Introduction to Quantum Computation

Department of Computer Science - École Polytechnique Fédérale de Lausanne

Author: Arthur Aimone

2024/2025

Contents

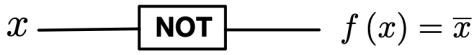
These lecture notes are based on the *Introduction to Quantum Computation* course given at EPFL for the academic year 2024/2025 by Prof. Olivier Lévêque and Prof. Rüdiger Urbanke. The content of the course was originally created by Prof. Nicolas Macris and is based on his lecture notes.

Chapter 1

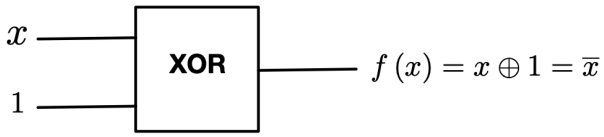
Classical Circuits

In this chapter, we present the fundamentals of classical circuits along with a description of the universal gates.
We consider $f:\{0,1\}^n \longrightarrow \{0,1\}^m$ be a Boolean function taking $(x_1,...,x_n) \mapsto (y_1,...,y_m) = f(x_1,...,x_n)$. One of the fundamental questions is to know if there exists a classical circuit computing in an automated manner the value of f for every input $(x_1,...,x_n)$. To answer this question, we first need to present some elementary gates. Recall first that $x \oplus y = \mathbb{1}_{\{x \neq y\}}$.

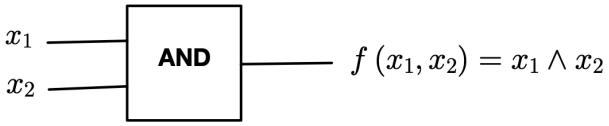
- **NOT** Gate: For this gate, the output is $f(x) = \overline{x} = \mathbb{1}_{\{x=0\}} \left(1 - \mathbb{1}_{\{\overline{x}=1\}}\right)$.



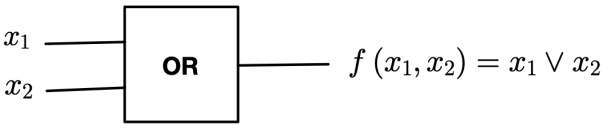
Note that this gate is equivalent to the **XOR** gate, given by a direct sum as the output:



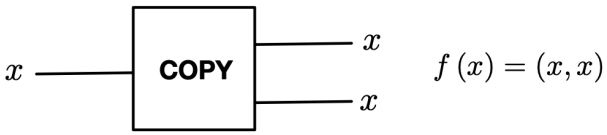
- **AND** Gate: For this gate, the output is $f(x) = x_1 \wedge x_2 = \mathbb{1}_{\{x_1=1 \text{ and } x_2=1\}}$.



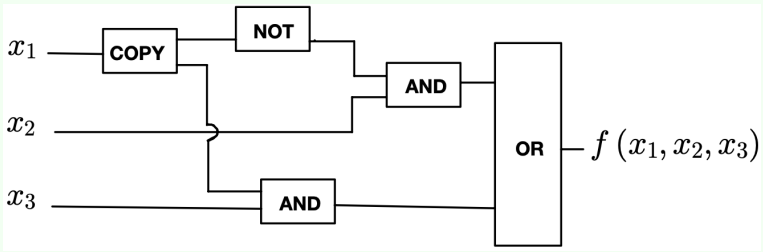
- **OR** Gate: For this gate, the output is $f(x) = x_1 \vee x_2 = \mathbb{1}_{\{x_1=1 \text{ or } x_2=1\}}$. This is the non-exclusive OR.



- **COPY** Gate: Although it is called a 'gate', it is not really one since this operation can be realized by physically joining two wires together.



Example 1.0.1. *The following circuit outputs* $f(x_1,x_2,x_3) = (\overline{x_1} \wedge x_2) \vee (x_1 \wedge x_3)$.



Definition 1.0.2. *A* ***Boolean circuit*** *is a directed (can only go from left to right), acyclic graph with n bits input and m bits output, whose vertices are logic gates and edges are wires.*

We can then answer the question given in introduction via the following theorem.

Theorem 1.0.3 (Emil Post, 1921)**.** *Every Boolean function f can be realized by a Boolean circuit made only of the elementary gates AND, OR, NOT and COPY (universal gates).*

Proof. Let $f : \{0,1\}^n \longrightarrow \{0,1\}^m$ be a Boolean function. First note that in general $f = (f_1, ..., f_m)$, however the theorem needs only to be proven for $m = 1$ as an inductive argument.
Then, consider those vectors $a^{(1)}, ..., a^{(k)} \in \{0,1\}^n$ such that $f\left(a^{(j)}\right) = 1$ for all $1 \leq j \leq k$ and $f(b) = 0$ for all $b \neq a^{(1)}, ..., a^{(k)}$. Furthermore, define $C_a(x) = \mathbb{1}_{\{x=a\}}$, where x and a are vectors. Then:

$$f(x) = C_{a^{(1)}}(x) \vee ... \vee C_{a^{(k)}}(x), \tag{1.1}$$

where the $\vee$ operations correspond to OR operations.
Observe now that for $a \in \{0,1\}^n$, we can write $C_a(x)$ as a series of AND operations:

$$C_a(x) = \phi_{a_1}(x_1) \wedge ... \wedge \phi_{a_n}(x_n), \tag{1.2}$$

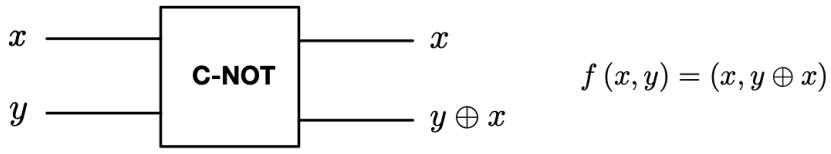
where $\phi_{a_j}(x_j) = x_j$ if $a_j = 1$ and $\phi_{a_j}(x_j) = \overline{x_j}$ if $a_j = 0$. Note that the last condition corresponds to a NOT operation. And so, the computation of $f(x_1, ..., x_n)$ can be realized exclusively with the gates AND, OR, COPY and NOT.

□

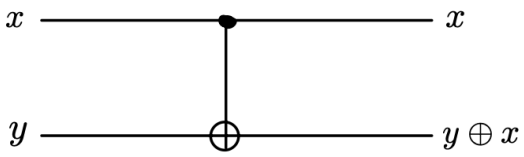
We end this chapter by discussing the irreversibility and reversibility of some gates. The gates AND, OR and COPY are irreversible, in the sense that we cannot recover the inputs from the outputs. However, the gate COPY is logically reversible but its inverse deletes a bit, where physically it dissipates heat. Now, in quantum circuits, irreversible gates are forbidden. Fortunately, the previous gates can be emulated by reversible gates.

• **NOT** Gate: The NOT gate is clearly reversible, since applying it twice recovers the original state.

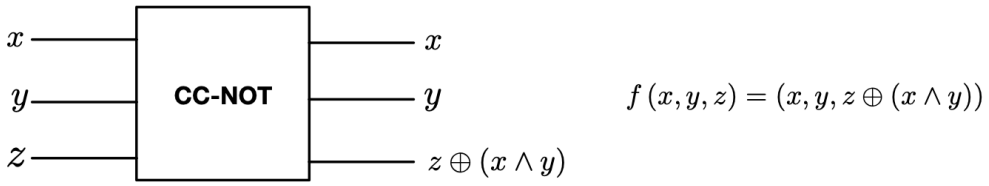
• **C-NOT** Gate: The Controlled-NOT gate has an output given by the XOR operator $f(x,y) = (x, y \oplus x)$, where $f(0,y) = (0,y)$ and $f(1,y) = (0, y \oplus 1) = (1, \overline{y})$. By applying it twice, we note that this gate is reversible.



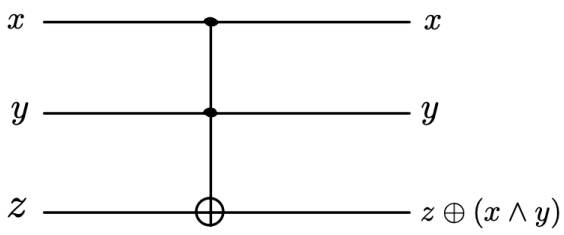
Equivalently, this can be represented as:



• **Toffoli/CC-NOT** Gate: The Controlled-Controlled-NOT gate has an output given by the XOR operator $f(x,y,z) = (x,y,z + (x \oplus y))$, where as long as $x = 0$ or $y = 0$ we have $f(x,y,z) = (x,y,z)$ and $f(1,1,z) = (x,y,z \oplus 1) = (x,y,\overline{z})$. By applying it twice, we note that this gate is reversible.



Equivalently, this can be represented as:

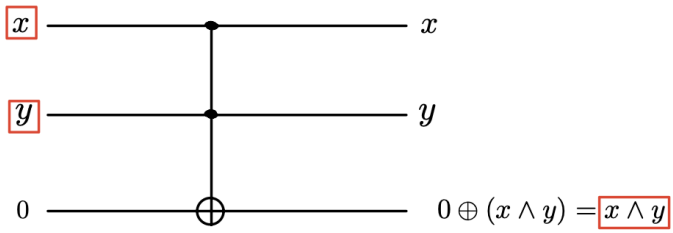


Proposition 1.0.4. *The set of gates {NOT,C-NOT,CC-NOT} is universal.*

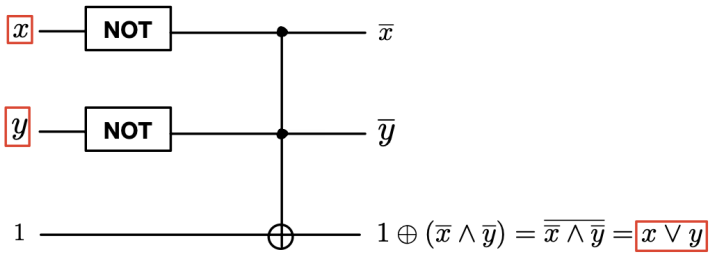
Proof. All gates {AND, OR, NOT, COPY} can be retrieved from the gates {NOT,C-NOT,CC-NOT}.

• **NOT** Gate: For the NOT gate, it is obvious.

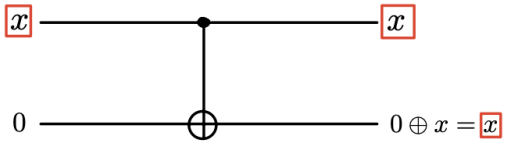
• **AND** Gate: For the AND gate, let $z = 0$ in a CC-NOT gate to find:



• **OR** Gate: For the OR gate, coupling two NOT gates in one CC-NOT gate and using De Morgan's Law in the output state, we find:



• **COPY** Gate: For the COPY gate, let $y = 0$ in a C-NOT gate to find:



As a result of Post's Theorem (1.0.3), the set of gates {NOT,C-NOT,CC-NOT} is universal. Remark that, actually, the NOT and C-NOT gates can themselves be retrieved from CC-NOT gates, but the reverse statement is false.

□

Chapter 2

Fundamentals of Quantum Mechanics and Quantum Circuits

In this chapter we present the fundamental notions of Quantum Mechanics that will be crucial to describe Quantum Circuits. To this end, we will also present how we can use these fundamentals to describe some elementary circuits.

2.1 Dirac's Notation

The state of a quantum system is described by a unit vector in a Hilbert space $\mathcal{H}$ (on $\mathbb{C}$). In this course, we will only consider the finite dimensional Hilbert space $\mathcal{H} = \mathbb{C}^N$ with $N = 2^n$ where n is the number of qubits. In particular, the state of a single qubit is a unit vector in $\mathbb{C}^2$.
The whole idea of quantum computation is to work with qubits in these superposed states in order to perform simultaneous computations.
In Dirac's notation, we consider the ket formulation for a column vector:

$$|\phi\rangle = \begin{pmatrix} \alpha_0 \\ \vdots \\ \alpha_{N-1} \end{pmatrix} \in \mathbb{C}^N, \tag{2.1}$$

and the bra formulation for a row vector:

$$\langle\phi| = \begin{pmatrix} \overline{\alpha}_0 & \cdots & \overline{\alpha}_{N-1} \end{pmatrix}. \tag{2.2}$$

We then define the scalar product between $|\phi\rangle = \begin{pmatrix} \alpha_0 & \cdots & \alpha_{N-1} \end{pmatrix}^T$ and $|\psi\rangle = \begin{pmatrix} \beta_0 & \cdots & \beta_{N-1} \end{pmatrix}^T$ by the braket:

$$\langle\phi|\psi\rangle = \sum_{i=0}^{N-1} \overline{\alpha}_i \beta_i, \tag{2.3}$$

with the associated norm $\||\phi\rangle\| = \sqrt{\langle\phi|\phi\rangle}$. We furthermore present some properties.

- Positivity: $\langle\phi|\phi\rangle = \sum_{i=0}^{N-1} |\alpha_i|^2 \geq 0$.
- Strict Positivity: $\langle\phi|\phi\rangle = 0$ if and only if $|\phi\rangle = \begin{pmatrix} 0 & \cdots & 0 \end{pmatrix}^T$.
- Symmetry: $\langle\psi|\phi\rangle = \sum_{i=0}^{N-1} \overline{\beta}_i \alpha_i = \overline{\sum_{i=0}^{N-1} \overline{\alpha}_i \beta_i} = \overline{\langle\phi|\psi\rangle}$.

- Bilinearity: On the one hand we have:

$$\langle\phi|\left(\alpha|\psi_1\rangle+\beta|\psi_2\rangle\right)=\sum_{i=0}^{N-1}\overline{\alpha}_i\left(\alpha\beta_{1i}+\beta\beta_{2i}\right)=\alpha\sum_{i=0}^{N-1}\overline{\alpha}_i\beta_{1i}+\beta\sum_{i=0}^{N-1}\overline{\alpha}_i\beta_{2i}=\alpha\langle\phi|\psi_1\rangle+\beta\langle\phi|\psi_2\rangle.$$

On the other hand we have:

$$\left(\alpha\langle\phi_1|+\beta\langle\phi_2|\right)|\psi\rangle=\sum_{i=0}^{N-1}\overline{\left(\alpha\alpha_{1i}+\beta\alpha_{2i}\right)}\beta_i=\overline{\alpha}\sum_{i=0}^{N-1}\overline{\alpha}_{1i}\beta_i+\overline{\beta}\sum_{i=0}^{N-1}\overline{\alpha}_{2i}\beta_i=\overline{\alpha}\langle\phi_1|\psi\rangle+\overline{\beta}\langle\phi_2|\psi\rangle.$$

2.2 Computational Basis of $\mathcal{H}=\mathbb{C}^N$

In this section, recall that $N=2^n$. Now, consider the set of vectors e_i for the basis of $\mathcal{H}$:

$$e_i=\begin{pmatrix}0 & \cdots & 0 & 1 & 0 & \cdots & 0\end{pmatrix}=|x_1x_2...x_n\rangle,\qquad 0\leq i\leq N-1 \tag{2.4}$$

where $x_1x_2...x_n$ is the binary representation of i and the 1 is in the i-th position of e_i. Observe that $\langle x_1'...x_n'|x_1...x_n\rangle=\delta_{x_1'x_1}...\delta_{x_n'x_n}$, and hence we have an orthogonal basis. Furthermore, any vector $|\phi\rangle\in\mathbb{C}^N$ can be written as:

$$|\phi\rangle=\sum_{x_1,...,x_n\in\{0,1\}}\alpha_{x_1...x_n}|x_1...x_n\rangle:=\sum_{x\in\{0,1\}^n}\alpha_x|x\rangle \tag{2.5}$$

where $\langle\phi|\phi\rangle=1$ if and only if $\sum_{x_1,...,x_n\in\{0,1\}}|\alpha_{x_1...x_n}|^2=1$.

Example 2.2.1 ($n=1$ - $N=2$)**.** *For $n=1$ we consider $e_0=\begin{pmatrix}1 & 0\end{pmatrix}^T=|0\rangle$ and $e_1=\begin{pmatrix}0 & 1\end{pmatrix}^T=|1\rangle$. Then any vector $|\phi\rangle$ can be written as:*

$$|\phi\rangle=\alpha_0e_0+\alpha_1e_1=\begin{pmatrix}\alpha_0 & \alpha_1\end{pmatrix}^T=\alpha_0|0\rangle+\alpha_1|1\rangle.$$

We have a unit vector if and only if $|\alpha_0|^2+|\alpha_1|^2=1$. Hence any vector can be represented on the unit circle with the $|0\rangle$ vector on the x-axis and the $|1\rangle$ vector on the y-axis.

Example 2.2.2 ($n=2$ - $N=4$)**.** *For $n=2$ we consider $e_0=\begin{pmatrix}1 & 0 & 0 & 0\end{pmatrix}^T=|00\rangle$, $e_1=\begin{pmatrix}0 & 1 & 0 & 0\end{pmatrix}^T=|01\rangle$, $e_2=\begin{pmatrix}0 & 0 & 1 & 0\end{pmatrix}^T=|10\rangle$ and $e_3=\begin{pmatrix}0 & 0 & 0 & 1\end{pmatrix}^T=|11\rangle$, where each ket vector has been written with the binary representation of i (e.g. 01 is the binary representation of $i=1$). Then any vector $|\phi\rangle$ can be written as:*

$$|\phi\rangle=\alpha_{00}|00\rangle+\alpha_{01}|01\rangle+\alpha_{10}|10\rangle+\alpha_{11}|11\rangle.$$

We have a unit vector if and only if $|\alpha_{00}|^2+|\alpha_{01}|^2+|\alpha_{10}|^2+|\alpha_{11}|^2=1$.

2.3 Tensor Product

Let $\mathcal{H}_1=\mathbb{C}^{2^{n_1}}$ be a Hilbert space for n_1 qubits and $\mathcal{H}_2=\mathbb{C}^{2^{n_2}}$ be a Hilbert space for n_2 qubits. Now consider $\mathcal{H}=\mathcal{H}_1\otimes\mathcal{H}_2=\mathbb{C}^{2^{n_1}}\otimes\mathbb{C}^{2^{n_2}}\cong\mathbb{C}^{2^{n_1+n_2}}$ (isomorphic). Here, $\mathcal{H}$ is a vector space of dimension

$2^{n_1+n_2}$ spanned by all basis elements $|x,y\rangle = |x\rangle \otimes |y\rangle$. For all $|\phi\rangle \in \mathcal{H}$ it holds that:

$$|\phi\rangle = \sum_{\substack{0\leq x\leq 2^{n_1}-1 \\ 0\leq y\leq 2^{n_2}-1}} \alpha_{x,y} |x,y\rangle , \tag{2.6}$$

and in particular we have a unit vector if and only if $\sum_{x,y}|\alpha_{x,y}|^2 = 1$.
We now make an important remark: every element $|\phi\rangle$ in $\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2$ can be written as a linear combination of the basis elements $|x,y\rangle$, however **not** every element $|\phi\rangle$ in $\mathcal{H}$ can be written in the product form $|\phi_1\rangle \otimes |\phi_2\rangle$ (those are called product states).
In $\mathcal{H}_1 \otimes \mathcal{H}_2$ the conjugation operation yields:

$$\overline{|\phi_1\rangle \otimes |\phi_2\rangle} = \langle\phi_1| \otimes \langle\phi_2| . \tag{2.7}$$

For the scalar product we obtain:

$$(\langle\phi_1| \otimes \langle\phi_2|)(|\psi_1\rangle \otimes |\psi_2\rangle) = \langle\phi_1|\psi_1\rangle \langle\phi_2|\psi_2\rangle . \tag{2.8}$$

Hence, $\langle x',y'|x,y\rangle = \langle x'|x\rangle \langle y'|y\rangle = \delta_{x'x}\delta_{y'y}$.

Example 2.3.1. *Consider $\mathcal{H}_1 = \mathbb{C}^2$ and $\mathcal{H}_2 = \mathbb{C}^2$, then $\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2 \cong \mathbb{C}^4$. Furthermore, let $|\phi_1\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle \in \mathcal{H}_1$ and $|\phi_2\rangle = \beta_0|0\rangle + \beta_1|1\rangle \in \mathcal{H}_2$. Hence in $\mathcal{H}$ we have:*

$$|\phi_1\rangle \otimes |\phi_2\rangle = \alpha_0\beta_0|0,0\rangle + \alpha_0\beta_1|0,1\rangle + \alpha_1\beta_0|1,0\rangle + \alpha_1\beta_1|1,1\rangle .$$

Explicitly, the binary representations are:

$$|0,0\rangle = |0\rangle \otimes |0\rangle = \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}1 & 0 & 0 & 0\end{pmatrix}^T = e_0,$$
$$|0,1\rangle = |0\rangle \otimes |1\rangle = \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0 & 1 & 0 & 0\end{pmatrix}^T = e_1,$$
$$|1,0\rangle = |1\rangle \otimes |0\rangle = \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0 & 0 & 1 & 0\end{pmatrix}^T = e_2,$$
$$|1,1\rangle = |1\rangle \otimes |1\rangle = \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0 & 0 & 0 & 1\end{pmatrix}^T = e_3.$$

2.4 Axioms of Quantum Mechanics

2.4.1 Axiom 1 - State of a Quantum System

The state of a quantum system (isolated from the environment) is represented by a unit vector $|\phi\rangle$ in a Hilbert space $\mathcal{H}$. In particular, the state of a system of n qubits is represented by a unit vector in $\mathcal{H} = \mathbb{C}^{2^n} \cong \bigotimes^{n \text{ times}} \mathbb{C}^2$. Note that in general, for n qubits, $\||\phi\rangle\|^2 \neq n$.

Now, in the computational basis, we consider a set of basis vectors in the form:

$$\Big\{ |x_1,...,x_n\rangle, x_i \in \{0,1\}, 1 \leq i \leq n \Big\}, \tag{2.9}$$

where each ket vector satisfies:

$$\langle x'_1,...,x'_n | x_1,...,x_n \rangle = \delta_{x'_1 x_1} ... \delta_{x'_n x_n}. \tag{2.10}$$

Then, every vector in the Hilbert space can be written as:

$$|\phi\rangle = \sum_{x_1,...,x_n \in \{0,1\}} \alpha_{x_1,...,x_n} |x_1,...,x_n\rangle, \tag{2.11}$$

where we naturally impose a normalizing condition given by:

$$1 = \langle \phi | \phi \rangle = \sum_{x_1,...,x_n \in \{0,1\}} |\alpha_{x_1,...,x_n}|^2. \tag{2.12}$$

Example 2.4.1 ($n = 1$)**.** *We can represent vectors in* $\mathbb{C}^2$ *on the unit circle:*

$$|\phi\rangle = \cos(\theta)|0\rangle + \sin(\theta)|1\rangle, \tag{2.13}$$

where of course $\cos(\theta)^2 + \sin(\theta)^2 = 1$*. We can note two particular cases of states that will be important for later:*

$$|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \quad \textit{and,} \quad |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle). \tag{2.14}$$

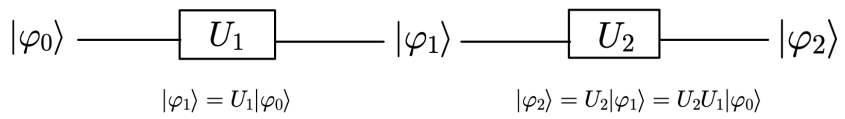
These are vectors at $\theta = +45^\circ$ *angle for* $|+\rangle$ *and at* $\theta = -45^\circ$ *angle for* $|-\rangle$ *on the unit circle.*

2.4.2 Axiom 2 - Time Evolution

An isolated quantum system evolves in time via unitary linear transformations U:

$$|\phi\rangle \longrightarrow U|\phi\rangle, \tag{2.15}$$

where U is a $2^n \times 2^n$ unitary matrix, hence satisfying $UU^\dagger = U^\dagger U = \mathbb{I}$ (i.e. $U^{-1} = U^\dagger$) for $U^\dagger$ being the adjoint of U (complex-conjugate transpose). In a quantum circuit, considering two evolutions gives:



We have norm conservation:

$$\langle \phi_1 | \phi_1 \rangle = \langle \phi_0 | U_1^\dagger U_1 | \phi_0 \rangle = \langle \phi_0 | \mathbb{I} | \phi_0 \rangle = \langle \phi_0 | \phi_0 \rangle = 1, \tag{2.16}$$

and similarly:

$$\langle \phi_2 | \phi_2 \rangle = \langle \phi_1 | U_2^\dagger U_2 | \phi_1 \rangle = \langle \phi_1 | \mathbb{I} | \phi_1 \rangle = \langle \phi_1 | \phi_1 \rangle = 1. \tag{2.17}$$

In words, $U = U_2U_1$ is also a unitary transformation, where more formally one can check that $UU^\dagger = U_2U_1U_1^\dagger U_2^\dagger = U_2U_2^\dagger = \mathbb{I}$. More generally, it is always the case that any quantum circuit can be represented by a single unitary transformation U.

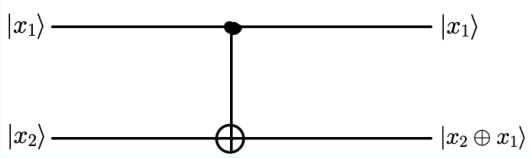
Example 2.4.2 (NOT Gate)**.** *The NOT gate acts on a single qubit in* $\mathbb{C}^2$*.*

The action of this gate on basis states is given by $NOT|0\rangle = |1\rangle$ *and* $NOT|1\rangle = |0\rangle$*. Then it follows that* $NOT(\alpha_0|0\rangle + \alpha_1|1\rangle) = \alpha_0|1\rangle + \alpha_1|0\rangle$ *(i.e. reflection with respect to the axis with angle* $45°$*). Let's now find the matrix representation of the NOT gate, where its matrix elements are:* $\langle 0|NOT|0\rangle = \langle 0|1\rangle = 0$*,* $\langle 0|NOT|1\rangle = \langle 0|0\rangle = 1$*,* $\langle 1|NOT|0\rangle = \langle 1|1\rangle = 1$ *and,* $\langle 1|NOT|1\rangle = \langle 1|0\rangle = 0$*. Therefore the matrix is:*

$$NOT = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = NOT^\dagger \Longrightarrow \textit{Hermitian.} \tag{2.18}$$

Furthermore, $NOT \cdot NOT^\dagger = NOT^\dagger \cdot NOT = \mathbb{I}$*, hence it is also unitary. We finally note that, with a simple calculation,* $NOT|+\rangle = |+\rangle$ *and* $NOT|-\rangle = -|-\rangle$*.*

Example 2.4.3 (C-NOT Gate)**.** *The C-NOT gate acts on two qubits in* $\mathbb{C}^2 \otimes \mathbb{C}^2 \cong \mathbb{C}^4$*.*



The action of this gate on basis states is given by $CNOT|00\rangle = |00\rangle$*,* $CNOT|01\rangle = |01\rangle$*,* $CNOT|10\rangle = |11\rangle$ *and* $CNOT|11\rangle = |10\rangle$*. Of course, this can be generalized as* $CNOT|x_1, x_2\rangle = |x_1, x_2 \oplus x_1\rangle$*. The matrix representation of the C-NOT gate in the basis* $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ *is given by:*

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} = CNOT^\dagger \Longrightarrow \textit{Hermitian.} \tag{2.19}$$

Furthermore, $CNOT \cdot CNOT^\dagger = CNOT^\dagger \cdot CNOT = \mathbb{I}$*, hence it is also unitary. We finally note that, with a simple calculation, for a state* $|\phi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$ *it holds that:* $CNOT|\phi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|11\rangle + \alpha_{11}|10\rangle$*.*

Remark 2.4.4. *Classically, a CNOT gate can emulate a COPY gate. But in the quantum world, copying a quantum state is impossible (**No Cloning Theorem**). Let us solve this apparent contradiction. To this end, consider* $|\phi\rangle \otimes |0\rangle$ *as input state to the CNOT gate, with* $|\phi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$*:*

$$
\begin{aligned}
CNOT(|\phi\rangle \otimes |0\rangle) &= CNOT\left((\alpha_0 |0\rangle + \alpha_1 |1\rangle) \otimes |0\rangle\right) \\
&= \alpha_0 CNOT\,|00\rangle + \alpha_1 CNOT\,|10\rangle \\
&= \alpha_0 |00\rangle + \alpha_1 |11\rangle = \text{Bell State} \neq |\phi\rangle \otimes |\phi\rangle .
\end{aligned}
$$

Only states in the computational basis can be copied.

2.4.3 Axiom 3 - Measurement Postulate

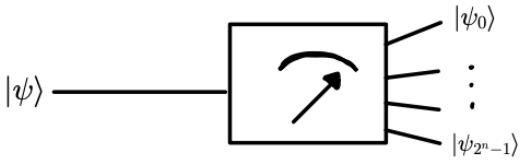
If an isolated quantum system is in state $|\psi\rangle \in \mathcal{H} = \mathbb{C}^{2^n}$ and one observes the system through a measurement equipment, described by an orthonormal basis $\{|\phi_0\rangle, |\phi_1\rangle, ..., |\phi_{2^n-1}\rangle\}$ of $\mathcal{H}$ (considering the computational basis), then the outcome of the measurement is given by $\{|\phi_i\rangle\}_{i=0}^{2^n-1}$ with probability $\mathbb{P}(i) = |\langle\phi_i|\psi\rangle|^2$. Note that:

$$
\sum_{i=0}^{2^n-1} \mathbb{P}(i) = \sum_{i=0}^{2^n-1} \overline{\langle\phi_i|\psi\rangle}\,\langle\phi_i|\psi\rangle = \sum_{i=0}^{2^n-1} \langle\psi|\phi_i\rangle\,\langle\phi_i|\psi\rangle = \langle\psi| \left(\sum_{i=0}^{2^n-1} |\phi_i\rangle\,\langle\phi_i| \right) |\psi\rangle = \langle\psi|\mathbb{I}|\psi\rangle = \langle\psi|\psi\rangle = 1.
$$

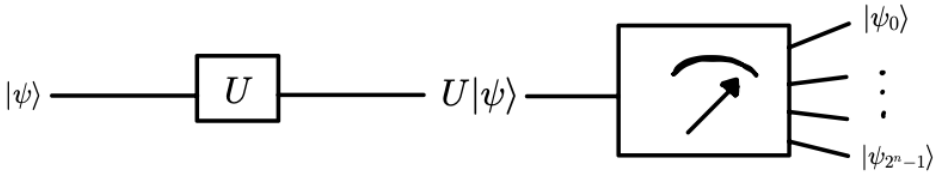
Observe that:

$$
|\phi_i\rangle\,\langle\phi_i| = \begin{pmatrix} 0 & & & & & \\ & \ddots & & & & \\ & & 0 & & 0 & \\ & & & 1 & & \\ & 0 & & & 0 & \\ & & & & & \ddots \\ & & & & & & 0 \end{pmatrix} \tag{2.20}
$$

where 1 is in the i-th row and i-th column, and this matrix is a rank-one matrix which is also a projector matrix (on $|\phi_i\rangle$). The graphical representation of a measurement is given by:



With the addition of a quantum circuit U, the probability is $\mathbb{P}(i) = |\langle\phi_i|\,U\,|\psi\rangle|^2$ and we have:



2.4.4 Axiom 4 - Composition of Quantum Systems

Consider two systems, one with n_1 qubits defined on a Hilbert space $\mathcal{H}_1 = (\mathbb{C}^2)^{\otimes n_1}$ (with dimension 2^{n_1}) and the other one with n_2 qubits defined on a Hilbert space $\mathcal{H}_2 = (\mathbb{C}^2)^{\otimes n_2}$ (with dimension 2^{n_2}). Then, we can set a global system consisting of $n_1 + n_2$ qubits defined on a Hilbert space $\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2 = (\mathbb{C}^2)^{\otimes (n_1+n_2)}$ (with dimension $2^{n_1+n_2}$). Now, not all states can be written as product states: in the form $|\phi_1\rangle \otimes |\phi_2\rangle$.

Example 2.4.5 (Product States)**.** *All the following are product states:*

- $|0,0\rangle = |0\rangle \otimes |0\rangle$.
- $\frac{1}{\sqrt{2}}(|0,1\rangle + |0,0\rangle) = |0\rangle \otimes \left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\right)$.
- $\frac{1}{2}(|0,0\rangle + |0,1\rangle + |1,0\rangle + |1,1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$.

If we don't have a product state we have an entangled state.

Example 2.4.6 (Entangled State)**.** *The Bell State is an entangled state, since:*

$$\frac{1}{\sqrt{2}}(|0,0\rangle + |1,1\rangle) \neq |\phi_1\rangle \otimes |\phi_2\rangle\,. \tag{2.21}$$

Proposition 2.4.7. *The state $|\phi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$ is a product state if and only if:*

$$det\begin{pmatrix} \alpha_{00} & \alpha_{01} \\ \alpha_{10} & \alpha_{11} \end{pmatrix} = 0. \tag{2.22}$$

2.5 Quantum Circuits

In this section, we present fundamental quantum gates with 1-qubit, 2-qubits and multiple qubits entries. Recall that a quantum circuit operating on n qubits can always be represented by a $2^n \times 2^n$ unitary matrix U.

2.5.1 1-Qubit Gates

Consider the Hilbert space $\mathcal{H} = \mathbb{C}^2$.

Definition 2.5.1. *The **NOT gate** is defined as:*

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}. \tag{2.23}$$

Definition 2.5.2. *The **Hadamard gate** is defined as:*

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{2.24}$$

Let's now observe the action of H on basis states $|0\rangle$ and $|1\rangle$. For vector $|0\rangle$:

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle, \tag{2.25}$$

and for vector $|1\rangle$:

$$H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |-\rangle. \tag{2.26}$$

And of course, with a simple calculation, for any state $|\phi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ it holds that:

$$H|\phi\rangle = \alpha_0|+\rangle + \alpha_1|-\rangle = \frac{\alpha_0+\alpha_1}{\sqrt{2}}|0\rangle + \frac{\alpha_0-\alpha_1}{\sqrt{2}}|1\rangle. \tag{2.27}$$

Finally note that H is Hermitian and unitary, since $H = H^\dagger$ and $HH^\dagger = H^\dagger H = \mathbb{I}$.

Definition 2.5.3. *The **Phase gates** Z, S and T are defined as:*

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad S = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{2}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \quad \text{and,} \quad T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{\sqrt{2}} + \frac{i}{\sqrt{2}} \end{pmatrix}.$$

Let's again observe the actions of Z, S and T on basis states $|0\rangle$ and $|1\rangle$. For vector $|0\rangle$:

$$Z|0\rangle = S|0\rangle = T|0\rangle = |0\rangle, \tag{2.28}$$

and for vector $|1\rangle$:

$$Z|1\rangle = -|1\rangle, \; S|1\rangle = i|1\rangle \;\text{ and }\; T|1\rangle = e^{i\frac{\pi}{4}}|1\rangle. \tag{2.29}$$

Of course, with a simple calculation, for any state $|\phi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ it holds that:

$$Z|\phi\rangle = \alpha_0|0\rangle - \alpha_1|1\rangle, \; S|\phi\rangle = \alpha_0|0\rangle + i\alpha_1|1\rangle \;\text{ and }\; T|\phi\rangle = \alpha_0|0\rangle + e^{i\frac{\pi}{4}}\alpha_1|1\rangle. \tag{2.30}$$

Finally note that Z, S and T are all unitary. Also remark that $Z = S^2 = T^4$ and $S = T^2$.

Theorem 2.5.4 (Without proof)**.** *Any 2×2 unitary matrix U can be approximated by a product of gates H, S and T in the following sense: for all $\delta > 0$, there exists V a product of $\mathcal{O}\left(\frac{1}{\delta}\right)$ matrices H, S and T such that $\|U - V\| < \delta$ (where $\|\cdot\|$ is some matrix norm).*

2.5.2 2-Qubits Gates

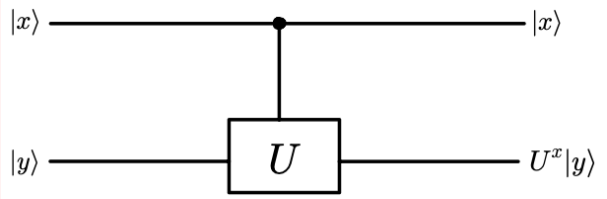
Consider the Hilbert space $\mathcal{H} = \mathbb{C}^4$ and the basis $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$.

Definition 2.5.5. *The **CNOT gate** is defined as:*

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{2.31}$$

We have already studied the action of the CNOT gate on basis states in Example (2.4.3).

Definition 2.5.6. *The **Controlled-U gate** is defined as:*

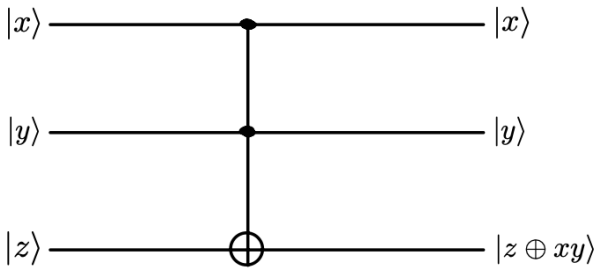


Here, U is a 2×2 unitary matrix and the output $U^x|y\rangle$ is given by:

$$U^x|y\rangle = \begin{cases} |y\rangle, & \text{if } x=0 \\ U|y\rangle, & \text{if } x=1 \end{cases} . \tag{2.32}$$

2.5.3 Multiple Qubits Gates

• We have already encountered the CCNOT gate defined on the Hilbert space $\mathcal{H}=\mathbb{C}^8$ and in the basis $\{|000\rangle, |001\rangle, |010\rangle, |011\rangle, |100\rangle, |101\rangle, |110\rangle, |111\rangle\}$.

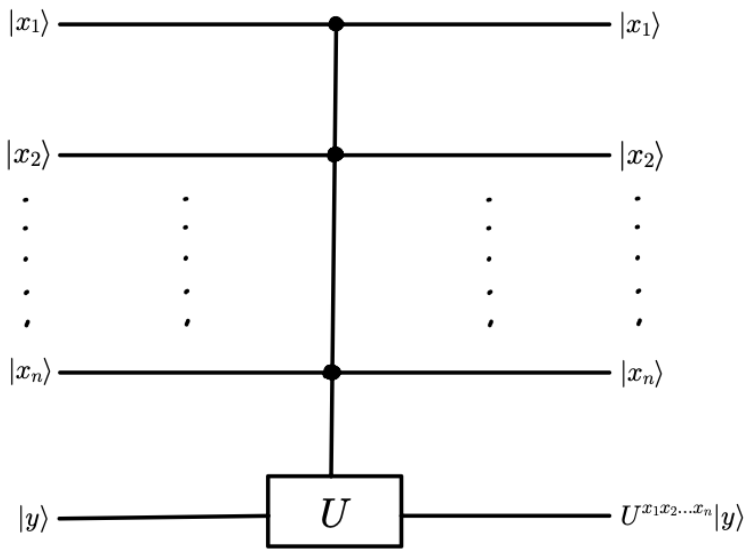


Definition 2.5.7. *The **CCNOT gate** is defined as:*

$$CCNOT = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} . \tag{2.33}$$

Remark 2.5.8. *Classically, it is not possible to create a Toffoli gate from CNOT and 1-bit gates. In the quantum world however, this is possible using the gates CNOT, H, T and S gates.*

• We can also consider multicontrol gates, where we now have to use a general Hilbert space $\mathcal{H}=\mathbb{C}^{2^{n+1}}$. In particular, we can build a circuit with a U gate where U acts on $|y\rangle$ only if $x_1=x_2=...=x_n=1$.



Theorem 2.5.9 (A.Barenco & al. - Without proof)**.** *Any $2^n \times 2^n$ unitary matrix U can be approximated (with arbitrary precision) by a circuit made only of gates T, S, H and CNOT. The number of gates needed for this approximation depends on the unitary matrix U (may be exponential in n).*

Remark 2.5.10. *Without the T gate, it can be shown that no quantum advantage can be obtained over classical circuits (Gottesman-Knill Theorem).*

Chapter 3

Deutsch's Model and a Quantum Algorithm

In this chapter, we present the model of Deutsch along with the so-called Deutsch's problem. We will then present a classical method of resolution, before moving on to the quantum algorithm called the Deutsch-Josza's algorithm.

3.1 Deutsch's Model of Quantum Circuits

As already mentioned, every circuit can be represented by a single unitary operation U and the extraction of information happens via a measurement in $\{|x_1...x_n\rangle, x_1, ..., x_n \in \{0,1\}\}$ with probability $\mathbb{P}(|x_1...x_n\rangle) = |\langle x_1...x_n|\psi_{\text{out}}\rangle|^2$. We can now mention two reasons for using a quantum circuit rather than a classical one:
(i) Simulate quantum physical systems.
(ii) Solve efficiently classical problems involving a Boolean function $f : \{0,1\}^n \longrightarrow \{0,1\}^m$.
Our aim will be to solve problems involving a Boolean function, and this will be done in 3 generic stages:
1) Any input of $f : \{0,1\}^n \longrightarrow \{0,1\}^m$ is a sequence of n bits $\{x_1, ..., x_n\}$, which can be encoded into a quantum state $|x_1...x_n\rangle$. We will construct superpositions of states given by:

$$|\psi\rangle = \sum_{x_1,...,x_n \in \{0,1\}} \alpha_{x_1...x_n} |x_1...x_n\rangle . \tag{3.1}$$

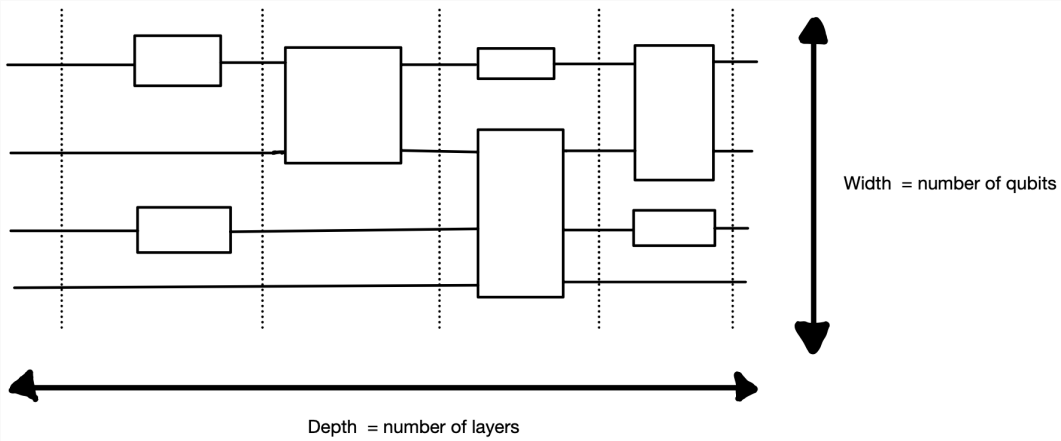
2) We consider a unitary operation $U^{(f)}$ performed on $|\psi\rangle$, where by linearity:

$$U^{(f)}|\psi\rangle = \sum_{x_1...x_n \in \{0,1\}} \alpha_{x_1...x_n} U^{(f)}|\psi\rangle . \tag{3.2}$$

3) For the measurement, the outcome $|x_1...x_n\rangle$ with probability $|\langle x_1...x_n|U^{(f)}|\psi\rangle|^2$ should be high (or at least > 0) for states $|x_1...x_n\rangle$ corresponding to the solution of the problem.

In what follows, we consider two assumptions without loss of generality. Note that the following assumptions may come with some additional cost on the circuit complexity:
- Initial state is $|0, 0, ..., 0\rangle$.
- The final measurement is performed in the computational basis $\{|x_1...x_n\rangle, x_1, ..., x_n \in \{0,1\}\}$.

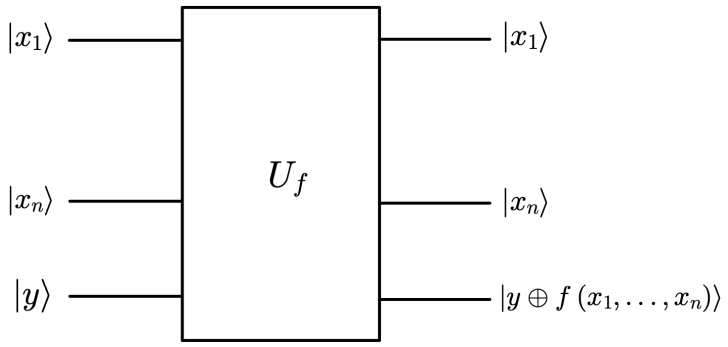
Remark 3.1.1. *The circuit complexity is set to be $C = width \times depth$.*



Before we proceed to the study of the quantum algorithm, let us introduce the quantum oracle gate U_f associated to a Boolean function $f : \{0,1\}^n \longrightarrow \{0,1\}$ (we consider the case $m = 1$ but this can be generalized). Observe first that unless $n = 1$ and f is bijective, the evaluation of a Boolean function f is in general irreversible. A reversible way of evaluating a function f is obtained by augmenting the memory with an ancilla bit:

$$\widetilde{f}(x_1, ..., x_n, y) = (x_1, ..., x_n, y \oplus f(x_1, ..., x_n)). \tag{3.3}$$

The corresponding quantum circuit, which roughly corresponds to a generalization of a Controlled-U gate, is given by:



Definition 3.1.2. *The quantum* ***oracle****, which has to be constructed for every f, acts as:*

$$U_f(|x_1...x_n\rangle \otimes |y\rangle) = |x_1...x_n\rangle \otimes |y \oplus f(x_1, ..., x_n)\rangle\,. \tag{3.4}$$

Proposition 3.1.3. *U_f is unitary.*

Proof. For all basis elements and for every f, we have:

$$\begin{aligned}\langle x'_1...x'_n| \otimes \langle y'| U_f^\dagger U_f |x_1...x_n\rangle \otimes |y\rangle &= \Big(\langle x'_1...x'_n| \otimes \langle y' \oplus f(x'_1,...,x'_n)|\Big)\Big(|x_1...x_n\rangle \otimes |y \oplus f(x_1,...,x_n)\rangle\Big) \\ &= \langle x'_1|x_1\rangle ... \langle x'_n|x_n\rangle \langle y' \oplus f(x'_1,...,x'_n)|y \oplus f(x_1,...,x_n)\rangle \\ &= \delta_{x'_1x_1}...\delta_{x'_nx_n} \langle y' \oplus f(x'_1,...,x'_n)|y \oplus f(x_1,...,x_n)\rangle \\ &= \delta_{x'_1x_1}...\delta_{x'_nx_n} \langle y' \oplus f(x_1,...,x_n)|y \oplus f(x_1,...,x_n)\rangle \\ &= \delta_{x'_1x_1}...\delta_{x'_nx_n}\delta_{y'y} = \mathbb{1}_{\{x'_1=x_1,...,x'_n=x_n,y'=y\}}.\end{aligned}$$

□

3.2 Deutsch's Problem and Classical Method of Resolution

We are given a Boolean function $f : \{0,1\}^n \longrightarrow \{0,1\}$ and an oracle capable of evaluating $f(x)$ for a given x at no cost. On top of that, we are informed that either f is constant (i.e. $f(x) = f(y)$ for all $x, y \in \{0,1\}^n$) or f is balanced (i.e. $f(x) = 1$ for half of the x's and $f(x) = 0$ for the other half). The **aim** of the problem is to decide between these two alternatives with the least possible number of calls to the oracle. Note that we do not know anything a priori about the structure of f.

Classical Method of Resolution
Call the oracle in k different points $x^{(1)},...,x^{(k)} \in \{0,1\}^n$:
- If $f\left(x^{(1)}\right) = ... = f\left(x^{(k)}\right)$ declare f is constant.
- Otherwise, declare f is balanced.

In the worst case, $k = 2^{n-1} + 1$ calls to the oracle are needed in order to obtain a correct answer with probability $p = 1$ (since k is greater than half the total number of points).

Probabilistic Algorithm (still classical)
Fix $k \geq 1$ and draw k iid points $x^{(1)},...,x^{(k)} \in \{0,1\}^n$ (with possible replacement). Again:
- If $f\left(x^{(1)}\right) = ... = f\left(x^{(k)}\right)$ declare f is constant.
- Otherwise, declare f is balanced.

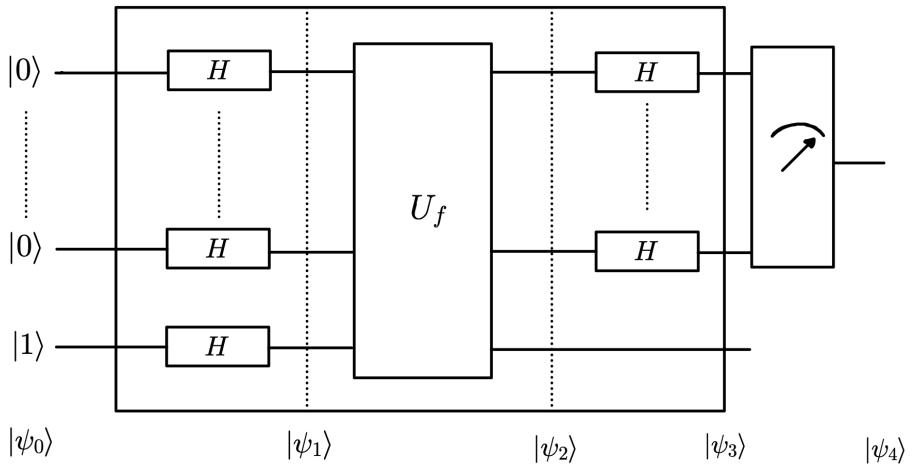
The probability of making an error, which can only happen in the first case, is $\frac{1}{2^{k-1}}$. This can be made as small as required in $\mathcal{O}(1)$ calls.

3.3 Deutsch-Josza's Quantum Algorithm

We now present the quantum alternative to Deutsch's algorithm, where we will make use of the Hadamard gate H, the quantum oracle U_f and finally making a measurement. We start with an initial state:

$$|\psi_0\rangle = \left(\bigotimes^{n \text{ times}} |0\rangle\right) \otimes |1\rangle := |0,0,...,0\rangle \otimes |1\rangle\,. \tag{3.5}$$

An extra ancilla qubit $|1\rangle$ is being added to the input to allow for computations later on.



Stage 1: State $|\psi_1\rangle$ corresponds to a superposition of states:

$$|\psi_1\rangle = \overset{(n+1) \text{ times}}{\bigotimes} H\,|\psi_0\rangle = H\,|0\rangle \otimes ... \otimes H\,|0\rangle \otimes H\,|1\rangle\,. \tag{3.6}$$

Now, recall that $H\,|0\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}} = \frac{1}{\sqrt{2}}\sum_{x_1\in\{0,1\}}|x_1\rangle$ and $H\,|1\rangle = \frac{|0\rangle-|1\rangle}{\sqrt{2}}$. Then we find:

$$|\psi_1\rangle = \frac{1}{\sqrt{2}}\sum_{x_1\in\{0,1\}}|x_1\rangle \otimes ... \otimes \frac{1}{\sqrt{2}}\sum_{x_n\in\{0,1\}}|x_n\rangle \otimes \frac{|0\rangle-|1\rangle}{\sqrt{2}} = \frac{1}{2^{\frac{n}{2}}}\sum_{x_1,...,x_n\in\{0,1\}}|x_1,...,x_n\rangle \otimes \frac{|0\rangle-|1\rangle}{\sqrt{2}}.$$

Stage 2: To find $|\psi_2\rangle$, state $|\psi_1\rangle$ has to go through the quantum oracle. Recall that:

$$U_f(|x_1...x_n\rangle \otimes |y\rangle) = |x_1...x_n\rangle \otimes |y \oplus f(x_1,...,x_n)\rangle\,. \tag{3.7}$$

Then:

$$\begin{aligned}
|\psi_2\rangle = U_f\,|\psi_1\rangle &= \frac{1}{2^{\frac{n}{2}}}\sum_{x_1,...,x_n\in\{0,1\}} U_f\left(|x_1,...,x_n\rangle \otimes \frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) \\
&= \frac{1}{2^{\frac{n}{2}}}\sum_{x_1,...,x_n\in\{0,1\}} |x_1,...,x_n\rangle \otimes \frac{|f(x_1,...,x_n)\rangle - \left|\overline{f(x_1,...,x_n)}\right\rangle}{\sqrt{2}} \\
&= \frac{1}{2^{\frac{n}{2}}}\sum_{x_1,...,x_n\in\{0,1\}} \begin{cases} |x_1,...,x_n\rangle \otimes \frac{|0\rangle-|1\rangle}{\sqrt{2}} & \text{if } f(x_1,...,x_n)=0 \\ |x_1,...,x_n\rangle \otimes \frac{|1\rangle-|0\rangle}{\sqrt{2}} & \text{if } f(x_1,...,x_n)=1 \end{cases} \\
&= \frac{1}{2^{\frac{n}{2}}}\sum_{x_1,...,x_n\in\{0,1\}} |x_1,...,x_n\rangle \otimes (-1)^{f(x_1,...,x_n)}\,.\,\frac{|0\rangle-|1\rangle}{\sqrt{2}} \\
&= \frac{1}{2^{\frac{n}{2}}}\sum_{x_1,...,x_n\in\{0,1\}} (-1)^{f(x_1,...,x_n)}\,|x_1,...,x_n\rangle \otimes \frac{|0\rangle-|1\rangle}{\sqrt{2}}.
\end{aligned}$$

The action of U_f on the ancilla qubit, which is in a superposition state, has now been transferred to the first n qubits. Note that, from now on, we could forget the ancilla qubit.

Stage 3: We are at the stage of the analysis of the output qubits. Set $\bigotimes^{n \text{ times}} H := H^{\otimes n}$. Then:

$$|\psi_3\rangle = \left(H^{\otimes n}\otimes \mathbb{I}\right)|\psi_2\rangle = \frac{1}{2^{\frac{n}{2}}}\sum_{x_1,...,x_n\in\{0,1\}} (-1)^{f(x_1,...,x_n)}\underbrace{H^{\otimes n}\,|x_1,...,x_n\rangle}_{(*)} \otimes \frac{|0\rangle-|1\rangle}{\sqrt{2}}. \tag{3.8}$$

Now, it holds that $H|x_1\rangle = \frac{|0\rangle+(-1)^{x_1}|1\rangle}{\sqrt{2}} = \frac{1}{\sqrt{2}}\sum_{z_1\in\{0,1\}}(-1)^{z_1x_1}|z_1\rangle$. Therefore:

$$(*) = H^{\otimes n}|x_1,...,x_n\rangle = H|x_1\rangle\otimes...\otimes H|x_n\rangle = \frac{1}{2^{\frac{n}{2}}}\sum_{z_1,...,z_n\in\{0,1\}}(-1)^{z_1x_1+...+z_nx_n}|z_1,...,z_n\rangle. \tag{3.9}$$

Hence, gathering everything together yields:

$$\begin{aligned}|\psi_3\rangle &= \frac{1}{2^{\frac{n}{2}}}\sum_{x_1,...,x_n\in\{0,1\}}(-1)^{f(x_1,...,x_n)}\left(\frac{1}{2^{\frac{n}{2}}}\sum_{z_1,...,z_n\in\{0,1\}}(-1)^{z_1x_1+...+z_nx_n}|z_1,...,z_n\rangle\right)\otimes\frac{|0\rangle-|1\rangle}{\sqrt{2}}\\ &= \sum_{z_1,...,z_n\in\{0,1\}}\underbrace{\left(\frac{1}{2^n}\sum_{x_1,...,x_n\in\{0,1\}}(-1)^{f(x_1,...,x_n)+z_1x_1+...+z_nx_n}\right)}_{:=\alpha_{z_1,...,z_n}}|z_1,...,z_n\rangle\otimes\frac{|0\rangle-|1\rangle}{\sqrt{2}}.\end{aligned}$$

Stage 4: This last stage is dedicated to the measurement of the first n qubits. The state $|z_1,...,z_n\rangle$ is observed with probability $|\alpha_{z_1,...,z_n}|^2$. Let us consider the particular state $|00...0\rangle$, then:

$$|\alpha_{00...0}|^2 = \left|\frac{1}{2^n}\sum_{x_1,...,x_n\in\{0,1\}}(-1)^{f(x_1,...,x_n)}\right|^2 = \begin{cases}1 & \text{if } f \text{ is constant}\\ 0 & \text{if } f \text{ is balanced}\end{cases}. \tag{3.10}$$

Hence if the output is $|00...0\rangle$, f is constant, otherwise f is balanced. Note that this operation has been made with a single call to the quantum oracle.

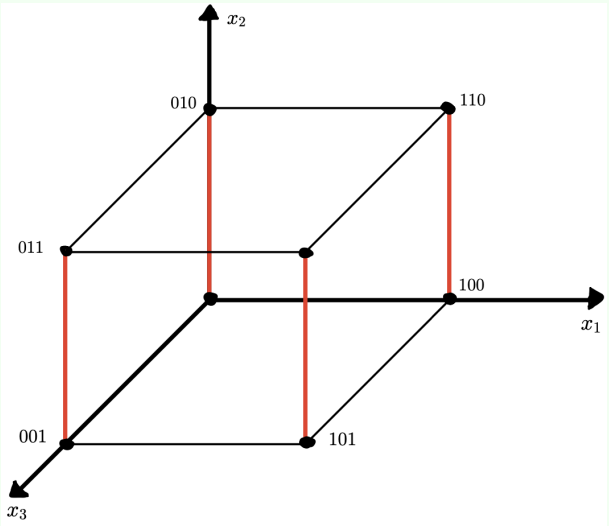
Remark 3.3.1. *In an actual quantum computer, there is noise, so the probability of a correct answer is never $p=1$. Furthermore, this problem is a toy problem since the full knowledge of f is required to build the gate U_f.*

Chapter 4

Simon's Algorithm

Let $f : \{0,1\}^n \longrightarrow X$ be a function such that $f(x) = f(y)$ if and only if either $x = y$ or $x \oplus a = y$ for some $a \in \{0,1\}^n \backslash \{0\}$. Note that X will be defined later and a is unknown.
The **aim** is to discover the value of $a \neq 0$ by asking as few questions as possible to the oracle f. Classically, we will see that this requires $\mathcal{O}(2^n)$ calls, whereas Simon's quantum algorithm finds the vector a with probability $p \geq 1 - \epsilon$ in a runtime of $poly(n) \cdot |\log(\epsilon)|$ and with a similar number of calls to the oracle.

Example 4.0.1 ($n = 3$). *Consider $f(x \oplus a) = f(x)$ for all $x \in \{0,1\}^3$. The image space X must be of cardinality 4, where we consider the vector a to be $a = (0,1,0)$.*



In fact, in general we require $|X| = \frac{2^n}{2} = 2^{n-1}$.

4.1 Classical Algorithm

The classical algorithm is constructed as follows: draw randomly pairs of points in $\{0,1\}^n$ (with replacement): $\left(x^{(1)}, y^{(1)}\right), \ldots, \left(x^{(q)}, y^{(q)}\right)$. If one such pair (say j), $f\left(x^{(j)}\right) = f\left(y^{(j)}\right)$, compute $a = x^{(j)} \ominus y^{(j)}$ $(= x^{(j)} \oplus y^{(j)})$ and declare success. On the contrary, if $f\left(x^{(j)}\right) \neq f\left(y^{(j)}\right)$ for all $1 \leq j \leq q$, then declare failure.

Proposition 4.1.1. *It holds that* $\mathbb{P}(\textit{success}) \leq \frac{q}{2^n-1}$.

Note that in order to ensure $\mathbb{P}(\text{success}) \geq 1-\epsilon$ we require $q \geq (2^n-1)(1-\epsilon)$ draws.

Proof. We remark that:

$$\mathbb{P}(\text{success}) = \mathbb{P}\left(\exists\, 1 \leq j \leq q : f\left(x^{(j)}\right) = f\left(y^{(j)}\right)\right) \leq \sum_{j=1}^{q} \mathbb{P}\left(f\left(x^{(j)}\right) = f\left(y^{(j)}\right)\right) \leq \frac{q}{2^n-1}. \quad (4.1)$$

We have used the fact that for a given x there is a unique corresponding y, hence:

$$\mathbb{P}\left(f\left(x^{(j)}\right) = f\left(y^{(j)}\right)\right) = \frac{1}{2^n-1}. \quad (4.2)$$

□

A slightly better classical algorithm can be related to the Birthday Problem. This simply corresponds to the random sampling in a set on N elements. The result is that in general the order of trials until we observe two identical elements is $\sqrt{N}$. As a result $\mathcal{O}\left(2^{\frac{N}{2}}\right)$ draws only are needed, however this is still exponential in N.

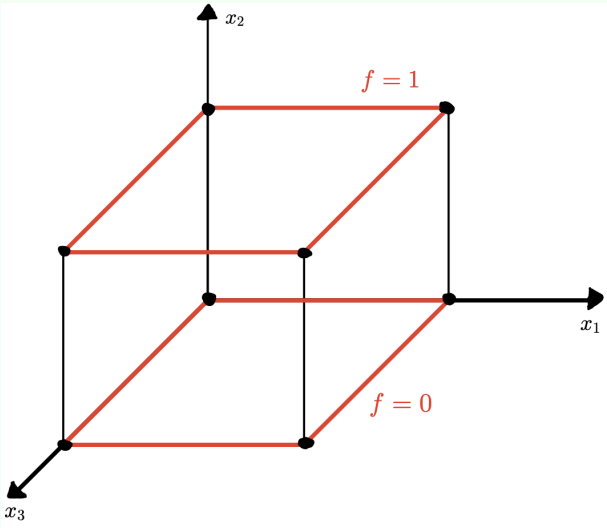
Towards a better construction, let us generalise this problem using notions of Group Theory. Consider $G = \{0,1\}^n$ be a group associated to a vector space, and let H be an unknown subgroup of G associated to a sub-vector space. In particular consider $H = span\left\{h^{(1)}, ..., h^{(k)}\right\}$ to be a k-dimensional subspace of linearly independent vectors. Then, Simon's Problem can be reformulated as follows: find the hidden subgroup $H \subset G$ with as few as possible calls to the oracle $f : \{0,1\}^n \longrightarrow X$ satisfying $f(x) = f(y)$ whenever $x \ominus y \in H$.

Concerning the cardinalities, $|G| = 2^n$, H is k-dimensional hence $|H| = 2^k$. So f possibly takes 2^{n-k} values, which corresponds to the order $|X|$. Using Lagrange's Theorem, a possible option for X is therefore $X = G\backslash H$ (i.e. a quotient group) with $|X| = |G\backslash H| = |G|\backslash|H| = 2^{n-k}$.
We have an equivalence relation $x \sim y$ if and only if $x \ominus y \in H$. The group G can then be divided into 2^{n-k} equivalence classes, namely there exists $\left\{V^{(1)}, ..., V^{(2^{n-k})}\right\}$ representatives of each class such that:

$$G = \bigsqcup_{j=1}^{2^{n-k}} \left\{V^{(j)} \oplus H\right\}. \quad (4.3)$$

Remark 4.1.2. *Note that, in this section, we have used the notation* $\ominus$. *In the context of the field* $\{0,1\}$, *the notation* $\ominus$ *is actually the same as* $\oplus$. *However, in a more general context and considering a field* $F = \{a_1, ..., a_k\}$, *the notation* $\ominus$ *is simply a subtraction modulo* k. *In other words,* $x \ominus y$ *is defined as* $x \oplus y^{-1}$, *where* y^{-1} *is the inverse element of* y *with respect to the set operation (i.e.* $y \cdot y^{-1} = id$, *with the group operation* $\cdot$*).*

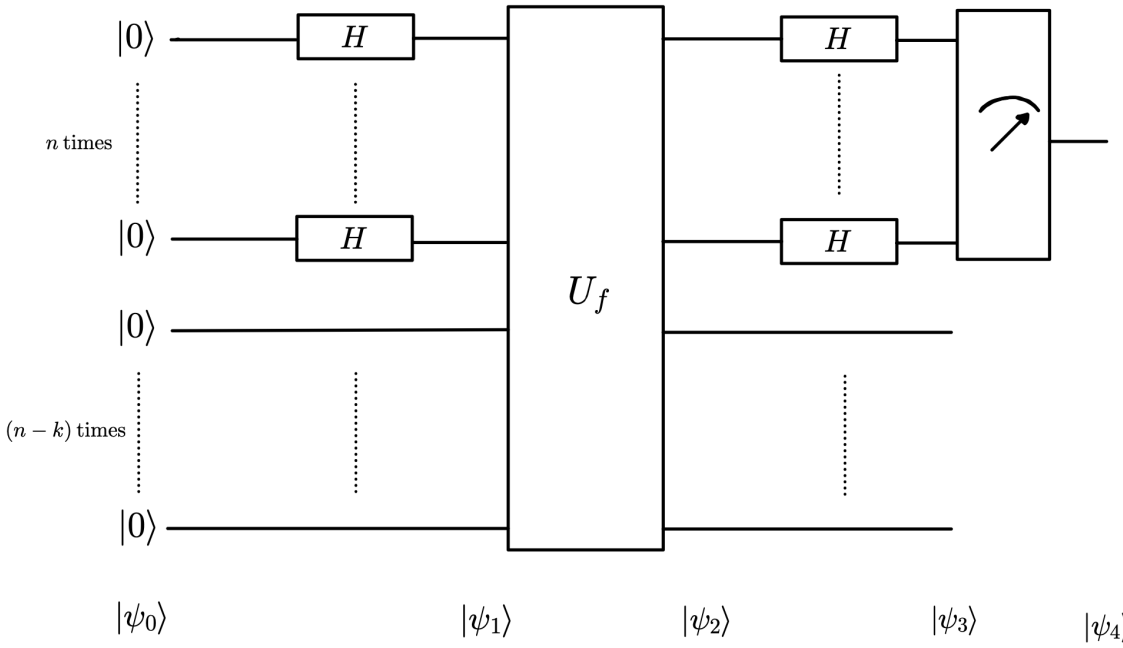
Example 4.1.3 ($n = 3$ and $k = 2$)**.** *In the following example,* $H = \{(0,0,0),(1,0,0),(0,0,1),(1,0,1)\}$ *and* $|X| = 2$*. The equivalence classes are* H *and* $H \oplus (0,1,0)$*.*



4.2 Simon's Quantum Algorithm

We now present the quantum alternative to Simon's algorithm, where we will make use of the Hadamard gate H, the quantum oracle U_f and finally making a measurement. We start with an initial state: (Note that $H^{\otimes n} \otimes \mathbb{I}_{n-k}$ is the same as $H^{\otimes n} \otimes \mathbb{I}^{\otimes(n-k)}$.)

$$|\psi_0\rangle = \left(\bigotimes^{n \text{ times}} |0\rangle\right) \otimes \left(\bigotimes^{(n-k) \text{ times}} |0\rangle\right) := \underbrace{|0\rangle \otimes \ldots \otimes |0\rangle}_{n \text{ times}} \otimes \underbrace{|0\rangle \otimes \ldots \otimes |0\rangle}_{(n-k) \text{ times}}. \tag{4.4}$$



Stage 1: Note that contrary to the Deutsch-Josza's algorithm, the $(n-k)$ ancilla qubits are left untouched before the passage through the oracle U_f.

$$|\psi_1\rangle = \left(H^{\otimes n} \otimes \mathbb{I}_{n-k}\right)|\psi_0\rangle = H^{\otimes n}|0...0\rangle \otimes |0...0\rangle = \frac{1}{2^{\frac{n}{2}}} \sum_{x_1,...,x_n \in \{0,1\}} |x_1...x_n\rangle \otimes |0...0\rangle = \frac{1}{2^{\frac{n}{2}}} \sum_{x \in \{0,1\}^n} |x\rangle \otimes |0...0\rangle .$$

Stage 2: To find $|\psi_2\rangle$, state $|\psi_1\rangle$ has to go through the quantum oracle. The oracle U_f is defined as:

$$U_f\big(|x\rangle \otimes |y\rangle\big) = |x\rangle \otimes |y \oplus f(x)\rangle , \tag{4.5}$$

where we note that $f(x)$ is modulo 2. Hence:

$$U_f U_f |x\rangle \otimes |y\rangle = U_f |x\rangle \otimes |y \oplus f(x)\rangle = |x\rangle \otimes |y \oplus f(x) \oplus f(x)\rangle = |x\rangle \otimes |y\rangle . \tag{4.6}$$

However remark that both y and $f(x)$ are $(n-k)$-dimensional. So:

$$|\psi_2\rangle = U_f |\psi_1\rangle = \frac{1}{2^{\frac{n}{2}}} \sum_{x \in \{0,1\}^n} |x\rangle \otimes |f(x)\rangle . \tag{4.7}$$

Stage 3: Following what was done for the Deutsch-Josza's algorithm we have:

$$H^{\otimes n}|x\rangle = \frac{1}{2^{\frac{n}{2}}} \sum_{y \in \{0,1\}^n} (-1)^{x \cdot y} |y\rangle . \tag{4.8}$$

So, for $|\psi_3\rangle$ we obtain:

$$|\psi_3\rangle = \left(H^{\otimes n} \otimes \mathbb{I}\right)|\psi_2\rangle = \frac{1}{2^n} \sum_{x,y \in \{0,1\}^n} (-1)^{x \cdot y} |y\rangle \otimes |f(x)\rangle . \tag{4.9}$$

Let us rewrite this using the set of the representatives of the equivalence classes of G: $\left\{V^{(1)},...,V^{(2^{n-k})}\right\}$:

$$\begin{aligned} |\psi_3\rangle &= \sum_{y \in \{0,1\}^n} \frac{1}{2^n} \sum_{j=1}^{2^{n-k}} \sum_{h \in H} (-1)^{\left(V^{(j)} \oplus h\right) \cdot y} |y\rangle \otimes \underbrace{\left|f\left(V^{(j)} + h\right)\right\rangle}_{= f\left(V^{(j)}\right) =: f_j} \\ &= \sum_{y \in \{0,1\}^n} \frac{1}{2^n} \sum_{j=1}^{2^{n-k}} (-1)^{V^{(j)} \cdot y} \left(\sum_{h \in H} (-1)^{h \cdot y} \right) |y\rangle \otimes |f_j\rangle . \end{aligned}$$

Now, the $(k \times n)$ matrix representation of H is given by $H = \begin{pmatrix} h^{(1)} & \cdots & h^{(k)} \end{pmatrix}^T$ whose kernel is defined by:

$$Ker = H^{\perp} = \{x \in \{0,1\}^n : H \cdot x = \underline{0}\} = \left\{x \in \{0,1\}^n : h^{(i)} \cdot x = 0 \text{ for all } i\right\}, \tag{4.10}$$

and is an $(n-k)$-dimensional subspace of $\{0,1\}^n$, remarking also that $(H^{\perp})^{\perp} = H$. Here, $h^{(i)} \cdot x$ is the usual dot product operation with $\oplus$ in between each term (addition modulo 2).
Observe now that $\sum_{h \in H} (-1)^{y \cdot h} \in \{0, 2^k\}$. Indeed we may consider the following two cases:

- If $y \in H^{\perp}$ then $y \cdot h = 0$ for all $h \in H$, so:

$$\sum_{h \in H} (-1)^{y \cdot h} = 2^k. \tag{4.11}$$

- If $y \notin H^\perp$ then there exists $h^{(0)} \in H$ such that $h^{(0)} \cdot y = 1$ and so:

$$\sum_{h\in H}(-1)^{y\cdot h} = \sum_{h'\in H}(-1)^{y\cdot\left(h^{(0)}+h'\right)} = -\sum_{h'\in H}(-1)^{y\cdot h'} \Longrightarrow \sum_{h\in H}(-1)^{y\cdot h} = 0. \tag{4.12}$$

Finally we obtain:

$$|\psi_3\rangle = \sum_{y\in H^\perp}\left(\frac{1}{2^{n-k}}\sum_{j=1}^{2^{n-k}}(-1)^{V^{(j)}\cdot y}\right)|y\rangle \otimes |f_j\rangle\,. \tag{4.13}$$

Stage 4: This last stage is dedicated to the measurement of the first n qubits. Here, the first n qubits are entangled with the last $(n-k)$ qubits in state $|\psi_3\rangle$. Hence the partial measurement of the first n qubits is more difficult to describe than in the case of Deutsch-Josza's algorithm.
In general, a measurement is described in quantum mechanics by a complete collection of orthogonal projectors $\{P_j : 1 \le j \le d\}$. In our context, the projectors admit the following properties:

- Applying the projection twice is the same as applying it once: $P_j^2 := P_j \cdot P_j = P_j$
- We consider orthogonal projections, i.e. for all x, y: $\langle P_j x, y - P_j y\rangle = 0$
- The projections are unitary. Recall that for all operators p, $\langle x, p^\dagger y\rangle = \langle px, y\rangle$. Then:

$$\langle P_j x, y - P_j y\rangle = \langle x, P_j^\dagger y - P_j^\dagger P_j y\rangle = \langle x, P_j y - P_j P_j y\rangle = \langle x, (P_j - P_j P_j)y\rangle = 0. \tag{4.14}$$

As a result and by definition it holds that $\langle P_j x, y - P_j y\rangle = 0 = \langle x - P_j x, P_j y\rangle$. Hence:

$$\langle x, P_j^\dagger y\rangle = \langle P_j x, y\rangle = \langle P_j x, P_j y\rangle = \langle x, P_j y\rangle \Longrightarrow P_j = P_j^\dagger, \quad \forall 1 \le j \le d. \tag{4.15}$$

- The set of projectors is complete: $\sum_{j=1}^d P_j = \mathbb{I}$.

Example 4.2.1. *$P_j = |\phi_j\rangle\langle\phi_j|$, where $\{|\phi_j\rangle, 1 \le j \le d\}$, is an orthonormal basis of the Hilbert space $\mathcal{H}$.*

Anyways, back to our measurement! Now, if the system is in state $|\psi\rangle$ before the measurement, the outcome state is:

$$|\psi'\rangle = \frac{P_j|\psi\rangle}{\|P_j|\psi\rangle\|} \quad \text{with probability} \quad \|P_j|\psi\rangle\|^2 = \langle\psi|P_j^\dagger P_j|\psi\rangle = \langle\psi|P_j|\psi\rangle\,.$$

In our case, the measurement of the first n qubits is described by the following complete collection of projectors:

$$\{P_y = |y\rangle\langle y| \otimes \mathbb{I}_{n-k}, y \in \{0,1\}^n\}\,. \tag{4.16}$$

For a given $y_0 \in \{0,1\}^n$, let us compute the outcome probability $\langle\psi_3|P_{y_0}|\psi_3\rangle$ of the state $\frac{P_{y_0}|\psi_3\rangle}{\|P_{y_0}|\psi_3\rangle\|} = |y_0\rangle \otimes$ (some state we do not care about). We then obtain:

$$\begin{aligned}\langle\psi_3|P_{y_0}|\psi_3\rangle &= \left(\sum_{y\in H^\perp}\frac{1}{2^{n-k}}\sum_{j=1}^{2^{n-k}}(-1)^{V^{(j)}\cdot y}\langle y|\otimes\langle f_j|\right)\left(|y_0\rangle\langle y_0|\otimes\mathbb{I}_{n-k}\right)\left(\sum_{y'\in H^\perp}\frac{1}{2^{n-k}}\sum_{j'=1}^{2^{n-k}}(-1)^{V^{(j')}\cdot y'}|y'\rangle\otimes|f_{j'}\rangle\right)\\ &= \sum_{y,y'\in H^\perp}\frac{1}{2^{2(n-k)}}\sum_{j,j'=1}^{2^{n-k}}(-1)^{V^{(j)}\cdot y+V^{(j')}\cdot y'}\langle y|y_0\rangle\langle y_0|y'\rangle\langle f_j|f_{j'}\rangle\\ &= \sum_{y,y'\in H^\perp}\frac{1}{2^{2(n-k)}}\sum_{j,j'=1}^{2^{n-k}}(-1)^{V^{(j)}\cdot y+V^{(j')}\cdot y'}\delta_{yy_0}\delta_{y_0y'}\delta_{jj'}.\end{aligned}$$

So, the above quadruple sum simplifies to two different results depending on the status of y_0:

- If $y_0\notin H^\perp$ then it is equal to 0 .

- If $y_0\in H^\perp$ then we obtain:

$$\frac{1}{2^{2(n-k)}}\sum_{j=1}^{2^{n-k}}(-1)^{V^{(j)}\cdot y_0+V^{(j)}\cdot y_0}=\frac{1}{2^{2(n-k)}}\sum_{j=1}^{2^{n-k}}(-1)^{2V^{(j)}\cdot y_0}=\frac{1}{2^{2(n-k)}}\sum_{j=1}^{2^{n-k}}1=\frac{2^{n-k}}{2^{2(n-k)}}=\frac{1}{2^{n-k}}. \quad (4.17)$$

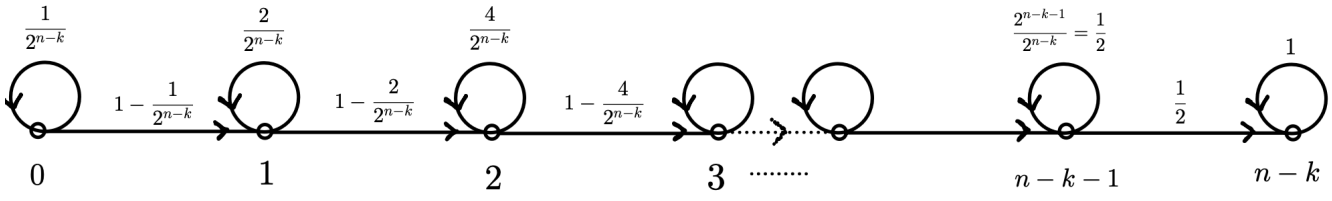
Hence, in the case of $y_0\in H^\perp$, the outcome probabilities are uniform over $H^\perp$.

In conclusion, Simon's algorithm is then the following:

- Run $(n-k)$ times the above circuit. This will output $y^{(1)},...,y^{(n-k)}$ uniformly and independently distributed on $H^\perp$.
- If $y^{(1)},...,y^{(n-k)}$ are linearly independent, then these form a basis of $H^\perp$ which is of dimension $(n-k)$. From this basis, we compute the basis of the dual space H via a classical algorithm (Gauss elimination - runtime $\mathcal{O}(n^3)$). In this case, we declare success.
- If $y^{(1)},...,y^{(n-k)}$ are not linearly independent the declare failure and restart the algorithm (note that in practice, one can do better than that).

Proposition 4.2.2. *It holds that* $\mathbb{P}(\textit{success})\geq\frac{1}{4}$.

We can represent the situation with a directed graph:



Proof. We have to consider the following probabilities:

$$\mathbb{P}\left(y^{(1)}\neq 0\right)=1-\frac{1}{2^{n-k}}. \quad (4.18)$$

$$\mathbb{P}\left(y^{(2)}\notin\operatorname{span}\left(y^{(1)}\right)\middle|y^{(1)}\neq 0\right)=\mathbb{P}\left(y^{(2)}\notin\left\{0,y^{(1)}\right\}\middle|y^{(1)}\neq 0\right)=1-\frac{2}{2^{n-k}}=1-\frac{1}{2^{n-k-1}}. \quad (4.19)$$

$$\mathbb{P}\left(y^{(3)} \notin \text{span}\left(y^{(1)}, y^{(2)}\right) \middle| y^{(1)}, y^{(2)} \text{ lin. indep.}\right) = 1 - \frac{4}{2^{n-k}} = 1 - \frac{1}{2^{n-k-2}}. \tag{4.20}$$

This process continues until the step $(n-k)$ which is given by:

$$\mathbb{P}\left(y^{(n-k)} \notin \text{span}\left(y^{(1)}, ..., y^{(n-k-1)}\right) \middle| y^{(1)}, ..., y^{(n-k-1)} \text{ lin. indep.}\right) = 1 - \frac{2^{n-k-1}}{2^{n-k}} = 1 - \frac{1}{2} = \frac{1}{2}.$$

Now, $\mathbb{P}(\text{success}) = \mathbb{P}\left(y^{(1)}, ..., y^{(n-k)} \text{ are lin. indep.}\right)$ and this can be represented by:

$$\mathbb{P}(\text{success}) = \prod_{i=1}^{n-k}\left(1 - \frac{2^{i-1}}{2^{n-k}}\right) = \prod_{i=0}^{n-k-1}\left(1 - \frac{1}{2^{n-k-i}}\right) = \prod_{i=1}^{n-k}\left(1 - \frac{1}{2^i}\right) = \exp\left(\sum_{i=1}^{n-k}\log\left(1 - \frac{1}{2^i}\right)\right).$$

One can plot the function $\log(1-x)$ and find a linear function $g(x)$ such that $\log(1-x) \geq g(x)$ on the interval $0 \leq x \leq \frac{1}{2}$. Given that the function $\log(1-x)$ is concave, with a simple analysis it is not difficult to show that the required function is $g(x) = -(2\log(2))x$. Therefore:

$$\mathbb{P}(\text{success}) \geq \exp\left(-2\log(2)\sum_{i=1}^{n-k}\frac{1}{2^i}\right) \geq \exp\left(-2\log(2)\right) = 2^{-2} = \frac{1}{4}, \tag{4.21}$$

where we have used the fact that $\sum_{i=1}^{n-k}\frac{1}{2^i} \leq 1$.

□

Of course, a success probability of only $\frac{1}{4}$ is not satisfactory; we would like a success probability $\mathbb{P} \geq 1-\epsilon$. Let us therefore repeat independently the whole algorithm T times:

$$\mathbb{P}(\text{failure after } T \text{ attempts}) = \mathbb{P}(\text{failure})^T \leq \left(\frac{3}{4}\right)^T \leq \epsilon \quad \text{if} \quad T \geq \frac{|\ln(\epsilon)|}{|\ln(3/4)|}. \tag{4.22}$$

In the end, we obtain a success probability $\mathbb{P} \geq 1-\epsilon$ after $\mathcal{O}((n-k)\cdot|\ln(\epsilon)|)$ calls to the quantum oracle U_f (and a polynomial runtime dominated by the $\mathcal{O}(n^3)$ computation of the dual basis). This is to be compared to the $\Omega(2^n)$ calls to the oracle f of any classical algorithm.

Chapter 5

Shor's Algorithm

Recall that for Simon's algorithm we were given a function $f : \{0,1\}^n \longrightarrow \{0,1\}^{n-1}$ such that there exists $a \in \{0,1\}^n$ with $f(x \oplus a) = f(x)$ for all x. The aim was to identify the vector a (i.e. the period of f). The output was a vector $y \in \{0,1\}^n$ uniformly distributed in the set $H^\perp = \{z \in \{0,1\}^n : z \cdot a = 0\}$ where $z \cdot a$ is the dot product $\sum_{i=1}^n z_i a_i$. Hence after sufficiently many runs of the algorithm we were able to identify the vector a.
In this chapter, we consider a slight variation of the problem. Given a periodic function $f : \mathbb{Z} \longrightarrow \mathbb{Z}$, we wish to identify the period $r \geq 1$ of the function f (i.e. the smallest value of $r \geq 1$ such that $f(x+r) = f(x)$ for all $x \in \mathbb{Z}$). However here $\mathbb{Z}$ is infinite, which makes the problem even more difficult! In particular, we will be interested in a function f defined by taking a (large) positive integer N and another number $a \in \{2, ..., N\}$ such that $\gcd(a, N) = 1$, and then for all $x \in \mathbb{Z}$:

$$f(x) = a^x \pmod N. \tag{5.1}$$

Finding the period of f amounts here to finding its order, i.e. the smallest value of $r \geq 1$ such that $a^r \pmod N = 1$. Now, in order to deal with the fact that $|\mathbb{Z}| = \infty$ we consider the following three assumptions:

- Let $n = \lceil \log_2(N) \rceil$ be the number of bits needed in the binary decomposition of numbers.
- Let $M = 2^m$ with $m \geq 1$ be such that $M \approx N^2$ (so that $M >> r$ also as $r \leq N$) and view the original f as $f : \{0, ..., M-1\} \longrightarrow \{0, ..., N-1\}$.
- Let us finally consider for now the strange assumption that $M = kr$ for some $k \geq 1$.

5.1 The Quantum Fourier Transform (QFT)

In this first section we present a powerful tool that will be useful for the description of Shor's algorithm and in general for multiple other applications, the Quantum Fourier Transform (QFT).

Definition 5.1.1. *The **Quantum Fourier Transform** (QFT) is the unitary transformation on m qubits, defined as:*

$$QFT\,|x\rangle = \frac{1}{2^{\frac{m}{2}}} \sum_{z=0}^{2^m-1} \exp\left(\frac{2\pi i x z}{2^m}\right) |z\rangle\,, \tag{5.2}$$

where $|x\rangle$ is an element of the computational basis (with $0 \leq x \leq 2^m - 1$). In this definition, xz is the classical multiplication of two numbers x and z.

Remark that the QFT is a "true" complex-valued transformation (except in the case $m=1$). Its usage may therefore lead to quantum algorithms outperforming classical ones.

Proposition 5.1.2. *The action of the QFT on a basis vector $|x\rangle$, with $x \in \{0,1\}^m$, may also be written as:*

$$QFT\,|x\rangle = \bigotimes_{j=1}^{m} \frac{1}{\sqrt{2}} \left(|0\rangle + \exp\left(\frac{2\pi i x}{2^j}\right) |1\rangle \right). \tag{5.3}$$

Proof. We start from the definition of the QFT:

$$QFT\,|x\rangle = \frac{1}{2^{\frac{m}{2}}} \sum_{z=0}^{2^m-1} \exp\left(\frac{2\pi i x z}{2^m}\right) |z\rangle. \tag{5.4}$$

Observe that $z = \sum_{j=1}^{m} z_j 2^{m-j}$, where $z_1 ... z_m$ is the binary decomposition of $z \in \{0, ..., 2^m - 1\}$. So we successively find that:

$$\exp\left(\frac{2\pi i x z}{2^m}\right) = \exp\left(2\pi i x \sum_{j=1}^{m} z_j 2^{-j}\right) = \prod_{j=1}^{m} \exp\left(2\pi i x z_j 2^{-j}\right)$$

$$\implies \exp\left(\frac{2\pi i x z}{2^m}\right) |z\rangle = \bigotimes_{j=1}^{m} \exp\left(2\pi i x z_j 2^{-j}\right) |z_j\rangle$$

$$\implies QFT\,|x\rangle = \bigotimes_{j=1}^{m} \left(\frac{1}{\sqrt{2}} \sum_{z_j \in \{0,1\}} \exp\left(2\pi i x z_j 2^{-j}\right) |z_j\rangle \right) = \bigotimes_{j=1}^{m} \frac{1}{\sqrt{2}} \left(|0\rangle + \exp\left(\frac{2\pi i x}{2^j}\right) |1\rangle \right).$$

□

We now move on to the construction of the circuit for the QFT. Let $x = \sum_{k=1}^{m} x_k 2^{m-k}$ where $x_1 ... x_m$ is the binary decomposition of $x \in \{0, ..., 2^m - 1\}$. Then:

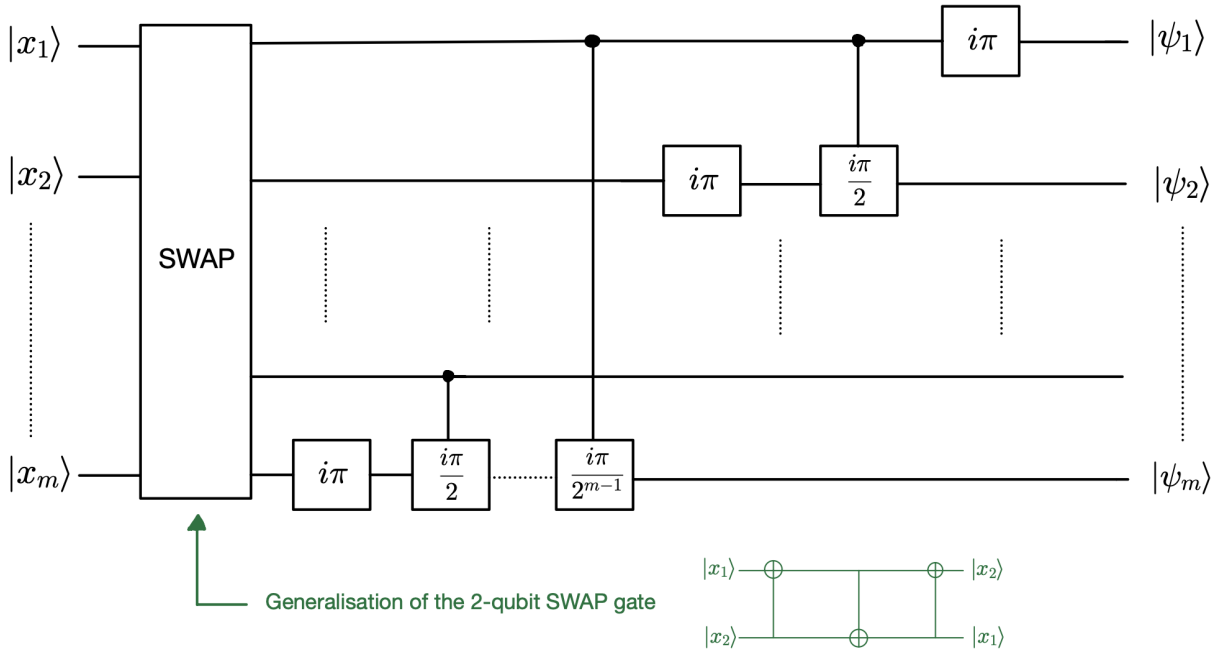
$$\exp\left(2\pi i x 2^{-j}\right) = \exp\left(2\pi i \sum_{k=1}^{m} x_k 2^{m-k-j}\right). \tag{5.5}$$

Observe that $x_k 2^{m-k-j}$ is an integer for $k \leq m-j$. Thus in this case we have that $\exp\left(2\pi i x_k 2^{m-k-j}\right) = 1$, and hence:

$$QFT\,|x\rangle = \bigotimes_{j=1}^{m} \frac{1}{\sqrt{2}} \left(|0\rangle + \exp\left(2\pi i \sum_{k=m-j+1}^{m} x_k 2^{m-k-j}\right) |1\rangle \right)$$

$$= \frac{1}{\sqrt{2}} \left(|0\rangle + \underbrace{\exp\left(2\pi i x_m 2^{-1}\right)}_{\text{rotation by } i\pi x_m} |1\rangle \right) \otimes \frac{1}{\sqrt{2}} \left(|0\rangle + \underbrace{\exp\left(2\pi i x_{m-1} 2^{-1}\right)}_{\text{rotation by } i\pi x_{m-1}} \underbrace{\exp\left(2\pi i x_m 2^{-2}\right)}_{\underline{\text{controlled}} \text{ rotation by } i\pi \frac{x_m}{2}} |1\rangle \right) \otimes ...$$

Let us now draw the circuit of the QFT:



The final task it to find the inverse circuit $QFT^\dagger$. We have that $QFT(|x_1\rangle \otimes ... \otimes |x_m\rangle) = |\psi_1\rangle \otimes ... \otimes |\psi_m\rangle$, or in short-hand notation $QFT\,|x\rangle = |\psi\rangle$. Now as QFT is unitary (we prove this fact below), it is possible to invert the QFT map and to have that $QFT^\dagger\,|\psi\rangle = |x\rangle$. The map $QFT^\dagger$ acting on a basis element $|z\rangle$ is given by:

$$QFT^\dagger\,|z\rangle = \frac{1}{2^{\frac{m}{2}}} \sum_{x=0}^{2^m-1} \exp\left(-\frac{2\pi i x z}{2^m}\right)|x\rangle\,. \tag{5.6}$$

Proposition 5.1.3. *The QFT is a unitary operation.*

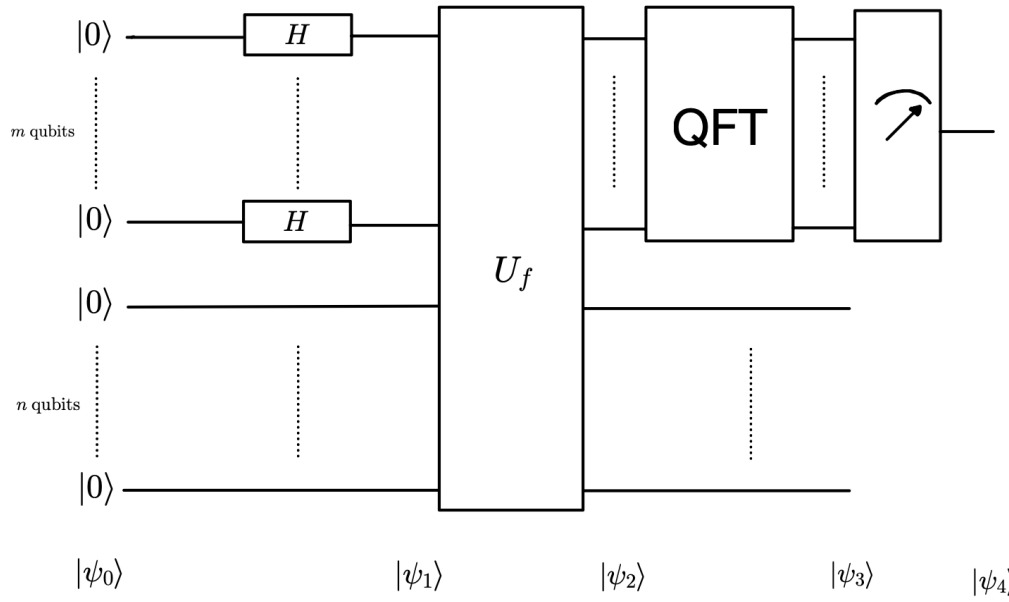
Proof. Directly from the definitions of QFT and $QFT^\dagger$ we have that:

$$\begin{aligned} QFT^\dagger \cdot QFT\,|x\rangle = \frac{1}{2^{\frac{m}{2}}} \sum_{y=0}^{2^m-1} \exp\left(\frac{2\pi i x y}{2^m}\right) QFT^\dagger\,|y\rangle &= \frac{1}{2^m} \sum_{y=0}^{2^m-1} \exp\left(\frac{2\pi i x y}{2^m}\right) \sum_{z=0}^{2^m-1} \exp\left(-\frac{2\pi i y z}{2^m}\right)|z\rangle \\ &= \sum_{z=0}^{2^m-1} \underbrace{\left(\frac{1}{2^m} \sum_{y=0}^{2^m-1} \exp\left(\frac{2\pi i (x-z) y}{2^m}\right)\right)}_{=\delta_{xz}} |z\rangle \\ &= \sum_{z=0}^{2^m-1} \delta_{xz}\,|z\rangle = |x\rangle\,. \end{aligned}$$

□

5.2 Shor's Quantum Algorithm ($M = kr$)

In this section, we delve into the details of Shor's quantum algorithm and in particular with the assumption that $M = kr$, where we first present the related quantum circuit.



Let us first remark that the input state is just $|\psi_0\rangle = |0\rangle^{\otimes m} \otimes |0\rangle^{\otimes n}$. As usual, let us now study the different layers of the algorithm.

Stage 1: State $|\psi_1\rangle$ is given (as usual) by:

$$|\psi_1\rangle = (H|0\rangle)^{\otimes m} \otimes |0\rangle^{\otimes n} = \frac{1}{2^{\frac{m}{2}}} \sum_{x_1,\dots,x_m \in \{0,1\}} |x_1 \dots x_m\rangle \otimes |0\rangle^{\otimes n} =: \frac{1}{2^{\frac{m}{2}}} \sum_{x=0}^{2^m-1} |x\rangle \otimes |0\rangle^{\otimes n}. \tag{5.7}$$

We recall that $x_1 \dots x_m$ is the binary representation of x, and $\{|x\rangle, 0 \leq x \leq 2^m - 1\}$ is the computational basis of $\mathbb{C}^{2^m}$.

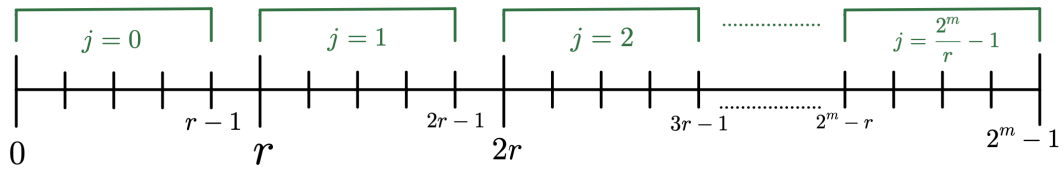
Stage 2: To find $|\psi_2\rangle$, state $|\psi_1\rangle$ has to go through the quantum oracle. Recall that:

$$U_f(|x\rangle \otimes |y\rangle) = |x\rangle \otimes |y \oplus f(x)\rangle. \tag{5.8}$$

Then:

$$|\psi_2\rangle = U_f|\psi_1\rangle = \frac{1}{2^{\frac{m}{2}}} \sum_{x=0}^{2^m-1} |x\rangle \otimes |f(x)\rangle = \frac{1}{2^{\frac{m}{2}}} \sum_{x_0=0}^{r-1} \sum_{j=0}^{\frac{2^m}{r}-1} |x_0 + jr\rangle \otimes \underbrace{|f(x_0 + jr)\rangle}_{= f(x_0)}. \tag{5.9}$$

Here, we have used the fact that $|f(x_0 + jr)\rangle = |f(x_0)\rangle$ by assumption, and also the following decomposition of the sum:



Stage 3: We are at the stage of the analysis of the output qubits. Then:

$$|\psi_3\rangle = (QFT \otimes \mathbb{I}_n)\,|\psi_2\rangle = \frac{1}{2^{\frac{m}{2}}} \sum_{x_0=0}^{r-1} \sum_{j=0}^{\frac{2^m}{r}-1} QFT\,|x_0 + jr\rangle \otimes |f(x_0)\rangle\,. \tag{5.10}$$

Here, the QFT acts in the following way:

$$QFT\,|x_0 + jr\rangle = \frac{1}{2^{\frac{m}{2}}} \sum_{y=0}^{2^m-1} \exp\left(\frac{2\pi i(x_0 + jr)y}{2^m}\right) |y\rangle\,. \tag{5.11}$$

Thus, gathering everything together yields:

$$\begin{aligned} |\psi_3\rangle &= \frac{1}{2^m} \sum_{x_0=0}^{r-1} \sum_{j=0}^{\frac{2^m}{r}-1} \sum_{y=0}^{2^m-1} \exp\left(\frac{2\pi i(x_0 + jr)y}{2^m}\right) |y\rangle \otimes |f(x_0)\rangle \\ &= \frac{1}{r} \sum_{x_0=0}^{r-1} \sum_{y=0}^{2^m-1} \exp\left(\frac{2\pi i x_0 y}{2^m}\right) \underbrace{\left(\frac{1}{(2^m/r)} \sum_{j=0}^{\frac{2^m}{r}-1} \exp\left(\frac{2\pi i j y}{2^m/r}\right)\right)}_{(*)} |y\rangle \otimes |f(x_0)\rangle\,. \end{aligned}$$

Now, we may note that:

$$(*) = \begin{cases} 1 & \text{if } y \text{ is a multiple of } 2^m/r \\ 0 & \text{otherwise} \end{cases}\,. \tag{5.12}$$

So in this case, we should only retain the terms $y = k\frac{2^m}{r}$ for $0 \le k \le r-1$. And hence, $|\psi_3\rangle$ becomes:

$$|\psi_3\rangle = \frac{1}{r} \sum_{x_0=0}^{r-1} \sum_{k=0}^{r-1} \exp\left(\frac{2\pi i x_0 k}{r}\right) \left|k\frac{2^m}{r}\right\rangle \otimes |f(x_0)\rangle\,. \tag{5.13}$$

Stage 4: This last stage is dedicated to the measurement of the first m qubits.

Proposition 5.2.1. *After the measurement of the first m qubits, the state $|\psi_3\rangle$ is projected uniformly at random onto one of the states $\left|k\frac{2^m}{r}\right\rangle$ with $0 \le k \le r-1$.*

Proof. Measuring the first m qubits in the computational basis leads to the output state:

$$|\psi_4\rangle = \frac{P_{y_0}\,|\psi_3\rangle}{\|P_{y_0}\,|\psi_3\rangle\|} \quad \text{where } P_{y_0} = |y_0\rangle\langle y_0| \otimes \mathbb{I}_n \text{ and for } 0 \le y_0 \le 2^m - 1. \tag{5.14}$$

This happens with probability $\mathbb{P}(y_0) = \langle\psi_3|\,P_{y_0}\,|\psi_3\rangle$. Note that (5.14) looks more complex than necessary since the output state of the first m qubits is simply $|y_0\rangle$. Now let us compute the probability, where we use the fact that f differs accross $0 \le x \le r-1$:

$$
\begin{aligned}
\mathbb{P}(y_0) &= \langle\psi_3| P_{y_0} |\psi_3\rangle \\
&= \left(\frac{1}{r}\sum_{x_0,k=0}^{r-1} \exp\left(-\frac{2\pi i x_0 k}{r}\right) \left\langle \frac{k2^m}{r}\right| \otimes \langle f(x_0)|\right) \left(|y_0\rangle\langle y_0| \otimes \mathbb{I}_n\right) \\
&\qquad\qquad \times \left(\frac{1}{r}\sum_{x_0',k'=0}^{r-1} \exp\left(\frac{2\pi i x_0' k'}{r}\right) \left|\frac{k'2^m}{r}\right\rangle \otimes |f(x_0')\rangle\right) \\
&= \frac{1}{r^2}\sum_{x_0,k,x_0',k'=0}^{r-1} \exp\left(\frac{2\pi i(x_0'k' - x_0 k)}{r}\right) \left\langle \frac{k2^m}{r}\middle| y_0\right\rangle \left\langle y_0 \middle| \frac{k'2^m}{r}\right\rangle \langle f(x_0)|f(x_0')\rangle \\
&= \frac{1}{r^2}\sum_{x_0,k,x_0',k'=0}^{r-1} \exp\left(\frac{2\pi i(x_0'k' - x_0 k)}{r}\right) \delta_{\frac{k2^m}{r},y_0}\delta_{\frac{k'2^m}{r},y_0}\delta_{x_0,x_0'}.
\end{aligned}
$$

Therefore, among the above four sums over x_0, k, x_0', k', only the one over x_0 remains. Hence:

$$
\mathbb{P}(y_0) = \begin{cases} \frac{1}{r^2}\sum_{x_0=0}^{r-1} 1 = \frac{1}{r} & \text{if there exists } 0 \le k \le r-1 \text{ with } y_0 = \frac{k2^m}{r} \\ 0 & \text{otherwise} \end{cases} . \tag{5.15}
$$

□

Proposition 5.2.2. *From this measurement, it is possible to extract the value of r with probability $\mathbb{P} \ge \frac{1}{4\ln\left(\ln(2^m)\right)}$.*

Proof. With $0 \le k \le r-1$, the output of the circuit is a number $y_0 = \frac{k2^m}{r}$. Hence $\frac{y_0}{2^m} = \frac{k}{r}$, where $\frac{y_0}{M}$ is known however k and r are not known a priori.

- If $\gcd(k,r) = 1$, then simplifying the fraction $\frac{y_0}{2^m}$ leads to $\frac{k}{r}$ and looking at the denominator we find r.
- If $\gcd(k,r) \neq 1$, then this procedure fails.

Note that in practice, we do not know a priori whether $\gcd(k,r) = 1$ or not (since we do not know k and r), but we can still simplify the fraction $\frac{y_0}{2^m}$ and test whether the resulting denominator is indeed a period of $f(x) = a^x (\mathrm{mod}\ N)$ or not (if $\gcd(k,r) \neq 1$ it won't be). Now, as $0 \le k \le r-1$ is uniform, the success probability of this procedure is:

$$
\mathbb{P}\big(\gcd(k,r) = 1\big) = \frac{\phi(r)}{r}, \tag{5.16}
$$

where $\phi(r) = |\{0 \le k \le r-1 : \gcd(k,r) = 1\}|$ is Euler's function (e.g. $\phi(10) = |\{1,3,7,9\}| = 4$). It can finally be shown that $\phi(r) \ge \frac{r}{4\ln\ln(r)}$ and hence:

$$
\mathbb{P}(\text{success}) \ge \frac{1}{4\ln\ln(r)} \ge \frac{1}{4\ln\ln(2^m)}. \tag{5.17}
$$

□

With these two propositions, we can declare success as $\mathcal{O}\big(\ln(\ln(2^m))\big)$ measurements will lead to the result with probability 1 (compare with Simon's algorithm).

5.3 Shor's Quantum Algorithm ($M \neq kr$)

Until now, we have considered the weird assumption $M = kr$. Let us get rid of this condition and see where this goes. So remember that we are looking for the smallest value of $r \geq 1$ such that $f(r) = a^r (\text{mod } N) = 1$ with $2 \leq a \leq N-1$ such that $\gcd(a, N) = 1$ and $M = 2^m$ with $m \geq 1$ is such that $M \geq N^2$. Here, f is viewed as $f : \{0, ..., M-1\} \longrightarrow \{0, ..., N-1\}$. Recall Shor's circuit in the beginning of the last section, where $n = \lceil \log_2(N) \rceil$. Considering $M \neq kr$, some things remain the same:

$$|\psi_0\rangle = |0...0\rangle \otimes |0...0\rangle . \tag{5.18}$$

$$|\psi_1\rangle = \frac{1}{\sqrt{M}} \sum_{x=0}^{M-1} |x\rangle \otimes |0...0\rangle . \tag{5.19}$$

$$|\psi_2\rangle = \frac{1}{\sqrt{M}} \sum_{x=0}^{M-1} |x\rangle \otimes |f(x)\rangle . \tag{5.20}$$

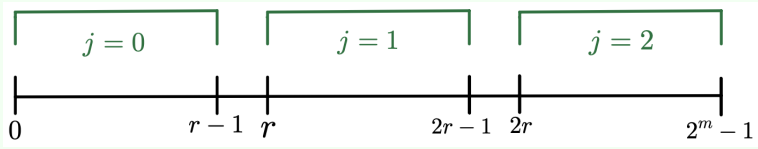
Define now, for $0 \leq x_0 \leq r-1$:

$$A(x_0) = \inf \{ j \geq 1 : x_0 + jr > M-1 \}. \tag{5.21}$$

Remark that when $M = kr$ we have that $A(x_0) = \frac{M}{r}$ for all x_0. Now we can split the sum as before:

$$|\psi_2\rangle = \frac{1}{\sqrt{M}} \sum_{x_0=0}^{r-1} \sum_{j=0}^{A(x_0)-1} |x_0 + jr\rangle \otimes |f(x_0 + jr)\rangle = \frac{1}{\sqrt{M}} \sum_{x_0=0}^{r-1} \sum_{j=0}^{A(x_0)-1} |x_0 + jr\rangle \otimes |f(x_0)\rangle . \tag{5.22}$$

Example 5.3.1. *Let $r = 6$ and $M = 16$. Considering the situation below we have that $A(x_0 = 1) = 3$ while $A(x_0 = 5) = 2$.*



Let us proceed now and compute $|\psi_3\rangle$:

$$|\psi_3\rangle = \frac{1}{r} \sum_{x_0=0}^{r-1} \sum_{y=0}^{M-1} \exp\left(\frac{2\pi i x_0 y}{M}\right) \underbrace{\left(\frac{r}{M} \sum_{j=0}^{A(x_0)-1} \exp\left(\frac{2\pi i j y}{M/r}\right) \right)}_{(**)} |y\rangle \otimes |f(x_0)\rangle . \tag{5.23}$$

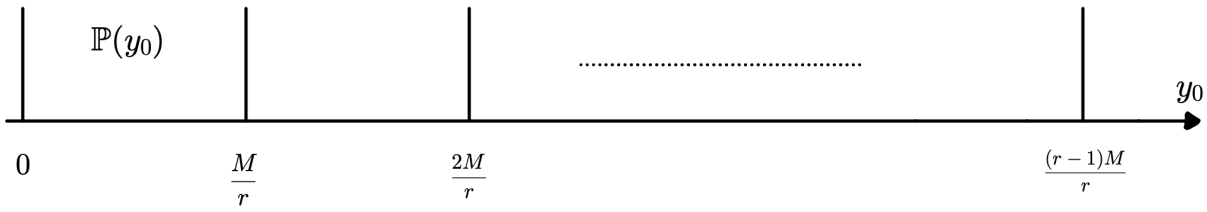
However note that $(**)$ is not 0 or 1 anymore! After the measurement of the first m qubits, the state of the first m qubits is $|y_0\rangle$ (with $0 \leq y_0 \leq M-1$) with probability:

$$\mathbb{P}(y_0) = \langle \psi_3 | \left(|y_0\rangle \langle y_0| \otimes \mathbb{I}_n \right) |\psi_3\rangle . \tag{5.24}$$

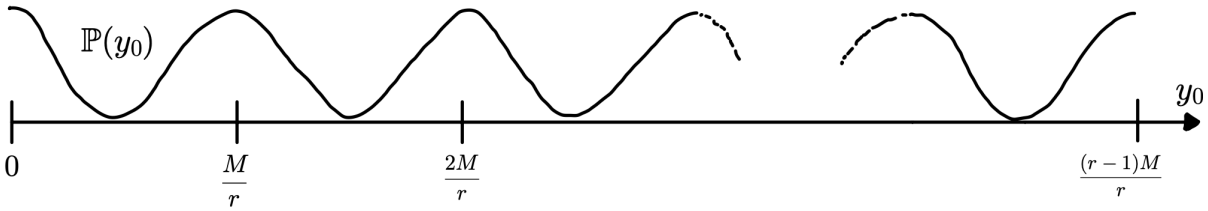
Let us now compute this probability:

$$\mathbb{P}(y_0) = \left(\frac{1}{r}\sum_{x_0=0}^{r-1}\sum_{y=0}^{M-1}\exp\left(-\frac{2\pi i x_0 y}{M}\right)\left(\frac{r}{M}\sum_{j=0}^{A(x_0)-1}\exp\left(-\frac{2\pi i j y}{M/r}\right)\right)\langle y|\otimes\langle f(x_0)|\right)(|y_0\rangle\langle y_0|\otimes\mathbb{I}_n)$$
$$\times\left(\frac{1}{r}\sum_{x_0'=0}^{r-1}\sum_{y'=0}^{M-1}\exp\left(\frac{2\pi i x_0' y'}{M}\right)\left(\frac{r}{M}\sum_{j'=0}^{A(x_0')-1}\exp\left(\frac{2\pi i j' y'}{M/r}\right)\right)|y'\rangle\otimes|f(x_0')\rangle\right)$$
$$= \frac{1}{r^2}\sum_{x_0,x_0'=0}^{r-1}\sum_{y,y'=0}^{M-1}\exp\left(\frac{2\pi i(x_0' y' - x_0 y)}{M}\right)\left(\frac{r}{M}\sum_{j=0}^{A(x_0)-1}\exp\left(-\frac{2\pi i j y}{M/r}\right)\right)\left(\frac{r}{M}\sum_{j'=0}^{A(x_0')-1}\exp\left(\frac{2\pi i j' y'}{M/r}\right)\right)$$
$$\times\,\delta_{y_0,y}\delta_{y_0,y'}\delta_{x_0,x_0'}$$
$$= \frac{1}{r^2}\sum_{x_0=0}^{r-1}\left(\frac{r}{M}\sum_{j=0}^{A(x_0)-1}\exp\left(-\frac{2\pi i j y_0}{M/r}\right)\right)\left(\frac{r}{M}\sum_{j'=0}^{A(x_0)-1}\exp\left(\frac{2\pi i j' y_0}{M/r}\right)\right)$$
$$= \frac{1}{r^2}\sum_{x_0=0}^{r-1}\underbrace{\left|\frac{r}{M}\sum_{j=0}^{A(x_0)-1}\exp\left(\frac{2\pi i j y_0}{M/r}\right)\right|^2}_{(***)}.$$

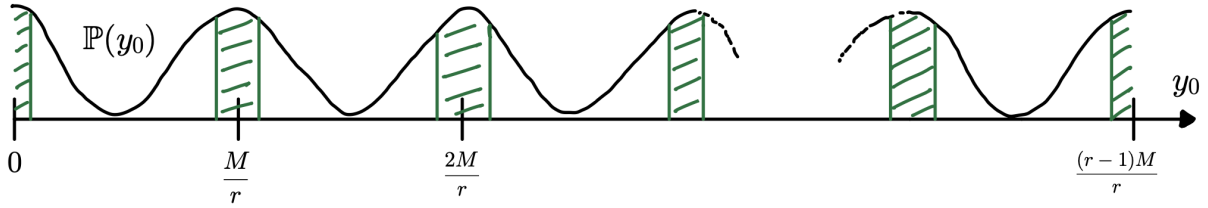
In the case $M = kr$, $(***)$ is equal to 1 for y_0 multiple of $\frac{M}{r}$ and 0 otherwise:



However in the general case, the situation is as follows:



So the output y_0 of the circuit is not necessarily a multiple of $\frac{M}{r}$ anymore. Nevertheless, one can show the following: Let $I = \bigcup_{k=0}^{r-1} I_k$ with $I_k = \left[k\frac{M}{r} - \frac{1}{2}, k\frac{M}{r} + \frac{1}{2}\right]$ (note $|I_k| = 1$ for all k). Then $\mathbb{P}(y_0 \in I) \geq \frac{2}{5}$.



Now, let us observe the possible deductions concerning the case where $y_0 \in I$. First note that $\left|\frac{y_0}{M} - \frac{k}{r}\right| \leq \frac{1}{2M}$ for some $k \geq 1$. However remember that we have chosen $M \geq N^2 > r^2$ and thus $\left|\frac{y_0}{M} - \frac{k}{r}\right| < \frac{1}{2r^2}$. In order to understand what to do with this, we need the notion of convergents.

5.4 Convergents and an Algorithm to find r

In this section, we first start by defining the notion of convergents and continued fractions, which will be best understood with an example. Suppose we want to approximate the real number $x = \frac{73}{31}$, whose true value is approximately $x \approx 2.355$. We may decompose x as follows:

- The first approximation gives:

$$\frac{73}{31} = 2 + \frac{11}{31} \xrightarrow{\text{approximate value}} 2. \tag{5.25}$$

- The second approximation gives:

$$\frac{73}{31} = 2 + \frac{1}{\left(\frac{31}{11}\right)} = 2 + \frac{1}{\left(2 + \frac{9}{11}\right)} \xrightarrow{\text{approximate value}} 2 + \frac{1}{2} = 2.5. \tag{5.26}$$

- The third approximation gives:

$$\frac{73}{31} = 2 + \frac{1}{2 + \frac{1}{\left(\frac{11}{9}\right)}} = 2 + \frac{1}{2 + \frac{1}{\left(1 + \frac{2}{9}\right)}} \xrightarrow{\text{approximate value}} 2 + \frac{1}{2 + \frac{1}{1}} = 2 + \frac{1}{3} = 2.\overline{3}. \tag{5.27}$$

- The fourth and last approximation gives:

$$\frac{73}{31} = 2 + \frac{1}{2 + \frac{1}{1 + \frac{1}{\left(\frac{9}{2}\right)}}} = 2 + \frac{1}{2 + \frac{1}{1 + \frac{1}{\left(4 + \frac{1}{2}\right)}}} \xrightarrow{\text{approximate value}} 2 + \frac{1}{2 + \frac{1}{1 + \frac{1}{4}}} = 2 + \frac{5}{14} \approx 2.357. \tag{5.28}$$

We have stopped the approximation since this last step yields a relatively simple calculation. Now, computing the convergents (which are the approximations at each step) of a real number x is therefore a way to obtain successive approximations of x. Moreover, the convergence is fast since one digit is essentially gained at each step. Finally note that if the initial x is irrational, then this procedure never stops.

Lemma 5.4.1 (Legendre's Lemma)**.** *Let $x \in \mathbb{R}$. If $\left|x - \frac{k}{r}\right| < \frac{1}{2r^2}$, then $\frac{k}{r}$ is a convergent of x.*

We can now present a small algorithm to find the value of r:

- If $y_0 \in I$ then $\left|\frac{y_0}{M} - \frac{k}{r}\right| < \frac{1}{2r^2}$ for some k.
- By Legendre's Lemma, $\frac{k}{r}$ is a convergent of $\frac{y_0}{M}$.
- Compute all the convergents of $x = \frac{y_0}{M}$ along with their denominators (say d_i): if one d_i is such that $a^{d_i} (\text{mod } N) = 1$ then $r = d_i$.

Finally remark that the above algorithm will almost surely fail to find the value of r if it turns out that $y_0 \notin I$. In this instance simply relaunch the whole problem, since $\mathbb{P}(y_0 \in I) \geq \frac{2}{5}$ then eventually success will occur.

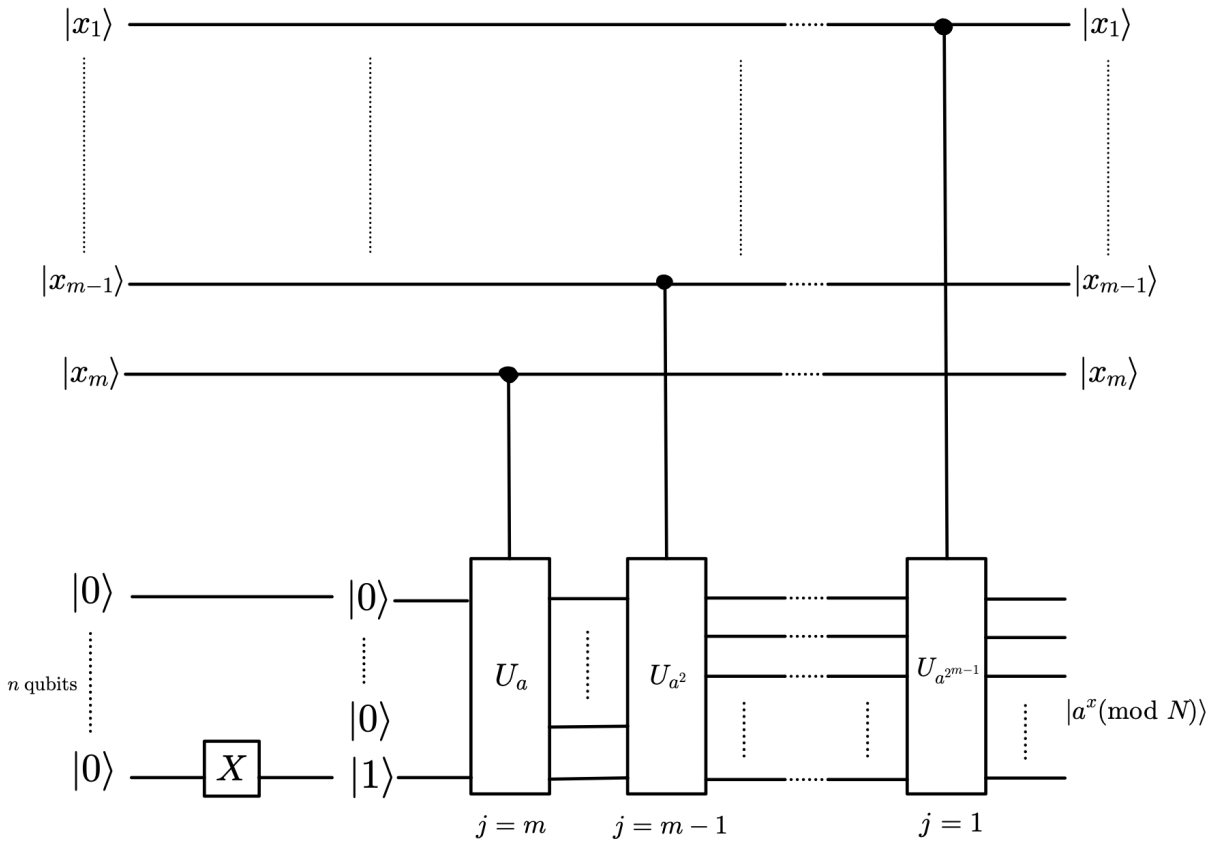
5.5 Circuit for U_f and the Shor's Factoring Algorithm

We now end this chapter by finding an explicit circuit for the oracle U_f and finally presenting the Shor's factoring algorithm.

Recall that x is represented on m bits, with $m = \log_2(M)$ and $M \geq N^2$. Also recall that $x = \sum_{k=1}^{m} x_k 2^{m-k}$ so that:

$$f(x) = a^x (\text{mod } N) = \left(\prod_{k=1}^{m} a^{x_k 2^{m-k}} \right) (\text{mod } N) = \prod_{k=1}^{m} \underbrace{\left(a^{x_k 2^{m-k}} (\text{mod } N) \right)}_{(*)} (\text{mod } N). \tag{5.29}$$

This last step comes from the fact that $(*)$ may be represented on n bits. This suggests the following circuit for U_f:



Let us make a comment about the gates $\{U_a, U_{a^2}, U_{a^4}, ...\}$, where U_a corresponds to a multiplication by $a(\text{mod } N)$. The input (say $|y\rangle$) is represented on n qubits. If $N \neq 2^n$, which is generally the case, then since a permutation gives a unitary operation we find:

$$U_a |y\rangle = \begin{cases} |ay \ (\text{mod } N)\rangle & \text{for } 0 \leq y \leq N-1 \\ |y\rangle & \text{for } N \leq y \leq 2^n - 1 \end{cases} . \tag{5.30}$$

Now, there is no need to know in advance the order r to be able to build the circuit U_f (i.e. no cheating to construct the gate). Furthermore, the above circuit performs efficiently the modular exponential with $\mathcal{O}(n^3) = \mathcal{O}(\log(N)^3)$ gates (equivalent to the runtime of the algorithm).

Proposition 5.5.1 (Factoring Problem)**.** *Given a (large) integer N, find a number $2 \leq a \leq N-1$ such that $a|N$. By repeatedly solving this problem, one finds the prime factor decomposition of N.*

A hardest instance of this problem is when $N = pq$ with p and q large prime numbers (since we have very few a's). We may now present the Shor's factoring algorithm.

Proposition 5.5.2 (Shor's Factoring Algorithm)**.** *This is given in five steps:*
(1) Choose $2 \leq a \leq N-1$ uniformly at random and compute $d = \gcd(a, N)$.
(2) If $d > 1$ then $a = d$ solves the factoring problem. Assume therefore $d = 1$ in the following.
(3) Compute the smallest value of $r \geq 1$ such that $a^r (mod\ N) = 1$.
(4) If r is odd, declare failure and restart the algorithm at (1).
(5) If r is even, then observe that:

$$a^r - 1 = \underbrace{(a^{r/2} - 1)}_{:=d_-}\underbrace{(a^{r/2} + 1)}_{:=d_+}. \tag{5.31}$$

Also by (3) we have $a^r - 1 = kN$ for some $k \in \mathbb{Z}$, thus $N | a^r - 1 \Longrightarrow N | d_- d_+$.
Then, three different options can happen:
- *Either $N | d_-$, but this is actually impossible as r is by assumption the smallest value such that $N | a^r - 1$.*
- *Or $N | d_+$ and in this case declare failure and restart the algorithm at (1).*
- *Or, by computing $\gcd(N, d_-)$ and $\gcd(N, d_+)$, we have that N shares non-trivial prime factors with both d_- and d_+. Declare success in this case.*

Remark that (1) requires $\mathcal{O}(\log(N)^3)$ runtime with Euclid's algorithm. In (2), the fact that $d > 1$ happens with low probability. Step (3) is done by using the algorithm described at the end of the previous section, which is at the heart of Shor's algorithm ($\mathcal{O}(\log(N)^3)$ runtime).
Note also that Robin & Miller showed in 1974 that the success probability of this algorithm is greater than or equal to $\frac{3}{4}$. Hence by repeating this algorithm T times, one can obtain an arbitrarily small error probability.

Example 5.5.3. *Let $N = 91$ and $a = 3$. Then by the above algorithm we have that $r = 6$. Now since r is even and by observing that $a^6 - 1 = (a^3 - 1)(a^3 + 1) = 26 \times 28$, we obtain the prime factorization with $\gcd(91, 26) = 13$ and $\gcd(91, 28) = 7$. We declare success since $91 = 13 \times 7$.*

Classically, for a and N with order m digits, finding the smallest value of $r \geq 1$ such that $a^r (\text{mod } N) = 1$ requires a runtime of order $\exp\left(\left(\frac{64m}{19}\right)^{\frac{1}{3}} \log(m)^{\frac{2}{3}}\right)$ with the best known algorithm (i.e. runtime superpolynomial in m). In comparison, the quantum order finding algorithm described in the previous section finds the minimum value of $r \geq 1$ such that $a^r (\text{mod } N) = 1$ with a runtime of order $\mathcal{O}(m^3) = \mathcal{O}(\log(N)^3)$, opening the door to a polynomial time resolution of the factoring problem (in theory!).

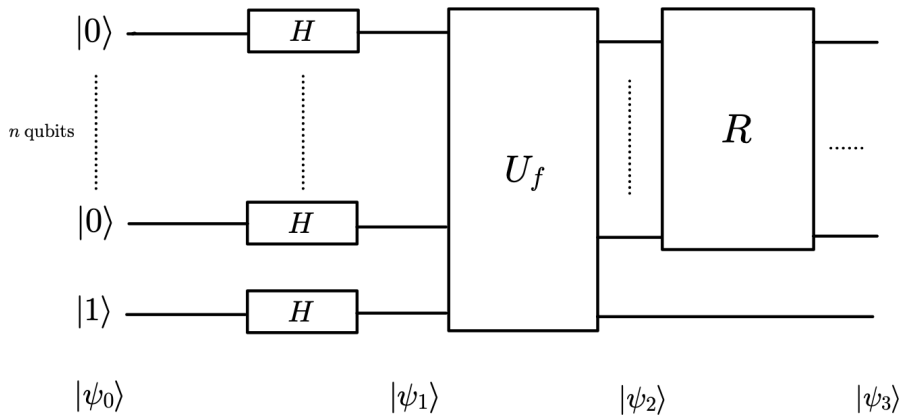
Chapter 6

Grover's Algorithm

Let $f : \{0,1\}^n \longrightarrow \{0,1\}$ be a Boolean function and $A = \{x \in \{0,1\}^n : f(x) = 1\}$. We are considering the **search problem**, namely to identify one element x of A with as few calls as possible to the oracle f. Consider first the case where $A = \{x^*\}$ is a singleton. Classically, identifying x^* may require up to $N = 2^n$ calls to the oracle f, in the worst case scenario. As we will see, Grover's quantum algorithm only requires $\sqrt{N} = 2^{\frac{n}{2}}$ calls to the oracle f to identify x^*.
Note that Grover's algorithm is usually presented solving the following problem: given a directory of N names in alphabetical order and corresponding phone numbers, it allows to recover the name corresponding to a given phone number in only $\sqrt{N}$ steps (instead of N steps classically). However, in order to work, Grover's algorithm requires us to build the gate U_f, which is not doable in the directory search problem, as we know nothing about the function f. Nevertheless, we will still look at some interesting applications of this algorithm later on.
Note also that we will consider, in general, functions f with $|A| = M \in \{1, ..., N\}$. Perhaps surprisingly, considering this generalization (without focusing solely on the case $M = 1$) will help us visualize better how the algorithm works!

6.1 Grover's Quantum Circuit: $|\psi_1\rangle$ and $|\psi_2\rangle$

We consider the following circuit, where R is a reflection gate.



Let us first remark that the input state is just $|\psi_0\rangle = |0\rangle^{\otimes n} \otimes |1\rangle$. As usual, let us now study the different layers of the circuit.

Stage 1: State $|\psi_1\rangle$ is given (as usual) by:

$$|\psi_1\rangle = \left(H|0\rangle\right)^{\otimes n} \otimes H|1\rangle = \frac{1}{\sqrt{N}} \sum_{x\in\{0,1\}^n} |x\rangle \otimes |-\rangle\,, \tag{6.1}$$

where $|-\rangle := \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$.

Stage 2: To find $|\psi_2\rangle$, state $|\psi_1\rangle$ has to go through the quantum oracle. Then:

$$|\psi_2\rangle = U_f|\psi_1\rangle = \frac{1}{\sqrt{N}} \sum_{x\in\{0,1\}} (-1)^{f(x)}|x\rangle \otimes |-\rangle\,. \tag{6.2}$$

So far, nothing is new. Now, remember that in our case, $M = |A| = \{x : f(x) = 1\}$ and thus $N - M = |A^c| = \{x : f(X) = 0\}$. Therefore:

$$\begin{aligned}|\psi_2\rangle = \frac{1}{\sqrt{N}} \sum_{x\in\{0,1\}} (-1)^{f(x)}|x\rangle \otimes |-\rangle &= \frac{1}{\sqrt{N}} \left(\sum_{x\in A^c} |x\rangle - \sum_{x\in A} |x\rangle\right) \otimes |-\rangle \\ &= \left[\sqrt{\frac{N-M}{N}} \left(\frac{1}{\sqrt{N-M}} \sum_{x\in A^c} |x\rangle\right) - \sqrt{\frac{M}{N}} \left(\frac{1}{\sqrt{M}} \sum_{x\in A} |x\rangle\right)\right] \otimes |-\rangle\,.\end{aligned}$$

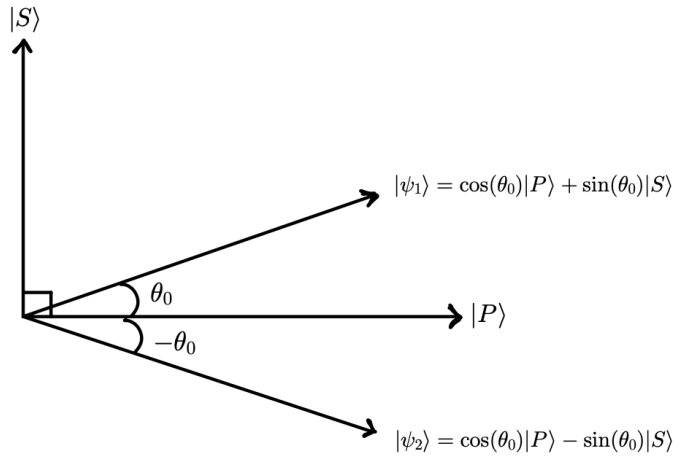
Let us now write $|P\rangle = \frac{1}{\sqrt{N-M}} \sum_{x\in A^c} |x\rangle$ and $|S\rangle = \frac{1}{\sqrt{M}} \sum_{x\in A} |x\rangle$. Both $|P\rangle$ and $|S\rangle$ are quantum states (normalized to 1) and $|\psi_2\rangle$ may be written as:

$$|\psi_2\rangle = \left(\sqrt{\frac{N-M}{N}}\,|P\rangle - \sqrt{\frac{M}{N}}\,|S\rangle\right) \otimes |-\rangle\,. \tag{6.3}$$

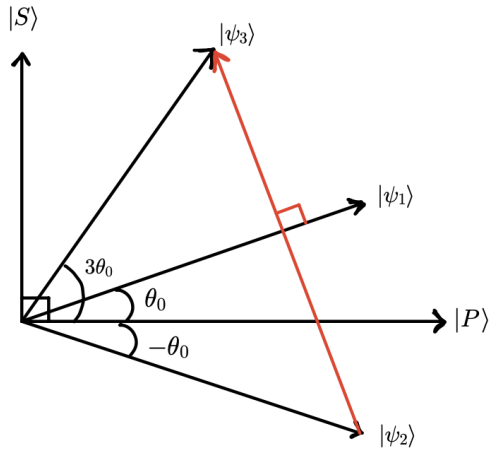
From now on, we forget about the extra $|-\rangle$ state. Note also that $\left(\sqrt{\frac{N-M}{N}}\right)^2 + \left(\sqrt{\frac{M}{N}}\right)^2 = 1$, hence there exists $\theta_0 \in [0, \frac{\pi}{2}]$ such that $\cos(\theta_0) = \sqrt{\frac{N-M}{N}}$ and $\sin(\theta_0) = \sqrt{\frac{M}{N}}$. Thus, we may write $|\psi_2\rangle$ as $|\psi_2\rangle = \cos(\theta_0)\,|P\rangle - \sin(\theta_0)\,|S\rangle$.

6.2 Geometric Interpretation and the Reflection Gate R

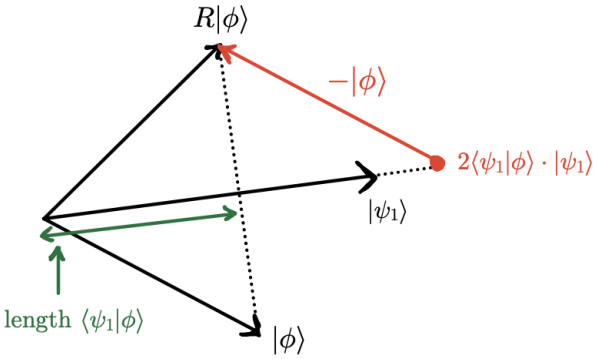
Since $|P\rangle$ and $|S\rangle$ share no common basis element, they are orthogonal. We then obtain the following picture:



The action of the gate U_f on state $|\psi_1\rangle$ can therefore be interpreted as a **reflection with respect to the axis $|P\rangle$**. However note that we do not know the axes $|P\rangle$ and $|S\rangle$ which is exactly what we are after. More precisely, the aim now is to push as much as possible the state of the system towards $|S\rangle$, which contains only elements $x \in A$.
A first step in this direction is done by applying the gate R which corresponds to another **reflection with respect to the state $|\psi_1\rangle$**.



Building such a gate R does <u>not</u> require using the function f again as we simply have that $|\psi_1\rangle = \frac{1}{\sqrt{N}}\sum_{x\in\{0,1\}^n}$. The following figure shows the geometric procedure to build R:

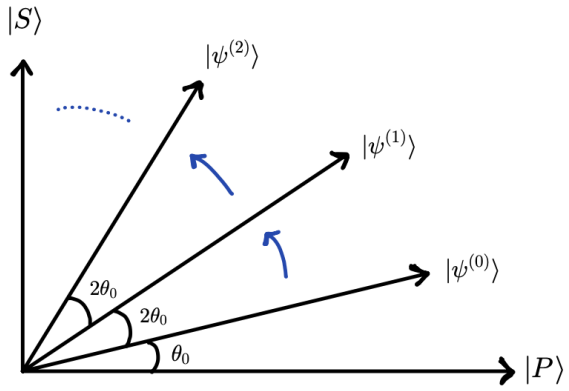


Thus we see that $R|\phi\rangle = 2\langle\psi_1|\phi\rangle|\psi_1\rangle - |\phi\rangle$. Explicitly, $R|\phi\rangle$ may be expanded as follows,

$$\begin{aligned} R|\phi\rangle = 2\langle\psi_1|\phi\rangle|\psi_1\rangle - |\phi\rangle = 2|\psi_1\rangle\langle\psi_1|\phi\rangle - |\phi\rangle &= \left(2|\psi_1\rangle\langle\psi_1| - \mathbb{I}_n\right)|\phi\rangle \\ &= \left(2H^{\otimes n}|\psi_0\rangle\langle\psi_0|H^{\otimes n} - \mathbb{I}_n\right)|\phi\rangle \\ &= H^{\otimes n}\left(2|\psi_0\rangle\langle\psi_0| - \mathbb{I}_n\right)H^{\otimes n}|\phi\rangle\,, \end{aligned}$$

with $|\psi_0\rangle = |0\rangle^{\otimes n}$ and where we forget again the state $|-\rangle$. Now, to proceed there on, with the successive application of U_f and R the states evolve from $|\psi_1\rangle = \cos(\theta_0)|P\rangle + \sin(\theta_0)|S\rangle$ (with angle $+\theta_0$) to $|\psi_2\rangle = \cos(\theta_0)|P\rangle - \sin(\theta_0)|S\rangle$ (with angle $-\theta_0$) and finally to $|\psi_3\rangle = \cos(3\theta_0)|P\rangle + \sin(3\theta_0)|S\rangle$ (with angle $3\theta_0$). Therefore, the successive application of U_f and R corresponds to a **rotation of angle $+2\theta_0$** (from $|\psi_1\rangle$ to $|\psi_3\rangle$) which brings the state closer to the state $|S\rangle$ (corresponding to our aim). By iterating this operation an appropriate number of times, we can get arbitrarily close to $|S\rangle$! For instance, after k iterations of the $G = RU_f$ gate, the state becomes:

$$\left|\psi^{(k)}\right\rangle = \cos\left((2k+1)\theta_0\right)|P\rangle + \sin\left((2k+1)\theta_0\right)|S\rangle\,. \tag{6.4}$$



6.3 Choosing the Number k of Iterations

In this section we investigate how to choose k in equation (6.4) so as to end up as close as possible to the state $|S\rangle$. We will investigate the two possibilities where M is known or unknown.

6.3.1 M is Known

Let us first assume that M is known. We again divide this case into three subcases.

- $M = 1$ (i.e. $A = \{x^*\}$) and N relatively large.

In this first case, $\sin(\theta_0) = \frac{1}{\sqrt{N}}$ and thus $\theta_0 \approx \frac{1}{\sqrt{N}}$. We target $\sin\left((2k+1)\theta_0\right) = 1$ to require $(2k+1)\theta_0 = \frac{\pi}{2}$. Therefore, we should choose $k = \left\lfloor \frac{\pi}{4}\sqrt{N} - \frac{1}{2} \right\rfloor$. Now, let x be the output state where we consider the previous choice of k. We obtain:

$$\mathbb{P}(x = x^*) = \left| \left\langle S \middle| \psi^{(k)} \right\rangle \right|^2 = \sin\left((2k+1)\theta_0\right)^2 = 1 - \mathcal{O}\left(\frac{1}{N}\right). \tag{6.5}$$

Thus, Grover's algorithm finds $x = x^*$ with high probability in $k = \mathcal{O}(\sqrt{N})$ calls to the oracle U_f. This is much less than the classical case of $\mathcal{O}(N)$ calls.

- Special case of $M = \frac{N}{4}$.

In this case, $\sin(\theta_0) = \sqrt{\frac{M}{N}} = \frac{1}{2}$ so that $\theta_0 = \frac{\pi}{6}$. Therefore, $\sin\left((2k+1)\theta_0\right) = \frac{\pi}{2}$ for $k = 1$! Thus a single iteration suffices to reach **exactly** the state $|S\rangle$, i.e. $\mathbb{P}(x \in A) = 1$.

- General M.

First, if $M \geq \frac{3}{4}N$ then $\mathbb{P}(\text{success}) \geq \frac{3}{4}$ with a classical algorithm and a single call to the oracle f. Assume therefore that $M < \frac{3}{4}N$, where this means that $\sin(\theta_0) < \frac{\sqrt{3}}{2}$ and hence $\theta_0 < \frac{\pi}{3}$. Then choose $k = \left\lfloor \frac{\pi}{4\theta_0} \right\rfloor$.

Proposition 6.3.1. *In this case,* $\mathbb{P}(\textit{success}) \geq \frac{1}{4}$.

Note that we can make this probability arbitrarily close to 1 by repeating the experiment multiple times.

Proof. By design, $k = \frac{\pi}{4\theta_0} - \frac{1}{2} + \delta$ with $|\delta| < \frac{1}{2}$. Hence, $(2k+1)\theta_0 = \frac{\pi}{2} + 2\delta\theta_0$ with $2|\delta|\theta_0 < 2|\delta|\frac{\pi}{3} < \frac{\pi}{3}$. In other words,

$$\mathbb{P}(\text{success}) = \sin\left((2k+1)\theta_0\right)^2 > \sin\left(\frac{\pi}{2} - \frac{\pi}{3}\right)^2 = \sin\left(\frac{\pi}{6}\right)^2 = \frac{1}{4}. \tag{6.6}$$

□

6.3.2 M is Unknown

Suppose now that M is unknown, then it seems that choosing k in this case is impossible. Let us apply the following algorithm:

- Choose $x \in \{0,1\}^n$ uniformly at random, where if it turns out that $x \in A$ then we are done.
- Choose $k \in \{0, ..., \sqrt{N} - 1\}$ uniformly at random and apply k iterations of $G = RU_f$. Finally, output the state measured.

Proposition 6.3.2. *In this case again,* $\mathbb{P}(\textit{success}) \geq \frac{1}{4}$.

Proof. If $M \geq \frac{3}{4}N$ then the first step is successful with probability $\mathbb{P} \geq \frac{3}{4} \geq \frac{1}{4}$. Assume therefore that $M < \frac{3}{4}N$. Then in this case we find:

$$\mathbb{P}(\text{success}) = \sum_{k=0}^{\sqrt{N}-1} \mathbb{P}(\text{success}|K=k)\mathbb{P}(K=k) = \frac{1}{\sqrt{N}} \sum_{k=0}^{\sqrt{N}-1} \mathbb{P}(\text{success}|K=k). \tag{6.7}$$

Remark now that $\mathbb{P}(\text{success}|K=k) = \sin\left((2k+1)\theta_0\right)^2$. Thus, it can be shown (by induction) that the following summation result holds (or by using trigonometric identities):

$$\mathbb{P}(\text{success}) = \frac{1}{\sqrt{N}} \sum_{k=0}^{\sqrt{N}-1} \sin\left((2k+1)\theta_0\right)^2 = \frac{1}{2} - \frac{\sin\left(4\theta_0\sqrt{N}\right)}{4\sqrt{N}\sin(2\theta_0)}. \tag{6.8}$$

But, $|\sin\left(4\theta_0\sqrt{N}\right)| < 1$ and $\sin(2\theta_0) = 2\sin(\theta_0)\cos(\theta_0) = 2\sqrt{\frac{M}{N}}\sqrt{\frac{N-M}{N}} > 2\sqrt{\frac{M}{N}} \cdot \frac{1}{4} > \sqrt{\frac{M}{N}} \geq \frac{1}{\sqrt{N}}$. Therefore,

$$\mathbb{P}(\text{success}) \geq \frac{1}{2} - \frac{1}{4} = \frac{1}{4}. \tag{6.9}$$

□

6.3.3 Conclusion and Applications

Even if M is not known, using Grover's circuit a random number of times ($< \sqrt{N}$) outputs a state $x \in A$ with probability $\mathbb{P} \geq \frac{1}{4}$. By repeating the experiment, this success probability can be amplified arbitrarily close to 1.

We can now mention two applications of this algorithm, where we have seen that we should be able to build the circuit U_f.

(1) SAT formulas: Let us consider a Boolean function f with $n = 4$ of the form:

$$f(x_1, x_2, x_3, x_4) = (x_1 \vee \overline{x}_2) \wedge (\overline{x}_1 \vee x_3 \vee x_4), \tag{6.10}$$

where as usual $\vee$ denotes an OR and $\wedge$ denotes an AND. Such Boolean functions are called *SAT formulas* (SAT as in "satisfiability"). When n is large and the number m of clauses (= expressions in parentheses) of the formula is also large, it is unclear how to find value(s) of x such that $f(x) = 1$. Nevertheless, it is straightforward to implement the circuit U_f associated to f.

(2) Factoring: There is a (non-trivial) way to apply Grover's algorithm in order to reduce the search space for factoring large values of N into product of primes. The improvement is not exponential, but still quadratic, which is noticeable.

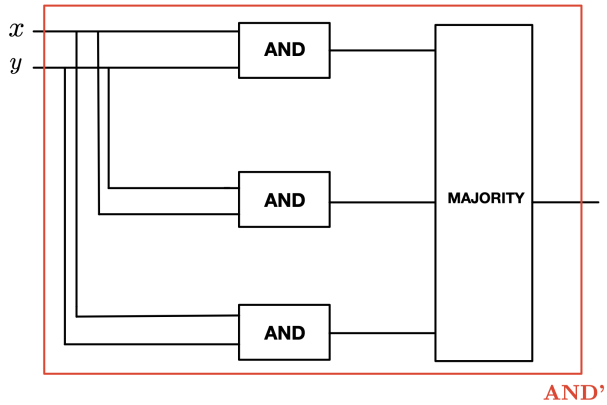
Chapter 7

Classical Error Correction

This chapter presents the fundamental ideas in classical error correction. We will first introduce the general problems that can arise in classical codes and some methods to solve these errors. We will then restrict ourselves to classical binary codes of length n by arguing that error correction is equivalent to solving a Sphere Packing Problem in n-dimensions. We will finally present some concrete ways to characterize a more restrictive class of codes, which are linear codes, and see how to construct them using the generator matrix point of view, the parity-check point of view and finally the construction of Hamming codes. We end this chapter by mentioning the syndrome decoding procedure for decoding linear codes.

7.1 Classical Error Correction

Consider a circuit with AND, OR, NOT gates where each component has probability p of failing (assume independence and that this probability is the same for all components). The first idea is to use a **repetition** procedure. For instance, consider the circuit below where we want to test an AND gate by repeating multiple times its outputs, after which we take the majority of outputs (i.e. majority of 1s or 0s) to then produce the most likely output.



In this circuit we test with three AND gates to produce a larger gate, called AND', and take the majority of outputs with the MAJORITY gate. Now, this new gate fails when two or three outputs from the AND gates yield the wrong true value. Thus, the probability of failure of this gate is $p' := cp^2$ for some constant c. We want $p' < p$, thus p must satisfy $p < \frac{1}{c}$. More precisely,

$$\mathbb{P}(\text{Failure of AND'}) = \mathbb{P}(\text{All three AND gates fail } \underline{\text{or}} \text{ two of them fail}) = p^3 + 3p^2(1-p) := p'_{\text{AND}}.$$

Theorem 7.1.1 (Threshold Theorem for an AND gate)**.** *If it is possible to build an AND gate with $p < \frac{1}{c}$, then it is possible to build an AND' with $p' < p$. We may also repeat this procedure an arbitrary number of times, to create the gates AND",...,AND$^{(k)}$ with respective probabilities $p'', ..., p^{(k)} \xrightarrow{k \longrightarrow \infty} 0$.*

We understand that the idea of this procedure is to repeat the process of gate testing in order to obtain smaller probabilities of failure. Note that, in the context of the above theorem, $p'' = c(p')^2 = c(cp^2)^2 = \frac{1}{c}(cp)^4$ and in general $p^{(k)} = \frac{1}{c}(cp)^{2^k}$. For our example, $p'_{\text{AND}} := p^3 + 3p^2(1-p) = 3p^2 - 2p^3 < p$ **if** $p < \frac{1}{2}$ (Threshold Theorem). Of course, one caveat in this story, we still need to find a way to construct/build the MAJORITY gate.

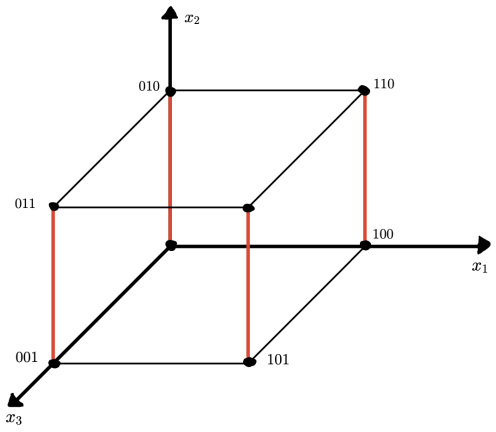
Now, instead of circuits, let us think about **transmission of information**. Suppose that an input $x \in \{0,1\}$ goes through a noisy channel C, where the output that you observe is a $y \in \{0,1\}$ with $\mathbb{P}(x = y) = 1 - p$ for $0 < p < \frac{1}{2}$ small. To study this noisy system, we can think in terms of **repetition codes** to introduce redundancy into the message. As an example, consider a repetition code of length 3 where we transform bit $0 \mapsto 000$ and $1 \mapsto 111$. These new inputs now go through the same channel C and we observe an output $y_1y_2y_3$ (independent for each bit). The question is, how to retrieve x from $y_1y_2y_3$? Here, we can apply the majority rule.

Example 7.1.2. *As an example of the majority rule, we can consider two cases.*
(a) $y_1y_2y_3 = 110$ thus the output is 1.
(b) $y_1y_2y_3 = 010$ thus the output is 0.

For the probability that we make a mistake, it will take into account the case of having three bit flips and the case of having two bit flips. Thus it is given by,

$$\mathbb{P}(\text{output} = 1 | x = 0 \text{ is sent}) = \mathbb{P}(\text{output} = 0 | x = 1 \text{ is sent}) = p^3 + 3p^2(1-p) < p, \quad \text{if } p < \frac{1}{2}. \tag{7.1}$$

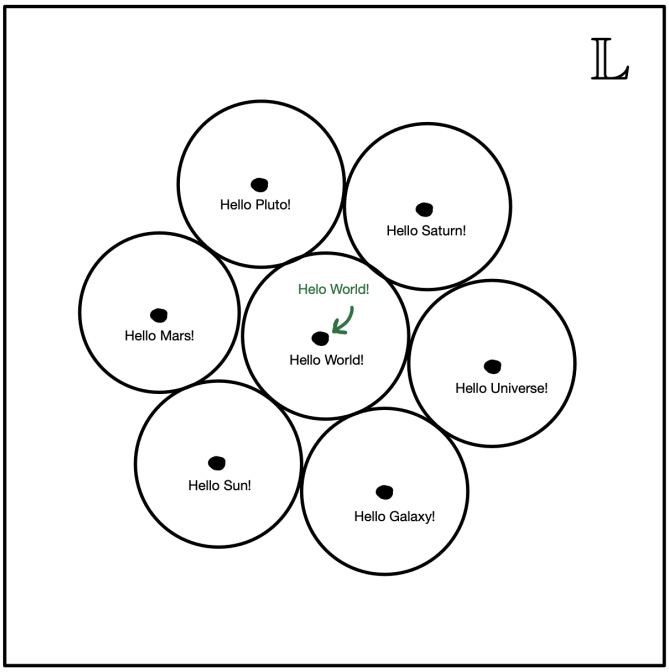
We may now introduce some parameters for more general models. Set $\boldsymbol{n}$ **= length of codewords** (in the example above $n = 3$), $\boldsymbol{r}$ **= rate** (in the example above $r = \frac{1}{3}$ where 3 bits were sent for 1 bit of information) and $\boldsymbol{d}$ **= distance** corresponding to the number of different bits in the codewords (in the example above $d = 3$). We want both a large r to obtain lots of information/sec and a large d for a good error correction. For this last parameter the image to have in mind is,



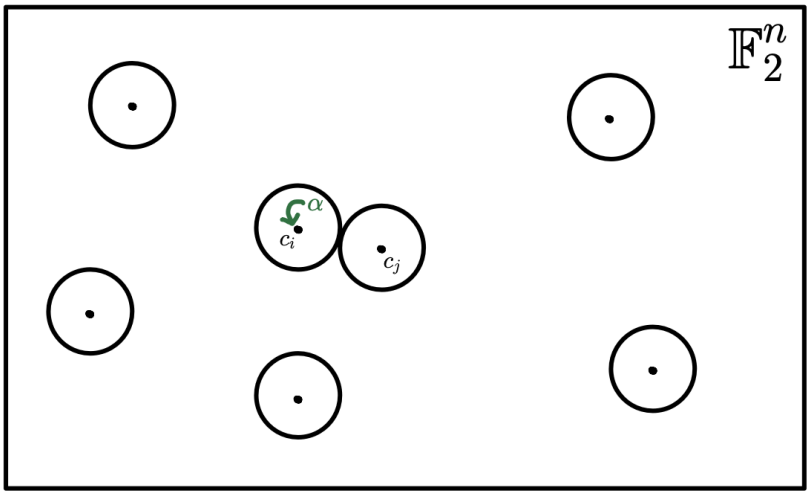
where the distance corresponds to the minimum number of edges to go from one codeword to an other (in the example, from 000 to 111 we need $d = 3$ edges).

7.2 Classical Binary Codes of Length n

Binary codes of length n are codes $\mathcal{C}$ corresponding to a subset of $\mathbb{F}_2^n = \{0,1\}^n$. In order to transmit k information bits we require $|\mathcal{C}| = 2^k$ where $k < n$. In this framework, codewords should be separated by distance $d \geq 2pn$ where pn is the average number of errors on one codeword. For the **decoding procedure**, we look for the nearest neighbor of the received sequence of bits. To gain some intuition on why we are looking for the nearest neighbor, let us forget binary for a second and concentrate on a space $\mathbb{L}$ of letters. The following figure shows that the procedure of error correction, seen here in the context of a typo in a sequence of letters, assigns the error "Helo World!" to the actual codeword "Hello World!".



Going back now to binary, a code can be seen as a collection of codewords $\mathcal{C} = \{c_1, ..., c_{2^k}\}$. Of course, to view the space $\mathbb{F}_2^n$ one would need to find a way to represent an n-dimensional cube, however we can simplify the visualization by viewing this space in 2-dimensions. In the following figure, $\{c_i\}_i$ are the codewords with their error balls around them and α is the received sequence of bits. Since it landed in the error ball of c_i, error correction has associated α with the codeword c_i.



In this configuration, we set the distance d to be the smallest distance between any two codewords c_i and c_j,

$$d = \min\{\text{distance}(c_i, c_j) : c_i, c_j \in \mathcal{C}, c_i \neq c_j\}. \tag{7.2}$$

Now, $\boldsymbol{\mathcal{C}}$ **can correct up to** $\boldsymbol{\left\lfloor \frac{d-1}{2} \right\rfloor}$ **errors**. The name of the game is to place the 2^k codewords in $\mathbb{F}_2^n$ so that the minimum distance d is the largest possible. The largest d becomes the more we can maximize the error ball around the codewords $\{c_i\}_i$ with no overlap between two error balls, so that we cover a larger portion of the space. From this point on, the game of arranging those error balls in $\mathbb{F}_2^n$ is a Sphere Packing Problem! Indeed, in the following figure, the packing on the right is more dense than the one one the left. Thus, it will cover a larger portion of errors in the space $\mathbb{F}_2^n$.

Remark 7.2.1. *Remark that this Sphere Packing formalism is a rather crude simplification of real life situations (it works well for small dimensions). In principle, especially in higher dimensions, one could think of more general partitionings of the space without explicitly considering n-dimensional spheres.*

We can then see that there are three important parameters for the code, which are (n, k, d) [we have also seen the rate of transmission $r = \frac{k}{n} \leq 1$]. It is crucial to note that we cannot choose (n, k, d) as we like as they are in a tradeoff situation.

Proposition 7.2.2 (Singleton bound)**.** *The tradeoff of (n, k, d) is given by $k + d \leq n + 1$.*

Since there are a lot of codewords in $\mathcal{C}$ we will need some structure for the rest of the theory. From now on, we will focus on **linear codes** which are codes that satisfy,

$$c_i, c_j \in \mathcal{C} \Longrightarrow c_i \oplus c_j \in \mathcal{C}, \tag{7.3}$$

which corresponds to a subspace of $\mathcal{C}$. Note that the XOR $\oplus$ of two codes will be the bit-wise XOR of the codes (e.g. $001 \oplus 101 = 100$).

7.3 Generator Matrix, Parity Check and Hamming Codes

As mentioned at the end of the previous section, we now focus on **linear codes** which are codes satisfying,

$$c_i, c_j \in \mathcal{C} \Longrightarrow c_i \oplus c_j \in \mathcal{C}. \tag{7.4}$$

In this section we will briefly present some characterization of linear codes (some ways of constructing them) with the generator matrix and a parity check before presenting the general ideas of Hamming codes.

- From the point of view of the **generator**, we define G to be a $k \times n$ generator matrix and consider

a code $\mathcal{C}$ to be the set $\mathcal{C} = \{c \in \mathbb{F}_2^n : c = u \cdot G, u \in \mathbb{F}_2^k\}$. Thus, a code $\mathcal{C}$ corresponds to the row space of G (i.e. the rows of G and linear combinations of them correspond to the codewords).

Example 7.3.1. *For the repetition (linear) code $\mathcal{C} = \{000, 111\}$ we have that $n = 3$ and $k = 1$. Consider then the matrix $G = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$ and take $u = \begin{pmatrix} 0 \end{pmatrix} \in \mathbb{F}_2$ or $u = \begin{pmatrix} 1 \end{pmatrix} \in \mathbb{F}_2$.*

• From the point of view of a **parity check**, we define H to be the $(n-k) \times n$ parity check matrix and consider a code $\mathcal{C}$ to be the set $\mathcal{C} = \{c \in \mathbb{F}_2^n : H \cdot c^T = 0\}$.

Example 7.3.2. *For the repetition (linear) code $\mathcal{C} = \{000, 111\}$ we have that $n = 3$, $k = 1$ and $n - k = 2$. Consider then the matrix*

$$H = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}, \tag{7.5}$$

and note that for both $c = \begin{pmatrix} 0 & 0 & 0 \end{pmatrix}$ and $c = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$ we have that $H \cdot c^T = 0$.

• We finally present the notion of **Hamming codes**, where we first define the *Hamming distance* in $\mathbb{F}_2^n$ to be,

$$d_H(c, c') = \big|\{1 \leq i \leq n : c_i \neq c'_i\}\big|, \quad \forall c, c' \in \mathbb{F}_2^n. \tag{7.6}$$

Thus, in other words, the Hamming distance between x and y is just the number of places where x and y differ (e.g. $d_H((1,1,0,0),(0,1,0,1)) = 2$). Also, we define the *distance of a code* $\widetilde{d}$ to be the minimum distance d between any two codewords,

$$\widetilde{d} = \min d(\mathcal{C}) = \min_{\substack{c_j, c_k \in \mathcal{C} \\ j \neq k}} d_H(c_j, c_k). \tag{7.7}$$

For linear codes, it turns out that the minimum distance corresponds to the minimum weight (i.e. number of 1s) of a non-zero codeword, as $d_H(c_i, c_j) = d(0, c_i \oplus c_j)$ for i, j where $c_i \oplus c_j \in \mathcal{C}$ and $c_i \neq c_j$ if and only if $c_i \oplus c_j \neq 0$. Mathematically, this is formally given by,

$$\widetilde{d} = \min_{\substack{c_i, c_j \in \mathcal{C} \\ i \neq j}} d_H(c_i, c_j) = \min_{\substack{c_i \in \mathcal{C} \\ c_i \neq 0}} |c_i|, \tag{7.8}$$

where $|x|$ is the *Hamming weight* of $x \in \mathbb{F}_2^n$ and corresponds to $|x| = \{\#1\text{s in vector } x\} = d(0, x)$. The idea of a Hamming code is to set a Hamming matrix where column j corresponds to the binary expansion of j.

Example 7.3.3. *Let $k = 4$, $n - k = 3$ and thus $n = 2^{n-k} - 1 = 7$. The Hamming matrix in this case is given by,*

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}. \tag{7.9}$$

Proposition 7.3.4. *For Hamming codes, the minimum distance of a code is $d = 3$.*

Proof. Recall that for linear codes, the minimum distance corresponds to the minimum weight (i.e. number of 1s) of a non-zero codeword, as $d_H(c_i, c_j) = d(0, c_i \oplus c_j)$ for i, j where $c_i \oplus c_j \in \mathcal{C}$ and $c_i \neq c_j$ if and only if $c_i \oplus c_j \neq 0$.
Now, $Hc^T = 0$ implies that at least weight$(c) \geq 1$ as H does not have a column of 0s. But is is also the case that weight$(c) \geq 2$ as H does not have identical columns. Finally, if weight$(c) = 3$ then it is indeed possible that $Hc^T = 0$ $\left[\text{e.g. take } c = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}\right]$. Thus, $\min d(\mathcal{C}) = 3$. □

Now, to perform error correction with this code we can use **syndrome decoding**. The idea is to assume that the received message is in the form $y = c + e$ where $c \in \mathcal{C}$ is a codeword and e is an error vector with a single 1 in it (to denote the bit flip). Note here that $c + e$ may be understood as $c \oplus e$ since we are working in $\mathbb{F}_2^n$. Then, recall that $Hc^T = 0$ and thus,

$$Hy^T = H(c^T + e^T) = \underbrace{Hc^T}_{=0} + He^T = He^T, \tag{7.10}$$

where He^T is the binary decomposition of the position of the errors. In other words, knowing the error e will tell us the binary decomposition of the position of the error. He^T is called the **syndrome**.

Example 7.3.5. *If $e = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$ then $He^T = \begin{pmatrix} 0 & 1 & 1 \end{pmatrix}^T$ is corresponding to the binary decomposition of the number 3. Thus in this case, we know that the error occurred in position 3.*

In conclusion, we have seen that for a $k \times n$ generator matrix G and an $(n-k) \times n$ parity-check matrix H such that $GH^T = 0$ we have that,

$$\mathcal{C} = \{x \in \mathbb{F}_2^n : x = u \cdot G, u \in \mathbb{F}_2^k\} = \{x \in \mathbb{F}_2^n : Hx^T = 0\}. \tag{7.11}$$

We have also seen that we could use Hamming codes for encoding and make use of syndrome decoding for the decoding procedure.

Chapter 8

Quantum Error Correction

This chapter explains how to do quantum information processing reliably in the presence of noise. We begin by developing the basic theory of quantum error-correcting codes with the bit-flip and the phase-flip error models. We will then introduce the Shor code to deal with both bit-flips and phase-flips. Finally, we will briefly study the Steane code which is more efficient than Shor's code.

One could think that an equivalent argument of performing repetition on classical code is to consider the entangled system $|\phi\rangle \otimes |\phi\rangle \otimes |\phi\rangle$ where $|\phi\rangle = \alpha|0\rangle + \beta|1\rangle$. However this cannot happen by the No-cloning Theorem (cf. Remark 2.4.4)! Instead, our repetition code will consist of taking $|0\rangle \mapsto |000\rangle$ and $|1\rangle \mapsto |111\rangle$. This isn't cloning since only the computational basis may be copied. Thus, for a general state $|\phi\rangle$, we will consider its repetition equivalent to be our "codewords" given by,

$$|\phi\rangle = \alpha|0\rangle + \beta|1\rangle \mapsto |\psi\rangle = \alpha|000\rangle + \beta|111\rangle. \tag{8.1}$$

In the first part of this chapter we consider two error models, one involving bit-flips (via an X gate) and the other involving phase flips (via a Z gate). Recall from Definition (2.5.3) that a Z gate is defined as,

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{8.2}$$

8.1 The Bit-Flip Error Model

This error model is very similar in its construction as its classical counterpart. We first start with error detection.

Example 8.1.1 (Bit-flip in position 1)**.** *Consider a bit flip in position 1, then for an input state* $|\psi\rangle = \alpha|000\rangle + \beta|111\rangle$ *the error will yield,*

$$|\psi\rangle \mapsto |\psi'\rangle = X_1|\psi\rangle = (X \otimes \mathbb{I} \otimes \mathbb{I})|\psi\rangle = \alpha|100\rangle + \beta|011\rangle. \tag{8.3}$$

Now, the quantum equivalent of syndromes are given by measurements using the observables $Z_1Z_2 := Z \otimes Z \otimes \mathbb{I}$ and $Z_2Z_3 := \mathbb{I} \otimes Z \otimes Z$ with possible eigenvalues $\lambda_\pm = \pm 1$. As an aside, remember the parity-check matrix of the classical repetition code,

$$H = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}, \tag{8.4}$$

where now the first row $\begin{pmatrix} 1 & 1 & 0 \end{pmatrix}$ corresponds to the observable Z_1Z_2 and the second row $\begin{pmatrix} 0 & 1 & 1 \end{pmatrix}$ corresponds to the observable Z_2Z_3.

• Assume no error has happened such that $|\psi'\rangle = \alpha|000\rangle + \beta|111\rangle$. Then, the actions of the observables Z_1Z_2 and Z_2Z_3 on $|\psi'\rangle$ are,

$$Z_1Z_2|\psi'\rangle = \alpha|000\rangle + (-1)(-1)\beta|111\rangle = \alpha|000\rangle + \beta|111\rangle = (+1)|\psi'\rangle. \tag{8.5}$$

$$Z_2Z_3|\psi'\rangle = \alpha|000\rangle + (-1)(-1)\beta|111\rangle = \alpha|000\rangle + \beta|111\rangle = (+1)|\psi'\rangle. \tag{8.6}$$

• Assume now that one bit was flipped (without loss of generality assume the first one) such that $|\psi'\rangle = \alpha|100\rangle + \beta|011\rangle$. Then, the actions of the observables Z_1Z_2 and Z_2Z_3 on $|\psi'\rangle$ are,

$$Z_1Z_2|\psi'\rangle = -\alpha|100\rangle - \beta|011\rangle = (-1)|\psi'\rangle. \tag{8.7}$$

$$Z_2Z_3|\psi'\rangle = \alpha|100\rangle + \beta|011\rangle = (+1)|\psi'\rangle. \tag{8.8}$$

In summary, with the measurements of the **stabilizers** Z_1Z_2 and Z_2Z_3 we obtain:

$$(Z_1Z_2, Z_2Z_3) \equiv (+1,+1) \longleftrightarrow \text{no bit flip}$$

$$(Z_1Z_2, Z_2Z_3) \equiv (-1,+1) \longleftrightarrow \text{bit flip in position 1}$$

$$(Z_1Z_2, Z_2Z_3) \equiv (-1,-1) \longleftrightarrow \text{bit flip in position 2}$$

$$(Z_1Z_2, Z_2Z_3) \equiv (+1,-1) \longleftrightarrow \text{bit flip in position 3}$$

The brackets (Z_1Z_2, Z_2Z_3) are considered as our new syndromes. Finally observe that $|\psi'\rangle$ is an eigenvector of Z_1, Z_2 and Z_3, as a result we have no state perturbation!
We now move on with error correction. In this case it is relatively straightforward.

$$(Z_1Z_2, Z_2Z_3) \equiv (+1,+1) \longrightarrow \text{do nothing}$$

$$(Z_1Z_2, Z_2Z_3) \equiv (-1,+1) \longrightarrow \text{apply } X_1$$

$$(Z_1Z_2, Z_2Z_3) \equiv (-1,-1) \longrightarrow \text{apply } X_2$$

$$(Z_1Z_2, Z_2Z_3) \equiv (+1,-1) \longrightarrow \text{apply } X_3$$

After applying X_1, X_2 or X_3 we will get back the state $|\psi\rangle = \alpha|000\rangle + \beta|111\rangle$. Note that X_1, X_2, X_3 need to be applied **after** the Zs to $|\psi'\rangle$, which is **not** an eigenvector of X_1, X_2, X_3.

In summary, for a bit-flip, $|\psi\rangle = \alpha|000\rangle + \beta|111\rangle \mapsto |\psi'\rangle = X_1|\psi\rangle = \alpha|100\rangle + \beta|011\rangle$. For error detection we find $Z_1Z_2|\psi'\rangle = (-1)|\psi'\rangle$ and $Z_2Z_3|\psi'\rangle = (+1)|\psi'\rangle$. Finally, for error correction set $X_1|\psi'\rangle = X_1X_1|\psi\rangle = X_1^2|\psi\rangle = |\psi\rangle$.

8.2 The Phase-Flip Error Model

Recall that the action of Z on the computational basis states is $Z|0\rangle = |0\rangle$ and $Z|1\rangle = -|1\rangle$. For a phase-flip, set a new code in the form $|0\rangle \mapsto |+++\rangle$ and $|1\rangle \mapsto |---\rangle$ such that $|\psi\rangle = \alpha|+++\rangle + \beta|---\rangle$. Now, the action of Z on $|+\rangle$ and $|-\rangle$ is $Z|+\rangle = |-\rangle$ and $Z|-\rangle = |+\rangle$ (we are in the same scenario as before).
Now, suppose that we have a phase-flip in the first qubit. Then,

$$Z_1|\psi\rangle = \alpha|-++\rangle + \beta|+--\rangle. \tag{8.9}$$

For error detection, we will use the observables X_1X_2 and X_2X_3. For error correction, if we have $(-1,+1)$ we apply Z_1 to recover the original state.

8.3 Shor's Code

In the two previous sections we have seen how to deal with bit-flip errors and phase-flip errors on their own. But how do we handle the situation if we have bit-flip and phase-flip errors together? For this, we will use Shor's code which corresponds to a concatenation of the two previous codes. The idea can be seen in two steps.

- The following transformations are useful to deal with phase-flips:

$$|0\rangle \mapsto |+++\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle+|1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle+|1\rangle}{\sqrt{2}} \tag{8.10}$$

$$|1\rangle \mapsto |---\rangle = \frac{|0\rangle-|1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle-|1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle-|1\rangle}{\sqrt{2}} \tag{8.11}$$

- The following transformations are useful to deal with bit-flips:

$$|0\rangle \mapsto \frac{|000\rangle+|111\rangle}{\sqrt{2}} \otimes \frac{|000\rangle+|111\rangle}{\sqrt{2}} \otimes \frac{|000\rangle+|111\rangle}{\sqrt{2}} =: |0\rangle_{\text{Shor}} \tag{8.12}$$

$$|1\rangle \mapsto \frac{|000\rangle-|111\rangle}{\sqrt{2}} \otimes \frac{|000\rangle-|111\rangle}{\sqrt{2}} \otimes \frac{|000\rangle-|111\rangle}{\sqrt{2}} =: |1\rangle_{\text{Shor}} \tag{8.13}$$

Here, we note that $k=1$ and $n=9$ and codewords are of the form $|\psi\rangle = \alpha|0\rangle_{\text{Shor}} + \beta|1\rangle_{\text{Shor}}$.

Proposition 8.3.1. *Shor's code protects against a bit-flip and/or a phase-flip.*

Proof. Consider an initial state $|\psi\rangle = \alpha|0\rangle_{\text{Shor}} + \beta|1\rangle_{\text{Shor}}$ and suppose that the output state $|\psi'\rangle$ suffered from a bit-flip and/or a phase-flip. For the bit-flip, we may consider the stabilizers $Z_1Z_2, Z_2Z_3, Z_4Z_5, Z_5Z_6, Z_7Z_8$ and Z_8Z_9. These measurements will not perturb the state and they will indeed provide some information on the bit-flip. Next we can consider the stabilizers $X_1X_2X_3X_4X_5X_6$ and $X_4X_5X_6X_7X_8X_9$ such that they will provide information on the phase-flip. Again, these stabilizers will not perturb the state.

□

Remark that these operators commute and $|\psi'\rangle$ is an eigenvector of all of them.

For error correction, the game is just to apply the correct Z gate or X gate. To see how this goes, let us consider some examples.

Example 8.3.2 (Bit-flip error on bit 3)**.** *In this case, only Z_2Z_3 has eigenvalue -1. Thus one applies X_3 to correct the error.*

Example 8.3.3 (Phase-flip error on bit 5)**.** *In this case, both $X_1X_2X_3X_4X_5X_6$ and $X_4X_5X_6X_7X_8X_9$ have eigenvalue -1. Thus one doesn't know where the phase-flip occurred among bits 4,5 or 6, however one can still correct the error by applying $Z_4Z_5Z_6$.*

Example 8.3.4 (Bit-flip and phase-flip error on bit 4)**.** *In this case, Z_4Z_5, $X_1X_2X_3X_4X_5X_6$ and $X_4X_5X_6X_7X_8X_9$ have eigenvalue -1. This bit-phase flip can be corrected by applying both X_4 and Z_4 [note that depending on the order of application, this might generate a global (-1) phase since $X_4Z_4 = -Z_4X_4$. However, the error will be corrected anyway.]*

8.4 Steane's Code

Recall the Hamming code Hamming$(7,4,3)$ with its parity-check matrix,

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}, \tag{8.14}$$

along with the generator matrix,

$$G = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}. \tag{8.15}$$

We recall the sets $\mathcal{C}_H = \{c = u \cdot G, u \in \mathbb{F}_2^4\} = \{c \in \mathbb{F}_2^7 : Hc^T = 0\}$ and $\mathcal{C}_H^\perp = \{c = u \cdot H, u \in \mathbb{F}_2^3\}$. Steane's code is more efficient than Shor's code. The insight is to take the classical Hamming code (which corrects single-bit flips) and turn it into a quantum code via the CSS (Calderbank–Shor–Steane) construction. The Hamming code (length 7, dimension 4) has a (3×7) parity-check matrix H. Its row-space (dimension 3) is a classical code of distance 4. In CSS language, one chooses two nested classical codes $C_2 \subset C_1$ here with $C_1 = \text{Hamming}(7,4,3)$ and $C_2 = C_1^\perp$. This nesting gives rise to a Steane$(7,1,3)$ quantum code. The transformations $|0\rangle_S$ and $|1\rangle_S$ are thus superpositions of the 8 classical length-7 codewords in C_2. We may set the following transformations for $|0\rangle_S$ and $|1\rangle_S$, where we are looking for the row-space of H:

$$\begin{aligned} |0\rangle \mapsto |0\rangle_S &= \frac{1}{\sqrt{8}} \sum_{c \in \mathcal{C}_H^\perp} |c\rangle \\ &= \frac{1}{\sqrt{8}} \big(|0000000\rangle + |1010101\rangle + |0110011\rangle + |0001111\rangle + |0111100\rangle + |1011010\rangle + |1101001\rangle + |1100110\rangle \big) \end{aligned}$$

$$|1\rangle \mapsto |1\rangle_S = X|0\rangle_S = \frac{1}{\sqrt{8}} \sum_{c\in\mathcal{C}_H^\perp} |c \oplus 1111111\rangle$$
$$= \frac{1}{\sqrt{8}}(\,|1111111\rangle + |0101010\rangle + |1001100\rangle + |1110000\rangle + |1000011\rangle + |0100101\rangle + |0010110\rangle + |0011001\rangle\,)$$

The corresponding stabilizers (operators that leave the code invariant) are given by $g_1 = X_4X_5X_6X_7$, $g_2 = X_2X_3X_6X_7$, $g_3 = X_1X_3X_5X_7$ and $g_4 = Z_4Z_5Z_6Z_7$, $g_5 = Z_2Z_3Z_6Z_7$, $g_6 = Z_1Z_3Z_5Z_7$ (these correspond to the 1's in the matrix H). Note that these stabilizers form a **group S**, which is a group generated by these stabilizers.

Proposition 8.4.1. *The stabilizers commute, i.e. $g_ig_j = g_jg_i$ for all i, j.*

Proof. We will show this for a specific example, where the rest may be shown in a similar way. Recall that $X_iZ_i = -Z_iX_i$,

$$g_2g_6 = X_2X_3X_6X_7Z_1Z_3Z_5Z_7 = Z_1X_2X_3Z_3Z_5X_6X_7Z_7 = Z_1X_2Z_3X_3Z_5X_6Z_7X_7 = g_6g_2. \quad (8.16)$$

In the second equality we have just rearranged the gates in order of their action. For the general case, remark that there is always an even number of common X's and Z's between two stabilizers. □

Proposition 8.4.2. *Codewords are invariant to stabilizers, i.e. $g_i|0\rangle_S = |0\rangle_S$ and $g_i|1\rangle_S = |1\rangle_S$.*

Proof. We again prove this for a specific example, where the rest may be shown in a very similar way.

$$g_1|0\rangle_S = g_1\frac{1}{\sqrt{8}}\sum_{c\in\mathcal{C}_H^\perp}|c\rangle = \frac{1}{\sqrt{8}}\sum_{c\in\mathcal{C}_H^\perp}|c\oplus 0001111\rangle = \frac{1}{\sqrt{8}}\sum_{c\in\mathcal{C}_H^\perp}|c\rangle\,,$$

where we have used the fact that $|c\oplus 0001111\rangle$ is an element of the group $\mathcal{C}_H^\perp$. For $|1\rangle_S$ one finds,

$$\begin{aligned} g_1|1\rangle_S = X_4X_5X_6X_7\frac{1}{\sqrt{8}}\sum_{c\in\mathcal{C}_H^\perp}|c\oplus 1111111\rangle &= \frac{1}{\sqrt{8}}\sum_{c\in\mathcal{C}_H^\perp}|c\oplus 1111111\oplus 0001111\rangle \\ &= \frac{1}{\sqrt{8}}\sum_{c\in\mathcal{C}_H^\perp}|c\oplus 0001111\oplus 1111111\rangle \\ &= \frac{1}{\sqrt{8}}\sum_{c\in\mathcal{C}_H^\perp}|c\oplus 1111111\rangle = |1\rangle_S\,, \end{aligned}$$

where we have used the fact that $c\oplus 0001111$ is an element of the group $\mathcal{C}_H^\perp$. Thus we have invariance for g_1, and the rest follows. We also have an identical reasoning for Z-stabilizers. □

Remark 8.4.3. *Note that the invariance property demonstrates that $|\psi\rangle$ is a codeword of the Steane code if and only if for all $g \in S$, $g(\alpha|0\rangle_S + \beta|1\rangle_S) = g|\psi\rangle = |\psi\rangle$.*

Proposition 8.4.4. *The Steane code can correct any single error, i.e. either* $\{X_i, Y_i, Z_i\}_{i=1}^{7}$.

Proof. We again prove this for a specific example, where the rest may be shown in a very similar way. Recall that the X and Z gates commute between each other. For $g_1 = X_4X_5X_6X_7$, consider a bit-flip in position 7, $|\psi'\rangle = X_7|\psi\rangle$. Then,

$$g_1|\psi'\rangle = g_1X_7|\psi\rangle = X_7g_1|\psi\rangle = X_7|\psi\rangle = |\psi'\rangle, \tag{8.17}$$

where we have used the invariance of codewords to stabilizers [Proposition (8.4.2)].

□

Example 8.4.5 (Syndrome for a bit-flip in position 7)**.** *Let* $|\psi'\rangle = X_7(\alpha|0\rangle_S + \beta|1\rangle_S)$. *Then:* $g_1|\psi'\rangle = (+1)|\psi'\rangle$, $g_2|\psi'\rangle = (+1)|\psi'\rangle$, $g_3|\psi'\rangle = (+1)|\psi'\rangle$, $g_4|\psi'\rangle = (-1)|\psi'\rangle$, $g_5|\psi'\rangle = (-1)|\psi'\rangle$ *and* $g_6|\psi'\rangle = (-1)|\psi'\rangle$. *Thus the syndrome is* $(1,1,1,-1,-1,-1)$. *From the three last positions in the syndrome* $(-1,-1,-1)$, *we get the error* 111 *corresponding to the binary decomposition of 7, i.e. X error in position 7.*

Example 8.4.6 (Other examples of syndromes)**.** *Consider the following:*

- X_3 *error* $\longrightarrow$ *syndrome* $= (1,1,1,\underbrace{1,-1,-1}) \longrightarrow 011 \longrightarrow X$ *error in position 3.*
- Z_4 *error* $\longrightarrow$ *syndrome* $= (\underbrace{-1,1,1},1,1,1) \longrightarrow 100 \longrightarrow Z$ *error in position 4.*
- $Y_6 = iX_6Z_6$ *error* $\longrightarrow$ *syndrome* $= (\underbrace{-1,-1,1}_{110},\underbrace{-1,-1,1}_{110}) \longrightarrow$ *error in position 6.*

8.5 Building Reliable Quantum Gates using the Steane Code

First recall the seven gates we will consider in this section,

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = iXZ = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix},$$

$$S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \quad T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{pmatrix}, \quad CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

With the Steane code we have the following encoding, yielding a 7 qubit state:

$$\alpha|0\rangle + \beta|1\rangle \xrightarrow{\text{encoding}} \alpha|0\rangle_S + \beta|1\rangle_S. \tag{8.18}$$

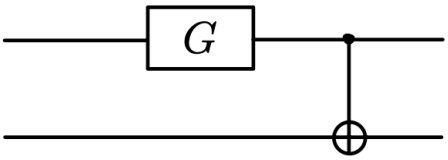
- The **idea** in this section is the following: On the one hand, for some gate G,

$$G(\alpha|0\rangle+\beta|1\rangle) \xrightarrow{\text{encoding}} |\psi\rangle_S, \tag{8.19}$$

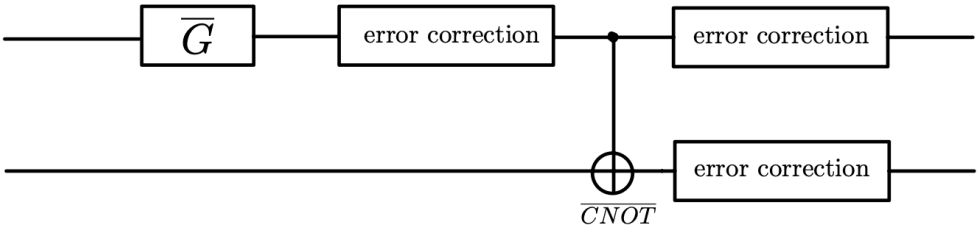
and on the other hand, for some gate $\overline{G}$,

$$\alpha|0\rangle+\beta|1\rangle \xrightarrow{\text{encoding}} \alpha|0\rangle_S+\beta|1\rangle_S \xrightarrow{\overline{G}} \overline{G}(\alpha|0\rangle_S+\beta|1\rangle_S). \tag{8.20}$$

- The **aim** is to find seven gates $\overline{G}=\{\overline{X},\overline{Y},\overline{Z},\overline{H},\overline{S},\overline{T},\overline{CNOT}\}$ that act on all the bits of the Steane code. We want the two outputs in (8.19) and (8.20) to match, such that $|\psi\rangle_S=\overline{G}(\alpha|0\rangle_S+\beta|1\rangle_S)$. We now understand that we want an alternative to first applying a gate G on a state $|\psi\rangle=\alpha|0\rangle+\beta|1\rangle$ and then encoding it, by first encoding the state $|\psi\rangle=\alpha|0\rangle+\beta|1\rangle$ and then applying some gate $\overline{G}$. From the point of view of a circuit, we are looking (for example) to go from a circuit of the form,



to one where we consider a gate $\overline{H}$ as a 7-qubit gate, and $\overline{CNOT}$ as a 14-qubit gate, [error correction gate = decoder gate]



Proposition 8.5.1. *For $U=\{X,Y,Z\}$, $\overline{U}$ is given by $\overline{U}=U_1U_2...U_7=U^{\otimes 7}$.*

Proof. We check this gate by gate.

- For gate X on the one hand, with $|\psi\rangle=\alpha|0\rangle+\beta|1\rangle$,

$$X(\alpha|0\rangle+\beta|1\rangle)=\alpha|1\rangle+\beta|0\rangle \xrightarrow{\text{Steane}} \alpha|1\rangle_S+\beta|0\rangle_S. \tag{8.21}$$

On the other hand,

$$\overline{X}(\alpha|0\rangle_S+\beta|1\rangle_S)=\alpha X^{\otimes 7}|0\rangle_S+\beta X^{\otimes 7}|1\rangle_S=\alpha|1\rangle_S+\beta|0\rangle_S. \tag{8.22}$$

- For gate Z on the one hand, with $|\psi\rangle=\alpha|0\rangle+\beta|1\rangle$,

$$Z(\alpha|0\rangle+\beta|1\rangle)=\alpha|0\rangle-\beta|1\rangle \xrightarrow{\text{Steane}} \alpha|0\rangle_S-\beta|1\rangle_S. \tag{8.23}$$

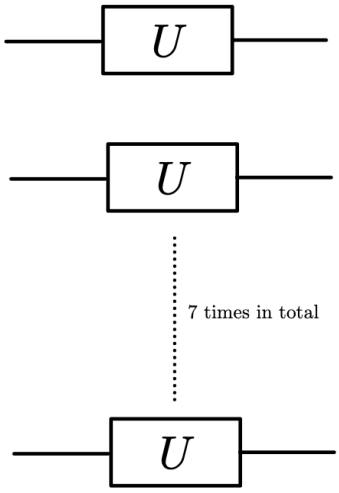
On the other hand,

$$\overline{Z}(\alpha|0\rangle_S+\beta|1\rangle_S)=\alpha Z^{\otimes 7}|0\rangle_S+\beta Z^{\otimes 7}|1\rangle_S=\alpha|0\rangle_S-\beta|1\rangle_S, \tag{8.24}$$

where we have used the facts that $|0\rangle_S$ has an even number of 1's and $|1\rangle_S$ has an odd number of 1's.

- $Y = iXZ$ is also fine since X and Z work independently.

□



Proposition 8.5.2. *For the gate S, $\overline{S}$ is given by $\overline{S} = Z_1 S_1 ... Z_7 S_7 = (ZS)^{\otimes 7}$.*

Proof. Since $S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$, we have for a state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$,

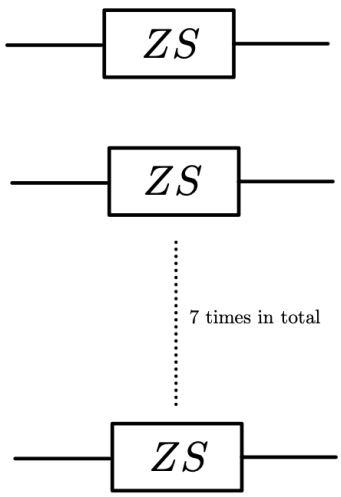
$$S(\alpha|0\rangle + \beta|1\rangle) = \alpha|0\rangle + i\beta|1\rangle \xrightarrow{\text{encoding}} \alpha|0\rangle_S + i\beta|1\rangle_S . \tag{8.25}$$

On the other hand, for $\alpha|0\rangle + \beta|1\rangle \xrightarrow{\text{encoding}} \alpha|0\rangle_S + \beta|1\rangle_S$, we need to find a gate $\overline{S}$ such that,

$$\overline{S}(\alpha|0\rangle_S + \beta|1\rangle_S) = \alpha|0\rangle_S + i\beta|1\rangle_S . \tag{8.26}$$

However, remark that $\overline{S} = S_1 ... S_7 = S^{\otimes 7}$ would be fine for the term $\alpha|0\rangle_S$ (since we get either a factor of $i^0 = 1$ or $i^4 = 1$) but not for $i\beta|1\rangle_S$ (since we would get $-i$ instead of i). Thus, by setting $\overline{S} = Z_1 S_1 ... Z_7 S_7 = (ZS)^{\otimes 7}$ will yield an extra minus sign.

□



Proposition 8.5.3. *For the gate H, $\overline{H}$ is given by $\overline{H} = H_1...H_7 = (H)^{\otimes 7}$.*

Proof. For gate H on the one hand, with $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$,

$$H(\alpha|0\rangle + \beta|1\rangle) = \alpha\left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right) + \beta\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) \xrightarrow{\text{Steane}} \alpha\left(\frac{|0\rangle_S + |1\rangle_S}{\sqrt{2}}\right) + \beta\left(\frac{|0\rangle_S - |1\rangle_S}{\sqrt{2}}\right). \quad (8.27)$$

On the other hand, for an encoding $\alpha|0\rangle + \beta|1\rangle \xrightarrow{\text{Steane}} \alpha|0\rangle_S + \beta|1\rangle_S$ we would like,

$$\overline{H}(\alpha|0\rangle_S + \beta|1\rangle_S) = \alpha\left(\frac{|0\rangle_S + |1\rangle_S}{\sqrt{2}}\right) + \beta\left(\frac{|0\rangle_S - |1\rangle_S}{\sqrt{2}}\right). \quad (8.28)$$

The tricky part here is to make sure that we would get constant flips in the signs (e.g. when it hits a qubit $|1\rangle$). For this, we have to verify two key points. In the following, we will use the conjugation relations $HX = ZH \iff HXH = Z$ and $HZ = XH \iff HZH = X$ (recall $H^2 = \mathbb{I}$).

(i) We need to make sure that $\overline{H}$ keeps codewords in the code (i.e. maps codewords into codewords). For $|\psi\rangle_S = \alpha|0\rangle_S + \beta|1\rangle_S$ we require, for all $g_i \in S$,

$$g_i\overline{H}|\psi\rangle_S = \overline{H}|\psi\rangle_S. \quad (8.29)$$

In other words, this is equivalent to verifying that $\overline{H}g_i\overline{H} \in S$. This is true, since for example $\overline{H}g_1\overline{H} = (H_1...H_7)X_4X_5X_6X_7(H_1...H_7) = (H_4X_4H_4)...(H_7X_7H_7) = Z_4Z_5Z_6Z_7 \in S$. It is not difficult to check that the remaining cases also hold.

(ii) The conjugation relations also hold for $\overline{H}$. Indeed,

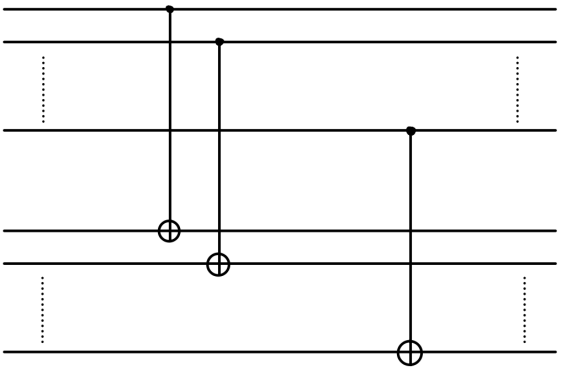
$$\overline{H}\,\overline{X}\,\overline{H} = (H_1...H_7)(X_1...X_7)(H_1...H_7) = (H_1X_1H_1)...(H_7X_7H_7) = Z_1...Z_7 = \overline{Z}. \quad (8.30)$$

$$\overline{H}\,\overline{Z}\,\overline{H} = (H_1...H_7)(Z_1...Z_7)(H_1...H_7) = (H_1Z_1H_1)...(H_7Z_7H_7) = X_1...X_7 = \overline{X}. \quad (8.31)$$

Therefore, from points (i) and (ii) we conclude that relation (8.28) holds.

□

We will not explicitly study $CNOT \mapsto \overline{CNOT}$, however we note that an error in $\overline{CNOT}$ might propagate but it will do so into two separate 7-qubit blocks (which is fine). The circuit for $\overline{CNOT}$ is given by,



Finally, we will not study either the gate $T \mapsto \overline{T}$ as it needs more attention than the other gates.