the protocol can continue.
6. Bob now deletes the respective bits on $\tilde{K}_a$ and begins a comparison of the result with $\tilde{K}'_b$, which will give a positive result. Then Bob accepts the commitment.

The procedure can be viewed in Figure 2.2.

2.3.1 Security

Given that the BB84 quantum bit commitment protocol will be dealt in detail in Section 6.1, we will examine in detail the security proof concerning the protocol.

First, as stated by Bennet and Brassard themselves in their paper, the protocol is **not** unconditionally secure given the *EPR-paradox*. In 1.3 we presented an EPR-pair, but not the paradox itself. The *Einstein–Podolsky–Rosen paradox* is an experiment, resulted in a famous paper [16], in which they questioned the completeness of quantum mechanics. Briefly, given two entangled system and supposing that there is no longer any interaction between the two parts, as in Equation 1.4, the paradox lies in the fact that the two system,

$\tilde{K}_a$	1	1	0	1	0	0	1	0
Alice prepares	$b_a = 1$							
$\tilde{q}$	$\vert-\rangle$	$\vert-\rangle$	$\vert+\rangle$	$\vert-\rangle$	$\vert+\rangle$	$\vert+\rangle$	$\vert-\rangle$	$\vert+\rangle$
$\tilde{B}_b$	0_+	$1_\times$	$1_\times$	0_+	0_+	$1_\times$	0_+	$1_\times$
Bob measures								
$\tilde{K}_b$	0	1	0	1	0	1	1	0
Basis compare	–	✓	✓	–	–	✓	–	✓
Threshold control ✓								
Consistency check ✓		✓	✓			✓		✓

Figure 2.2: Example of QBC execution

with a measurement on one of them, can simultaneously have exact values of position and momentum contradicting Heisenberg's uncertainty principle, viewed in Section 1.1. A famous attack expects Alice to prepare two strings of qubits in which each qubit of the first string is entangled with the corresponding qubit in the second one. Then she will send one of the two strings and keep the remaining one. In step 5, Alice then measures the remaining string of qubits, allowing her to know the measurement values obtained by Bob. This enables Alice to send the dual of the committed value together with a consistent bitstring, which will be considered valid by Bob and will allow Alice to cheat with probability of success $p_{win} \approx 1$.

An even more interesting and general result was given independently by Meyers [35] and Lo and Chau [32] with their famous *No-Go Theorem*[1].

Theorem 2.3.2. *Unconditionally secure (non-relativistic) quantum bit commitment is* ***impossible****.*

[1]In the field of theoretical physics, a no-go theorem is a particular theorem that states that a specific situation is not physically possible.

Proof. We follow the specific analysis of the *ideal* case, which includes BB84, from Lo and Chau [31] that proceeds as follows. In the *ideal* case, given $\tilde{q}$, Bob has absolutely no information about the value of the commitment b_a. In this case, the density matrix ρ associated with the possible values of the commitment are the same:

$$Tr_A \left|0\right\rangle \left\langle 0\right| \equiv \rho_0^{\tilde{q}} = \rho_1^{\tilde{q}} \equiv Tr_A \left|1\right\rangle \left\langle 1\right| . \tag{2.8}$$

Then, using *Schmidt decomposition* [25]:

$$\left|0\right\rangle = \sum_k \sqrt{\lambda_k} \left|\hat{e}_k\right\rangle_A \otimes \left|\hat{\phi}_k\right\rangle_{\tilde{q}} \qquad \left|1\right\rangle = \sum_k \sqrt{\lambda_k} \left|\hat{e}'_k\right\rangle_A \otimes \left|\hat{\phi}_k\right\rangle_{\tilde{q}} . \tag{2.9}$$

With λ_k as the eigenvalues of $Tr_A \left|0\right\rangle \left\langle 0\right|$ and $Tr_A \left|1\right\rangle \left\langle 1\right|$, and $\left|\hat{e}_k\right\rangle_A , \left|\hat{e}'_k\right\rangle_A$ and $\left|\hat{\phi}_k\right\rangle_{\tilde{q}}$ as the orthonormal bases of the corresponding Hilbert spaces. Note that all the $\left|\hat{\phi}_k\right\rangle_{\tilde{q}}$ and λ_k are *the same* for the two states. The only visible difference resides in Alice's $\left|\hat{e}_k\right\rangle_A , \left|\hat{e}'_k\right\rangle_A$. Now, if we consider a unitary transformation U which maps$\left|\hat{e}_k\right\rangle_A$ to $\left|\hat{e}'_k\right\rangle_A$, it acts *only* on Alice's system, and clearly transform $\left|0\right\rangle$ to $\left|1\right\rangle$. Therefore, is clear that Alice has no physical constraint that prevents her from cheating and changing the value of b_a in the unveil phase.

The non-ideal case will not be treated because it is applied to different, non-thorough commitment schemes; however, it can be found in [31]. □

Following this result, it is clear that if Alice has a quantum computer – a challenging hypothesis from a technological perspective, but not forbidden by the laws of physics – she can cheat with probability 1 on an ideal case like the one discussed earlier and with large probability in the non-ideal one.

2.4 Kent Relativistic Bit Commitment

When Bennet and Brassard proposed the BB84 bit commitment protocol, they were conscious about its insecurity degree, given the EPR-paradox, but considering the technology of the time it was stated to be secure because of the difficulty to replicate and the impossibility of preserving the state for a considerable amount of time. Later attempts to find unconditionally secure protocols were made [10, 9, 1], but all these schemes then fell due to Theorem 2.3.2.

However, the world is *relativistic*. Indeed, classical relativistic protocols have been proven to be secure against Mayer and Lo-Chau attack and are *conjectured* to be *unconditionally secure*.

For the conjecture of *unconditional security*, a couple of assumptions need to be declared. The provided quantum theory needs to be correct, and the background space-time is approximately Minkowski. That means that an adversary cannot surreptitiously make major changes to the local space-time geometry.

The proposed quantum relativistic bit commitment protocol relies on the fact that the committer is bounded, by Minkowski causality, to choose a specific commitment at the space-time point where the protocol starts. This protocol relies on the impossibility of completing a non-local measurement on a distributed state outside the joint future light cone of its components.

The implementation of the protocol requires only that the receiver send a quantum state to the committer, who needs to carry out individual measurements on them **as soon as they are received**. The basic protocol with security parameter n and using BB84 basis goes as follows:

Definition 2.4.1. Two points are said to be **lightlike separated** if the *spacetime interval* among them is zero.

In other words, this means that a beam of light could travel directly from one event to the other, and there will be an agreement about the order in which these events occurred.

Protocol 3 Kent 2012 Relativistic Quantum Bit Commitment

```
KENT12(n):
1: (Commit) Alice and Bob agree on a space-time point $P$, a set of coordinates $(x, y, z, t)$ for Minkowski space, with $P$ as the origin, and two points $\mathcal{Q}_0 = (x, 0, 0, x)$ and $\mathcal{Q}_1 = (-x, 0, 0, x)$ lightlike separated from $P$. They each have agents, separated in secure laboratories, adjacent to each of the points $P, Q_0, Q_1$.
2: Bob randomly chooses two sets of strings $\tilde{K}_b$ and $\tilde{B}_b$ of size $n$.
3: Bob prepares a string of Qubit $\tilde{q}$ of size $n$ using $\tilde{K}_b$ as values, $\tilde{B}_b$ as bases, according to the BB84 basis.
4: Bob sends $\tilde{q}$ to Alice to arrive at the space-time point $P$.
5: Alice's measures the qubits string according to the value she wants to commit. If she wants to commit 0 she measures $\tilde{q}$ with base 0, otherwise if she wants to commit 1 she uses the base 1 and sends the outcome over secure classical channel to her agents at $\mathcal{Q}_0$ and $\mathcal{Q}_1$, according to the BB84 bases.
6: (Unveil) To unveil her committed bit, Alice's agents at $\mathcal{Q}_0$ and $\mathcal{Q}_1$ reveal the measurement outcome to Bob's agents.
7: Bob's agents compare the revealed data to check that the outcomes declared by Alice's agents are both the same and consistent with the list of states sent by Bob at $P$. If this is true, Bob accepts the commitment; otherwise, Bob has detected Alice cheating.
```

Example 2.4.2. As usual, we will give an example of correct execution, using the same random values of the other examples:

1. *(Commit)* Omitting step 1 and assuming $n = 8$ Bob randomly choose $\tilde{K}_a = 11010010$ and $\tilde{B}_a = 10110001$.
2. The preparation of $\tilde{q}$ leads to $q_1 = |-\rangle, q_2 = |1\rangle, q_3 = |+\rangle, q_4 = |-\rangle\ q_5 = |0\rangle, q_6 = |0\rangle, q_7 = |1\rangle, q_8 = |+\rangle$.

$\tilde{K}_b$	1	1	0	1	0	0	1	0
$\tilde{B}_b$	0_+	$1_\times$	$1_\times$	0_+	0_+	$1_\times$	0_+	$1_\times$
Bob prepares								
$\tilde{q}$	$\lvert -\rangle$	$\lvert -\rangle$	$\lvert +\rangle$	$\lvert -\rangle$	$\lvert +\rangle$	$\lvert +\rangle$	$\lvert -\rangle$	$\lvert +\rangle$
Alice choose and measures	$b = 1$							
$\tilde{K}_b$	0	1	0	1	0	1	1	0
Space-time check ✓								
Consistency check $b = 0$ ✗	✗			✗	✗			✗
Consistency check $b = 1$ ✓		✓	✓			✓		✓

Figure 2.3: Example of Kent 2012 execution

3. Assuming a committing value $b_a = 1$, the measurement leads to $K_a = 01010110$ with positions 1,2,4,5 to be random due to basis discrepancy. Then Alice sends $K_a = 01010110$ to her two agents.

4. *(Unveil)* After the send operation by Alice's agents, The consistency check done by Bob lead to an acceptation of the committed value.

Also in this case, Figure 2.3 represents this execution.

2.4.1 Security

Shortly after the publication of the protocol's paper, two independent analyses has been proposed: the first from Kaniewski et al. [27], the second from Croke and Kent itself [11]. To discuss the conjecture of unconditional security, we follow step by step the second one, presenting the lemmas, the main theorem and the respective proofs. First, it is clear that the protocol is secure against Bob, who learns nothing about the committed values until Alice performs the unveiling procedure. For Alice, the key is that she is constrained to reveal the values *to both* Bob's agents at Q_0 and Q_1. This is because, by Minkowski causality, the unveiling of data consistent with a 0 or 1 at a determined space-time point Q_0 depends entirely on the series of operations done on the "line" $P \to Q_0$. Specifically, calling a strategy $\mathcal{S}$ a series of operations, if Alice follows a strategy $\mathcal{S}_0$ and a strategy $\mathcal{S}_1$, respectively on the line $(P \to Q_0]$ and $(P \to Q_1]$, Alice has probability p_i of producing data *consistent* with the values $b_a = i$ at Q_i. This implies that, with probability at least p, by combining the data obtained after the execution of $\mathcal{S}_0$ and $\mathcal{S}_1$, Alice can produce data consistent with *both sets of measurements in complementary bases*. That is, for each qubit $|q_i\rangle$ she can choose a subset of *two* states from the BB84 bases $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$, one from each basis, which must include $|q_i\rangle$. But how much is the probability p? And how is related with the security parameter n? Before arriving to it, we need a couple of lemmas, taken directly from [11].

Lemma 2.4.3. *Given a single qubit $|q\rangle$, randomly chosen from the uniform distribution, Alice's probability p of choosing one of the subsets $B_0 = \{|0\rangle, |+\rangle\}, B_1 = \{|+\rangle, |1\rangle\}, B_1 = \{|1\rangle, |-\rangle\}, B_3 = \{|-\rangle, |0\rangle\}$, which includes $|q\rangle$ is $p \leq \frac{1}{2}(1 + \frac{1}{\sqrt{2}})$, for any strategy $\mathcal{S}$.*

Proof. Recalling the standard discrimination problem, in which Bob chooses a state $\hat{\sigma}_j$ from the set of possible one, with associated probability p_j. Alice's measurement, meant to determine the state, can be described by a *positive operator-valued measure* (POVM) $\{\hat{\pi}_j\}$ where the outcome $\hat{\pi}_j$ leads to the choice of the state $\hat{\sigma}_j$. Then the probability that Alice identified the correct state is

$$p_{correct} = \sum_j p_j \cdot Tr(\hat{\sigma}_j \hat{\pi}_j). \tag{2.10}$$

It is clearly visible that the probability p for our protocol is a variation of the standard discrimination problem. Here, Bob prepares a random state from the BB84 set of states, while Alice get two non-orthogonal guesses at the state. We write the BB84 states as follows:

$$|s_0\rangle = |0\rangle \qquad |s_1\rangle = |+\rangle \qquad |s_2\rangle = |1\rangle \qquad |s_3\rangle = |-\rangle\,. \tag{2.11}$$

In this way, we can define as $G_i = \{|s_i\rangle\,, |s_{i+1}\rangle\}$, for $i = 0\ldots 3$ and $|s_{3+1}\rangle = |s_i\rangle$, Alice's possible guesses. Each of these guesses corresponds to a measurement outcome, so we can associate each one with a POVM element $\hat{\pi}_i$. Then, using $\hat{\rho}_j$ as the corresponding density matrix of $|s_j\rangle$, the probability that Alice wins is:

$$\begin{aligned} p_{win} &= \frac{1}{4}\cdot\sum_i Tr\left(\hat{\rho}_j(\hat{\pi}_i + \hat{\pi}_{i+1})\right) \\ &= \frac{1}{2}\cdot\sum_i Tr\left(\frac{1}{2}\hat{\pi}_j(\hat{\rho}_j + \hat{\rho}_{i+1})\right)\,. \end{aligned} \tag{2.12}$$

Therefore, up to a factor of 2, the problem is equivalent to that of discriminating between a set of equiprobable states, and maximizing the probability of Alice's victory corresponds to minimizing the probability of error in discriminating those states.

Then, given that a necessary and sufficient condition on a POVM, realizing a minimum error measurement in the standard case is:

$$\forall j, \quad \hat{\Gamma} - p_j\hat{\rho}_j \geq 0, \tag{2.13}$$

with

$$\hat{\Gamma} = \sum_i p_i\hat{\rho}_i\hat{\pi}_i \qquad \text{and} \qquad p_{correct} = Tr(\hat{\Gamma}) \tag{2.14}$$

Returning to our problem:

$$\hat{\Gamma} = \frac{1}{4}\sum_{i=1}^{4} \hat{\pi}_i \left(\frac{1}{2}\hat{\rho}_i + \frac{1}{2}\hat{\rho}_{i+1}\right). \tag{2.15}$$

For our set then

$$\hat{\Gamma} = \frac{1}{8}\left(1 + \frac{1}{\sqrt{2}}\right)\hat{I}. \tag{2.16}$$

Recalling the factor of 2, Alice's optimal guessing probability is

$$p_{win} = 2Tr(\hat{\Gamma}) = \frac{1}{2}\left(1 + \frac{1}{\sqrt{2}}\right). \tag{2.17}$$

□

Lemma 2.4.4. *Given a string of qubits $|\tilde{q}\rangle$ of length n, randomly chosen from the uniform distribution and using any strategy $\mathcal{S}$, Alice's win probability $p_{i_0,...,i_{n-2};j_0,...,j_{n-2}} \leq \frac{1}{2}\left(1 + \frac{1}{\sqrt{2}}\right)$ for any $i_0, ..., i_{n-2}; j_0, ..., j_{n-2}$ consistent with S .*

Proof. First, $p_{i_1,...,i_{n-2};j_1,...,j_{n-2}}$ is Alice's probability of choosing a subset of from $G_0 = \{|0\rangle, |+\rangle\}, G_1 = \{|+\rangle, |1\rangle\}, G_2 = \{|1\rangle, |-\rangle\}$, and $G_3 = \{|-\rangle, |0\rangle\}$ that includes $|q_{n-1}\rangle$, conditioned on the first $n-1$ qubits being $|q_{i_0}\rangle, ..., |q_{i_{n-2}}\rangle$, and her guesses $G_{j_0}, ..., G_{jn-2}$. Then, suppose by contradiction that a strategy $\mathcal{S}$ violates this bound for some values $i_0, ..., i_{n-2}; j_0, ..., j_{n-2}$. Having this, Alice can proceed as follows.

1. Prepare an entangled state of two qubits and a string of length $n-1$ of BB84 states $|s_{i_0}\rangle, ..., |s_{i_{n-2}}\rangle$.
2. Apply the strategy $\mathcal{S}$ to the string of qubits and one of the two qubit of the entangled state, ignoring the knowledge of her own bistring.
3. For the first $n-1$ qubits, check the guesses produced by $\mathcal{S}$. If the result does not agree with $G_{j_0}, ..., G_{jn-2}$, return to step 1, otherwise proceed to the next step.

4. Apply a teleportation operation on the unknown BB84 state $|q_{n-1}\rangle$ of length n and the other entangled qubit, obtaining teleportation unitary U.

5. Examine the output of the strategy $\mathcal{S}$, obtaining a guess at a subset containing the teleported unknown qubit $U|q_{n-1}\rangle$, and apply the inverse $U\dagger$ to obtain a guess G_i containing $|q_{n-1}\rangle$.

The guess obtained in the last step is, by assumption, correct with probability $p_{i_1,\ldots,i_{n-2};j_1,\ldots,j_{n-2}} > \frac{1}{2}\left(1+\frac{1}{\sqrt{2}}\right)$.

Note that in this iteration, $|q_{n-1}\rangle$ is left *isolated* until step 5. Hence, Alice would have a strategy that produces a guess for *any* unknown state $|q_{n-1}\rangle$ with probability $p > \frac{1}{2}\left(1+\frac{1}{\sqrt{2}}\right)$, violating Lemma 2.4.3. □

Having given the two previous lemma's proofs, we can now go to present the main theorem.

Theorem 2.4.5. *Given a bistring of qubits $|\tilde{q}\rangle$ of length n, randomly chosen from the uniform distribution, Alice's probability of being able to produce data consistent with measurements in complementary BB84 bases is*

$$p_n \leq \left(\frac{1}{2}(1+\frac{1}{\sqrt{2}})\right)^n$$

Proof. The proof follows directly from Lemma 2.4.1. □

Chapter 3

Process Algebras

Another step in introducing our research work is dedicated to *process algebras*. We start this chapter with a short historical background, passing through a well-known process algebra and ending with various notions of *behavioral equivalence*. A more in-depth view of the topics covered in this chapter can be found in [21].

3.1 Background

The process algebra approach to the verification of concurrent systems has been very successful in the field of theoretical computer science. At the beginning, various *sequential* formalisms capturing the concept of computable function were proposed, among the most famous we can find the *Turing machine* [45] or the *λ-calculus* [8]. The journey of models for concurrent computation started around the 60s with the so-called *Petri nets* [37] and began to arouse great interest in the scientific community between the 70s and the early 80s with the work of Robin Milner [36], Tony Hoare [24], Jan Bergstra and Jan Willem Klop [7]. But what are process algebras? Process algebras are a family of mathematically rigorous frameworks that provide a way for modelling and verifying properties of concurrent communicating systems, using algebraic laws for a formal manipulation and analysis.

In other words, a process algebra (or process calculus) provides not only the constructs to describe a system, but also a formal semantics to describe its evolution. Note that in the continuation of the chapter, we use the terms "system", "program" and "agent" equivalently.

But what is a process algebra made of?

- A *syntax* is the basic component of a process algebra. It is the set of rules that defines the correct combination of symbols into terms, or processes.
- A *semantics*, which can be of three different types:
  - *Operational*, in which a system is modeled as a *labelled transition system.*
  - *Algebraic*, in which the definition of a system is given by a set of algebraic laws.
  - *Denotational*, which maps a system to a mathematical model in which the *denotation* of any system is determined directly from the denotation of a subset of its components.

For this brief introduction we prefer to present a process algebra using the operational approach, and therefore it is important to give the definition of labelled transition system.

Definition 3.1.1. A **labelled transition system** (LTS) is a triple $\langle \mathcal{S}, Act, \rightarrow \rangle$ where:

1. $\mathcal{S}$ is a finite set of states.
2. Act is a set of transition labels.
3. $\rightarrow$ is a transition relation, that is, a subset of $\mathcal{S} \times Act \times \mathcal{S}$.

The set S contains process algebra terms, while the labels contained in Act

represent the actions carried out by the transition relation $\rightarrow$, which allow the system to evolve.

3.2 CCS

The *Calculus of Communicating Systems* (CCS) is one of the most successful process calculi. It was introduced by Robin Milner in the 80s [36] and it is still being extended as of today to model different types of systems, such as quantum systems, as we will see in Chapter 5. To introduce CCS, we start with the syntax, and then we pass to the operational semantics.

3.2.1 Syntax

The building blocks of the syntax are *actions* and *operators*. Actions represent an *atomic* and *instantaneous* execution step and are used to represent *internal* or *external* computation steps. External actions are in turn divided into two types, *input* and *output* actions.

Example 3.2.1. For clarity, thinking about a coffee machine, an input action can be used to represent the insertion of a coin, an internal action to represent the preparation of a coffee and an output action to represent the dispensing of that coffee.

The main difference between internal and external actions is that the former are *invisible* to an external observer. This peculiarity will be crucial for the continuation of the thesis. To formalize these intuitions, let us introduce the set Act_{CCS} of CCS's actions. Assume a countably infinite set $\mathcal{A}$ of labels not containing the symbol τ. The actions in CCS are given by the following BNF:

$$\begin{aligned}
\alpha ::= {} & \tau, && \text{representing an internal action.} \\
\mid {} & a, \text{ with } a \in \mathcal{A}, && \text{representing the input action on port } a. \\
\mid {} & \overline{a}, \text{ with } a \in \mathcal{A}, && \text{representing the output action on port } a.
\end{aligned} \tag{3.1}$$

Then we can formally define Act_{CCS} as

$$Act_{CCS} = \mathcal{A} \cup \{\overline{a} \,|\, a \in \mathcal{A}\} \cup \{\tau\}. \tag{3.2}$$

Having presented the actions, we can now delve into the operators, defining the CCS's processes that represent a system. Being the main purpose of CCS that of modelling concurrent systems, the main operation that a process must be able to perform is the parallel execution of two different systems, often called *parallel composition.* In addition to this, the processes must also be able to represent the classic operations, like the execution of actions, also called *prefixing*, the *nondeterministic choice*, or situations in which determined actions are not allowed, often called *restriction.* Given this, assume a set $\mathcal{S}_{CCS}$ of valid CCS's processes, ranged over by $P, Q, ...$ and a *relabelling function* f, then we can define, using the usual Backus-Naur form, CCS's syntax as

$$\begin{array}{ll}
P, Q ::= \mathbf{nil}, & \text{representing the terminated process.} \\
\mid \;\; \alpha.P, \text{ with } \alpha \in Act_{CCS}, & \text{representing the prefixing operation.} \\
\mid \;\; P + Q, & \text{representing the \textit{choice} operation.} \\
\mid \;\; P \,\|\, Q, & \text{representing the \textit{parallel composition.}} \\
\mid \;\; P \backslash L, \text{ with } L \subseteq \mathcal{A}, & \text{representing the \textit{restriction} operation.} \\
\mid \;\; P[f], & \text{representing the \textit{relabelling} operation.}
\end{array} \tag{3.3}$$

Going into more detail:

- $\alpha.P$ allows an action α to be executed *before* a process description P.
- The choice operator $P + Q$ allows to describe a system that can behave nondeterministically like either P or Q, depending on external details.
- $P \,\|\, Q$ defines a system that is composed by processes P and Q, running in parallel.
- The restriction operator $P \backslash L$, with $L \subseteq \mathcal{A}$, allows describing a system

in which the actions contained in L are not permitted for the process P.

- $P[f]$ with the relabelling function $f : \mathcal{A} \to \mathcal{A}$, allows defining a system that behaves like P, in which, for each α that P executes, $P[f]$ executes $f(\alpha)$.

Example 3.2.2. We can define a coffee machine using CCS syntax in this way:

$$P := \text{coin}.(\overline{\text{coffee}}.\mathbf{nil} + \overline{\text{tea}}.\mathbf{nil}) \tag{3.4}$$

With coin, $\overline{\text{coffee}}$ and $\overline{\text{tea}} \in Act_{CCS}$.

3.2.2 Operational Semantics

The last step in defining a process algebra is to describe the meaning of the operators with a precise semantics. For this step, we use the operational approach. But what is an operational semantics? In Section 3.1 we introduced it as a means to model a system as a labelled transition system. More precisely, using Definition 3.1.1, we can think of the states $\mathcal{S}$ of the transition system as just processes of our process algebra, while the transition label *Act* with the transition relation $\to$ models the reduction steps that take one state to another.

Before talking about the operational semantics of CCS, let us define briefly what an *inference system* is.

Definition 3.2.3.

An **inference systems** is a set of inference rules of the form:

$$\frac{p_1, p_2, ..., p_n}{c}$$

where c is the *conclusion* and $p_1, p_2, ..., p_n$ are the *premises.* If all $p_1, p_2, ..., p_n$ are true, then c is true.

Having this, we can specify the semantics of CCS in the form of a ternary relation $\rightarrow_{CCS}$ from $\mathcal{S}_{CCS}$ to $\mathcal{S}_{CCS}$ inductively, using a collection of *inference rules*:

- **Prefix Operator**

$$\frac{}{\alpha.P \xrightarrow{\alpha} P} \tag{3.5}$$

This rule says that processes in the form $\alpha.P$ may perform α and then behave like P. It is the operator used for the sequential composition.

- **Choice Operator**

$$\frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'}, \qquad \frac{Q \xrightarrow{\alpha} Q'}{P + Q \xrightarrow{\alpha} Q'} \tag{3.6}$$

These two rules describe the choice operator: given an action α that brings process P (or Q) into process P' (or Q'), the composite system $P + Q$ can perform the same α and behave like P' (or Q'). In other words, the choice operation serves as an abstraction to express nondeterministic alternatives among possible behaviors. This nondeterminism can come for internal, external or mixed factors.

- **Parallel Composition Operator**

$$\frac{P \xrightarrow{\alpha} P'}{P \| Q \xrightarrow{\alpha} P' \| Q}, \qquad \frac{Q \xrightarrow{\alpha} Q'}{P \| Q \xrightarrow{\alpha} P \| Q'} \tag{3.7}$$

These rules of parallel composition describe a situation in which, given an action α that brings process P (or Q) into process P' (or Q'), the composite system $P \,\|\, Q$ can perform the same α and behave like $P' \| Q$ (or $P \| Q'$). Parallel composition allows processes P and Q to perform computations simultaneously, together with interaction within synchronous or asynchronous channels. This is clearly the key transition that differentiates process algebras like CCS from sequential models of computation.

- **Restriction Operator**

$$\frac{\alpha \notin L \quad P \xrightarrow{\alpha} P'}{P \backslash L \xrightarrow{\alpha} P' \backslash L} \tag{3.8}$$

The rule for the restriction operator describes that, with the same premise of the choice or parallel operator, a process $P\backslash L$, performing $\alpha \notin L$ may behave like $P'\backslash L$. This means that the action restriction persists after the transition.

- **Relabelling Operator**

$$\frac{P \overset{\alpha}{\to} P'}{P[f] \xrightarrow{f(\alpha)} P'[f]} \tag{3.9}$$

This rule says that, given an action α that brings process P into process P', a process subjected to relabelling $P[f]$ may behave like $P'[f]$ performing the action given by application of the function f on α. Also in this case this means that the relabelling operation persists after the transition.

Recalling Definition 3.1.1, we can view a system described by CCS terms as an LTS. Recall that $\mathcal{S}_{CCS}$ is the set of valid CCS processes and Act_{CCS} is the set of valid actions:

$$System_0 := \langle \mathcal{S}_{CCS}, Act_{CCS}, \rightarrow_{CCS} \rangle \tag{3.10}$$

is a valid LTS.

Example 3.2.4. Abstracting away the overline for actions, the process from Example 3.2.2 can be described by Figure 3.1, with $P_1 =$ (coffee.**nil**+tea.**nil**), and $P_2 = P_3 =$ **nil**

3.3 Behavioral Equivalence

When we talk about software or system verification, it is fundamental to have a rich theory which can be used to compare the behavior of different systems. Thanks to the operational approach used in the previous section for describing CCS, we can describe concurrent systems as LTSs, and then

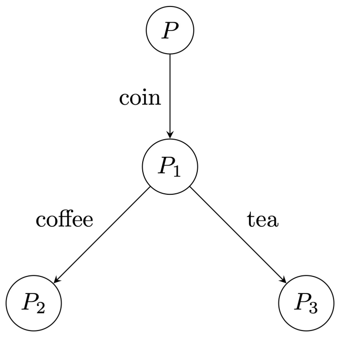


Figure 3.1: LTS of the coffee machine

we can inherit the different notions of behavioral equivalence proposed in the literature over them. For this chapter, we will refer to [21] for proofs and definitions.

But first, what is an *equivalence*?

Definition 3.3.1. A binary relation $\mathcal{R}$ is an **equivalence relation** on a set $\mathcal{S}$ if, for $s_1, s_2, s_3 \in \mathcal{S}$, $\mathcal{R}$ is:

- *Reflexive*, if $s_1 \mathcal{R} s_1$.
- *Symmetric*, when $s_1 \mathcal{R} s_2$ if and only if $s_2 \mathcal{R} s_1$.
- *Transitive*, if $s_1 \mathcal{R} s_2$ and $s_2 \mathcal{R} s_3$, then $s_1 \mathcal{R} s_3$.

Two small remarks: in the continuation, we will talk only about **deterministic LTSs**, furthermore, all the equivalence notions presented are supposed to be *strong equivalences.* A strong equivalences is a particular type of equivalence in which there is no distinction between internal (τ) and external actions, in fact all the actions are considered *observable.*

Definition 3.3.2. Let $L = \langle \mathcal{S}, Act, \rightarrow \rangle$ be an LTS, then L is **deterministic** if, for all $P \in \mathcal{S}$ and $\alpha \in Act$, if $P \xrightarrow{\alpha} P'$ and $P \xrightarrow{\alpha} P''$ then $P' = P''$.

3.3.1 Isomorphism Equivalence

As LTSs can be viewed as an *edge-labelled directed graph*, the notion of graph isomorphism leads to the definition of *isomorphism equivalence.*

Definition 3.3.3.

- Given two LTSs, $L_0 = \langle \mathcal{S}_0, Act, \rightarrow_0 \rangle$ and $L_1 = \langle \mathcal{S}_1, Act, \rightarrow_1 \rangle$, an **isomorphism** between them is a bijection $f : \mathcal{S}_0 \rightarrow \mathcal{S}_1$ such that, for all $P, Q \in \mathcal{S}_0$ and $\alpha \in A$:

$$P \xrightarrow{\alpha}_0 Q \text{ if and only if } f(P) \xrightarrow{\alpha}_1 f(Q).$$

- If there exists an **isomorphism** between two LTSs L_0, L_1, we say that they are **isomorphic**, written:

$$L_0 \cong L_1.$$

Proposition 3.3.4. *The isomorphism on LTSs is an equivalence relation.*

Proof. Given Definition 3.3.1, we need to prove that $\cong$ is *reflexive*, *symmetric* and *transitive*. The first property is clearly trivial. For *symmetry*, being f a bijection means that there exists an inverse function of f, $f' : \mathcal{S}_1 \rightarrow \mathcal{S}_0$, that is a bijection, and therefore $\cong$ is symmetric. Lastly, *transitivity* is also simple. Given an additional LTS $L_2 = \langle \mathcal{S}_2, Act, \rightarrow_2 \rangle$, if $L_0 \cong L_1$ then there is a bijection f such that, given $P, Q \in \mathcal{S}_0$ and $\alpha \in Act$:

$$P \xrightarrow{\alpha}_0 Q \text{ if and only if } f(P) \xrightarrow{\alpha}_1 f(Q).$$

$L_1 \cong L_2$ means that there is a bijection f' such that, given $P', Q' \in \mathcal{S}_1$ and $\alpha \in Act$:

$$P' \xrightarrow{\alpha}_0 Q' \text{ if and only if } f'(P') \xrightarrow{\alpha}_1 f'(Q').$$

Thus, combining f, f' we obtain a new function $f'' = f' \circ f$ that is clearly a bijection, and therefore $L_0 \cong L_2$. $\square$

Given that whatever is done on S_0 can be done on S_1, any two isomorphic LTSs are then *indistinguishable* by any observer. However, the isomorphism equivalence is somewhat too discriminating, as we show in the example below:

Example 3.3.5. Consider the two LTSs in Figure 3.2. The two LTSs are not isomorphic because there is no bijection between the set of states included in their definition, but they clearly behave similarly.

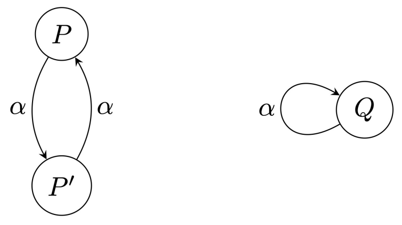


Figure 3.2: Two non-isomorphic LTSs

Furthermore, when we come to finding an isomorphism between two LTSs, we need to recall that graph isomorphism is an *NP* problem [3], even if it has recently been proved to be solvable in quasipolynomial time [2, 23]. These problems lead us to the next equivalence, which is weaker.

3.3.2 Trace Equivalence

Since LTSs are very similar to automata, we could also consider equivalences defined on automata, like *trace equivalence*.

Definition 3.3.6. Given an LTS $L_0 = \langle \mathcal{S}_0, Act, \rightarrow_0 \rangle$ and a state $P \in \mathcal{S}_0$:

- A **trace** of P is a sequence of actions $\alpha_0, \alpha_1, ..., \alpha_n \in Act$, such that $P \xrightarrow{\alpha_0} P' \xrightarrow{\alpha_2} ... \xrightarrow{\alpha_n} P''$
- The **set of traces** of P, denoted as $Tr(P)$ is:

$$Tr(P) = \{seq \in Act^* \mid \exists P' \in \mathcal{S}_0.\, P \xrightarrow{seq}{}^{*} P'\}$$

- Two LTSs L_0 and L_1 are **trace equivalent** if:

$$Tr(L_0) = Tr(L_1)$$

Following the last section, we give an example to clarify the notion of trace equivalence and to compare it with isomorphic equivalence.

Example 3.3.7. The two LTSs in Figure 3.2 are trace equivalent. It is clear that isomorphic equivalence is *finer* than trace equivalence.

The limits of trace equivalence relies on the fact that, like language equivalence over automata, the verification of trace equivalence is *PSPACE-complete* [21]. For this reason, we continue by introducing a more checkable alternative for behavioral equivalence.

3.3.3 Simulation Equivalence

Simulation equivalence is the last notion we cover in this chapter, but it will also be the starting point for Chapter 4.

Definition 3.3.8. Given an LTS $L = \langle \mathcal{S}, Act, \rightarrow \rangle$, two processes $P, Q \in \mathcal{S}$ and a binary relation $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S}$:

- $\mathcal{R}$ is a **simulation** if $(P, Q) \in \mathcal{R}$ implies, for all $\alpha \in Act$, whenever $P \xrightarrow{\alpha} P'$, then $Q \xrightarrow{\alpha} Q'$, for some Q' such that $(P', Q') \in \mathcal{R}$.
- If there exists a simulation $\mathcal{R}$ such that $(P, Q) \in \mathcal{R}$, process P is said to be **simulated** by process Q, written:

$$P \lesssim Q.$$

- If $P \lesssim Q$ and $Q \lesssim P$, P and Q are said to be *simulation equivalent*, written:

$$P \simeq Q.$$

The name *simulation equivalence* derives directly from the intuition that a state can simulate whatever action another state performs, reaching another state that it is still able to simulate.

Note that the notion of simulation equivalence is defined only on states, but it can be lifted to LTSs easily.

Corollary. *Given two LTSs,* $L_0 = \langle \mathcal{S}_0, Act_0, \rightarrow_0 \rangle$ *and* $L_1 = \langle \mathcal{S}_1, Act_1, \rightarrow_1 \rangle$, *with* $\mathcal{S}_0 \cap \mathcal{S}_1 = \emptyset$, *we can create another LTS* $L_2 = L_1 \cup L_2$ *in this way:* $L_2 \langle \mathcal{S}_0 \cup \mathcal{S}_0, Act_0 \cup Act_1, \rightarrow_0 \cup \rightarrow_1 \rangle$. *And then a simulation* $\mathcal{R} : \mathcal{S}_0 \times \mathcal{S}_1$ *is also a simulation on* $(\mathcal{S}_0 \cup \mathcal{S}_1) \times (\mathcal{S}_0 \cup \mathcal{S}_1)$

To delve into the notion of simulation, we expose some interesting proprieties taken directly from [21].

Proposition 3.3.9. *Given an LTS* $L = \langle \mathcal{S}, Act, \rightarrow \rangle$, *the following relations are all simulations:*

- $Id_P = \{(P, P) \,|\, \forall P \in \mathcal{S}\}$.
- $\mathcal{R}_0\mathcal{R}_1 = \{(P, Q) \,|\, \exists R.(P, R) \in \mathcal{R}_0 \wedge (R, Q) \in \mathcal{R}_1\}$, *for all simulations* $\mathcal{R}_0, \mathcal{R}_1$.
- $\bigcup_{i \in I} \mathcal{R}_i$ *if* $\mathcal{R}_i$ *is a simulation, for all* $i \in I$.

Proof. We shall just prove $\mathcal{R}_0\mathcal{R}_1$ and $\bigcup_{i \in I} \mathcal{R}_i$, because the proof of Id_P is immediate.

- For $\mathcal{R}_0\mathcal{R}_1$, given a pair (P, Q) in it, there exists a state R such that

  $$(P, R) \in \mathcal{R}_0 \text{ and } (R, Q) \in \mathcal{R}_1.$$

  Now let $P \xrightarrow{\alpha} P'$. Then there exists R' such that

  $$R \xrightarrow{\alpha} R' \text{ and } (P', R') \in \mathcal{R}_0.$$

  But, as $(R, Q) \in R_2$ we have also, for some Q':

  $$Q \xrightarrow{\alpha} Q' \text{ and } (R', Q') \in \mathcal{R}_1.$$

Then $(P', Q') \in \mathcal{R}_0\mathcal{R}_1$, and $\mathcal{R}_0\mathcal{R}_1$ is a simulation.

- For $\bigcup_{i \in I} \mathcal{R}_i$, assuming $(P, Q) \in \bigcup_{i \in I} \mathcal{R}_i$, there exists a $j \in I$ such that $(P, Q) \in \mathcal{R}_j$. Now let $P \xrightarrow{\alpha} P'$, then must exist a Q' such that

$$Q \xrightarrow{\alpha} Q' \text{ and } (P', Q') \in \mathcal{R}_j.$$

Therefore $(P', Q') \in \bigcup_{i \in I} \mathcal{R}_i$ and $\bigcup_{i \in I} \mathcal{R}_i$ is a simulation.

□

A characterization of Definition 3.3.8 related to the $\lesssim$ relation is the following one.

Definition 3.3.10.

$$\lesssim = \bigcup\{\mathcal{R} \,:\, \mathcal{R} \text{ is a simulation}\}.$$

This derives from the fact that a process Q simulates the process P, written $P \lesssim Q$, if there exists a simulation $\mathcal{R}$ containing (P, Q). This clearly means that $\lesssim$ is the union of all simulations, as stated in Definition 3.3.10.

Furthermore, as for the other notions, we prove that the relation $\simeq$ is an equivalence relation.

Definition 3.3.11. A binary relation $\mathcal{R}$ is a **preorder** on a set $\mathcal{S}$ if, for any $s_1, s_2, s_3 \in \mathcal{S}$, $\mathcal{R}$ is:

- *Reflexive*: $s_1 \mathcal{R} s_1$.
- *Transitive*: if $s_1 \mathcal{R} s_2$ and $s_2 \mathcal{R} s_3$, then $s_1 \mathcal{R} s_3$.

Proposition 3.3.12. *For any LTS $\langle \mathcal{S}, Act, \rightarrow \rangle$:*

1. *The relation $\lesssim$ is a preorder*
2. *The relation $\simeq$ is an equivalence relation*

Proof.

1. As Definition 3.3.11 states, we need to prove that $\lesssim$ is reflexive and transitive.
   - Following Proposition 3.3.9, the identity relation Id_P is a simulation. Then by Definition 3.3.10, $\lesssim$ is reflexive.
   - For transitivity, assume that $P, Q, R \in \mathcal{S}$, then if $P \lesssim R$ there exists a simulation $\mathcal{R}_0$ containing (P, R), and if $R \lesssim Q$ there exists a simulation $\mathcal{R}_1$ containing (R, Q). Also in this case, following Proposition 3.3.9, $\mathcal{R}_0\mathcal{R}_1$ is a simulation containing (P, Q). Then by Definition 3.3.10, $\lesssim$ is transitive.
2. Given that $P \simeq Q$ means that $P \lesssim Q$ and $Q \lesssim P$, reflexivity and transitivity of $\simeq$ are already proven from the previous point of this proof. For symmetry, trivially note that $Q \simeq P$ implies $Q \lesssim P$ and $P \lesssim Q$, and thus $P \simeq Q$

□

Example 3.3.13. The LTSs in Figure 3.2 are also simulation equivalent.

To conclude the section, we remark that in the analyzed case of deterministic LTSs, the trace equivalence and simulation equivalence notions coincide.

Chapter 4

Bisimulation on Abstract Labelled Transition Systems

The last step before getting into our research work is to introduce the notion of *bisimulation*. More precisely, we will first talk about bisimulation in the area of nondeterministic LTSs, pointing out the differences between the deterministic version analyzed in the last chapter, and then we will pass into a probabilistic version of LTSs, which, with small reinterpretations related to the quantum context, will be the protagonist of the continuation of this thesis.

The material in this chapter is based on standard textbooks [36, 42, 12].

4.1 Bisimulation Equivalence

In the previous chapter, we introduced the notions of simulation and simulation equivalence and, like the name suggests, the notion of bisimulation equivalence is based on them. It was proposed in [39] and [36] and it combines the notion of simulation with its inverse to define a *finer* equivalence. In this section, in contrast with the last chapter, we move our attention to **nondeterministic LTSs**, in which we present two distinct formulations of bisimulation. The former is called *strong bisimulation*, and like in the previous chapter, it treats the internal action τ like all the others. The latter, much more appreciated over the years, differentiates the τ-action from the others, defining an interesting way to view this relation.

Example 4.1.1. Having moved into the world of non-deterministic LTSs, it can be useful to remember that simulation equivalence and trace equivalence *do not coincide here*, contrary to deterministic LTSs, as can be seen in Figure 4.1.

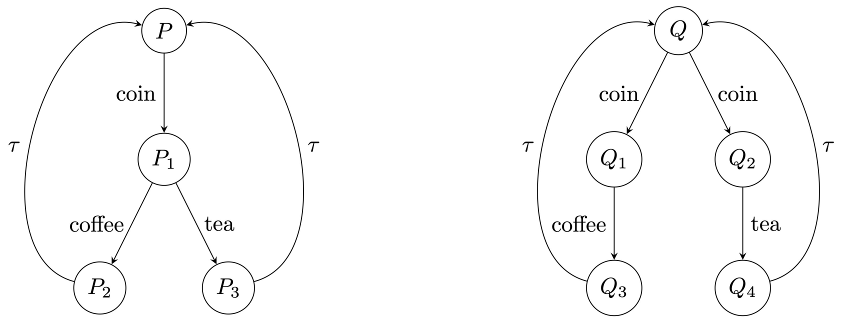


Figure 4.1: Two trace equivalent but **not** simulation equivalent LTSs

4.1.1 Strong Bisimulation

The notion of *strong bisimulation*, as already said, comes by combining a simulation relation $\mathcal{R}$ with its inverse $\mathcal{R}^{-1}$.

Definition 4.1.2. Given an LTS $L = \langle \mathcal{S}, Act, \rightarrow \rangle$, two processes $P, Q \in \mathcal{S}$, a binary relation $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S}$, is a **strong bisimulation** if $(P, Q) \in \mathcal{R}$ implies, for all $\alpha \in Act$:

- For each $P \xrightarrow{\alpha} P'$, then $Q \xrightarrow{\alpha} Q'$, for some Q' such that $(P', Q') \in \mathcal{R}$.
- For each $Q \xrightarrow{\alpha} Q'$, then $P \xrightarrow{\alpha} P'$, for some P' such that $(P', Q') \in \mathcal{R}$.

Then if $\mathcal{R}$ and its inverse $\mathcal{R}^{-1}$ are both simulations, $\mathcal{R}$ is a bisimulation. Given that, the derivation of the term *bi*simulation is a clear consequence.

Example 4.1.3. As stated before, bisimulation equivalence is finer than simulation equivalence, but the difference between the two is very subtle and to discuss it we avail ourselves of Figure 4.2. In this figure $P \simeq Q$, that means that $P \lesssim Q$ and $Q \lesssim P$. More precisely:

- $P \lesssim Q$ because there exists a relation $\mathcal{R}_0 : \{(P, Q), (P_1, Q_1), (P_2, Q_2), (P_3, Q_2), (P_4, Q_3)\}$ that is a simulation.
- $Q \lesssim P$ because there exists a relation $\mathcal{R}_1 : \{(Q, P), (Q_1, P_1), (Q_2, P_3), (Q_3, P_4)\}$ that is a simulation.

As can be seen, the inverse of $\mathcal{R}_0$, defined as

$$\mathcal{R}_0^{-1} : \{(Q, P), (Q_1, P_1), (Q_2, P_2), (Q_2, P_3)(Q_3, P_4)\}$$

is different from $\mathcal{R}_1$ and moreover *it is not a simulation.* In fact, there is no strong bisimulation containing P and Q, so $P \not\sim Q$.

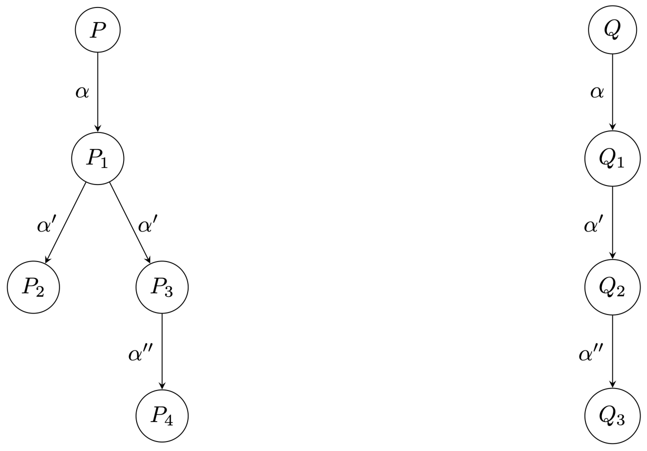


Figure 4.2: Two simulation equivalent but **not** bisimilar LTSs

Going forward, just like in Chapter 3.3.3, we proceed to expose a series of interesting proprieties of bisimulation.

Proposition 4.1.4. *Given an LTS $L = \langle \mathcal{S}, Act, \rightarrow \rangle$, the following relations are all strong bisimulations:*

1. *$Id_P = (\{(P, P) \mid \forall P \in \mathcal{S}\}$.*
2. *$\mathcal{R}_0\mathcal{R}_1 = \{(P, Q) \mid \exists R.(P, R) \in \mathcal{R}_0 \wedge (R, Q) \in \mathcal{R}_1\}$, for all strong bisimulations $\mathcal{R}_0, \mathcal{R}_1$.*
3. *$\bigcup_{i \in I} \mathcal{R}_i$, if $\mathcal{R}_i$ is a strong bisimulation, for all $i \in I$.*
4. *$\mathcal{R}^{-1} = \{(Q, P) \mid (P, Q) \in \mathcal{R}\}$, for any strong bisimulation $\mathcal{R}$.*

Proof. The proofs for (1), (2), (3) derive directly from the proof of Proposition 3.3.9. (4) is trivially true given Definition 4.1.2. □

Definition 4.1.5. Given an LTS $L = \langle \mathcal{S}, Act, \rightarrow \rangle$ and $P, Q \in \mathcal{S}$, P and Q are **strongly bisimilar**, written $P \sim Q$, if there exists a strong bisimulation $\mathcal{R}$ containing (P, Q). An equivalent definition may be

$$\sim = \bigcup \{\mathcal{R} : \mathcal{R} \text{ is a strong bisimulation}\}.$$

Then, using Proposition 4.1.4 and Definition 4.1.5, we can derive the following proposition.

Proposition 4.1.6. *For any LTS $\langle \mathcal{S}, Act, \rightarrow \rangle$:*

1. *The relation $\sim$ is an equivalence relation.*
2. *The relation $\sim$ is the largest strong bisimulation.*

Proof.

1. To prove that $\sim$ is an equivalence relation, we rely on Proposition 4.1.4. Given $P, Q, R \in \mathcal{S}$
   - For reflexivity, Proposition 4.1.4(1) tell us that for every $P \in \mathcal{S}$, $P \sim P$.
   - For symmetry, if $P \sim Q$, then there exists a strong bisimulation $\mathcal{R}$ containing (P, Q). Proposition 4.1.4(4) states that $\mathcal{R}^{-1}$ is also a strong bisimulation and therefore $Q \sim P$.

   - Lastly, for transitivity, if $P \sim Q$ and $Q \sim R$, then there exist two strong bisimulation $\mathcal{R}_0, \mathcal{R}_1$ with $(P, Q) \in \mathcal{R}_0$ and $(Q, R) \in \mathcal{R}_1$. Thus $(P, R) \in \mathcal{R}_0\mathcal{R}_1$ and, by Proposition 4.1.4(2), $P \sim R$.

2. The proof that $\sim$ is the largest strong bisimulation is trivial and derives directly from Definition 4.1.5. Given that $\sim$ is a strong bisimulation and include any other such, it is the largest one.

$\square$

Another interesting idea that we want to present is a generalization of the notion of strong bisimulation, called *strong bisimulation up to* $\approx$.

Definition 4.1.7. Given an LTS $\langle \mathcal{S}, Act, \rightarrow \rangle$ and a binary relation $\mathcal{R}$:

- The relation $\sim \mathcal{R} \sim$ is a composition of binary relations such that, given $(P, Q) \in \mathcal{R}$, $P \sim \mathcal{R} \sim Q$ means that there exist P', Q' such that:

$$P \sim P',\ Q \sim Q',\ P'\mathcal{R}Q'.$$

- $\mathcal{R}$ is a **strong bisimulation up to** $\sim$ if, given $(P, Q) \in \mathcal{R}$ and $\alpha \in Act$, $P\mathcal{R}Q$ implies:

  – For each $P \overset{\alpha}{\rightarrow} P'$, then $Q \overset{\alpha}{\rightarrow} Q'$, for some Q' such that $P' \sim \mathcal{R} \sim Q'$.

  – For each $Q \overset{\alpha}{\rightarrow} Q'$, then $P \overset{\alpha}{\rightarrow} P'$, for some Q' such that $P' \sim \mathcal{R} \sim Q'$.

Lemma 4.1.8. *If $\mathcal{R}$ is a strong bisimulation up to $\sim$, then $\sim \mathcal{R} \sim$ is a strong bisimulation.*

Proof. Given Definition 4.1.11, two processes $P, Q \in \mathcal{S}$ and $P \sim \mathcal{R} \sim Q$, thanks to the symmetry of strong bisimulation, we need only need to prove that, if $P \overset{\alpha}{\rightarrow} P'$, there must be a Q' such that $Q \overset{\alpha}{\rightarrow} Q'$ and $P' \sim \mathcal{R} \sim Q'$. To prove this, we proceed by parts, noting that, for some P_1 and Q_1, $P \sim P_1\mathcal{R}Q_1 \sim Q$:

- If $P \sim P_1$ and $P \xrightarrow{\alpha} P'$ then $P_1 \xrightarrow{\alpha} P'_1$ and $P' \sim P'_1$.
- If $Q_1 \sim Q$ and $Q_1 \xrightarrow{\alpha} Q'_1$ then $Q' \xrightarrow{\alpha} Q'$ and $Q'_1 \sim Q'$.
- If $P_1 \mathcal{R} Q_1$ and $P_1 \xrightarrow{\alpha} P'_1$ then $Q_1 \xrightarrow{\alpha} Q'_1$ and $P'_1 \sim \mathcal{R} \sim Q'_1$.

Composing these three elements and recalling the transitivity of $\sim$, we obtain that $P' \sim \mathcal{R} \sim Q'$ and then $\sim \mathcal{R} \sim$ is a strong bisimulation.

$\square$

Example 4.1.9. To make clear why strong bisimulation up to $\sim$ is more general than strong bisimulation, we rely on a simple example, using Figure 4.3. In this figure are represented the LTSs of two bisimilar processes P, Q. Then, we can define a relation $\mathcal{R}$ that is a bisimulation in this way:

$$\mathcal{R} : \{(P, Q), (P_1, Q_1), (P_2, Q_2), (P_3, Q_2)\}$$

Note that $\mathcal{R}$ contains what may seem to be a redundancy: the pairs (P_2, Q_2) and (P_3, Q_2) have the same second member, and their first member are also bisimilar because they are both dead process and the only transition that leads to them comes from the same process and with the same action. The intuition here, derived from [36], is that with the bisimulation up to $\sim$ we can use relations which do not require to list members of bisimulation which are identical "up to $\sim$". In fact, we can define a relation $\mathcal{R}'$ that is a bisimulation up to $\sim$ for P and Q in this way:

$$\mathcal{R}' : \{(P, Q), (P_1, Q_1), (P_2, Q_2)\}$$

This can be very useful in the presence of large LTSs.

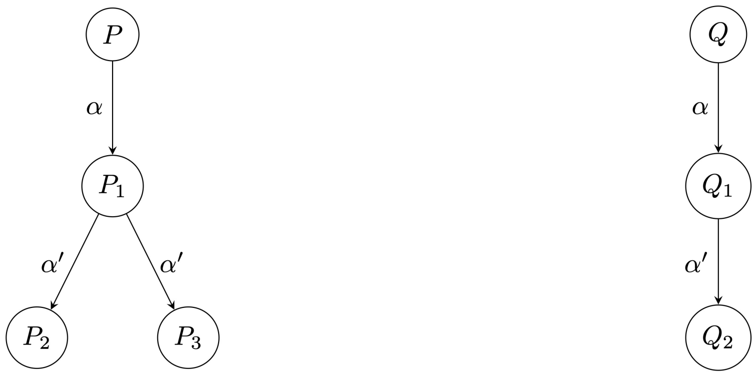


Figure 4.3: Two bisimulation equivalent LTSs

To conclude the section, we can summarize all the various strong equivalences relations introduced in Chapters 3 and 4 using Figure 4.4.

In Figure 4.4 the arrows from equivalences notions means that the former is finer (included) than the latter.

$$\begin{tikzcd}
\text{Isomorphism equivalence } (\cong) \arrow[d] \\
\text{Bisimulation equivalence } (\sim) \arrow[d] \\
\text{Simulation equivalence } (\simeq) \arrow[d] \\
\text{Trace equivalence } (=)
\end{tikzcd}$$

Figure 4.4: Relation between equivalences relations

4.1.2 Weak Bisimulation

The notion of *weak bisimulation* or *observational equivalence* is probably one of the most used equivalences in literature. The main intuition here relies on the fact that the τ-actions are internal actions and then an external observer cannot distinguish between them. Note that this matches much more closely the real world, in which we often want to equate systems with different internal behavior, but with the same external behavior.

To introduce the concept of weak bisimulation, we need a few preliminary definitions.

Definition 4.1.10. Given an LTS $L = \langle \mathcal{S}, Act, \rightarrow \rangle$, two processes $P, Q \in \mathcal{S}$ and a sequence of actions $s = \alpha_1, \alpha_2, ..., \alpha_n$:

- If $s \in Act^*$, then $\hat{s} \in (Act - \{\tau\})^*$ is the sequence of actions obtained by deleting all the τ-actions from s.
- If $s \in Act^*$, then $P \xrightarrow{s} Q$ if:

$$P \xrightarrow{\alpha_1} \dots \xrightarrow{\alpha_n} Q$$

- Let $s \in (Act - \{\tau\})^*$, then $P \overset{s}{\Rightarrow} Q$ if:

$$P \xrightarrow{\tau^*}\xrightarrow{\alpha_1}\xrightarrow{\tau^*} \dots \xrightarrow{\tau^*}\xrightarrow{\alpha_n}\xrightarrow{\tau^*} Q$$

Intuitively, $P \overset{\alpha}{\Rightarrow} P'$ means that the system described by the process P turns into the system described by the process P' performing a sequence of invisible τ-actions and exactly one α action, with $\alpha \neq \tau$.

Then we can give the definition of weak bisimulation.

Definition 4.1.11. Given an LTS $L = \langle \mathcal{S}, Act, \rightarrow \rangle$ and two processes $P, Q \in \mathcal{S}$, a binary relation $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S}$, is a **weak bisimulation** if $(P, Q) \in \mathcal{R}$ implies, for all $\alpha \in Act$:

- For each $P \xrightarrow{\alpha} P'$, then $Q \overset{\hat{\alpha}}{\Rightarrow} Q'$, for some Q' such that $(P', Q') \in \mathcal{R}$.

- For each $Q \overset{\alpha}{\rightarrow} Q'$, then $P \overset{\hat{\alpha}}{\Rightarrow} P'$, for some P' such that $(P', Q') \in \mathcal{R}$.

Recalling the last section, we can give the analogous version of definition and proposition for weak bisimulation.

Proposition 4.1.12. *Given an LTS $L = \langle \mathcal{S}, Act, \rightarrow \rangle$, the following relation are all weak bisimulations:*

1. $Id_P = (\{(P, P) \mid \forall P \in \mathcal{S}\}$.
2. $\mathcal{R}_0\mathcal{R}_1 = \{(P, Q) \mid \exists R.(P, R) \in \mathcal{R}_0 \wedge (R, Q) \in \mathcal{R}_1\}$, *for all weak bisimulations* $\mathcal{R}_0, \mathcal{R}_1$.
3. $\bigcup_{i \in I} \mathcal{R}_i$, *if* $\mathcal{R}_i$ *is a weak bisimulation, for all* $i \in I$.
4. $\mathcal{R}^{-1} = \{(Q, P) \mid (P, Q) \in \mathcal{R}\}$, *for all weak bisimulation* $\mathcal{R}$.

Proof. Also in this case, the proof derives directly from Proposition 4.1.4. The only difference is that for $\mathcal{R}_0\mathcal{R}_1$, we need a further result: If $(R, Q) \in \mathcal{R}_i$ and $R \overset{\hat{\alpha}}{\Rightarrow} R'$, then for some Q', $Q \overset{\hat{\alpha}}{\Rightarrow} Q'$ and $(R', Q') \in \mathcal{R}_i$. □

The definition of weakly bisimilar processes is the following.

Definition 4.1.13. Given an LTS $L = \langle \mathcal{S}, Act, \rightarrow \rangle$ and two processes $P, Q \in \mathcal{S}$, P and Q are **weakly bisimilar**, written $P \approx Q$, if there exists a weak bisimulation $\mathcal{R}$ containing (P, Q). An equivalent definition may be

$$\approx = \bigcup \{\mathcal{R} : \mathcal{R} \text{ is a weak bisimulation}\}.$$

Having these two alternative definitions for weak bisimulation, we can prove that it is an equivalence relation.

Proposition 4.1.14. *For any LTS $\langle \mathcal{S}, Act, \rightarrow \rangle$:*

1. *The relation $\approx$ is an equivalence relation.*
2. *The relation $\approx$ is the largest weak bisimulation.*

Proof. The proof follows identically the proof of Proposition 4.1.6, using Proposition 4.1.12 and Definition 4.1.13. □

It is easy to see that the definition of weak bisimulation is coarser than Definition 4.1.2.

Example 4.1.15. To give a clear vision of how different are the notion of weak and strong bisimulation, we can use Figure 4.5. In this simple case, $P \approx Q$ but $P \not\sim Q$, due to the τ-action from P_1 to P_2.

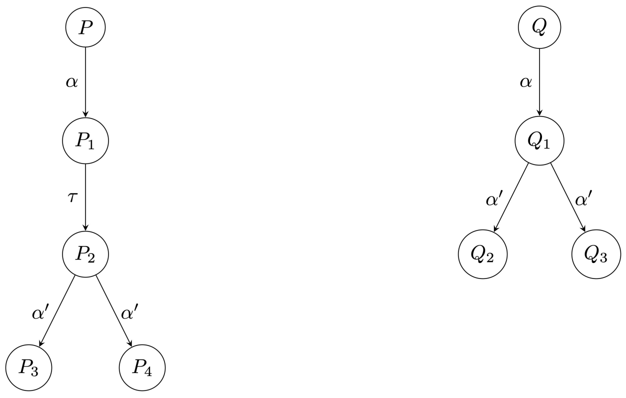


Figure 4.5: Two weak bisimilar LTSs

As stated in [43] and contrary to what it seems, the *bisimulation up to* technique applies well in the strong bisimulation case, but does not directly generalize to the weak case. For this reason, we introduce a notion very similar to weak bisimilarity, called *expansion*, which allows us to derive an up-to method for weak bisimilarity as well.

Let us start by giving the definition of expansion.

Definition 4.1.16. Given an LTS $L = \langle \mathcal{S}, Act, \rightarrow \rangle$ and two processes $P, Q \in \mathcal{S}$, a binary relation $\mathcal{T} \subseteq \mathcal{S} \times \mathcal{S}$, is an **expansion** if $(P, Q) \in \mathcal{T}$ implies, for all $\alpha \in Act$:

- For each $P \xrightarrow{\alpha} P'$, then $Q \overset{\alpha}{\Rightarrow} Q'$, for some Q' such that $(P', Q') \in \mathcal{T}$.
- For each $Q \xrightarrow{\alpha} Q'$, then $P \xrightarrow{\hat{\alpha}} P'$, for some P' such that $(P', Q') \in \mathcal{T}$.

We can state that expansion is a preorder derived from $\approx$ by the comparison of the number of silent actions.

Definition 4.1.17. Given an LTS $L = \langle \mathcal{S}, Act, \rightarrow \rangle$ and two processes $P, Q \in \mathcal{S}$, we say that Q **expands** P, written $P \lessapprox Q$, if there exists an expansion $\mathcal{T}$ containing (P, Q) An equivalent definition may be

$$\lessapprox = \bigcup \{\mathcal{T}\text{, with } \mathcal{T} \text{ expansion }\}.$$

As can be noted, the expansion relation is an asymmetric version of $\approx$, indeed $P \lessapprox Q$ means that $P \approx Q$ and also that Q uses at least as many resources as P, i.e., P uses *no more* τ-actions than Q. Moreover, as the next theorem proves, this relation resides "between" strong and weak bisimulation.

Theorem 4.1.18. *It holds that* $\sim \subset \lessapprox$ *and* $\lessapprox \subset \approx$.

Proof. The inclusion $\lessapprox \subseteq \approx$ is trivial. For $\sim \subseteq \lessapprox$ simply note that if $\overset{\alpha}{\Rightarrow}$ is a transition that includes 0 or more τ-actions, then $\overset{\alpha}{\rightarrow} \subseteq \overset{\alpha}{\Rightarrow}$, and the same reasoning also applies to $\overset{\hat{\alpha}}{\rightarrow}$. For the strictness note that given a process $P \in \mathcal{S}$:

$$P \not\sim \tau.P, \qquad P \lessapprox \tau.P, \qquad \tau.P \not\lessapprox P, \qquad \tau.P \approx P.$$

□

Since $\lessapprox \subset \approx$ we can now define the *expansion up to* $\lessapprox$.

Definition 4.1.19. $\mathcal{T}$ is an **expansion up to** $\lessapprox$ if, given $(P, Q) \in \mathcal{S}$ and $\alpha \in Act$, $P\mathcal{T}Q$ implies:

- For each $P \overset{\alpha}{\rightarrow} P'$, then $Q \overset{\alpha}{\Rightarrow} Q'$, for some Q' such that $P' \sim \mathcal{R} \lessapprox Q'$.
- For each $Q \overset{\alpha}{\rightarrow} Q'$, then $P \overset{\hat{\alpha}}{\rightarrow} P'$, for some Q' such that $P' \lessapprox \mathcal{R} \lessapprox Q'$.

Lemma 4.1.20. *If* $\mathcal{T}$ *is an expansion up to* $\lessapprox$*, then* $\mathcal{T}$ *is an expansion.*

Proof. The proof derives directly from the proof of Lemma 4.1.8, obviously with the respective changes due to the definition of expansion. □

4.2 Probabilistic Bisimulation

In the last section, we have presented the notion of bisimulation in two different versions. A strong one, in which τ-actions are treated like all the other actions, and a weak one, in which instead τ-actions are indistinguishable to an external observer. In this section, we want to take another step to get closer to our research work, presenting the notion of bisimulation based on a probabilistic version of LTSs, called *probabilistic labelled transition systems* (pLTSs). Clearly, we must start by presenting the basic definitions and lemmas about pLTSs. However, we will give particular emphasis to those which are called *lifted relations*, and then we conclude the section talking about probabilistic bisimulation itself.

First, let us prepare the scene by giving some basic definitions regarding the field of probability, starting from the definition of *discrete probability distribution* and *point distribution.*

Definition 4.2.1.

- A **discrete probability distribution** over a set S, is a function $\nu : S \to [0,1]$ with $\sum_{s\in S} \nu(s) = 1$, with countable support $\lceil \nu \rceil = \{s \in S \,|\, \nu(s) > 0\}$.
- The **point distribution**, denoted by $\bar{s}$ is a distribution that assigns 1 to s and 0 to all other elements of S, so that $\lceil \bar{s} \rceil = \{s\}$.

Now using $\mathcal{D}(\mathcal{S})$ to denote the set of all the probability distributions over $\mathcal{S}$, ranged over by $\lambda, \gamma \ldots$, we can proceed giving the definition of probabilistic labelled transition system.

Definition 4.2.2. A **probabilistic labelled transition system** is a triple $\langle \mathcal{S}, Act, \to \rangle$ where:

1. $\mathcal{S}$ is a finite set of states.

2. *Act* is a set of transition labels, containing the silent transition τ.

3. $\rightarrow$ is a transition relation, that is, a subset of $\mathcal{S} \times Act \times \mathcal{D}(\mathcal{S})$.

4.2.1 Lifting relations

When we move into the probabilistic setting, systems are often modelled as distributions over states. Then we need to *lift* the relation presented on state to distribution, in order to compare different systems.

Definition 4.2.3. Let $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{D}(\mathcal{S})$ be a relation from states to distributions. We use $\mathcal{R}^\dagger$ to indicate the corresponding **lifted relation** over distributions. Here $\mathcal{R}^\dagger$ is the smallest relation $\mathcal{R}^\dagger \subset \mathcal{D}(\mathcal{S}) \times \mathcal{D}(\mathcal{S})$ that satisfies:

1. $s\mathcal{R}\gamma$ implies $\bar{s}\mathcal{R}^\dagger\gamma$, where $\bar{s}$ is the point distribution.

2. Linearity: given a finite set I, if $\lambda_i\mathcal{R}^\dagger\gamma_i$ for all $i \in I$, then $(\sum_{i\in I} p_i \cdot \lambda_i)\mathcal{R}^\dagger(\sum_{i\in I} p_i \cdot \gamma_i)$ for any choice of $p_i \in [0,1]$, such that $\sum_{i\in I} p_i = 1$.

Now we can give an alternative characterization of lifting, taken by [13], which will be useful in the following pages.

Lemma 4.2.4. $\lambda\mathcal{R}^\dagger\gamma$ *if and only if there is a finite index set I, such that:*

1. $\lambda = \sum_{i\in I} p_i \cdot \bar{s}_i$.

2. $\gamma = \sum_{i\in I} p_i \cdot \gamma_i$.

3. $s_i\mathcal{R}\gamma, \quad \forall i \in I$.

Proof.
($\Leftarrow$) Suppose that there is an index set I that respects (1), (2), (3). By clause (3) and 4.2.3(1) we have $\bar{s}_i\mathcal{R}^\dagger\gamma_i$ for each $i \in I$. Using also the linearity rule in 4.2.3 we obtain that $(\sum_{i\in I} p_i \cdot \bar{s}_i)\mathcal{R}^\dagger(\sum_{i\in I} p_i \cdot \gamma_i)$.

($\Rightarrow$) Now let us proceed by induction:

- if $\lambda \mathcal{R}^\dagger \gamma$ because $\lambda = \bar{s}$ and $s \mathcal{R} \gamma$ then we only take I to be the set $\{i\}$ with $p_i = 1$ and $\gamma_i = \gamma$.

- if $\lambda \mathcal{R}^\dagger \gamma$ because $\lambda = \sum_{i \in I} p_i \cdot \lambda_i$, $\gamma = \sum_{i \in I} p_i \cdot \gamma_i$ and $\lambda_i \mathcal{R}^\dagger \gamma_i$ for each $i \in I$ then, by induction hypothesis, there are index sets K_i such that $\lambda_i = \sum_{k \in K_i} p_{ik} \cdot \bar{s_{ik}}$, $\gamma_i = \sum_{k \in K_i} p_{ik} \cdot \gamma_{jk}$ and $s_{ik} \mathcal{R} \gamma_{jk}$ for each $i \in I$ and $k \in K_i$. It follows that $\lambda = \sum_{i \in I} \sum_{k \in K_i} p_i p_{ik} \cdot \bar{s}_{ik}$, $\gamma = \sum_{i \in I} \sum_{k \in K_i} p_i p_{ik} \cdot \gamma_{jk}$ and $s_{ik} \mathcal{R} \gamma_{jk}$ for each $i \in I$ and $k \in K_i$. In conclusion, it suffices to take $\{ik \,|\, i \in I, k \in K\}$ as the index set and $\{p_i p_{ik} \,|\, i \in I, k \in K\}$ as the collection of probabilities.

□

We proceed by defining the notion that allows us to "*decompose*" the states of a distribution, called *left-decomposability.*

Definition 4.2.5. Given a finite set I, a relation $\mathcal{R} \subset \mathcal{D}(\mathcal{S}) \times \mathcal{D}(\mathcal{S})$ over distributions is called **left-decomposable** if $(\sum_{i \in I} p_i \cdot \lambda_i) \mathcal{R} \gamma$ implies that γ can be written as $(\sum_{i \in I} p_i \cdot \gamma_i)$ for every $i \in I$ such that $\lambda_i \mathcal{R} \gamma_i$.

Given the definition of left-decomposability, the next lemma tells us that **any** lifted relation is left-decomposable.

Lemma 4.2.6. *For any $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{D}(\mathcal{S})$, the lifted relation $\mathcal{R}^\dagger$ over distribution is left-decomposable.*

Proof. Given a finite set I, suppose that $\lambda = \sum_{i \in I} p_i \cdot \lambda_i$ and $\lambda \mathcal{R}^\dagger \gamma$, we need to find a family of γ_i such that:

1. $\lambda_i \mathcal{R}^\dagger \gamma_i$.

2. $\gamma = \sum_{i \in I} p_i \cdot \gamma_i$,

From the alternative definition on lifting given as Lemma 4.2.4 we know that:

$$\lambda = \sum_{k \in K} p_k \cdot \bar{s}_k, \qquad s_k \mathcal{R} \gamma_j, \qquad \gamma = \sum_{k \in K} p_k \cdot t_k.$$

Now let us define γ_i to be

$$\sum_{s\in\langle\lambda_i\rangle} \lambda_i(s)\cdot\left(\sum_{\{k\in K\,|\,s=s_k\}} \frac{p_k}{\lambda(s)}\cdot\gamma_j\right). \tag{4.1}$$

We can simply note that λ_s can be written as $\sum_{\{k\in K\,|\,s=s_k\}} p_k$ and consequently:

$$\lambda_i = \sum_{s\in\langle\lambda_i\rangle} \lambda_i(s)\cdot\left(\sum_{\{k\in K\,|\,s=s_k\}} \frac{p_k}{\lambda(s)}\cdot\bar{s}_j\right). \tag{4.2}$$

Since $s_j\mathcal{R}\gamma_k$, this proved (1).

Let us go into (2). First, we are shortening the sum $\sum_{\{k\in K\,|\,s=s_k\}} \frac{p_k}{\lambda(s)}\cdot\gamma_j$ to $\Xi(s)$. Hence, $\sum_{i\in I} p_i\cdot\gamma_i$ can be written as:

$$\begin{aligned}&\sum_{s\in\langle\lambda\rangle}\sum_{i\in I} p_i\cdot\lambda_i(s)\cdot\Xi(s)\\ &=\sum_{s\in\langle\lambda\rangle}\lambda(s)\cdot\Xi(s).\end{aligned} \tag{4.3}$$

Lastly, $\lambda(s)\cdot\Xi(s)=\sum_{\{k\in K\,|\,s=s_k\}} p_k\cdot t_k$ and thus we have:

$$\begin{aligned}\sum_{i\in I} p_i\cdot\gamma_i &= \sum_{s\in\lambda}\sum_{\{k\in K\,|\,s=s_k\}} p_j\cdot\gamma_j\\ &= \sum_{k\in K} p_k\cdot t_k = t.\end{aligned} \tag{4.4}$$

□

When we are considering the transition relation $\rightarrow$ over distributions, it is clear that there will be some small differences compared to a standard LTS. The next definition helps us understand those differences.

Definition 4.2.7. Considering a pLTS $\langle\mathcal{S}, Act, \rightarrow\rangle$, we establish the following transition over distributions:

1. $\xrightarrow{\hat{\tau}}^{\dagger}$. Let $s\xrightarrow{\hat{\tau}}\lambda$ if either $s\xrightarrow{\tau}\lambda$ or $\lambda = s$, and lift it to distributions.

2. $\xrightarrow{\hat{\alpha}}^{\dagger}$. Let $s \xrightarrow{\hat{\alpha}} \lambda$ if $s \xrightarrow{\alpha} \lambda$ for $\alpha \in Act$ and $\alpha \neq \tau$, and lift it to distributions.

3. $\overset{\hat{\tau}}{\Longrightarrow}^{\dagger}$. The transition $\overset{\hat{\tau}}{\Longrightarrow}^{\dagger}$ is equal to $(\xrightarrow{\hat{\tau}}^{\dagger})^{*}$, the reflexive and transitive closure of $\xrightarrow{\hat{\tau}}$.

4. $\overset{\hat{\alpha}}{\Longrightarrow}^{\dagger}$.The transition $\overset{\hat{\alpha}}{\Longrightarrow}^{\dagger}$ is equal to $(\overset{\hat{\tau}}{\Longrightarrow}\xrightarrow{\hat{\alpha}}\overset{\hat{\tau}}{\Longrightarrow})^{\dagger}$ for $\alpha \in Act$ and $\alpha \neq \tau$.

   For point distributions, we write $s \overset{\hat{\alpha}}{\Longrightarrow}^{\dagger} P$ rather than $\bar{s} \overset{\hat{\alpha}}{\Longrightarrow}^{\dagger} \gamma$.

It is important to point out that here $\overset{\hat{\alpha}}{\Longrightarrow}^{\dagger}$ **is not** a lifted transition. However, the following proposition and lemma show that it is still both linear and left-decomposable.

Proposition 4.2.8. *If two binary relations $\mathcal{R}_1$and $\mathcal{R}_2$ are both linear and left-decomposable, then so is $\mathcal{R}_1\mathcal{R}_2$.*

Proof. Let us suppose that

$$(\lambda, \gamma) \in \mathcal{R}_1\mathcal{R}_2.$$

Then for some Q we have

$$(\lambda, \nu) \in \mathcal{R}_1 \text{ and } (\nu, \gamma) \in \mathcal{R}_2.$$

Now, since $\mathcal{R}_1$ is linear and left-decomposable, $(\sum_{i\in I} p_i \cdot \lambda_i)\mathcal{R}_1\nu$ implies that ν can be written as $(\sum_{i\in I} p_i \cdot \nu_i)$. Following the same argument since $\mathcal{R}_2$ is also linear and left-decomposable and $\nu = (\sum_{i\in I} p_i \cdot \nu i)$, implies that γ can be written as $(\sum_{i\in I} p_i \cdot \gamma_i)$. □

Lemma 4.2.9. *The transition relations $\overset{\hat{\alpha}}{\Longrightarrow}^{\dagger}$ are both left-decomposable and linear.*

Proof. The proof relies on the fact that both properties are preserved by composition. So if two relations $\mathcal{R}_1, \mathcal{R}_2$ are both linear and left-decomposable, then so is $\mathcal{R}_1 \cdot \mathcal{R}_2$. Then since $\overset{\hat{\alpha}}{\Longrightarrow}^{\dagger}$ is formed by recurrent composition of $\xrightarrow{\hat{\tau}}^{\dagger}$

and exactly one $\xrightarrow{\hat{\alpha}}^{\dagger}$ which we know are both linear and left-decomposable, then also $\overset{\hat{\alpha}}{\Longrightarrow}^{\dagger}$ it is. □

Note that in the continuation we will use $\mathcal{R}$ and $\mathcal{R}^{\dagger}$ interchangeably, e.g., we will usually write $\xrightarrow{\hat{\alpha}}$ for $\xrightarrow{\hat{\alpha}}^{\dagger}$ and so on.

4.2.2 Probabilistic Bisimulation Equivalences

Having presented probabilistic LTSs and the lifted relations on them, we can go forward presenting briefly the notions of *strong probabilistic bisimulation* and *weak probabilistic bisimulation* starting, as in the previous chapter, with the notion of *probabilistic simulation.*

Definition 4.2.10. Given an pLTS $L = \langle \mathcal{S}, Act, \rightarrow \rangle$, two processes $P, Q \in \mathcal{S}$, a distribution $\lambda \in \mathcal{D}(\mathcal{S})$ and a binary relation $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S}$, $\mathcal{R}$ is a **probabilistic simulation** if $(P, Q) \in \mathcal{R}$ implies for all $\alpha \in Act$:

- For each $P \xrightarrow{\alpha} \lambda$, then $Q \xrightarrow{\alpha} \gamma$, for some $\gamma \in \mathcal{D}(\mathcal{S})$ such that $(\lambda, \gamma) \in \mathcal{R}$.

The intuition here, similarly as simulation equivalence on LTSs, is that Q can simulate the behavior of P if whatever action of P can lead to a distribution λ, then also Q, performing the same action, can lead to a distribution γ and so on.

The definition of probabilistic simulation brings us directly to the notion of strong probabilistic bisimulation.

Definition 4.2.11. Given a pLTS $L = \langle \mathcal{S}, Act, \rightarrow \rangle$, two processes $P, Q \in \mathcal{S}$, a binary relation $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S}$, is a **strong probabilistic bisimulation** if $(P, Q) \in \mathcal{R}$ implies, for all $\alpha \in Act$:

- For each $P \xrightarrow{\alpha} \lambda$, then $Q \xrightarrow{\alpha} \gamma$, for some $\gamma \in \mathcal{D}(\mathcal{S})$ such that $(\lambda, \gamma) \in \mathcal{R}$.
- For each $Q \xrightarrow{\alpha} \gamma$, then $P \xrightarrow{\alpha} \lambda$, for some $\lambda \in \mathcal{D}(\mathcal{S})$ such that $(\lambda, \gamma) \in \mathcal{R}$.

Note that also in this case, the bisimulation notion derive from the composition of a simulation $\mathcal{R}$ and its inverse $\mathcal{R}^{-1}$.

The differentiation between τ-actions and normal action lead us to the last definition of this chapter.

Definition 4.2.12. Given a pLTS $L = \langle \mathcal{S}, Act, \rightarrow \rangle$ and two distributions $\lambda, \gamma \in \mathcal{S}$, a binary relation $\mathcal{R} \subseteq \mathcal{D}(\mathcal{S}) \times \mathcal{D}(\mathcal{S})$, is a **weak probabilistic bisimulation** if $(\lambda, \gamma) \in \mathcal{R}$ implies, for all $\alpha \in Act$ and all finite set of probabilities $\{p_i \,|\, i \in I\}$ with $\sum_{i \in I} p_i = 1$:

- For each $\lambda \overset{\alpha}{\Rightarrow} \sum_{i \in I} p_i \cdot \lambda_i$, then $\gamma \overset{\alpha}{\Rightarrow} \sum_{i \in I} p_i \cdot \gamma_i'$, for some γ_i such that $(\lambda_i, \gamma_i) \in \mathcal{R}$ for each $i \in I$.
- For each $\gamma \overset{\alpha}{\Rightarrow} \sum_{i \in I} p_i \cdot \gamma_i$, then $\lambda \overset{\alpha}{\Rightarrow} \sum_{i \in I} p_i \cdot \lambda_i'$, for some λ_i such that $(\lambda_i, \gamma_i) \in \mathcal{R}$ for each $i \in I$.

All the equivalence notions presented in this section were dealt with quickly because they derive entirely from the nondeterministic case. Furthermore, they will only serve us as a baseline to better understand the successive notions in the next chapter.

Chapter 5

Bisimulation on Quantum Processes

After presenting quantum computing, process algebras and bisimulation, we are ready to present all notions used for the cryptographic proof in Chapter 6. In this chapter, we start by presenting a quantum extension of CCS, called *quantum CCS* (qCCS), of which we will present the modified syntax and semantics. Then, we present three different bisimulation notions used in the literature to compare quantum processes.

5.1 qCCS

As in the theory of the concurrent systems, the process algebra's approach has also landed in the verification of quantum systems. Several proposals have been made, like QPAlg [26] or CQP [20]. For our research work, having presented CCS, we use a quantum extension of it, called qCCS [18, 49].

5.1.1 Syntax

Having moved into the quantum world, qCCS syntax needs to define actions and operators to model quantum systems. Being qCCS an extension of CCS, most of the operators and actions derive from it. But, when we come to define the actions, we must consider not only the input and output of *classical* variables through classical channels, but also of *quantum* variables and their related communication channels. Furthermore, we also need to be able to model the application of quantum gates and the measurement.

More formally, in qCCS three types of data are considered: Real for real numbers, Bool for booleans and Qbt for qubits. Therefore, $cVar$, ranged over by $x, y, \ldots$, and $qVar$, ranged over $q, r, \ldots$, are respectively the countably infinite sets that represent the classical variables and the quantum variables. Let us assume a set Exp, ranged over by $e, e', \ldots$, which includes $cVar$ as a subset, of classical expressions and a set $bExp$, ranged over by $b, b', \ldots$, of Boolean valued expressions. Let $cChan$, ranged over by $c, d, \ldots$, and $qChan$, by $\mathrm{c}, \mathrm{d}, \ldots$, be the sets of classical and quantum channels. A relabelling function f is defined as a map on $cChan \cup qChan$ to itself such that $f(cChan) \subseteq cChan$ and $f(qChan) \subseteq qChan$. Sequences of classical or quantum variables are often abbreviated with $\tilde{x}$ or $\tilde{q}$ respectively. Consider also a set of *process constants*, i.e., names for processes, ranged over by $A, B, \ldots$, assigned to each of them, there are two non-negative integers $ar_c(A)$ and $ar_q(A)$. If, for example, $|\tilde{x}| = ar_c(A)$ and $|\tilde{q}| = ar_q(A)$ then $A(\tilde{x}, \tilde{q})$ is called a process constant. Furthermore, when $ar_q(A)$ and $ar_c(A)$ are both equal to 0, then we also use A as the only process constant produced by A.

Thanks to the notation presented erlier, we can now easily introduce the actions of qCCS, with the following BNF:

$$\begin{array}{llr}
\alpha ::= \tau, & \text{for the internal action.} & \\
\quad \mid \; c!e, \; \text{with } c \in \mathit{cChan} \text{ and } e \in \mathit{Exp}, & \text{for the classical output.} & \\
\quad \mid \; c?x, \; \text{with } c \in \mathit{cChan} \text{ and } x \in \mathit{cVar}, & \text{for the classical input.} & \\
\quad \mid \; \mathrm{c}!q, \; \text{with } \mathrm{c} \in \mathit{qChan} \text{ and } q \in \mathit{qVar}, & \text{for the quantum output.} & \\
\quad \mid \; \mathrm{c}?q, \; \text{with } \mathrm{c} \in \mathit{qChan} \text{ and } q \in \mathit{qVar}, & \text{for the quantum input.} & \\
\quad \mid \; \Psi[\tilde{q}], & \text{for the super-operator application.} & \\
\quad \mid \; M[\tilde{q}; x], & \text{for the measurement operation.} & \\
 & & (5.1)
\end{array}$$

Then, we can define Act_{qCCS} as

$$\begin{aligned}
Act_{qCCS} \; &= \{c!x, c?x \mid c \in \mathit{cChan}, x \in \mathit{cVar}\} \\
&\cup \; \{\mathrm{c}!q, \mathrm{c}?q \mid \mathrm{c} \in \mathit{qChan}, q \in \mathit{qVar}\} \\
&\cup \; \{\tau\}.
\end{aligned} \tag{5.2}$$

Remark. The application of the quantum super-operator and the quantum measurement *are not* in Act_{qCCS}, because those actions are *invisible* from the outside.

Thus, assuming a set S_{qCCS} of valid qCCS processes, ranged over by $P, Q, \ldots,$ we can define qCCS syntax. Note that we present an extension of qCCS, in which the probabilistic choice is added directly to the syntax, as proposed in [17].

$$
\begin{array}{lll}
P, Q ::= \textbf{nil}, & & \text{for the terminated process.} \\
\quad \mid & \alpha.P, & \text{for the prefixing operation.} \\
\quad \mid & P + Q, & \text{for the choice operation.} \\
\quad \mid & P \parallel Q, & \text{for the parallel composition.} \\
\quad \mid & P \backslash L, & \text{for the restriction operation.} \\
\quad \mid & P[f], & \text{for the relabelling operation.} \\
\quad \mid & A(\tilde{e}, \tilde{q}), & \text{for the process constant.} \\
\quad \mid & \textbf{if } b \textbf{ then } Q, \text{ with } b \in bExp & \text{for the conditional operation.} \\
\quad \mid & \sum_{i \in I} p_i \, P_i, & \text{for the probabilistic choice operation.}
\end{array}
\tag{5.3}
$$

Furthermore, let qv be a function from $qProc \to 2^{qVar}$ that represents the sets of free quantum variables. We can define $qv(P)$ with $P \in qProc$ inductively as follows:

$$
\begin{array}{rlrl}
qv(\textbf{nil}) &= \emptyset & qv(\tau.P) &= qv(P) \\
qv(c!e.P) &= qv(P) & qv(c?x.P) &= qv(P) \\
qv(\mathrm{c}!q.P) &= qv(P) \cup \{q\} & qv(\mathrm{c}?q.P) &= qv(P) - \{q\} \\
qv(\Psi[\tilde{q}].P) &= qv(P) \cup \tilde{q} & qv(M[\tilde{q}; x].P) &= qv(P) \cup \tilde{q} \\
qv(P + Q) &= qv(P) \cup qv(Q) & qv(P \parallel Q) &= qv(P) \cup qv(Q) \\
qv(P \backslash L) &= qv(P) & qv(P[f]) &= qv(P) \\
qv(A(\tilde{e}, \tilde{q})) &= \tilde{q} & qv(\textbf{if } b \textbf{ then } P) &= qv(P)
\end{array}
\tag{5.4}
$$

To prevent physically unimplementable quantum processes, we add the following requirement: In $\mathrm{c}!q.P$, $q \notin qv(P)$ and $qv(P) \cap qv(Q) = \emptyset$ in $P \parallel Q$.

Example 5.1.1. As for Chapter 3, we give an example to understand how a simple quantum system can be modeled as a qCCS process. The circuit in

Figure 1.3, assuming $q_0 = q_1 = |0\rangle$, can be defined as

$$P = H[q_0].Cnot[q_0, q_1].M[q_0; x_0].M[q_1; x_1].out!(x_0, x_1).\mathbf{nil}$$

It is also easy to check $qv(P)$. Given that $qv(out!(x_0, x_1).\mathbf{nil}) = qv(\mathbf{nil}) = \emptyset$, and $qv(H[q_0].Cnot[q_0, q_1].M[q_0; x_0].M[q_1; x_1]) = \{q_0, q_1\}$, $qv(P) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$.

5.1.2 Transitional Semantics

Before presenting the transitional semantics of qCCS we need to prepare the scene. First, a process P is called *closed* if $fv(P) = \emptyset$, or rather it contains no free *classical* variable. Let us assume an *evaluation* function $\psi\{v/x\}$ to be a function that maps x to v. Let us also assume that each quantum variable $q \in qVar$ has an associated 2-dimensional Hilbert space $\mathcal{H}_q$ that represents the state space of the system, also called a q-system. That means for any $\tilde{q} \subset qVar$, $\mathcal{H}_{\tilde{q}} = \bigotimes_{q \in \tilde{q}} \mathcal{H}_q$, and this is valid for every subset of $qVar$. In particular, the state space of the whole environment is defined as $\mathcal{H} = \mathcal{H}_{qVar}$, which is a countably infinite dimensional Hilbert space. Supposing that P is a closed quantum process, a pair in the form $\langle P, \rho \rangle$ is called a *configuration*, where $\rho \in \mathcal{D}(\mathcal{H})$ is a density operator on $\mathcal{H}$. Given that $\mathcal{H}$ is infinite dimensional, ρ should be understood as a density operator on some *finite dimensional* subspace of $\mathcal{H}$ which contains $\mathcal{H}_{qv(P)}$. The set of all configurations is denoted by $\mathcal{CON}$, and is ranged over by $B, C \ldots$. Lastly, we use $\mathcal{D}(\mathcal{CON})$ for denoting the set of all the probability distributions over $\mathcal{CON}$ and we overload the notation using $\lambda, \gamma, \ldots$, like with $\mathcal{D}(\mathcal{S})$, for ranging over the set. More precisely:

$$\mathcal{D}(\mathcal{CON}) = \{\lambda : \mathcal{CON} \to [0,1] \mid \sum_{\lambda(C)>0} \lambda(C) = 1\} \tag{5.5}$$

The transitional semantics of qCCS is hence given by the following pLTS:

$$\langle \mathcal{CON}, Act_{qCCS}, \rightarrow_{qCCS} \rangle, \tag{5.6}$$

with $\rightarrow \subseteq \mathcal{CON} \times Act \times \mathcal{D}(\mathcal{CON})$ is the smallest relation satisfying the following inference rules:

- **Classical channel rules**:

$$\mathit{Inp}_C \quad \frac{v \in \mathsf{Real}}{\langle c?x.P, \rho\rangle \xrightarrow{c?v} \langle P\{v/x\}, \rho\rangle}$$

$$\mathit{Out}_C \quad \frac{v \in \mathsf{Real}}{\langle c!e.P, \rho\rangle \xrightarrow{c!v} \langle P, \rho\rangle}$$

$$\mathit{Com}_C \quad \frac{\langle P_1, \rho\rangle \xrightarrow{c?v} \langle P_1', \rho\rangle, \langle P_2, \rho\rangle \xrightarrow{c!v} \langle P_2', \rho\rangle}{\langle P_1 \| P_2, \rho\rangle \xrightarrow{\tau} \langle P_1' \| P_2', \rho\rangle}$$

The three rules just defined describe, like the in the standard description of CCS, the action of communication using a classical variable. The value v in the rule Out_C represents the result of the evaluation of e. Note also that the rules do not modify in any way the context ρ because they only deal with classical variables.

- **Quantum channel rules**:

$$\mathit{Inp}_Q \quad \frac{r \notin qv(\mathrm{c}?q.P)}{\langle \mathrm{c}?q.P, \rho\rangle \xrightarrow{\mathrm{c}?r} \langle P\{r/q\}, \rho\rangle}$$

$$\mathit{Out}_Q \quad \frac{}{\langle \mathrm{c}!q.P, \rho\rangle \xrightarrow{\mathrm{c}!q} \langle P, \rho\rangle}$$

$$\mathit{Com}_Q \quad \frac{\langle P_1, \rho\rangle \xrightarrow{\mathrm{c}?r} \langle P_1', \rho\rangle \quad \langle P_2, \rho\rangle \xrightarrow{\mathrm{c}!r} \langle P_2', \rho\rangle}{\langle P_1 \| P_2, \rho\rangle \xrightarrow{\tau} \langle P_1' \| P_2', \rho\rangle}$$

The rules above in this case describe communication involving quantum variables. It is important to note that also in this case, the rules do not modify the context ρ. At first sight this appears to be contradictory with the input rule, but the intuition behind it is that it is intended to describe the input of a qubit from inside the context, and when the qubit is already described in it, the input action is only a declaration.

- **Quantum rules**:

$$Op\ \frac{}{\langle \Psi[\tilde{r}].P, \rho\rangle \xrightarrow{\tau} \langle P, \Psi_{\tilde{r}}(\rho)\rangle}$$

$$Meas\ \frac{M = \sum_{i\in I} c_i E^i, p_i = \operatorname{tr}\left(E^i_{\tilde{r}}\rho\right) > 0}{\langle M[\tilde{r}; x].P, \rho\rangle \xrightarrow{\tau} \sum_{i\in I} p_i \left\langle P\left\{c_i/x\right\}, E^i_{\tilde{r}}\rho E^i_{\tilde{r}}/p_i\right\rangle}$$

The rules *Meas* and *Op* describe respectively the operations of the measurement and the application of a quantum operator on a sequence of qubits $\tilde{r}$. As can be seen, the act of measuring or applying the quantum operator are modelled using τ-actions, which are by definition invisible from the outside.

- **Distribution rule**:

$$Dist\ \frac{}{\left\langle \sum_{i\in I} p_i P_i, \rho\right\rangle \xrightarrow{\tau} \sum_{i\in I} p_i \left\langle P_i, \rho\right\rangle}$$

The distribution rule is an extension of qCCS proposed in [17] to allow the probabilistic choice directly at the syntax level, which will be of crucial in the following chapter.

- **Standard rules**:

$$\textit{Sum}\ \frac{\langle P,\rho\rangle \xrightarrow{\alpha} \lambda}{\langle P+Q,\rho\rangle \xrightarrow{\alpha} \lambda} \qquad \textit{Par}\ \frac{\langle P_1,\rho\rangle \xrightarrow{\alpha} \lambda, qbv(\alpha)\cap qv\,(P_2)=\emptyset}{\langle P_1\|P_2,\rho\rangle \xrightarrow{\alpha} \lambda\|P_2}$$

$$\textit{Cho}_1\ \frac{\langle P,\rho\rangle \xrightarrow{\alpha} \lambda, [\![b]\!]=\text{true}}{\langle \text{ if } b \text{ then } P,\rho\rangle \xrightarrow{\alpha} \lambda} \qquad \textit{Cho}_2\ \frac{\langle P,\rho\rangle \xrightarrow{\alpha} \lambda, [\![b]\!]=\text{false}}{\langle \text{ if } b \text{ then } P,\rho\rangle \xrightarrow{\alpha} \mathbf{nil}}$$

$$\textit{Rel}\ \frac{\langle P,\rho\rangle \xrightarrow{\alpha} \lambda}{\langle P[f],\rho\rangle \xrightarrow{f(\alpha)} \lambda[f]} \qquad \textit{Res}\ \frac{\langle P,\rho\rangle \xrightarrow{\alpha} \lambda, cn(\alpha)\cap L=\emptyset}{\langle P\backslash L,\rho\rangle \xrightarrow{\alpha} \lambda\backslash L}$$

$$\textit{Def}\ \frac{\langle P\{\tilde{v}/\tilde{x},\tilde{r}/\tilde{q}\},\rho\rangle \xrightarrow{\alpha} \lambda, A(\tilde{x},\tilde{q}):=P, \tilde{v}=[\![\tilde{e}]\!]}{\langle A(\tilde{e},\tilde{r}),\rho\rangle \xrightarrow{\alpha} \lambda}$$

For clarity, in the premise of *Cho₁* and *Cho₂*, the term $[\![b]\!] = (\text{true/false})$ means that the evaluation of the expression $b \in bExp$ leads to the result of (true/false); Furthermore the function $qbv(\alpha)$ is used to represent the bounded quantum variables of α, i.e. $qbv(\text{c}.q) = \{q\}$, for $\alpha \neq \text{c}.q$, $qbv(\alpha) = \emptyset$. Lastly, the symmetric form of *Par* and *Sum* has been omitted.

Example 5.1.2. Now, with a complete vision of qCCS, we can describe the system in Example 5.1.1 more accurately. Given the same process

$$P = H[q_0].Cnot[q_0,q_1].M[q_0;x_0].M[q_1;x_1].out!(x_0,x_1).\mathbf{nil},$$

we can associate to P a density operator $\rho = |00\rangle\langle 00|$ to form the configuration $B = \langle P,\rho\rangle$. This configuration can be viewed in Figure 5.1.

With the subscript 0 or 1 in B_0, B_1 we mean that the first measurement lead to a result of 0 or 1. In the same way, the second digit in the subscript of $B_{0,0}, B_{1,1}$ correspond to the second measurement operation on q_1. Note that all the quantum operation and measurement are treated like τ-actions.

5.2 Quantum Bisimulation Equivalences

In Chapter 4 we have presented various notions of weak and strong bisimulation, starting from nondeterministic LTSs and continuing to pLTSs. Having

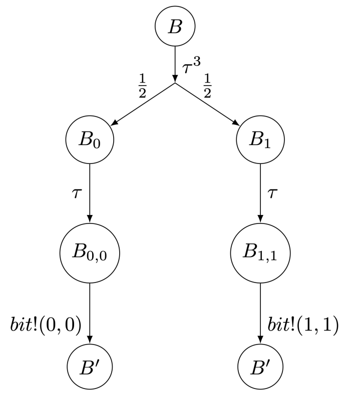


Figure 5.1: pLTS for EPR-pair and measuring

seen in Equation 5.6 that pLTSs, when acting on configurations, can be used to represent quantum systems, in this section we investigate the various bisimulation notions based on it. Note that all the notions proposed are intended to be *weak bisimulations*.

5.2.1 State-Based Bisimulation

The notion of bisimulation, when it is applied thinking about the state space of the system associated to a determined configuration, leads us to the first equivalence of this section, the so called *state-based bisimulation*. We call this type of bisimulation state-based, as in [17], because it only compares *individual configurations* and not combinations of them. Let us prepare the scene, assume $B = \langle P, \rho \rangle$ is a configuration, we refer to $qv(P)$ as $qv(B)$, and $env(B) = tr_{qv(P)}(\rho)$ as the quantum environment of the process P in B.

Definition 5.2.1. Given a pLTSs $\langle \mathcal{CON}, Act, \rightarrow \rangle$ and two configurations $B, C \in \mathcal{CON}$, a binary relation $\mathcal{R} \subseteq \mathcal{CON} \times \mathcal{CON}$ is a **state-based ground bisimulation** if $(B, C) \in \mathcal{R}$ implies, for all $\alpha \in Act$:

- $qv(B) = qv(C)$, $env(B) = env(C)$.
- For each $B \xrightarrow{\alpha} \lambda$ then $C \xRightarrow{\hat{\alpha}} \gamma$, for some γ such that $(\lambda, \gamma) \in \mathcal{R}$.

- For each $C \xrightarrow{\alpha} \gamma$ then $B \overset{\hat{\alpha}}{\Longrightarrow} \lambda$, for some λ such that $(\lambda, \gamma) \in \mathcal{R}$.

The next definition is used to characterize the *closure under super-operators*.

Definition 5.2.2. A relation $\mathcal{R} \subseteq \mathcal{CON} \times \mathcal{CON}$ is said to be **closed under super-operators** if, $(B, C) \in \mathcal{R}$ implies $(\Phi(B), \Phi(C)) \in \mathcal{R}$ for any $\Phi \in \mathcal{SO}(\mathcal{H}_{\overline{qv(B) \cup qv(C)}})$.

Combining the definitions of state-ground bisimulation and closure under super-operators, we arrive at the main definition of the section.

Definition 5.2.3. A binary relation $\mathcal{R} \subseteq \mathcal{CON} \times \mathcal{CON}$ is a **state-based bisimulation** if it is closed under super-operators and it is a state-based ground bisimulation.

Thus, we can define when two configurations are state-based bisimilar.

Definition 5.2.4. Given a pLTSs $\langle \mathcal{CON}, Act, \rightarrow \rangle$ and two quantum configurations $B, C \in \mathcal{CON}$, B and C are **state-based bisimilar**, written $B \approx_s C$ if there exists a state-based bisimulation $\mathcal{R}$ containing (B, C).

Note that the definition of state-based bisimilarity can be easily lifted to quantum processes.

Definition 5.2.5. Two quantum process terms P and Q are **state-based bisimilar**, written $P \approx_s Q$ if for any evaluation ψ and for any quantum state $\rho \in \mathcal{D}(\mathcal{H})$, $\langle P\psi, \rho \rangle \approx_s \langle Q\psi, \rho \rangle$.

Example 5.2.6. For a better comprehension of state-based bisimilarity, we make a simple example of two state-based bisimilar qCCS configurations. Given $L_{qCCS} = \langle \mathcal{CON}, Act_{qCCS}, \rightarrow_{qCCS} \rangle$, let us define two processes P, Q:

$$P = \text{in}?q.M[q;x].out!x.\mathbf{nil} \backslash \{\text{in}\} \qquad Q = H[q].M[q;x].out!x.\mathbf{nil}$$

Then, given a density operator ρ, we can define the following two state-based bisimilar configurations $B, C \in \mathcal{CON}$:

$$B = \langle P, |+\rangle_q \langle +| \otimes \rho \rangle \qquad C = \langle Q, |0\rangle_q \langle 0| \otimes \rho \rangle$$

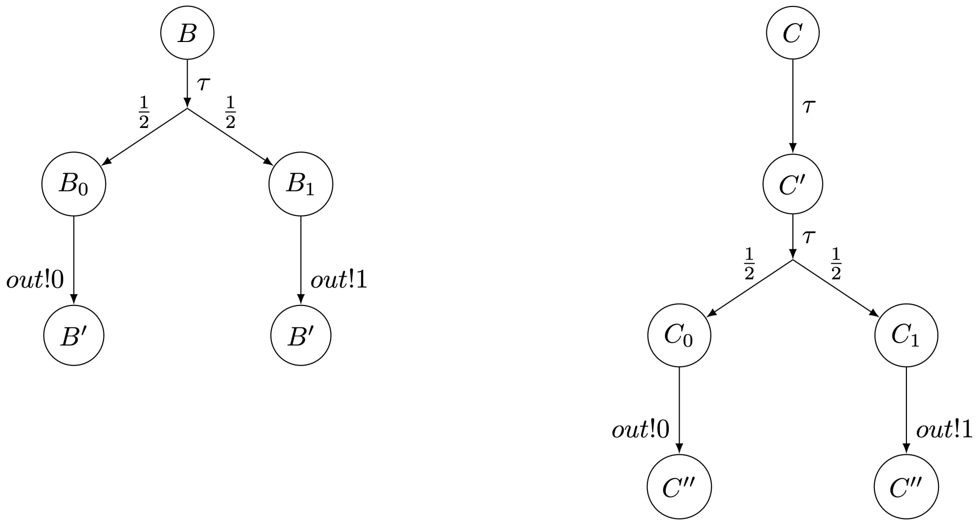


Figure 5.2: The pLTSs representing two state-based bisimilar configurations

The pLTSs that describe B and C are represented in Figure 5.2.

Given this, we can proceed by giving a proposition that further characterize the notion of state-based bisimulation.

Proposition 5.2.7. *Given $\lambda, \gamma \in \mathcal{D}(\mathcal{CON})$ and supposing that $\lambda \approx_s \gamma$ and $\lambda \overset{\hat{\alpha}}{\Longrightarrow} \lambda'$. Then there exists some γ' such that $\gamma \overset{\hat{\alpha}}{\Longrightarrow} \gamma'$ and $\lambda' \approx_s \gamma'$*

Proof. By Lemmas 4.2.4, 4.2.6, 4.2.9, the following propriety holds:

($*$) If $\lambda \approx_s \gamma$ and $\lambda \overset{\hat{\tau}}{\Longrightarrow} \lambda'$ then there is some γ' with $\gamma \overset{\hat{\tau}}{\Longrightarrow} \gamma'$ and $\lambda' \approx_s \gamma'$

Suppose that $\lambda \overset{\alpha}{\Longrightarrow} \lambda'$ and $\lambda \approx_s \gamma$. If α is τ then γ' follows by the application of propriety (*). Differently, by definition we know $\lambda \overset{\hat{\tau}}{\Longrightarrow} \lambda_1$, $\lambda_1 \overset{\alpha}{\longrightarrow} \lambda_2$ and $\lambda_2 \overset{\hat{\tau}}{\Longrightarrow} \lambda'$. The application of propriety (*) gives us a γ_1 such that $\gamma \overset{\hat{\tau}}{\Longrightarrow} \gamma_1$ and $\lambda_1 \approx_s \gamma_1$, the application of Lemma 4.2.6 gives us a γ_2 such that $\gamma_1 \overset{\hat{\alpha}}{\Longrightarrow} \gamma_2$ and $\lambda_2 \approx_s \gamma_2$. Lastly, a further application of propriety (*) gives us $\gamma_2 \overset{\hat{\tau}}{\Longrightarrow} \gamma'$, such that $\lambda' \approx_s \gamma'$. The result follows from the transitivity of $\overset{\hat{\tau}}{\Longrightarrow}$. □

Lastly, we prove that $\approx_s$ is an equivalence relation.

Theorem 5.2.8. *For any pLTSs $\langle \mathcal{CON}, Act, \rightarrow \rangle$:*

1. *The relation $\approx_s$ is the largest state-based bisimulation on $\mathcal{CON}$.*
2. *The relation $\approx_s$ is an equivalence relation.*
3. *The lifted relation $\approx_s \subset \mathcal{D}(\mathcal{CON}) \times \mathcal{D}(\mathcal{CON})$ is both linear and left-decomposable.*

Proof.

1. $\approx_s$ it is certainly a state-based bisimulation, it remains to be proven the fact that it is the largest:
   Definition 5.2.4 may be equivalently expressed as follows:
   $$\approx_s = \bigcup \{\mathcal{R} : \mathcal{R} \text{ is a state-based bisimulation}\}.$$
   That means that $\approx_s$ includes any state-based bisimulation, therefore is the largest one.
2. *Reflexivity* and *symmetry* of $\approx_s$ are trivial and will not be discussed. Coming to *transitivity*, we only need to show that $\approx_s \cdot \approx_s$ is a state based bisimulation. Suppose $B_1 \approx_s B_2$ and $B_2 \approx_s B_3$, then if $B_1 \xrightarrow{\alpha} \lambda_1$ there exists some $B_2 \overset{\hat{\alpha}}{\Longrightarrow} \lambda_2$ such that $\lambda_1 \approx_s \lambda_2$, since $B_1 \approx_s B_2$. From $B_2 \approx_s B_3$ and Proposition 5.2.7, it follows that $B_3 \overset{\hat{\alpha}}{\Longrightarrow} \lambda_3$ and $\lambda_2 \approx_s \lambda_3$. Therefore, we see that $\lambda_1(\approx_s \cdot \approx_s)\lambda_3$.
3. $\approx_s \subset \mathcal{D}(\mathcal{CON}) \times \mathcal{D}(\mathcal{CON})$ it is trivially linear by Definition 4.2.3 and it is left-decomposable by Lemma 4.2.6.

□

5.2.2 Distribution-Based Bisimulation

As stated in [15, 17], state-based bisimulation is sometimes too discriminating when it is applied to probabilistic automata. When pLTSs and distributions of quantum configurations are considered, state-based bisimulations are predictably also too discriminating, in the sense that it can distinguish distributions which will never be distinguished by any outside observer. This leads to a characterization of an intruder with unrealistic power. Furthermore, in this setting, the quantum states "accompanying" the processes are very likely to be combined when interacting with each other, as pointed out in the following example, taken by [17].

Example 5.2.9. Given a density operator ρ, a super-operator Ψ which carries out a measurement operation, and a two-outcome measurement $O = c_0 |0\rangle \langle 0| + c_1 |1\rangle \langle 1|$, we can declare two configurations B and C as $B := \langle \Psi[q].\mathbf{nil}, |+\rangle_q \langle +| \otimes \rho \rangle$ and $C := \langle M[q;x].\mathbf{nil}, |+\rangle_q \langle +| \otimes \rho \rangle$ with $|+\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$. Since the two configurations are both performing a measurement and ignoring the output, we definitely want $B \approx_s C$. However, this is not true. Let $B_0 = \langle \mathbf{nil}, |0\rangle_q \langle 0| \otimes \rho \rangle$, $B_1 = \langle \mathbf{nil}, |1\rangle_q \langle 1| \otimes \rho \rangle$, $B_I = \langle \mathbf{nil}, I_q/2 \otimes \rho \rangle$ and $\lambda = \frac{1}{2}B_0 + \frac{1}{2}B_1$, then $B_I \not\approx_s \lambda$ because otherwise, using the left-decompositivity of $\approx_s$, we would need that $B_I \approx_s B_0, B_I \approx_s B_1$ which is obviously false.

The evidence in Example 5.2.9, applies to **every** state-based bisimulation because of the result of Lemma 4.2.6. The rest of this section proposes definitions and theorems directly on distributions, in order to make indistinguishable configurations, like those in the previous example, bisimilar.

As usual, let us begin with some basic definitions.

Definition 5.2.10. Given a pLTS $\langle \mathcal{CON}, Act, \rightarrow \rangle$, a distribution $\lambda \in \mathcal{D}(\mathcal{CON})$ is **transition consistent** if for any λ' in his support and for $\alpha \in Act$, with $\alpha \neq \tau$, $\lambda' \overset{\hat{\alpha}}{\Longrightarrow} \gamma'$ for some γ' implies $\lambda \overset{\hat{\alpha}}{\Longrightarrow} \gamma$ for some γ.

Definition 5.2.11. A decomposition $\lambda = \sum_{i \in I} p_i \cdot \lambda_i$ with $p_i > 0$, for each $i \in I$, is called a **tc-decomposition** of λ if, for each $i \in I$, λ_i is transition consistent.

Having given the context, we can proceed and give the definition of a notion of bisimulation directly on distributions, called *distribution-based bisimulation*, and some related theorems. Note that for a distribution $\lambda = \sum_i p_i \lambda_i$, for each i, $qv(\lambda) = \bigcup_i qv(\lambda_i)$, $env(\lambda) = \sum_i p_i \cdot env(\lambda_i)$ and $\Psi(\lambda) = \sum_i p_i \Psi(\lambda_i)$.

Definition 5.2.12. Given a pLTS $\langle \mathcal{CON}, Act, \rightarrow \rangle$ and two distributions $\lambda, \gamma \in \mathcal{D}(\mathcal{CON})$, a binary relation $\mathcal{R} \subseteq \mathcal{D}(\mathcal{CON}) \times \mathcal{D}(\mathcal{CON})$ is called a **distribution-based ground bisimulation**, if $(\lambda, \gamma) \in \mathcal{R}$ implies, for all $\alpha \in Act$:

1. $qv(\lambda) = qv(\gamma)$, $env(\lambda) = env(\gamma)$.
2. For each $\lambda \xrightarrow{\hat{\alpha}} \lambda'$, then $\gamma \overset{\hat{\alpha}}{\Rightarrow} \gamma'$, for some γ' such that and $(\lambda', \gamma') \in \mathcal{R}$.
3. For each $\gamma \xrightarrow{\hat{\alpha}} \gamma'$, then $\lambda \overset{\hat{\alpha}}{\Rightarrow} \lambda'$, for some λ' such that and $(\lambda', \gamma') \in \mathcal{R}$.
4. if λ is not transition consistent and $\lambda = \sum_{i \in I} p_i \cdot \lambda_i$ is a tc-decomposition, then there exists a transition $\gamma \overset{\hat{\tau}}{\Rightarrow} \sum_{i \in I} p_i \cdot \gamma_i$ such that for each i, $(\lambda_i, \gamma_i) \in \mathcal{R}$.

It may be interesting to note that, in contrast with Definition 5.2.1, when we move to distributions we have a further clause. The intuition here comes from the fact that the transition described in 5.2.12(2) is possible **only** when λ is transition consistent. This means that all the configurations contained in the support of the starting distribution, λ in the definition, can perform *weak* α-transitions because of the splitting into transition consistent components stated in 5.2.12(4).

As in the previous section, combining the definition of closure under super-operators and Definition 5.2.12 we arrive at the definition of *distribution-based bisimulation.*

Definition 5.2.13. A binary relation $\mathcal{R} \subseteq \mathcal{D}(\mathcal{CON}) \times \mathcal{D}(\mathcal{CON})$ is a **distribution-based bisimulation** if it is a distribution-based ground bisimulation and it is also closed under super-operators.

Then we proceed, as usual, to define when distributions on configurations are distribution-based bisimilar.

Definition 5.2.14. Given a pLTSs $\langle \mathcal{CON}, Act, \rightarrow\rangle$ and two quantum distributions $\lambda, \gamma \in \mathcal{D}(\mathcal{CON})$, λ and γ are **state-based bisimilar**, overloading the notation, written $\lambda \approx \gamma$, if there exists a distribution-based bisimulation $\mathcal{R}$ containing (λ, γ).

Lastly, we conclude the section with the following two theorems.

Theorem 5.2.15. *For any pLTS $\langle \mathcal{CON}, Act, \rightarrow\rangle$:*

1. *The relation $\approx$ is the largest bisimulation on $\mathcal{D}(\mathcal{CON})$.*
2. *The relation $\approx$ is an equivalence relation.*

Proof. The proof for (1) and (2) follows exactly the proof of (1) and (2) of Theorem 5.2.8. □

Theorem 5.2.16. *Given $\lambda, \gamma \in \mathcal{D}(\mathcal{CON})$, thus $\lambda \approx_s \gamma$ implies $\lambda \approx \gamma$, but $\lambda \approx \gamma$ does not imply $\lambda \approx_s \gamma$.*

Proof. Just observing the definitions of state-based and distribution-based ground bisimulation, it is easy to note that the three clauses in Definition 5.2.3 correspond to the first three clauses in Definition 5.2.13, so the former implication is trivially proven. For the latter we need a counterexample: Example 5.2.9 shows two configurations, B and C, that are distribution-based bisimilar but not state-based bisimilar. Recalling that a configuration can be expressed as a point distribution then we have proven the non-implication. □

Theorem 5.2.16 only states that the definition of distribution-based bisimulation is coarser then its state-based counterpart.

5.2.3 Bisimulation Metrics

All the bisimulation relations defined in the previous section and chapters represent *exact* bisimulations. That is, they distinguish two processes as bisimilar or not bisimilar. To close the gap between these dual statements and capture the idea that a process *approximately* implements its specification, in [14], the notion of *approximate bisimulation* is proposed in the context of pLTSs. This idea has been used successfully in [49] in the context of quantum processes, and finally it has been extended in [17] to a distribution-based setting.

Let us start as always by giving some basic definitions.

Definition 5.2.17. A **metric** on a set $\mathcal{X}$ is a function $d : \mathcal{X} \times \mathcal{X} \to [0, \infty)$, where $[0, \infty)$ is the set of non-negative real numbers, which satisfies:

1. *Triangle inequality*: $d(x, y) \leq d(x, z) + d(z, y)$, for any z.
2. *Symmetry*: $d(x, y) = d(y, x)$.
3. *Identity of indiscernibles*: $d(x, y) = 0 \Leftrightarrow x = y$.

Definition 5.2.18. A **pseudometric** on a set $\mathcal{X}$ is a function $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that satisfies:

1. *Triangle inequality*: $d(x, y) \leq d(x, z) + d(z, y)$, for any z.
2. *Symmetry*: $d(x, y) = d(y, x)$.
3. $d(x, x) = 0$

Note that the difference between a metric d and a pseudometric d' is that if $d(x, y) = 0$ then $x = y$, whereas, if $d'(x, y) = 0$ there can exist some x, y such that $x \neq y$. Having this, we can go directly to the notion of approximate bisimulation.

Definition 5.2.19. Given a pLTS $\langle \mathcal{CON}, Act, \to \rangle$, two distributions $\lambda, \gamma \in \mathcal{D}(\mathcal{CON})$ and $\mu \in [0, 1]$, a binary relation $\mathcal{R} \subseteq \mathcal{D}(\mathcal{CON}) \times \mathcal{D}(\mathcal{CON})$ closed under super-operators, is called **μ-bisimulation**, if $(\lambda, \gamma) \in \mathcal{R}$ implies, for all $\alpha \in Act$:

1. $qv(\lambda) = qv(\gamma)$, $d(env(\lambda), env(\gamma)) \leq \mu$.

2. For each $\lambda \xrightarrow{\hat{\alpha}} \lambda'$, then $\gamma \overset{\hat{\alpha}}{\Longrightarrow} \gamma'$, for some γ' such that and $(\lambda', \gamma') \in \mathcal{R}$.
3. For each $\gamma \xrightarrow{\hat{\alpha}} \gamma'$, then $\lambda \overset{\hat{\alpha}}{\Longrightarrow} \lambda'$, for some λ' such that and $(\lambda', \gamma') \in \mathcal{R}$.
4. if λ is not transition consistent and $\lambda = \sum_{i \in I} p_i \cdot \lambda_i$ is a tc-decomposition, then exist a transition $\gamma \overset{\hat{\tau}}{\Longrightarrow} \sum_{i \in I} p_i \cdot \gamma_i$ such that, for each i such that $(\lambda_i, \gamma_i) \in \mathcal{R}$, $\sum_i p_i \geq 1 - \mu$.

A pair of observations can be done: first, with $d(env(\lambda), env(\gamma))$ in Definition 5.2.19(1) we intend the *trace distance* between the two environments. Then, as can be easily seen, Definition 5.2.19 follows directly from Definition 5.2.13, with the differences given by the trace distance.

As usual, we define when two quantum distributions are stated to be μ-bisimilar.

Definition 5.2.20. Given a pLTSs $\langle \mathcal{CON}, Act, \rightarrow \rangle$ and two quantum distributions $\lambda, \gamma \in \mathcal{D}(\mathcal{CON})$, λ and γ are μ-**bisimilar**, written $\lambda \overset{\mu}{\approx} \gamma$ if there exists a μ-bisimulation $\mathcal{R}$ containing (λ, γ).

Moreover, using the previous definitions, we can introduce the concept of *bisimulation distance.*

Definition 5.2.21. Given a pLTSs $\langle \mathcal{CON}, Act, \rightarrow \rangle$ and two distributions $\lambda, \gamma \in \mathcal{D}(\mathcal{CON})$, the **bisimulation distance** between distributions is a function $db : \mathcal{D}(\mathcal{CON}) \times \mathcal{D}(\mathcal{CON}) \rightarrow]0, 1]$, defined as follows:

$$db(\lambda, \gamma) = inf\,\{\mu \geq 0 \,|\, \lambda \overset{\mu}{\approx} \gamma\}.$$

Note that the bisimulation distance function can be defined also on processes.

Definition 5.2.22. Given a pLTSs $\langle \mathcal{CON}, Act, \rightarrow \rangle$ and two configurations $B = \langle P, \rho \rangle$, $C = \langle Q, \rho \rangle \in \mathcal{CON}$, the **bisimulation distance** between processes is a function $db : qProc \times qProc \rightarrow]0, 1]$ defined as follows:

$$db(P, Q) = inf\{\mu \geq 0 \,|\, \forall \psi,\ \rho \in \mathcal{D}(\mathcal{H}),\ \langle P\psi, \rho \rangle \overset{\mu}{\approx} \langle Q\psi, \rho \rangle\}.$$

Then, the next theorem shows that the bisimulation distance is a pseudo-metric on $\mathcal{D}(\mathcal{CON})$.

Theorem 5.2.23. *The bisimulation distance d_b is a pseudo-metric on $\mathcal{D}(\mathcal{CON})$.*

Proof. We need to prove that the bisimulation distance satisfies the clauses of Definition 5.2.18. Note that the bisimulation distance *db* between two distributions is the smallest μ for which exists a μ-bisimulation containing (λ, γ). Furthermore, given a μ-bisimulation containing (λ, γ), μ is the upper-bound of the trace distance between $env(\lambda)$ and $env(\gamma)$ that is a metric on Hilbert space. Therefore, given the close link between the trace distance and the bisimulation distance, the latter is clearly a pseudometric. Lastly, note that d_b is not a metric on $\mathcal{D}(\mathcal{CON})$ because there exists infinite number of configurations with the same *env*. $\square$

Example 5.2.24. As we have already said, the notion of approximate bisimulation can be used to capture the idea that a process approximately implements its specification. To give an example of two approximately bisimilar configurations, given $L_{qCCS} = \langle \mathcal{CON}, Act_{qCCS}, \rightarrow_{qCCS} \rangle$, let us define two processes $P_{test}(n), P_{spec}(n)$ as follows:

$$P_{test}(n) = H[\tilde{q}].M[\tilde{q}; \tilde{x}].(\textbf{if } eq_1(x) \textbf{ then } out!x.\textbf{nil} \textbf{ else } fail!0.\textbf{nil})$$
$$P_{spec}(n) = H[\tilde{q}].M[\tilde{q}; \tilde{x}].out!x.\textbf{nil}$$

Here, the function symbol eq_1 returns *true* if every bit of the input string is equal to 1, while, the parameter n is used to define the length of the string $\tilde{q}$. Then, given a density operator ρ, we can define the configurations $B, C \in \mathcal{D}(\mathcal{CON})$:

$$B = \langle P_{test}(n), \rho \rangle \qquad C = \langle P_{spec}(n), \rho \rangle$$

Here, we have $B \overset{\mu}{\approx} C$ with $\mu = \frac{1}{2^n}$. Note that this is true because the only visible action that can lead to an inconsistent transition between B and C is the output action *fail!0*, which is performed with probability $\frac{1}{2^n}$.

Lastly, we conclude the section and the chapter with a theorem that shows how the approximate bisimulation relates to the distribution-based bisimulation.

Theorem 5.2.25. *For any* $\lambda, \gamma \in \mathcal{D}(\mathcal{CON})$*,* $\lambda \approx \gamma$ *iff* $d_b(\lambda, \gamma) = 0$

Proof. If $d_b(\lambda, \gamma) = 0$ then there exists a μ-bisimulation with $\mu = 0$ containing (λ, γ). Recalling that $\lambda \approx \gamma$ if there is a distribution-state bisimulation $\mathcal{R}$ containing (λ, γ) and observing the definitions of μ-bisimulation and distribution-state bisimulation, note that the only difference resides in the respective clauses (2), in which in Definition 5.2.12 we have $env(\lambda) = env(\lambda)$, while in Definition 5.2.19 we have $d(\lambda, \gamma) \leq \mu$ with d as the trace distance. Being $\mu = 0$ then $d(\lambda, \gamma) = 0$ and this implies that $env(\lambda) = env(\gamma)$. $\square$

Chapter 6

Cryptographic Proofs in qCCS

In this chapter we use the previously introduced notions of bisimulation on quantum systems to verify the correctness and the security degree of the two bit commitment schemes presented in Chapter 2: the BB84 quantum bit commitment and the Kent relativistic bit commitment. Note that we investigate different notions of security, based on the intruder capabilities.

6.1 BB84 Quantum Bit Commitment

For the analysis of the BB84 quantum bit commitment scheme, we represent the protocol as a quantum process in which the two co-protagonists, Alice and Bob, are seen as parallel quantum processes. We follow the method in [17] starting by "breaking" the protocol into two phases: in the former, we abstract the security of the protocol, analyzing only its correctness, in the latter, instead, we analyze the security degree, modifying Alice's process to become the intruder and using the notion of μ-bisimulation to analyze the *asymptotic security* of the protocol. Note that the variable of the possession of a quantum computer exposed at the end of Section 2.3 allows us to define two different types of intruders.

6.1.1 Correctness

Following [17], the protocol is initially described without the privacy amplification phase, and **only** its correctness is analyzed.

As already said, we model the BB84 quantum bit commitment protocol as a quantum process, called *QBC84*(n), with n as the security parameter. The process is designed to simply run in parallel two quantum processes that impersonate Alice and Bob. As in [17], we design the protocol so that returns a substring of Bob's bitstring that matches with Alice's base, that is up to immediately before step 6 of Protocol 2. We do this in order to later put BB84 in a test environment. Let us describe *QBC84*(n):

$$\begin{aligned}
\mathit{Alice}(n) &:= \sum_{b_a \in \{0,1\}} \sum_{\tilde{K}_a \in \{0,1\}^n} \frac{1}{2^{n+1}} \,\mathrm{Set}_{\tilde{K}_a}[\tilde{q}].H_{b_a}[\tilde{q}].\mathrm{A2B}!\tilde{q}.\,\mathit{Unveil}_A(b_a, \tilde{K}_a)\\
\mathit{Unveil}_A(b_a, \tilde{K}_a) &:= b2a?u.a2b!(b_a, \tilde{K}_a).\mathbf{nil}\\
\mathit{Bob}(n) &:= \mathrm{A2B}?\tilde{q}. \sum_{\tilde{B}_b \in \{0,1\}^n} \frac{1}{2^n} M_{\tilde{B}_b}[\tilde{q}; \tilde{K}_b].\,\mathrm{Set}_{\tilde{0}}[\tilde{q}].b2a!1.\,\mathit{Unveil}_B(\tilde{B}_b, \tilde{K}_b)\\
\mathit{Unveil}_B(\tilde{B}_b, \tilde{K}_b) &:= a2b?(b_a, \tilde{K}_a).out!(\mathrm{cmp}(\tilde{K}_a, b_a, \tilde{B}_b), \mathrm{cmp}(\tilde{K}_b, b_a, \tilde{B}_b), b_a).\mathbf{nil}\\
\mathit{QBC84}(n) &:= \mathit{Alice}(n) \| \mathit{Bob}(n)
\end{aligned} \tag{6.1}$$

The extensive use of the probabilistic choice is clearly visible directly at syntax level. For example, we design the choice of Alice's commitment b_a as a probabilistic process, described by the first use of the probabilistic choice operator. The second probabilistic choice, instead, describes the choice of the random bitstring $\tilde{K}_a$. The fraction $\frac{1}{2^{n+1}}$ represents the probability of each distribution, and it is the combination of the two previous probabilities. Continuing our analysis of Alice's process, the operator $\mathrm{Set}_{\tilde{K}_a}$ is used to prepare the string of qubits $\tilde{q}$, setting the i-th qubit to the state $|K_a\rangle_i$. The operator $H_{b_a}[\tilde{q}]$ instead applies the Hadamard gate H on $\tilde{q}$ if $b_a = 1$, otherwise it does nothing. Moving to the analysis of the process that models Bob, also in this case we use a probabilistic choice to model the random bitstring of bases $\tilde{B}_b$ generated by him, together with the associated probability. The operator $M_{\tilde{B}_b}[\tilde{q}; \tilde{K}_b]$ performs a measurement on each of the qubits of $\tilde{q}$ using as bases the corresponding basis in $\tilde{B}_b$. Lastly, during the unveil phase, Bob sends through the classical channel *out* the substrings of $\tilde{K}_a$ and $\tilde{K}_b$ using the function symbol cmp, together with the value of the commitment b_a. Here, the function $\mathrm{cmp}(\tilde{x}, y, \tilde{z})$ returns a substring of $\tilde{x}$: for $i = 0 \dots n-1$, if the i-th bit of $\tilde{z}$ corresponds to y it appends the i-th bit of $\tilde{x}$ to the substring. For the sake of showing the correctness of the processes just described, we put *QBC84*(n) in a test environment:

$$\begin{aligned} \mathit{Test} &:= \mathit{out}?(\tilde{K}_a, \tilde{K}_b, b_a).(\mathbf{if}\ \tilde{K}_a = \tilde{K}_b\ \mathbf{then}\ \mathit{bit}!b_a.\mathbf{nil}\ \mathbf{else}\ \mathit{fail}!0.\mathbf{nil}) \\ \mathit{QBC84}_{test}(n) &:= (\mathit{QBC84}(n) \| \mathit{Test}) \setminus \{a2b, b2a, \mathrm{A2B}, \mathit{out}\} \end{aligned} \tag{6.2}$$

The process *Test* simply give in output the value of the commitment if the protocol execution was successful, otherwise it uses the channel *fail* to output 0, representing the failure of the protocol execution. We also remark that we have left the part of privacy amplification for a second part.
Having this, we need to design an *ideal specification*, with which we can compare *QBC84*$_{test}(n)$. Our specification needs to model what we want *QBC84*(n) to do ideally: first, we want our protocol to never fail, then, having modelled the choice of the commitment value as a probabilistic choice, we want it to be uniformly distributed. Now we are ready to present our ideal specification of BB84 quantum bit commitment:

$$\mathit{QBC84}_{spec}(n) := \sum_{i=0}^{n} \sum_{\tilde{x} \in \{0,1\}^i} \frac{\binom{n}{i}}{2^{n+i}} \mathrm{Set}_{\tilde{0}}[\tilde{q}].\mathit{bit}!\,\mathrm{first}(\tilde{x}).\mathbf{nil} \tag{6.3}$$

With $\mathrm{first}(\tilde{x})$ to be a function that takes a string of bits and return the first bit of the bitstring. Note that, both in the ideal specification and in the process *Bob*(n) we have used the operator $\mathrm{Set}_{\tilde{0}}[\tilde{q}]$ only for technical reasons. This way, we are able to equate the environments related to the configurations in which the processes must be placed.

We are now ready to compare the BB84 quantum bit commitment in a test environment, $QBC84_{test}(n)$, with our ideal specification, $QBC84_{spec}(n)$, using the notion of distributed-based bisimulation.

Proposition 6.1.1. *Given a density operator ρ, it holds that:*

$$\langle QBC84_{test}(n), \rho\rangle \approx \langle QBC84_{spec}(n), \rho\rangle$$

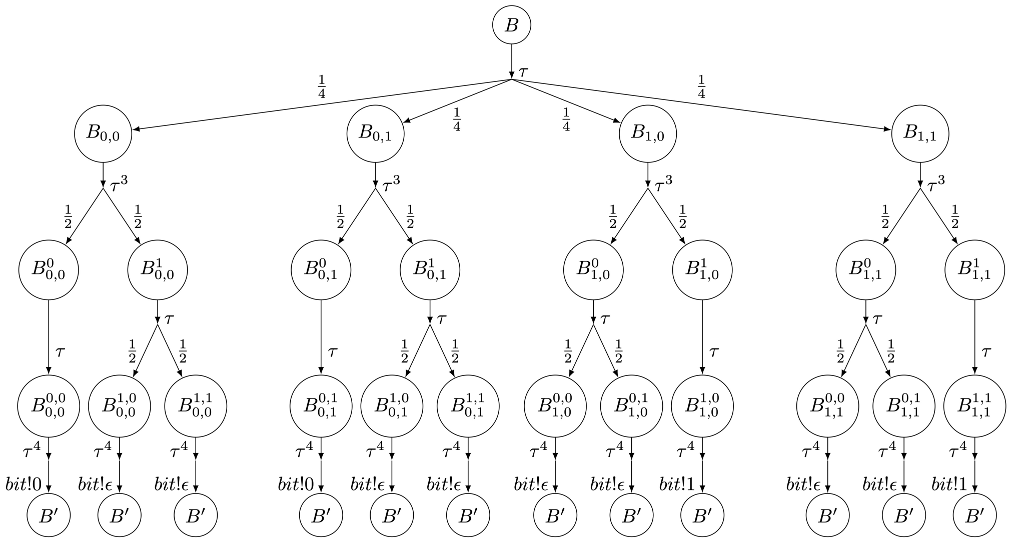

B
τ
$\frac{1}{4}$
$B_{0,0}$ $B_{0,1}$ $B_{1,0}$ $B_{1,1}$
τ^3
$\frac{1}{2}$
$B^0_{0,0}$ $B^1_{0,0}$ $B^0_{0,1}$ $B^1_{0,1}$ $B^0_{1,0}$ $B^1_{1,0}$ $B^0_{1,1}$ $B^1_{1,1}$
$B^{0,0}_{0,0}$ $B^{1,0}_{0,0}$ $B^{1,1}_{0,0}$ $B^{0,1}_{0,1}$ $B^{1,0}_{0,1}$ $B^{1,1}_{0,1}$ $B^{0,0}_{1,0}$ $B^{0,1}_{1,0}$ $B^{1,0}_{1,0}$ $B^{0,0}_{1,1}$ $B^{0,1}_{1,1}$ $B^{1,1}_{1,1}$
τ^4
$bit!0$ $bit!\epsilon$ $bit!1$
B'

Figure 6.1: pLTS representing $QBC84_{test}(1)$

Proof. The proof is very tedious, to simplify it we first analyze a simplified version in which the security parameter n is fixed to 1, and then we use the same concepts for the analysis of the protocol for every n. Figure 6.1 represents the pLTSs of $QBC84_{test}(1)$ and Figure 6.2 represents the pLTSs of $QBC84_{spec}(1)$, together with a quantum state $\rho \in \mathcal{D}(\mathcal{H})$ where $B = \langle QBC84_{test}(1), \rho\rangle$ and $C = \langle QBC84_{spec}(1), \rho\rangle$. In the figure, the subscript of the B-configurations denotes the choices for b_a and $\tilde{K}_a$, while the superscript denotes the choices for $\tilde{B}_b$ and $\tilde{K}_b$. The proof of distribution-based bisimilarity is given, starting from the bottom, as follows:

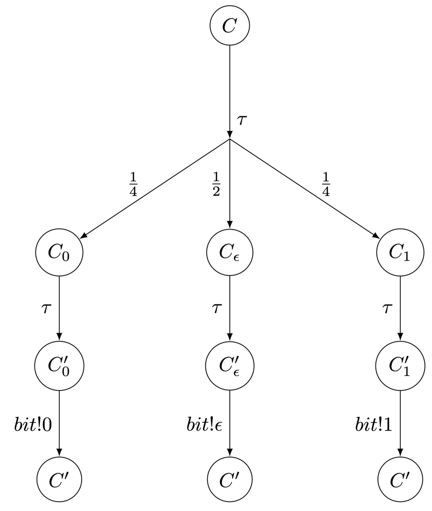


Figure 6.2: pLTS representing $QBC84_{spec}(1)$

1. B' and C' are clearly bisimilar as they are both dead, with $qv = 0$ and the accompanied quantum stares are $|0\rangle_{\tilde{q}}\langle 0| \otimes tr_q\rho$.

2. For every $i \in \{0, 1\}$, the configurations $B_{0,i}^{0,i}, B_{1,i}^{1,i}$ and C'_i are bisimilar, as they all have $qv = 0$, and the only visible action that they can perform is $bit!i$, through which they become B' and C'.

3. Almost the same as in case (2), all the configuration in $\{B_{i,j}^{k,h} \mid i, j, k, h \in 0, 1, i \neq k\}$ are bisimilar to C'_ϵ.

4. For every $i \in \{0, 1\}$, the configurations $B_{0,i}^0$, B_{1i}^1 and C_i are bisimilar, as they all have $qv = \tilde{q}$ and the only action they can perform is τ, through which they become a configuration in (2).

5. As in (4), all the configuration in $\{B_{i,j}^k \mid i, j, k \in 0, 1, i \neq k\}$ are bisimilar to C_ϵ.

6. $B_{0,0}, B_{0,1}$ are bisimilar to $(\frac{1}{2}C_0 + \frac{1}{2}C_\epsilon)$, as the all have $qv = \tilde{q}$ and they can perform action τ through which become a distribution of (4) or (5) with half probability for each one.

7. $B_{1,0}, B_{1,1}$ are bisimilar to $(\frac{1}{2}C_1 + \frac{1}{2}C_\epsilon)$ as in (6)

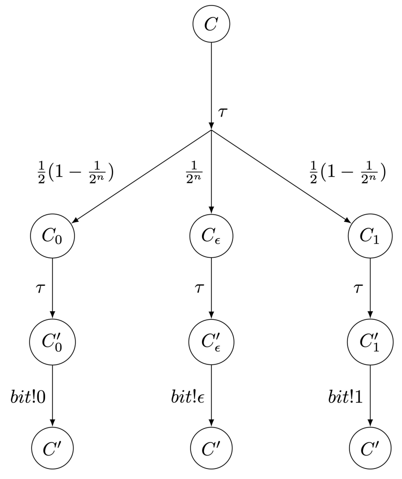


Figure 6.3: $QBC84_{spec}(n)$

8. Lastly, B is bisimilar to C as $B \xrightarrow{\tau} \sum_{i,j=0}^{1} \frac{1}{4} B_{i,j}$ while $C \xrightarrow{\tau} \frac{1}{4}C_0 + \frac{1}{2}C_\epsilon + \frac{1}{4}C_1$, and from (6) and (7), the resulting distributions are bisimilar.

For $QBC84_{test}(n)$ and $QBC84_{spec}(n)$, the proof is a little longer and more convoluted, but it follows the same methodology used in the previous case.

Figure 6.3 represents the pLTS of $QBC84_{spec}(n)$, together with a quantum state $\rho \in \mathcal{D}(\mathcal{H})$, where $C = \langle QBC84_{spec}(n), \rho \rangle$. Note that the only case in which the channel *bit* receives the empty string ϵ is when every bit of $\tilde{B}_b$ is different from b_a, and this happens $\frac{1}{2^n}$ of the times.

For $QBC84_{test}(n)$, the analysis needs to be done by parts because the full graph representing it is too large to be shown. Figures 6.4, 6.5, 6.6, 6.7, represents the parts of the pLTS of $QBC84_{test}(n)$, together with the same quantum state ρ, where $B = \langle QBC84_{test}(n), \rho \rangle$. Exactly as in the case with $n = 1$, the subscripts separated by the comma for the B-configurations denote the choices of $b_a, \tilde{K}_a$, while the superscripts denote $\tilde{B}_b, \tilde{K}_b$. In particular:

- Figure 6.4 represents the initial part of the graph. As n increases, the number of the possible equiprobable states grows exponentially.

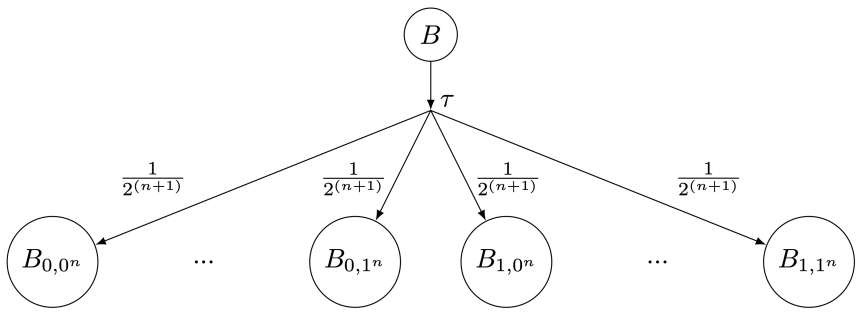


Figure 6.4: $QBC84_{test}(n)(1)$

- Figure 6.5 represents the next part of the process, for the single state $B_{0,0^n}$, and depicts the choice of Bob's string of bases. Also in this case, the number of possible equiprobable states grows exponentially with n.

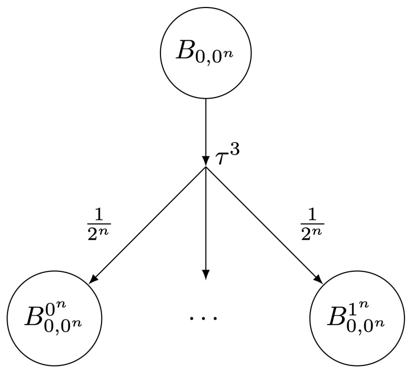


Figure 6.5: $QBC84_{test}(n)(2)$

- Figure 6.6 proceeds to the next part, taking into consideration the case in which at least one of the bases in the string chosen match with Alice's basis – in this case all of them – and this happens with probability $p = 1 - \frac{1}{2^n}$.

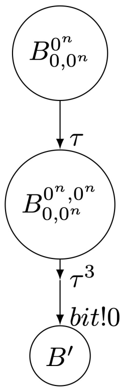


Figure 6.6: $QBC84_{test}(n)(3)$

- Lastly, Figure 6.7 shows the case in which every single bit of Bob's bitstring representing the bases does not match Alice's. In this case, the measurement returns a completely random string and the protocol returns the empty string.

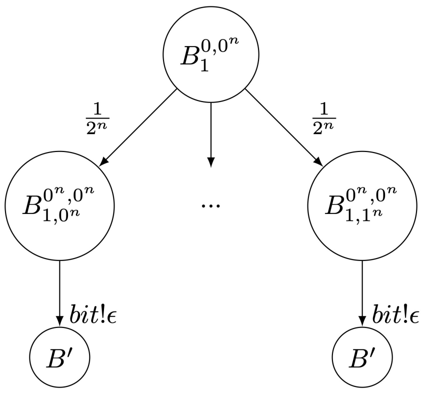


Figure 6.7: $QBC84_{test}(n)(4)$

The proof of distribution-based bisimilarity between $QBC84_{test}(n)$ and $QBC84_{spec}(n)$ is given, as always starting from the bottom:

1. Like before, B' and C' are bisimilar as they are both dead, they are both closed and the accompanied quantum states are $|0\rangle_{\tilde{q}}\langle 0| \otimes tr_q\rho$.

2. For every $i \in \{0,1\}$, $\tilde{j}, \tilde{k}, \tilde{h} \in \{0,1\}^n$, the configurations $B^{\tilde{k},\tilde{h}}_{i,\tilde{j}}$ with at least a bit of $\tilde{k}$ matching i are bisimilar to C'_i

3. As in (2), all the configurations in $\{B^{\tilde{k},\tilde{h}}_{i,\tilde{j}} \mid i \in \{0,1\}, \tilde{j}, \tilde{k}, \tilde{h} \in \{0,1\}^n, i \neq \tilde{k}\}$ are bisimilar to C'_ϵ. Note that with $i \neq \tilde{k}$ we mean that every bit of $\tilde{k}$ is different from i.

4. For every $i \in \{0,1\}$, $\tilde{j}, \tilde{k} \in \{0,1\}^n$, the configurations $B^{\tilde{k}}_{i,\tilde{j}}$ with at least a bit of $\tilde{k}$ matching with i are bisimilar to C_i as they all have $qv = \tilde{q}$ and the only action they can perform is τ, through which they become a configuration in (2).

5. As in (4), all the configurations in $\{B^{\tilde{k}}_{i,\tilde{j}} \mid i \in \{0,1\}, \tilde{j}, \tilde{k} \in \{0,1\}^n, i \neq \tilde{k}\}$ are bisimilar to C_ϵ.

6. For every $\tilde{j}$, the configurations $B_{0,\tilde{j}}$ with $\tilde{j} \in \{0,1\}^n$ are bisimilar to $\frac{1}{2^n}C_\epsilon + (1 - \frac{1}{2^n}C_0)$ as they all have $qv = \tilde{q}$ and they can perform action τ through which become a distribution of (4) or (5) with the respective probability.

7. As in (6), For every $\tilde{j}$, the configurations $B_{1,\tilde{j}}$ with $\tilde{j} \in \{0,1\}^n$ are bisimilar to $\frac{1}{2^n}C_\epsilon + (1 - \frac{1}{2^n}C_1)$

8. Lastly B is bisimilar to C as $B \xrightarrow{\tau} \sum_{i=0}^{n} \sum_{\tilde{j}=0^n}^{1^n} \frac{1}{2^{n+1}} B_{i,\tilde{j}}$, while $C \xrightarrow{\tau} \frac{1}{2^n}C_\epsilon + \frac{1}{2}(1 - \frac{1}{2^n})C_0 + \frac{1}{2}(1 - \frac{1}{2^n})C_1$, and from (6) and (7), the resulting distributions are bisimilar.

□

6.1.2 Security Against a Passive Attacker

At the end of Section 2.3, we mentioned the fact the variable deriving from the possession or non-possession of a quantum computer by Alice leads to the definition of two different types of intruders. In this subsection we investigate the notion of security considering the case in which Alice *does not have* a quantum computer, and we call her a *passive attacker*. Note that in this case we need to prove that the protocol is *secure* against Alice. As already said, Alice here becomes the intruder and her job is simply trying to commit the **dual** value of the real bit commitment. It is also important to note that in this part we add the privacy amplification phase. The modified protocol is the following:

$$\begin{aligned}
Alice(n)' &:= \sum_{b_a \in \{0,1\}} \sum_{\tilde{K}_a \in \{0,1\}^n} \frac{1}{2^{n+1}} \,\mathrm{Set}_{\tilde{K}_a}[\tilde{q}].H_{b_a}[\tilde{q}].\mathrm{A2B}!\tilde{q}.\, Unveil_A{}'(b_a, \tilde{K}_a) \\
Unveil_A{}'(b_a, \tilde{K}_a) &:= b2a?u.a2b!(\neg b_a, \tilde{K}_a).\mathbf{nil} \\
Bob(n) &:= \mathrm{A2B}?\tilde{q}. \sum_{\tilde{B}_b \in \{0,1\}^n} \frac{1}{2^n} M_{\tilde{B}_b}[\tilde{q}; \tilde{K}_b].\, \mathrm{Set}_{\tilde{0}}[\tilde{q}].b2a!1.\, Unveil_B(\tilde{B}_b, \tilde{K}_b) \\
Unveil_B(\tilde{B}_b, \tilde{K}_b) &:= a2b?(b_a, \tilde{K}_a).out!(\mathrm{cmp}(\tilde{K}_a, b_a, \tilde{B}_b), \mathrm{cmp}(\tilde{K}_b, b_a, \tilde{B}_b), b_a).\mathbf{nil} \\
Bob'(n) &:= (Bob(n) \| out?(\tilde{K}'_a, \tilde{K}'_b, b_a).(\mathbf{if}\ \ \mathrm{len}(\tilde{K}'_a = \tilde{K}'_b \geq n/2) \\
&\qquad\qquad \mathbf{then}\ out_b!(\tilde{K}'_a, \tilde{K}'_b, b_a).\mathbf{nil}) \\
QBC84'(n) &:= (Alice(n)' \| Bob'(n))
\end{aligned} \tag{6.4}$$

Note that there are few changes, indeed, the modified $Unveil_A{}'$ simply sends the dual of b_a, modeling the fact that the opponent tries to cheat by changing the value of the commitment in the middle of the protocol execution. The modified process impersonating Bob, called $Bob'(n)$, instead, adds the missing parts of the protocol, related to the privacy amplification phase. Note that these changes do not invalidate our distribution-based bisimulation proof of Proposition 6.1.1, because the most significant change is represented by process $Bob'(n)$ and the added if-then statement, that is modeled as a τ-action and does not lead to any inconsistency in the distributions.

Again, the protocol is put in a test environment. This time, the environment must contain a way to detect Alice's cheating because our intent is not to evaluate the similarity of the protocol with its specification, but to evaluate its security aspects. The test environment is the following:

$$\begin{aligned} \mathit{Test}' &:= \mathit{out}_b?(\tilde{K}_a, \tilde{K}_b, b_a).(\textbf{if } \tilde{K}_a = \tilde{K}_b \textbf{ then } \mathit{hacked}!0.\textbf{nil}) \\ \mathit{QBC84}_{\mathit{test}}{}'(n) &:= (\mathit{QBC84}'(n) \| \mathit{Test}') \backslash \{a2b, b2a, \mathrm{A2B}, \mathit{out}, \mathit{out}_b\} \end{aligned} \tag{6.5}$$

Here, if $\tilde{K}_a$ and $\tilde{K}_b$ coincide, it means that Bob would consider the commitment of the dual value as valid and then the protocol would be compromised. To investigate the security degree of the protocol we prove the following proposition:

Proposition 6.1.2. *Given a density operator ρ, it holds that:*

$$\langle \mathit{QBC84}_{\mathit{test}}{}'(n), \rho \rangle \overset{c^n}{\approx} \langle \mathrm{Set}_{\tilde{0}}[\tilde{q}].\boldsymbol{nil}, \rho \rangle$$

Proof. Let suppose that $B = \langle \mathit{QBC84}_{\mathit{test}}{}'(n), \rho \rangle$ and $C = \langle \mathrm{Set}_{\tilde{0}}[\tilde{q}].\textbf{nil}, \rho \rangle$. For the proof of $B \overset{c^n}{\approx} C$, we only need to compute the probability of B performing *hacked*!0, because this is the only visible action that B can do which contributes to a possible transition inconsistency of distributions obtained from C. If the total probability of their appearance is upper bounded by c^n with $c = \frac{1}{2}$, then $B \overset{c^n}{\approx} C$. Analyzing the protocol, for each qubit sent by the evil Alice, Bob chooses the wrong basis with probability $\frac{1}{2}$, making half of the bases incorrect on average. For each of the wrong bases, the probability of measuring the same string that Alice want to encode is exactly $\frac{1}{2}$. Therefore, the total probability of B performing *hacked*!0 is $p = \frac{1}{2}^{\frac{n}{2}}$. □

As the process $\mathrm{Set}_{\tilde{0}}[\tilde{q}].\textbf{nil}$ never perform *hacked*!0, this indicates the insecurity degree of *QBC84*(n) to be at most c^n, which decreases exponentially to 0 when $n \to \infty$.

6.1.3 Security Against an Active Attacker

In this case instead, we analyze the case in which Alice has a quantum computer. This setting includes an active version of the attacker, impersonated by Alice, who wants to cheat on Bob using the EPR-paradox described in

Chapter 2. Given that the attack modifies in an important way the behavior of Alice, we prefer to modify the whole protocol and start again from the correctness test, analyzing first if the protocol follows its specification and then evaluating its security properties.

The modified protocol is the following:

$$\begin{aligned}
\mathit{Alice}''(n) &:= \sum_{b_a\in\{0,1\}} \sum_{\tilde{K}_a\in\{0,1\}^n} \frac{1}{2^{n+1}} \operatorname{Set}_{\tilde{K}_a}[\tilde{q}]. \operatorname{Set}_{\tilde{K}_a}[\tilde{r}].H_{b_a}[\tilde{q}]. \operatorname{Cnot}[\tilde{q},\tilde{r}].\\
&\qquad .\mathrm{A2B}!\tilde{q}.\,\mathit{Unveil_A}''(b_a,\tilde{K}_a)\\
\mathit{Unveil_A}''(b_a,\tilde{K}_a) &:= \mathit{b2a}?u.M_{b_a}[\tilde{r};\tilde{K}'_a]. \operatorname{Set}_{\tilde{0}}[\tilde{r}].\mathit{a2b}!(b_a,\tilde{K}'_a).\mathbf{nil}\\
\mathit{Bob}''(n) &:= \mathrm{A2B}?\tilde{q}. \sum_{\tilde{B}_b\in\{0,1\}^n} \frac{1}{2^n} M_{\tilde{B}_b}[\tilde{q};\tilde{K}_b].\mathit{b2a}!1.\,\mathit{Unveil_B}'(\tilde{B}_b,\tilde{K}_b)\\
\mathit{Unveil_B}'(\tilde{B}_b,\tilde{K}_b) &:= \mathit{a2b}?(b_a,\tilde{K}_a). \operatorname{Set}_{\tilde{0}}[\tilde{q}].\\
&\qquad .\mathit{out}!(\operatorname{cmp}(\tilde{K}_a,b_a,\tilde{B}_b),\operatorname{cmp}(\tilde{K}_b,b_a,\tilde{B}_b),b_a).\mathbf{nil}\\
\mathit{QBC84}''(n) &:= \mathit{Alice}''(n)\|\mathit{Bob}''(n)
\end{aligned} \tag{6.6}$$

Note that in this design choice we model the exact attack described by Bennet and Brassard, but for the moment Alice remains honest. In $\mathit{Alice}''(n)$ we have added a Set operator that prepares another string of qubits $\tilde{r}$ in the same way as $\tilde{q}$. After the application of the Hadamard gate, a CNot gate is applied to $[\tilde{q},\tilde{r}]$. The meaning of the operation is to create a series of entangled pairs of qubits by applying the CNot gate to every pair $|q\rangle_i, |r\rangle_i$.

Thus, we follow the steps of Section 6.1.1 and we put $\mathit{QBC84}''(n)$ in the same test environment:

$$\begin{aligned}
\mathit{Test}'' &:= \mathit{out}?(\tilde{K}_a,\tilde{K}_b,b_a).(\mathbf{if}\ \tilde{K}_a=\tilde{K}_b\ \mathbf{then}\ \mathit{bit}!b_a.\mathbf{nil}\ \mathbf{else}\ \mathit{fail}!0.\mathbf{nil})\\
\mathit{QBC84_{test}}''(n) &:= (\mathit{QBC84}''(n)\|\,\mathit{Test}'')\setminus\{\mathit{a2b},\mathit{b2a},\mathrm{A2B},\mathit{out}\}
\end{aligned} \tag{6.7}$$

The ideal specification of the protocol here changes slightly because we must consider that the second string of qubits changes the environment of the processes.

$$QBC84_{spec}{}'(n) := \sum_{i=0}^{n} \sum_{\tilde{x}\in\{0,1\}^i} \frac{\binom{n}{i}}{2^{n+i}} \, \mathrm{Set}_{\tilde{0}}[\tilde{q}].\, \mathrm{Set}_{\tilde{0}}[\tilde{r}].bit!\, \mathrm{first}(\tilde{x}).\mathbf{nil} \tag{6.8}$$

We are now ready to compare this modified version of BB84 quantum bit commitment in a test environment with this new ideal specification using, as usual, the notion of distributed-based bisimulation.

Proposition 6.1.3. *Given a density operator ρ, it holds that:*

$$\langle QBC84_{test}{}''(n), \rho\rangle \approx \langle QBC84'_{spec}(n), \rho\rangle$$

Proof. The proof follows from the proof of Proposition 6.1.1: no new visible action has been added, and the operation $\mathrm{Set}_{\tilde{0}}[\tilde{r}]$ added in the new specification balances the new string of qubits introduced for the EPR attack in $Alice''$. □

To analyze the security propriety related to this particular setting, we remark that the BB84 quantum bit commitment protocol is unsafe if the attacker has quantum memory and uses the EPR-paradox to build some EPR-pairs. For this reason, our goal here is to prove that the bisimilarity relation found in the passive attacker case turns out to be **impossible** in this setting.

Let us define the last modification of the BB84 quantum bit commitment scheme, adding the privacy amplification phase and modifying Alice's unveiling process to make her cheat:

$$\begin{aligned}
\mathit{Alice}'''(n) &:= \sum_{b_a\in\{0,1\}} \sum_{\tilde{K}_a\in\{0,1\}^n} \frac{1}{2^{n+1}} \operatorname{Set}_{\tilde{K}_a}[\tilde{q}].\operatorname{Set}_{\tilde{K}_a}[\tilde{r}].H_{b_a}[\tilde{q}].\operatorname{Cnot}[\tilde{q},\tilde{r}].\\
&\qquad .\mathrm{A2B}!\tilde{q}.\mathit{Unveil_A}'''(b_a,\tilde{K}_a)\\
\mathit{Unveil_A}'''(b_a,\tilde{K}_a) &:= \mathit{b2a}?u.M_{\neg b_a}[\tilde{r};\tilde{K}'_a].\operatorname{Set}_{\tilde{0}}[\tilde{r}].\mathit{a2b}!(\neg b_a,\tilde{K}'_a).\mathbf{nil}\\
\mathit{Bob}''(n) &:= \mathrm{A2B}?\tilde{q}.\sum_{\tilde{B}_b\in\{0,1\}^n}\frac{1}{2^n}M_{\tilde{B}_b}[\tilde{q};\tilde{K}_b].\mathit{b2a}!1.\mathit{Unveil_B}'(\tilde{B}_b,\tilde{K}_b)\\
\mathit{Unveil_B}'(\tilde{B}_b,\tilde{K}_b) &:= \mathit{a2b}?(b_a,\tilde{K}_a).\operatorname{Set}_{\tilde{0}}[\tilde{q}].\\
&\qquad .\mathit{out}!(\operatorname{cmp}(\tilde{K}_a,b_a,\tilde{B}_b),\operatorname{cmp}(\tilde{K}_b,b_a,\tilde{B}_b),b_a).\mathbf{nil}\\
\mathit{Bob}'''(n) &:= (\mathit{Bob}''(n)\|\mathit{out}?(\tilde{K}'_a,\tilde{K}'_b,b_a).(\mathbf{if}\ \operatorname{len}(\tilde{K}'_a=\tilde{K}'_b\geq n/2)\\
&\qquad \mathbf{then}\ \mathit{out_b}!(\tilde{K}'_a,\tilde{K}'_b,b_a).\mathbf{nil})\\
\mathit{QBC84}'''(n) &:= (\mathit{Alice}'''(n)\|\mathit{Bob}'''(n))
\end{aligned} \tag{6.9}$$

Then, we put $\mathit{QBC84}'''(n)$ under test, using the same test environment of Section 6.1.2.

$$\begin{aligned}
\mathit{Test}''' &:= \mathit{out_b}?(\tilde{K}_a,\tilde{K}_b,b_a).\ (\mathbf{if}\ \tilde{K}_a=\tilde{K}_b\ \mathbf{then}\ \mathit{hacked}!0.\mathbf{nil})\\
\mathit{QBC84_{test}}'''(n) &:= (\mathit{QBC84}'''(n)\|\mathit{Test}''')\setminus\{\mathit{a2b},\mathit{b2a},\mathrm{A2B},\mathit{out},\mathit{out_b}\}
\end{aligned} \tag{6.10}$$

Then, to prove the protocol is unsecure in this setting, we need to prove the following proposition:

Proposition 6.1.4. *Given a density operator ρ, it holds that:*

$$\langle \mathit{QBC84_{test}}'''(n),\rho\rangle \overset{c^n}{\not\approx} \langle \operatorname{Set}_{\tilde{0}}[\tilde{q}].\operatorname{Set}_{\tilde{0}}[\tilde{r}].\boldsymbol{nil},\rho\rangle$$

Proof. To prove this proposition for every c^n, we need to find a distribution that is inconsistent between the two. Let then suppose $B=\langle \mathit{QBC84_{test}}'''(n),\rho\rangle$ and $C=\langle\operatorname{Set}_{\tilde{0}}[\tilde{q}].\operatorname{Set}_{\tilde{0}}[\tilde{r}].\mathbf{nil},\rho\rangle$. As in the proof of Proposition 6.1.1, we need to compute the probability of B performing $\mathit{hacked}!0$, because C does not contain any visible action. By the EPR-paradox, when Bob performs the measurement of $\tilde{q}$, it modifies $\tilde{r}$ such that Alice can use it to perform a new measurement that returns a bistring consistent with the new value of the commitment and also consistent with the result of Bob's bitstring with probability $p_{win} = 1$. Then B and C are not approximately bisimilar for any n and the resulting protocol is not secure under the active attacker assumption. □

6.2 Kent Relativistic Bit Commitment

For the analysis of the Kent relativistic bit commitment protocol, we will follow step by step Section 6.1. In fact, we model the protocol as a concurrent execution of quantum processes, which in this case model the activity of Alice and Bob, together with their respective agents. In the first part we design a light version of the protocol, in which we do not consider Bob's consistency check. This first version is then put in a classical test environment to check if there exists a bisimilarity relation with its ideal specification. Lastly, we analyze the security of the protocol under two different assumptions: in the former the adversary is designed as a passive attacker, in the latter instead we give more power to the adversary by modeling them as an active attacker.

6.2.1 Correctness

Being already familiar with qCCS, we can directly describe the Kent relativistic bit commitment protocol as a process, *KENT12*(n), that returns the two bitstrings obtained by Bob's agents. Note also that this model underlies the first step of agreement between the parties. The protocol design is the following:

$$\begin{aligned}
Bob(n) &:= \sum_{\tilde{K}_b, \tilde{B}_b \in \{0,1\}^n} \frac{1}{2^{2n}} \, \mathrm{Set}_{\tilde{K}_b}[\tilde{q}].H_{b_a}[\tilde{q}].\mathrm{B2A}!\tilde{q}.out!(\tilde{K}_b, \tilde{B}_b).\mathbf{nil} \\
Alice(n) &:= \mathrm{B2A}?\tilde{q}. \sum_{b_a \in \{0,1\}} \frac{1}{2} M_{b_a}[\tilde{q}; \tilde{K}_a]. \, \mathrm{Set}_{\tilde{0}}[\tilde{q}]. \\
&\qquad .a2a_1!(\tilde{K}_a, b_a).a2a_2!(\tilde{K}_a, b_a).\mathbf{nil} \\
Alice1 &:= a2a_1?(\tilde{K}_{a_1}, b_{a_1}).a_1 2b_1!(\tilde{K}_{a_1}, b_{a_1}).\mathbf{nil} \\
Alice2 &:= a2a_2?(\tilde{K}_{a_2}, b_{a_2}).a_2 2b_2!(\tilde{K}_{a_2}, b_{a_2}).\mathbf{nil} \\
Bob1 &:= a_1 2b_1?(\tilde{K}_{a_1}, b_{a_1}).out!(\tilde{K}_{a_1}, b_{a_1}).\mathbf{nil} \\
Bob2 &:= a_2 2b_2?(\tilde{K}_{a_2}, b_{a_2}).out!(\tilde{K}_{a_2}, b_{a_2}).\mathbf{nil} \\
KENT12(n) &:= Bob(n) \| Alice(n) \| Alice1 \| Alice2 \| Bob1 \| Bob2
\end{aligned} \tag{6.11}$$

Firstly, note that the protocol design is particularly simple because with qCCS we can abstract away the relativistic part by the syntax. Differently from the BB84 quantum bit commitment scheme, here it is Bob that starts the protocol by randomly generating the two bitstrings which he needs to prepare the qubit string. This double random generation leads to a much larger number of distributions, as can be seen by the probability parameter $\frac{1}{2^{2n}}$. Note also that we have added the commitment value b_a in the classical communication channel from Alice's agents to Bob's agents. This has no effect on the security properties of the protocol and does not add any constraints, as demonstrated in [33], but it will help us in the following proofs.

To show the correctness of the protocol we put it in a new test environment, and we modify the syntax introducing the following syntax sugar:

$$\textbf{switch } x \textbf{ case } y : P_1 \textbf{ case } z : P_2 \ldots \textbf{ otherwise} : P_n,$$

that correspond to:

$$\textbf{if } b_1 \textbf{ then } P_1 \textbf{ else if } b_2 \textbf{ then } P_2 \ldots \textbf{ else } P_n,$$

with $x, y, z \in \textsc{Real}$, $b_1 = (x = y), b_2 = (x = z) \in \textsc{Bool}$ and $P_1, P_2, P_n \in qProc$.

Thus, the test environment is the following:

$$\begin{aligned}
\mathit{Test} := \mathit{out}?(\tilde{K}_b, \tilde{B}_b, \tilde{K}_{a_1}, \tilde{K}_{a_2}, b_{a_1}, b_{a_2}).(&\textbf{if } \tilde{K}_{a_1} = \tilde{K}_{a_2} \textbf{ then} \\
&(\textbf{if } b_{a_1} = b_{a_2} \textbf{ then } (\textbf{switch}\, \mathrm{cons}\left(b_{a_1}, \tilde{K}_{a_1}, \tilde{K}_b, \tilde{B}_b\right) \\
&\qquad \textbf{case } 0 : \mathit{bit}!0.\textbf{nil} \\
&\qquad \textbf{case } 1 : \mathit{bit}!1.\textbf{nil} \\
&\qquad \textbf{case } \epsilon : \mathit{bit}!\epsilon.\textbf{nil} \\
&\qquad \textbf{otherwise: } \mathit{fail}!0.\textbf{nil}) \\
&\textbf{else } \mathit{fail}!0.\textbf{nil}) \textbf{ else } \mathit{fail}!0.\textbf{nil})
\end{aligned}$$

$$\mathit{KENT12}_{\mathit{test}}(n) := (\mathit{KENT12}(n) \,\|\, \mathit{Test}) \setminus \{\text{B2A}, a2a_1, a2a_2, a_12b_1, a_22b_2, \mathit{out}\} \tag{6.12}$$

The function $\mathrm{cons}(x, \tilde{y}, \tilde{z}, \tilde{w})$ checks the consistency of the value $\tilde{x}$, representing the basis, and the string $\tilde{y}$, with respect to the string $\tilde{z}$, using the string $\tilde{w}$ as bases, and returns the value obtained by the consistency check. If the check fails because the string is entirely different from the committed value, it returns the empty string. Instead, if the check fails because x is inconsistent with the other values, it returns an error value, which falls into the **otherwise** case into the switch operator.

Having this we can proceed using the same ideal specification of Section 6.1:

$$\mathit{KENT12}_{\mathit{spec}}(n) := \sum_{i=0}^{n} \sum_{\tilde{x} \in \{0,1\}^i} \frac{\binom{n}{i}}{2^{n+i}} \mathrm{Set}_{\tilde{0}}[\tilde{q}].\mathit{bit}!\,\mathrm{first}(\tilde{x}).\textbf{nil} \tag{6.13}$$

With $\mathrm{first}(\tilde{x})$ to be a function that takes a string of bits and returns its first bit.

We compare $\mathit{KENT12}_{\mathit{test}}(n)$ and $\mathit{KENT12}_{\mathit{spec}}(n)$ using the notion of distributed-based bisimulation.

Proposition 6.2.1. *Given a density operator ρ, it holds that:*

$$\langle KENT12_{test}(n), \rho\rangle \approx \langle KENT12_{spec}(n), \rho\rangle$$

Proof. As in Proposition 6.1.1, to simplify the proof, we start our analysis using $n = 1$. Note that the pLTS representing $KENT12_{spec}(1)$ is very similar to the one that represent $QBC84_{spec}(1)$, because of the formalization of the two processes. The only visible difference is given by the inversion of roles concerning the act of sending $\tilde{q}$, as can be viewed in Figure 6.8. Furthermore, the pLTS representing $KENT12_{test}(1)$ is equal to the pLTS of $QBC84_{test}(1)$ because the formalization of the specification is the same.

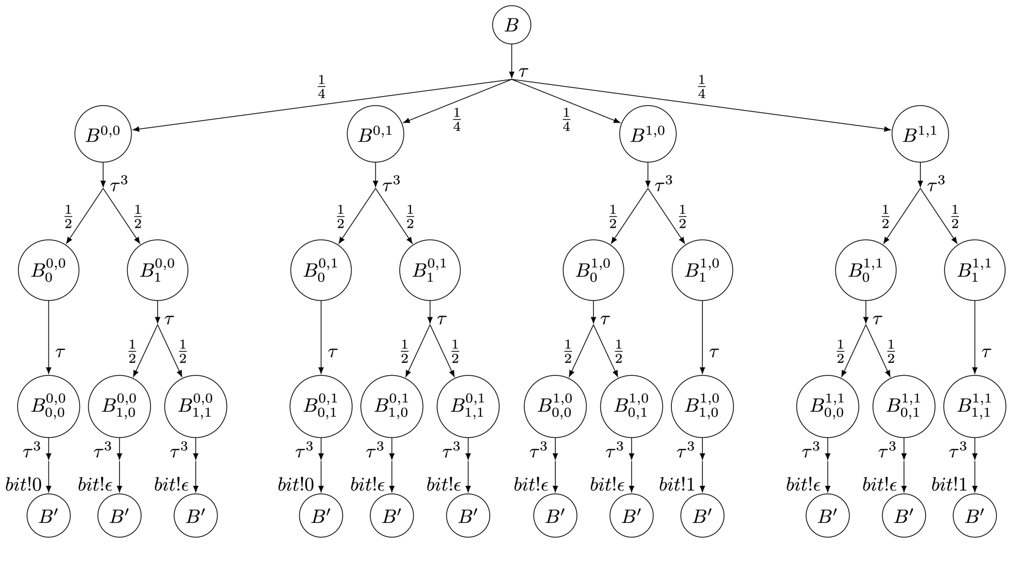


Figure 6.8: pLTS of $KENT12_{spec}(1)$

Remember that the superscripts of the B-configurations denote Bob's choices of $\tilde{B}_b$ and $\tilde{K}_b$, while the subscripts denote Alice's choices of b_a and the obtained measurement $\tilde{K}_a$. Then we can follow the same steps of the proof of Proposition 6.1.1:

1. B' and C' are clearly bisimilar because they are both dead, with $qv = 0$ and the accompanied quantum stares are $|0\rangle_{\tilde{q}}\langle 0| \otimes tr_q\rho$.

2. For every $i \in \{0,1\}$, the configurations $B^{0,i}_{0,i}, B^{1,i}_{1,i}$ and C'_i are bisimilar, as they all have $qv = 0$, and the only visible action that they can perform is $bit!i$, through which they become B' and C'.

3. Almost the same as in point (2), all the configuration in $\{B^{i,j}_{k,h} \mid i,j,k,h \in 0,1, i \neq k\}$ are bisimilar to C'_ϵ.

4. For every $i \in \{0,1\}$, the configurations $B^{0,i}_0$, B^{1i}_1 and C_i are bisimilar, as they all have $qv = \tilde{q}$ and the only action they can perform is τ, through which they become a configuration in (2).

5. As in (4), all the configurations in $\{B^{i,j}_k \mid i,j,k \in 0,1, i \neq k\}$ are bisimilar to C_ϵ.

6. $B^{0,0}, B^{0,1}$ are bisimilar to $(\frac{1}{2}C_0 + \frac{1}{2}C_\epsilon)$, as they all have $qv = \tilde{q}$ and they can perform action τ, through which become a distribution of (4) or (5) with half probability for each one.

7. $B^{1,0}, B^{1,1}$ are bisimilar to $(\frac{1}{2}C_1 + \frac{1}{2}C_\epsilon)$, as in (6)

8. In the end, B is bisimilar to C as $B \xrightarrow{\tau} \sum_{i,j=0}^{1} \frac{1}{4}B^{i,j}$ while $C \xrightarrow{\tau} \frac{1}{4}C_0 + \frac{1}{2}C_\epsilon + \frac{1}{4}C_1$, and from (6) and (7), the resulting distributions are bisimilar.

Instead, when we move to the proof of bisimilarity between $KENT12_{test}(n)$ and $KENT12_{spec}(n)$, things differ sightly from the $QBC84_{test}(n)$ proof. This happens because of the change of the number of possible distributions, given by the choice of $\tilde{K}_b$ and $\tilde{B}_b$ by Bob. However, even in this case we do an analysis by parts, given the exponential number of states resulting from the increase in the number of distributions. Then:

- Figure 6.9 represents the initial part of the graph. As n increases, the number of the possible equiprobable states grows exponentially due to the two random bitstrings.

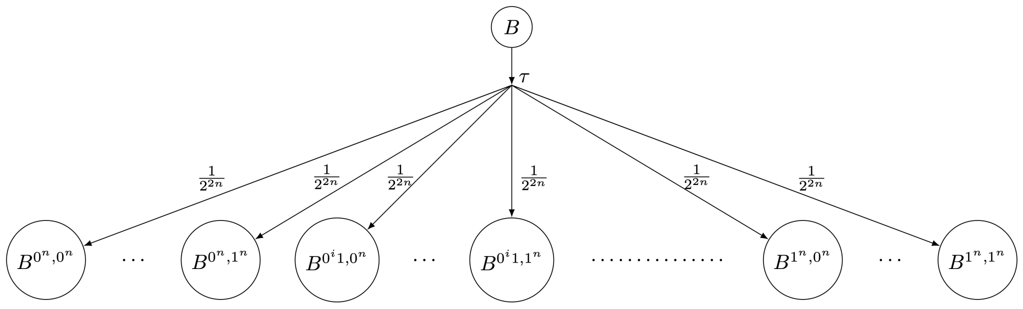


Figure 6.9: $\mathit{KENT12}_{test}(n)(1)$

- Figure 6.10 represents the following part of the pLTS, focusing on the individual state $B^{0^n,0^n}$. This part depicts the choice of the Alice's commitment b_a.

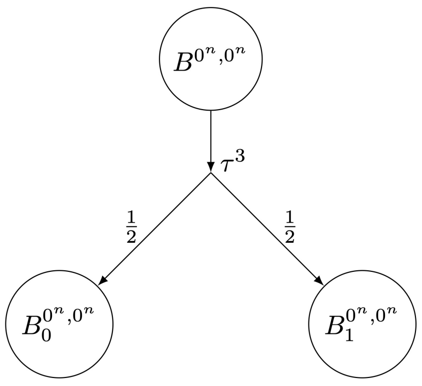


Figure 6.10: $\mathit{KENT12}_{test}(n)(2)$

- Figure 6.11 represents the next part, considering the case in which Alice's commitment matches with at least one basis in the bitstring of bases chosen by Bob. Remember that this happens with probability $p = 1 - \frac{1}{2^n}$.

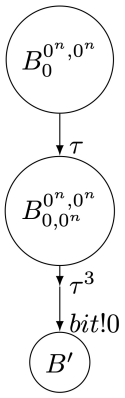


Figure 6.11: $KENT12_{test}(n)(3)$

- Lastly, Figure 6.12 shows the case in which Alice's commitment does not match with any single bit of the Bob's bitstring representing the bases. In this case, the measurement returns a completely random bitstring and the protocol returns the empty string because Bob cannot check the consistency of Alice's measurement bistring.

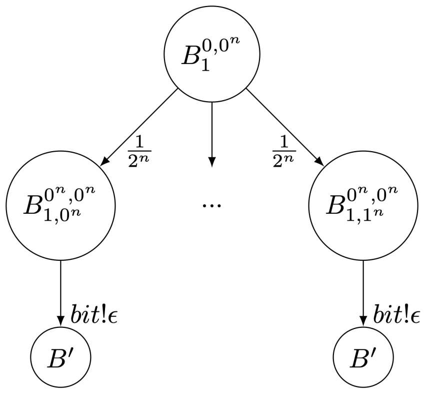


Figure 6.12: $KENT12_{test}(n)(4)$

Given that $KENT12_{spec}(n)$ follows exactly Figure 6.3, we can proceed to the proof, starting from the bottom:

1. As above, B' and C' are bisimilar as they are both dead, they are both

closed and the accompanied quantum stares are $|0\rangle_{\tilde{q}}\langle 0| \otimes tr_q\rho$.

2. For every $k \in \{0,1\}$, $\tilde{i}, \tilde{j}, \tilde{h} \in \{0,1\}^n$, the configurations $B^{\tilde{i},\tilde{j}}_{k,\tilde{h}}$ with at least a bit of $\tilde{i}$ matching k are bisimilar to C'_k

3. As in (2), all the configurations in $\{B^{\tilde{i},\tilde{j}}_{k,\tilde{h}} \,|\, k \in \{0,1\}, \tilde{i}, \tilde{j}, \tilde{h} \in \{0,1\}^n, k \neq \tilde{i}\}$ are bisimilar to C'_ϵ.

4. For every $k \in \{0,1\}$, $\tilde{i}, \tilde{j} \in \{0,1\}^n$, the configurations $B^{\tilde{i},\tilde{j}}_k$ with at with at least a bit of $\tilde{i}$ matching with k are also bisimilar to C_k as they all have $qv = \tilde{q}$ and the only action they can perform is τ, through which they become a configuration in (2).

5. As in (4), all the configurations in $\{B^{\tilde{i},\tilde{j}}_k \,|\, k \in \{0,1\}, \tilde{i}, \tilde{j} \in \{0,1\}^n, \tilde{i} \neq k\}$ are bisimilar to C_ϵ.

6. For every $\tilde{i}$ and $\tilde{j} \in \{0,1\}^n$, the distributions $B^{\tilde{i},\tilde{j}}$ are bisimilar to $\frac{1}{2^n}C_\epsilon + \frac{1}{2}(1 - \frac{1}{2^n})C_0 + \frac{1}{2}(1 - \frac{1}{2^n})C_1$ as they all have $qv = \tilde{q}$ and they can perform action τ through which become a distribution of (4) or (5) with the respective probability.

7. Lastly B is bisimilar to C as $B \xrightarrow{\tau} \sum_{i=0}^{n} \sum_{\tilde{j}=0^n}^{1^n} \frac{1}{2^{2n}} B^{\tilde{i},\tilde{j}}$, while $C \xrightarrow{\tau} \frac{1}{2^n}C_\epsilon + \frac{1}{2}(1 - \frac{1}{2^n})C_0 + \frac{1}{2}(1 - \frac{1}{2^n})C_1$, and from (6) the resulting distributions are bisimilar.

□

6.2.2 Security against Passive Attacker

As in the previous section, we want to build progressively stronger intruders. We start with an investigation of the notion of security of the Kent relativistic bit commitment protocol, considering the case in which Alice is a passive attacker. As we know, Kent's protocol is conjectured to be unconditionally secure, so we need to prove this using the approximate bisimulation technique. Note that here, like in Section 6.1.2, the only changes to Kent's processes derive from Alice's attempt to commit the opposite value with respect to that used for the measurement. The modified protocol is the following:

$$\begin{aligned}
\mathit{Bob}(n) &:= \sum_{\tilde{K}_b,\tilde{B}_b\in\{0,1\}^n} \frac{1}{2^{2n}}\,\mathrm{Set}_{\tilde{K}_b}[\tilde{q}].H_{b_a}[\tilde{q}].\mathrm{B2A}!\tilde{q}.\mathit{out}!(\tilde{K}_b,\tilde{B}_b).\mathbf{nil}\\
\mathit{Alice}'(n) &:= \mathrm{B2A}?\tilde{q}.\sum_{b_a\in\{0,1\}}\frac{1}{2}M_{b_a}[\tilde{q};\tilde{K}_a].\,\mathrm{Set}_{\tilde{0}}[\tilde{q}].\\
&\qquad\qquad .a2a_1!(\tilde{K}_a,\neg b_a).a2a_2!(\tilde{K}_a,\neg b_a).\mathbf{nil}\\
\mathit{Alice1} &:= a2a_1?(\tilde{K}_{a_1},b_{a_1}).a_12b_1!(\tilde{K}_{a_1},b_{a_1}).\mathbf{nil}\\
\mathit{Alice2} &:= a2a_2?(\tilde{K}_{a_2},b_{a_2}).a_22b_2!(\tilde{K}_{a_2},b_{a_2}).\mathbf{nil}\\
\mathit{Bob1} &:= a_12b_1?(\tilde{K}_{a_1},b_{a_1}).\mathit{out}!(\tilde{K}_{a_1},b_{a_1}).\mathbf{nil}\\
\mathit{Bob2} &:= a_22b_2?(\tilde{K}_{a_2},b_{a_2}).\mathit{out}!(\tilde{K}_{a_2},b_{a_2}).\mathbf{nil}\\
\mathit{KENT12}(n)' &:= \mathit{Bob}(n)\|\mathit{Alice}'(n)\|\mathit{Alice1}\|\mathit{Alice2}\|\mathit{Bob1}\|\mathit{Bob2}
\end{aligned} \tag{6.14}$$

Then we can put the protocol into a test environment:

$$\begin{aligned}
\mathit{Test}' &:= \mathit{out}?(\tilde{K}_b,\tilde{B}_b,\tilde{K}_{a_1},\tilde{K}_{a_2}b_{a_1},b_{a_2}).(\mathbf{if}\ \tilde{K}_{a_1}=\tilde{K}_{b_2}\ \mathbf{then}\\
&\qquad (\mathbf{if}\ b_{a_1}=b_{a_2}\ \mathbf{then}\ (\mathbf{switch}\,\mathrm{cons}(b_{a_1},\tilde{K}_{a_1},\tilde{K}_b,\tilde{B}_b)\\
&\qquad\qquad \mathbf{case}\ 0: \mathit{hacked}!0.\mathbf{nil}\\
&\qquad\qquad \mathbf{case}\ 1: \mathit{hacked}!0.\mathbf{nil}\\
&\qquad\qquad \mathbf{otherwise:}\ \mathbf{nil})))\\
\mathit{KENT12}'_{\mathit{test}}(n) &:= (\mathit{KENT12}'(n)\,\|\,\mathit{Test}')\backslash\,\{\mathrm{B2A},a2a_1,a2a_2,a_12b_1,a_22b_2,\mathit{out}\}
\end{aligned} \tag{6.15}$$

Thus, given the test environment, we use the following proposition to show the security degree of $\mathit{KENT12}_{\mathit{test}}'(n)$.

Proposition 6.2.2. *Given a density operator ρ, it holds that:*

$$\langle \mathit{KENT12}_{\mathit{test}}'(n),\rho\rangle \overset{c^n}{\approx} \langle \mathrm{Set}_{\tilde{0}}[\tilde{q}].\boldsymbol{nil},\rho\rangle$$

Proof. Also in this case, the proof follows directly from 6.1.2. Supposing $B = \langle \mathit{KENT12}_{test}''(n), \rho\rangle$ and $C = \langle \mathrm{Set}_{\tilde{0}}[\tilde{q}].\mathbf{nil}, \rho\rangle$, we need to compute the probability of B performing *hacked*!0. Analyzing this modified protocol, the probability of Alice's victory, resulting in *hacked*!0 in the test environment, depends solely on the probability with which the bits obtained by the qubit measured with a wrong basis are consistent with respect to Bob's corresponding bit. We know that a measurement performed with the wrong basis has an equal probability of returning 0 or 1. Then, given that Bob's bases are chosen randomly, on average the probability of Alice's victory is $p_{hacked!0} = \frac{1}{2}^{n/2}$. Therefore $B \overset{\frac{1}{2}^{n/2}}{\approx} C$. □

6.2.3 Security against Active Attacker

Thanks to [11], in this case we can analyze the environment in which the intruder, Alice, can perform *every* strategy based on classical computation she wants. We start by modifying the protocol:

$$\begin{aligned}
\mathit{Bob}(n) &:= \sum_{\tilde{K}_b, \tilde{B}_b \in \{0,1\}^n} \frac{1}{2^{2n}} \mathrm{Set}_{\tilde{K}_b}[\tilde{q}].H_{b_a}[\tilde{q}].\mathrm{B2A}!\tilde{q}.out!(\tilde{K}_b, \tilde{B}_b).\mathbf{nil} \\
\mathit{Alice}''(n) &:= \mathrm{B2A}?\tilde{q}. \sum_{b_a \in \{0,1\}} \frac{1}{2} M_{b_a}[\tilde{q}; \tilde{K}_a].\,\mathrm{Set}_{\tilde{0}}[\tilde{q}]. \\
&\qquad .a2a_1!\mathrm{strat}_1(\tilde{K}_a, \neg b_a).a2a_2!\mathrm{strat}_2(\tilde{K}_a, \neg b_a).\mathbf{nil} \\
\mathit{Alice1} &:= a2a_1?(\tilde{K}_{a_1}, b_{a_1}).a_1 2b_1!(\tilde{K}_{a_1}, b_{a_1}).\mathbf{nil} \\
\mathit{Alice1} &:= a2a_2?(\tilde{K}_{a_2}, b_{a_2}).a_2 2b_2!(\tilde{K}_{a_2}, b_{a_2}).\mathbf{nil} \\
\mathit{Bob1} &:= a_1 2b_1?(\tilde{K}_{a_1}, b_{a_1}).out!(\tilde{K}_{a_1}, b_{a_1}).\mathbf{nil} \\
\mathit{Bob2} &:= a_2 2b_2?(\tilde{K}_{a_2}, b_{a_2}).out!(\tilde{K}_{a_2}, b_{a_2}).\mathbf{nil} \\
\mathit{KENT12}(n)'' &:= \mathit{Bob}(n) \| \mathit{Alice}''(n) \| \mathit{Alice1} \| \mathit{Alice2} \| \mathit{Bob1} \| \mathit{Bob2}
\end{aligned} \tag{6.16}$$

Alice, in this case, after having measured the strings of qubits with the commitment value, attempts to cheat using two ideal strategies represented by the two functions strat$_1$ and strat$_2$, which take as input the string resulting from the measurement operation and the dual of the value of the commitment, trying to return a string of bits consistent with the latter. Note that strat$_1$ and strat$_2$ can represent every strategy based on classical computation.

As usual, we put *KENT12*$''(n)$ in the same test environment of the previous section.

$$\begin{aligned}
\mathit{Test}' := \mathit{out}?(\tilde{K}_b, \tilde{B}_b, \tilde{K}_{a_1}, \tilde{K}_{a_2} b_{a_1}, b_{a_2}).(\textbf{if } \tilde{K}_{a_1} = \tilde{K}_{b_2} \textbf{ then}\\
(\textbf{if } b_{a_1} = b_{a_2} \textbf{ then } (\textbf{switch}\, \mathrm{cons}(b_{a_1}, \tilde{K}_{a_1}, \tilde{K}_b, \tilde{B}_b)\\
\textbf{case } 0 : \mathit{hacked}!0.\textbf{nil}\\
\textbf{case } 1 : \mathit{hacked}!0.\textbf{nil}\\
\textbf{otherwise: nil})))\\
\mathit{KENT12}''_{test}(n) := (\mathit{KENT12}''(n) \,\|\, \mathit{Test}') \backslash \{\mathrm{B2A}, a2a_1, a2a_2, a_1 2b_1, a_2 2b_2, \mathit{out}\}
\end{aligned} \tag{6.17}$$

As in Proposition 6.1.2, to show the security degree of *KENT12*$''_{test}(n)$ it suffices to prove the following property:

Proposition 6.2.3. *Given a density operator ρ, it holds that:*

$$\langle \mathit{KENT12}_{test}''(n), \rho\rangle \overset{c^n}{\approx} \langle \mathrm{Set}_{\tilde{0}}[\tilde{q}].\boldsymbol{nil}, \rho\rangle$$

Proof. The proof here follows the security proofs in Section 2.4.1. Let us suppose $B = \langle \mathit{KENT12}_{test}''(n), \rho\rangle$ and $C = \langle \mathrm{Set}_{\tilde{0}}[\tilde{q}].\mathbf{nil}, \rho\rangle$, thanks to Lemma 2.4.3 we know that for a single qubit, the probability of Alice winning is $p \leq \frac{1}{2}(1 + \frac{1}{\sqrt{2}})$, and thanks to Lemma 2.4.1 and Theorem 2.4.5 we know that the probability of Alice winning for n qubits is $p_n \leq (\frac{1}{2}(1 + \frac{1}{\sqrt{2}}))^n$. Then, assuming that the strategy $\mathcal{ST} = \{\mathrm{strat}_1, \mathrm{strat}_2\}$ is optimal, $p_n(\mathcal{ST}) = (\frac{1}{2}(1+ \frac{1}{\sqrt{2}}))^n$. Remembering that the only case in which there is an inconsistency of transitions between B and C concerns the transition *hacked*!0, and that this transition occurs only when Alice wins, given that there are not modifications on $env(B)$ or $env(C)$ with respect to Proposition 6.2.2, $B \overset{c^n}{\approx} C$ with $c = \frac{1}{2}(1 + \frac{1}{\sqrt{2}})$. □

Conclusions

The first part of this thesis has been used to introduce all the topics needed for the comprehension of our research work. We started by giving a brief introduction to quantum computing, presenting the physical principles that acts on qubits together with some famous quantum gates and algorithms. We consequently presented quantum cryptography, laying the foundations of perfect secrecy and introducing the two examined protocols: the BB84 quantum bit commitment and Kent's relativistic bit commitment protocol. Subsequently, we presented CCS and various notions of behavioral equivalence used in the literature to compare different systems, immediately followed by the notion of bisimulation, declined in several ways, up to its probabilistic version. Afterwards, we presented a quantum extension of CCS, called qCCS, with its syntax and its operational semantics. Relatively to qCCS, we presented the notions of state-based bisimulation, distribution-based bisimulation and approximate bisimulation, giving examples using qCCS. Lastly, we presented the core of our research work, some cryptographic proofs of the aforementioned protocols, using qCCS and the notions of distributed-based bisimulation and approximate bisimulation.

For each of the two protocols, we followed the same pattern:

- Firstly, we analyzed only the correctness of the desired protocols, by initially describing a simpler version of them and putting it in a test environment. This environment was then compared with an ideal specification of the protocol itself, using the notion of distribution-based bisimulation.

- Next, we introduced an intruder, impersonated by Alice and modelled like a passive attacker, and added the missing parts from the initial design. Thus, we added the protocol to a modified test environment, and we used the notion of approximate bisimulation to analyze the security of the protocol.

- Lastly, we modeled a more powerful intruder, called active attacker, and added the protocol to a test environment. Finally, we used again the notion of approximate-bisimulation to analyze the security of the protocol under this new assumption. Note that we were able to demonstrate that the equivalence notions used on qCCS can capture both the case in which a protocol is secure, as in Kent's protocol, and the case in which the protocol is not secure, as in BB84 under the active attacker hypotesis.

To the best of our knowledge, this is the first time that the security of quantum bit commitment protocols has been examined using a quantum process algebra.

Related and Future Work

With this thesis, we have brought the process algebra approach to bit commitment protocols, including a state-of-the-art relativistic protocol that is conjectured to be unconditionally secure. Furthermore, we have defined opponents that lead to increasingly powerful security notions. In our opinion, there are mainly two directions for future work.

First, bisimilarity checking is a long and tedious task, as can be seen in the proof of propositions 6.1.1 or 6.2.2. This clearly becomes a problem when the number of parties increases or when more complex protocols are considered. Given these aspects, a formal framework for the semi-automated verification of security proofs of quantum cryptographic protocols has been proposed in

[30]. In this paper the syntax of qCCS is partially changed, but it is still the closest proposal to our work, and with some minimal modification the models that we have presented can be proven in that framework. It is also important to point out that our approach is not the only direction taken by the scientific community for the verification of quantum cryptographic protocols. For example, following the work in [4, 5], in [46] the first *quantum relational Hoare Logic* (qRHL) is proposed, together with a tool used for the verification of the security proof of different quantum cryptographic protocols.

Lastly, in this thesis we have pursued various notion of security which, however, are weaker than the notion of *mutual information between the states* normally used in the security proofs of quantum cryptographic protocols. Even if we went further than the work started in [17], both defining a more powerful intruder from a quantum perspective in Section 6.1.3 and using generic classical (optimal) strategies in Section 6.2.3, these security notions may not be enough. An interesting direction of research, from our viewpoint, could be that of defining a new action that represents a *generic* sequence of quantum operations *bounded in space*. This way, we would be able to define a powerful version of the intruder that is capable of carrying out *every* quantum operation (bounded in space) and every classical operation, taking another step forward and getting closer to the notion of mutual information between the states. Note that, for us, this is possible given the current use of the operator Set_0, for example in equations 6.1 and 6.6, which is used to "balance" two environments.

Bibliography

[1] M Ardehali. “Quantum bit commitment based on EPR”. In: *Arxiv preprint quant-ph/9505019* (1995).

[2] László Babai. “Graph isomorphism in quasipolynomial time”. In: *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*. 2016, pp. 684–697.

[3] László Babai and Eugene M Luks. “Canonical labeling of graphs”. In: *Proceedings of the fifteenth annual ACM symposium on Theory of computing*. 1983, pp. 171–183.

[4] Gilles Barthe et al. “Probabilistic relational reasoning for differential privacy”. In: *Proceedings of the 39th annual ACM SIGPLAN-SIGACT symposium on Principles of programming languages*. 2012, pp. 97–110.

[5] Gilles Barthe et al. “Probabilistic relational verification for cryptographic implementations”. In: *ACM SIGPLAN Notices* 49.1 (2014), pp. 193–205.

[6] Charles H Bennett and Gilles Brassard. *Proceedings of the ieee international conference on computers, systems and signal processing*. 1984.

[7] Jan A. Bergstra and Jan Willem Klop. “Algebra of communicating processes with abstraction”. In: *Theoretical computer science* 37 (1985), pp. 77–121.

[8] Paul Bernays. “Alonzo Church. An unsolvable problem of elementary number theory. American journal of mathematics, vol. 58 (1936), pp. 345–363.” In: *The Journal of Symbolic Logic* 1.2 (1936), pp. 73–74.

[9] G Brassard et al. “Proceedings of the 34th Annual IEEE Symposium on the Foundation of Computer Science”. In: (1993).

[10] Gilles Brassard and Claude Crépeau. “Quantum bit commitment and coin tossing protocols”. In: *Conference on the Theory and Application of Cryptography*. Springer. 1990, pp. 49–61.

[11] Sarah Croke and Adrian Kent. “Security details for bit commitment by transmitting measurement outcomes”. In: *Physical Review A* 86.5 (2012), p. 052309.

[12] Yuxin Deng. *Semantics of Probabilistic Processes: An Operational Approach*. Springer, 2015.

[13] Yuxin Deng and Yuan Feng. “Open bisimulation for quantum processes”. In: *IFIP International Conference on Theoretical Computer Science*. Springer. 2012, pp. 119–133.

[14] Alessandra Di Pierro, Chris Hankin, and Herbert Wiklicky. “Measuring the confinement of probabilistic systems”. In: *Theoretical Computer Science* 340.1 (2005), pp. 3–56.

[15] Laurent Doyen, Thomas A Henzinger, and Jean-François Raskin. “Equivalence of labeled Markov chains”. In: *International journal of foundations of computer science* 19.03 (2008), pp. 549–563.

[16] Albert Einstein, Boris Podolsky, and Nathan Rosen. “Can quantum-mechanical description of physical reality be considered complete?” In: *Physical review* 47.10 (1935), p. 777.

[17] Yuan Feng and Mingsheng Ying. “Toward automatic verification of quantum cryptographic protocols”. In: *arXiv preprint arXiv:1507.05278* (2015).

[18] Yuan Feng et al. “Probabilistic bisimulations for quantum processes”. In: *Information and Computation* 205.11 (2007), pp. 1608–1639.

[19] Richard P Feynman. “Simulating physics with computers”. In: *Feynman and computation*. CRC Press, 2018, pp. 133–153.

[20] Simon J Gay and Rajagopal Nagarajan. “Communicating quantum processes”. In: *Proceedings of the 32nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming languages*. 2005, pp. 145–157.

[21] Roberto Gorrieri and Cristian Versari. *Introduction to concurrency theory: transition systems and CCS*. Springer, 2015.

[22] Werner Heisenberg. “Über den anschaulichen Inhalt der quantentheoretischen Kinematik und Mechanik”. In: *Original Scientific Papers Wissenschaftliche Originalarbeiten*. Springer, 1985, pp. 478–504.

[23] Harald Andrés Helfgott, Jitendra Bajpai, and Daniele Dona. “Graph isomorphisms in quasi-polynomial time”. In: *arXiv preprint arXiv:1710.04574* (2017).

[24] Charles Antony Richard Hoare. “Communicating sequential processes”. In: *Communications of the ACM* 21.8 (1978), pp. 666–677.

[25] Lane P Hughston, Richard Jozsa, and William K Wootters. “A complete classification of quantum ensembles having a given density matrix”. In: *Physics Letters A* 183.1 (1993), pp. 14–18.

[26] Philippe Jorrand and Marie Lalire. “Toward a quantum process algebra”. In: *Proceedings of the 1st Conference on Computing Frontiers*. 2004, pp. 111–119.

[27] Jędrzej Kaniewski et al. “Secure bit commitment from relativistic constraints”. In: *IEEE Transactions on Information Theory* 59.7 (2013), pp. 4687–4699.

[28] Adrian Kent. “Unconditionally secure bit commitment by transmitting measurement outcomes”. In: *Physical review letters* 109.13 (2012), p. 130501.

[29] Takahiro Kubota et al. “Application of a process calculus to security proofs of quantum protocols”. In: *Proceedings of the International Conference on Foundations of Computer Science (FCS)*. The Steering Committee of The World Congress in Computer Science, Computer . . . 2012, p. 1.

[30] Takahiro Kubota et al. "Semi-automated verification of security proofs of quantum cryptographic protocols". In: *Journal of Symbolic Computation* 73 (2016), pp. 192–220.

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

[32] Hoi-Kwong Lo and Hoi Fung Chau. "Why quantum bit commitment and ideal quantum coin tossing are impossible". In: *Physica D: Nonlinear Phenomena* 120.1-2 (1998), pp. 177–187.

[33] Tommaso Lunghi et al. "Experimental bit commitment based on quantum communication and special relativity". In: *Physical review letters* 111.18 (2013), p. 180504.

[34] Dominic Mayers. "Unconditional security in quantum cryptography". In: *Journal of the ACM (JACM)* 48.3 (2001), pp. 351–406.

[35] Dominic Mayers. "Unconditionally secure quantum bit commitment is impossible". In: *Physical review letters* 78.17 (1997), p. 3414.

[36] Robin Milner. *A calculus of communicating systems*. Springer, 1980.

[37] Tadao Murata. "Petri nets: Properties, analysis and applications". In: *Proceedings of the IEEE* 77.4 (1989), pp. 541–580.

[38] Michael A Nielsen and Isaac Chuang. *Quantum computation and quantum information*. 2002.

[39] D Park. "A new equivalence notion for communicating systems". In: *Bulletin EATCS* 14 (1981), pp. 78–80.

[40] John Preskill. "Quantum computing and the entanglement frontier". In: *arXiv preprint arXiv:1203.5813* (2012).

[41] Ajith Ramanathan et al. "Probabilistic bisimulation and equivalence for security analysis of network protocols". In: *International Conference on Foundations of Software Science and Computation Structures*. Springer. 2004, pp. 468–483.

[42] Davide Sangiorgi. *Introduction to bisimulation and coinduction*. Cambridge University Press, 2011.

[43] Davide Sangiorgi and Robin Milner. “Techniques of ‘weak bisimulation up to’”. In: *Proc. CONCUR*. Vol. 92. Citeseer. 1992, pp. 32–46.

[44] Claude E Shannon. “Communication theory of secrecy systems”. In: *The Bell system technical journal* 28.4 (1949), pp. 656–715.

[45] Alan Mathison Turing et al. “On computable numbers, with an application to the Entscheidungsproblem”. In: *J. of Math* 58.345-363 (1936), p. 5.

[46] Dominique Unruh. “Quantum relational Hoare logic”. In: *Proceedings of the ACM on Programming Languages* 3.POPL (2019), pp. 1–31.

[47] William K Wootters and Wojciech H Zurek. “A single quantum cannot be cloned”. In: *Nature* 299.5886 (1982), pp. 802–803.

[48] Noson S Yanofsky and Mirco A Mannucci. *Quantum computing for computer scientists*. Cambridge University Press, 2008.

[49] Mingsheng Ying et al. “An algebra of quantum processes”. In: *ACM Transactions on Computational Logic (TOCL)* 10.3 (2009), pp. 1–36.

Ringraziamenti

Se devo essere sincero, non avrei mai pensato di trovarmi a scrivere dei ringraziamenti così sentiti per la fine del mio secondo ciclo di studi universitari, ma durante la stesura di questa tesi e in generale durante tutta il mio percorso di laurea magistrale, ho affrontato situazioni che mai e poi mai mi sarei aspettato di affrontare. Questi momenti li ho superati perché ho anche ricevuto un incredibile sostegno.

Voglio innanzitutto ringraziare il mio relatore di tesi, il Professor Ugo Dal Lago. La sua esperienza, i suoi consigli e la sua pazienza sono stati fondamentali per portare a termine questo lavoro di tesi e di certo saranno di grandissima importanza nei miei studi futuri. Senza di lei e senza il suo corso di Crittografia sicuramente i miei studi avrebbero preso una direzione differente.

Voglio ringraziare il mio correlatore di tesi, il Dottor Andrea Colledan, soprattutto per i suggerimenti che mi hanno permesso di migliorare vari punti di questa tesi e più in generale la mia scrittura scientifica.

Ci tenevo anche a ringraziare tutto il corpo docenti della Laurea Triennale e Magistrale in Informatica dell'Università di Bologna. Ricorderò voi e tutto il tempo passato in facoltà come uno dei periodi più belli e stimolanti della mia vita.

Un enorme ringraziamento va alla mia ragazza Giulia e alla mia famiglia per l'incredibile supporto e per la loro vicinanza durante questo difficilissimo periodo storico. Senza di voi non ce l'avrei mai fatta.

Per ultimi, ma non di certo per importanza, voglio ringraziare tutti i miei amici. Nello specifico il mio migliore amico di sempre Alessandro, la mia compagnia che ormai è diventata come una seconda famiglia e per finire gli amici che ho conosciuto casualmente per questioni lavorative e che sono diventati ormai parte integrante della mia vita.